UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Definición de una ontología integrada de procesos y recursos, para el desarrollo colaborativo de planes de proceso

TESIS DOCTORAL

Presentada por:
Lorenzo Solano García
Dirigida por:
Pedro Rosado Castellano
Fernando Romero Subirón

Valencia, Noviembre 2015

Resumen

La planificación de procesos, como nexo de unión entre el diseño y la fabricación, es un elemento clave para asegurar que las características de los productos fabricados satisfacen las necesidades del cliente. En las tareas de planificación del proceso de fabricación confluyen múltiples factores que, al conjugarse con la diversidad de estrategias y enfoques posibles, configuran un escenario particularmente complejo.

Por otra parte, la evolución de un entorno cada vez más competitivo y globalizado ha obligado a aumentar la flexibilidad y agilidad de los sistemas productivos. Esto ha sido especialmente crítico en las pequeñas y medianas empresas que para sobrevivir han tenido que organizarse creando estructuras de colaboración. Estructuras colaborativas que aprovechan las tecnologías de información y comunicación, y permiten salvar las dificultades originadas por la deslocalización. Un caso extremo se presenta en las empresas de tipo OKP (One-of-a-Kind Production) virtual, en las que el desarrollo de nuevos productos es el proceso clave y está sujeto a grandes exigencias de inmediatez y diversidad de producción, que solo pueden alcanzarse con una estrecha cooperación/colaboración entre los socios.

Tomando como punto de partida lo anterior, esta tesis realiza una aportación en el ámbito de la planificación de procesos colaborativa. Para ello se propone una ontología que da soporte y consistencia a las herramientas de co-planificación empleadas en la creación de planes de proceso, especialmente en la toma de decisiones vinculadas con la asignación óptima y dinámica de los recursos.

La ontología propuesta en primer lugar, ontología de dominio PPDRC (Product and Processes Development Resource Capabilities), es una ontología genérica capaz de soportar cualquier tipo de planificación de proceso que sea ejecutada por agentes inteligentes en un contexto colaborativo. Una generalidad que se valida en el trabajo, con su aplicación a la planificación del propio proceso de desarrollo de productos, procesos y recursos, y con la especialización de la misma, ontología MIRC (Manufacturing and Inspection Resource Capabilities), para la planificación de los procesos de mecanizado e inspección.

La ontología PPDRC presenta toda una serie de características singulares, como: el carácter social y agentivo de los recursos implicados en la planificación; la posibilidad de representar planes de proceso no lineales; el concepto de capacidad de recurso basado en sus habilidades para la realización de actividades; o la utilización de conceptos presentes en ontologías de base, que facilitan su interoperabilidad con otras ontologías. La ontología se muestra especialmente eficaz para el establecimiento y validación de planes de proceso en base a las capacidades de los recursos, al permitir mantener la información y conocimiento sobre sus capacidades. Un conocimiento que se enriquece por inferencia a partir de los datos, predicados y reglas que forman parte de dicha ontología.

Por su parte, la ontología MIRC es una propuesta que reúne todas las características de la ontología PPDRC y que presta una especial atención a las actividades de preparación realizadas sobre los recursos, pues éstas condicionan en gran medida sus capacidades para la ejecución de las actividades de tipo operación (mecanizado e inspección). Se trata de una característica que la diferencia de otras, al considerar que las actividades de preparación son claves para la correcta selección y asignación de los recursos y que deben considerarse durante la validación de estos planes de proceso.

La tesis se ha redactado en base a dos artículos, en los que se describen las mencionadas ontologías (PPDRC y MIRC) y se presentan sendos casos de estudio que constatan su validez y muestran el alcance de las mismas. Para facilitar su seguimiento, contiene unos capítulos adicionales, en los que se relata el planteamiento general y las bases del trabajo, y se discuten los resultados y trabajos futuros.

Resum

La planificació de processos, com a nexe d'unió entre el disseny i la fabricació, és un element clau per a assegurar que les característiques dels productes fabricats satisfan les necessitats del client. En les tasques de planificació del procés de fabricació conflueixen múltiples factors, que al conjugar-se amb la diversitat d'estratègies i enfocaments possibles configuren un escenari particularment complex.

D'altra banda, l'evolució d'un entorn, cada vegada més competitiu i globalitzat ha obligat a augmentar la flexibilitat i agilitat dels sistemes productius. Açò ha sigut especialment crític en les xicotetes i mitjanes empreses, que per a poder sobreviure han hagut d'organitzar-se, creant estructures de col·laboració. Estructures de col·laboració que aprofiten les tecnologies d'informació i comunicació, i permeten salvar les dificultats originades per la deslocalització. Un cas extrem es presenta en les empreses de tipus OKP (One-of-a-Kind Production) virtual, en les que el desenrotllament de nous productes és el procés clau i està subjecte a grans exigències d'immediatesa i diversitat, que només poden aconseguir-se amb una estreta cooperació/col·laboració entre els socis.

Prenent com a punt de partida l'anterior, esta tesi realitza una aportació en l'àmbit de la planificació de processos col·laborativa. Per a això es proposa una ontologia que dóna suport i consistència a les ferramentes de co-planificació empleades en la creació de plans de procés, especialment en la presa de decisions vinculades amb l'assignació òptima i dinàmica dels recursos.

L'ontologia proposada en primer lloc, ontologia de domini PPDRC (Product and Processes Development Resource Capabilities), és una ontologia genèrica capaç de suportar qualsevol tipus de planificació de procés que siga executada per agents intel·ligents en un context col·laboratiu. Una generalitat que es valida en el treball, amb la seua aplicació a la planificació del propi procés de desenrotllament de productes, processos i recursos i amb l'especialització de la mateixa, ontologia MIRC (Manufacturing and Inspection Resource Capabilities), per a la planificació dels processos de mecanitzat i inspecció.

L'ontologia PPDRC presenta tota una sèrie de característiques singulars, com: el caràcter social i agentiu dels recursos implicats en la planificació; la possibilitat de representar plans de procés no lineals; el concepte de capacitat de recurs basat en les seues habilitats per a la realització de activitats; o la utilització de conceptes presents en ontologies de base, que facilita la seua interoperabilitat amb altres ontologies. L'ontologia es mostra especialment eficaç per a l'establiment i validació de plans de procés basant-se en les capacitats dels recursos, al permetre mantenir la informació i coneixement sobre les seues capacitats. Un coneixement que s'enriqueix per inferència a partir de les dades, predicats i regles que formen part de la dita ontologia.

Per la seua banda, l'ontologia MIRC és una proposta que reuneix totes les característiques de l'ontologia PPDRC i que presta una especial atenció a les activitats de preparació realitzades sobre els recursos, perquè estes condicionen en gran manera les seues capacitats per a l'execució de les activitats d'operació (mecanitzat i inspecció). Es tracta d'una característica que la diferència d'altres, al considerar que les activitats de preparació són claus per a la correcta selecció i assignació dels recursos i que han de considerar-se durant la validació d'estos plans de procés.

La tesi s'ha redactat basant-se en dos articles, en les que es descriuen les mencionades ontologies (PPDRC i MIRC) i es presenten sengles casos d'estudi que constaten la seua validesa i mostren l'abast de les mateixes. Per a facilitar el seu seguiment, conté uns capítols addicionals, en els que es relata el plantejament general i les bases del treball, i es discuteixen els resultats i treballs futurs.

Summary

Process planning, as a link between design and manufacturing, is a key function to ensuring that the characteristics of manufactured products meet customer needs. Multiple factors converge in the tasks of manufacturing process planning. When these factors are combined with the diversity of possible strategies and approaches, they form a particularly complex scenario.

Moreover, the development of an environment increasingly competitive and globalized forces an increase of flexibility and agility of production systems. This has been especially critical in small and medium enterprises. In order to survive, these enterprises have had to improve their organization creating collaborative structures. Collaborative structures that take advantage of information and communication technologies, allowing overcome the difficulties caused by their location all over the world. An extreme case occurs in the virtual OKP (One-of-a-Kind Production) companies, in which the development of new products is the key process and it is subject to heavy demands of immediacy and diversity in terms of production. These demands can only be achieved through a close cooperation/collaboration between company partners.

Taking as a starting point the above, this thesis makes a contribution in the field of collaborative process planning. For this, an ontological approach is proposed. This ontology gives support and consistency to the co-planning tools used in creating process plans, especially in decision-making related to the optimal and dynamics resource allocation.

The first proposed ontology, the domain PPDRC (Product and Processes Development Resource Capabilities) ontology, is a generic ontology able to support any kind of process planning to be executed by intelligent agents in a collaborative context. Its generality is validated in this work, by means of its application to the process planning of the development process of products, processes and resources and its specialization, the MIRC (Manufacturing and Inspection Resource Capabilities) ontology, for machining and inspection process planning.

The PPDRC ontology presents a number of particular characteristics, such as: social and agentive character of the resources involved in the process planning; the possibility for representing nonlinear process plans; the concept of resource capability based on its skills to perform a specific activity; or the fact that it has been built using concepts of foundational ontologies, facilitating the interoperability with other ontologies. The ontology is particularly effective for the establishment and validation of process plans based on the capabilities of the resources involved, allowing to maintain the information and knowledge about the capabilities of these resources. A knowledge that is enriched by inference from the data, predicates and rules that are part of the ontology.

On the other hand, the MIRC ontology is a proposal that has all the characteristics of PPDRC ontology and pays special attention to preparation activities of the resources, because they largely determine their capability to implement operation activities (machining and inspection). This is a characteristic that differentiates it from others, considering that preparation activities are critical to the correct selection and allocation of resources that should be considered during the validation of these process plans.

The thesis has been written on the basis of two articles that describe the aforementioned ontologies (PPDRC and MIRC) and they present two separate case studies which

demonstrate their validity and scope. To facilitate the reading, this document contains some additional chapters. These additional chapters relate the general approach and the basis of the work, and discuss the results and future works.

(Dedicatoria)

A todos los que han contribuido a la realización de esta tesis. Especialmente, como no, a mis directores, compañeros y amigos Pedro y Fernando, por su inestimable ayuda, apoyo y paciencia.

A todos los miembros de la ontología familiar Solano-Calabuig. Especialmente, a las entidades de primer nivel Yolanda, David y Raúl, que también han tenido que aguantar lo suyo.

Contenido

Listado de Figuras

Listado de Figures (capítulos 5 y 6)

Listado de Tablas

Listado de Tables (capítulos 5 y 6)

Capítulo 1. PLANTEAMIENTO GENERAL Y OBJETIVOS

1. Introducción

El entorno competitivo en el que desarrollan su actividad las empresas de fabricación mecánica está caracterizado por un nivel de exigencia creciente, tanto en las especificaciones de calidad y coste del producto como en las condiciones de los procesos de ejecución, entrega/instalación, mantenimiento, etc. Estas exigencias han conducido, sobre todo a las pequeñas y medianas empresas, a una búsqueda permanente de soluciones eficientes, como medio para garantizar su supervivencia. Esta dinámica de cambio permanente, ha dado lugar a una fuerte evolución, orientada por nuevos paradigmas de fabricación que proponen el incremento de la flexibilidad y agilidad en base a conseguir altos niveles de integración en los propios sistemas de fabricación. Unos cambios que imponen mayores exigencias sobre la operación y el funcionamiento del sistema, que solo son posibles gracias a las mejoras organizativas y a la incorporación masiva de las Tecnologías de la Información y la Comunicación (TIC) (Camarinha-Matos, Afsarmanesh y Rabelo 2003; Prince y Kay 2003; Buyukozkan, Dereli y Baykasoglu 2004; Camarinha-Matos y Afsarmanesh 2013).

Este complejo escenario, en el que se desarrolla la actividad de las empresas de fabricación, afecta de manera especial a uno de los procesos de empresa más determinantes para asegurar la supervivencia y alcanzar el éxito: el Desarrollo Integrado de Producto, Proceso y Recursos. Se trata de un enfoque que se contrapone al enfoque clásico, caracterizado por flujos de trabajo secuenciales, y que ha sido objeto de numerosos trabajos en las tres últimas décadas (Armstrong 2001; Xie et al. 2003; McGrath 2004; Mostafei, Bouras y Batouche 2005; Browning, Fricke y Negele 2006; Cheung et al. 2006; Buyukozkan y Arsenyan 2012; Paashuis 2013). Todos ellos resaltan la importancia del mismo para aquellas empresas que basan su estrategia en la innovación y el desarrollo de nuevos productos. Estos trabajos también inciden en las dificultades presentes en su implantación, derivadas de la gran cantidad y diversidad de factores que intervienen, así como del carácter dinámico que presentan muchos de ellos.

Con la implementación de estos enfoques, las empresas consiguieron en las décadas de los 80 y 90 una mejora sustancial, que se vio favorecida por la aparición de herramientas informáticas de apoyo a cada una de las actividades implicadas (CAX). Unas mejoras de tipo técnico a las que se sumaron otras de carácter organizativo, que posibilitaban una gestión diferente del flujo de trabajo y que dieron lugar a las prácticas de la Ingeniería Concurrente (Carter y Baker 1991). Estas prácticas conllevan la creación de equipos de trabajo y la ejecución de tareas en paralelo, y precisan de una gran sincronización y de un continuo intercambio de información y conocimiento tanto entre los miembros de los equipos de trabajo como entre las aplicaciones expertas que realizan las distintas tareas.

A finales del siglo pasado, debido a la gran presión de la globalización de los mercados y la incesante deslocalización de los negocios, dichas prácticas, aplicadas en principio dentro de la empresa (intra-empresa), se extendieron a las redes colaborativas de empresas, dando paso a la Ingeniería Concurrente Inter-empresa o Ingeniería Colaborativa (Buyukozkan, Dereli y Baykasoglu 2004; Wang et al. 2005). Una extensión que impone nuevos y mayores requerimientos a la cooperación y al intercambio de información, debido a la diversidad y dispersión de los agentes implicados. En este sentido, la Ingeniería Colaborativa exige un alto grado de integración que, dada la heterogeneidad de los sistemas y aplicaciones utilizados, solo puede garantizarse asegurando una interoperabilidad a nivel semántico basada en el uso de conceptos compartidos entre socios y, consecuentemente, entre aplicaciones.

Todo lo anterior es particularmente relevante en el caso de aquellas empresas virtuales que adoptan 'One-of-a-Kind Production', denominadas empresas OKP virtuales (Xie y Tu 2011; Bernard 2014). Este tipo de empresas colaborativas presentan un escenario en el que se reúne una combinación de factores adversos que obligan a optimizar la eficiencia del sistema, justificando la orientación de este trabajo hacia dichas empresas. En las empresas con un sistema de producción de tipo OKP es esencial la agilidad, para lo que se requieren altos niveles de integración y colaboración entre los diferentes subsistemas vinculados al diseño y desarrollo de productos, la planificación de procesos y la planificación, programación y control de la producción. Para ello, se precisa una infraestructura tecnológica que soporte la interacción entre socios y la interoperabilidad entre aplicaciones (software) y posibilite los altos niveles de cooperación y coordinación necesarios. Unas necesidades de interoperabilidad, técnica y semántica, que se pueden conseguir con la incorporación de planteamientos ontológicos, soportados por la Web, que mejoren aquellos basados en repositorios de modelos de metadatos.

Los planteamientos ontológicos permiten establecer y compartir una semántica de forma rigurosa, precisa y sin ambigüedades (Calì et al. 2005). Este tipo de planteamientos están basados en una ontología explícita, que define formal y explícitamente los conceptos y sus relaciones en un determinado ámbito. A su vez, la utilización de este tipo de planteamientos implica una separación entre la funcionalidad de la aplicación y los conceptos y relaciones manejados en la misma, que no se presenta en aplicaciones basadas en ontologías no explicitadas. De este modo, diferentes aplicaciones puede intercambiar información independientemente de las ontologías no explícitas que les dan soporte. Además, la riqueza semántica de las ontologías permite mejorar la flexibilidad y versatilidad de las aplicaciones que forman el sistema. La posibilidad de manejar estas ontologías explicitadas en red, permite acceder a fuentes de información deslocalizadas, gestionar el conocimiento

de forma distribuida y que los diferentes agentes participen de forma síncrona o asíncrona en el desarrollo de los productos y los procesos y recursos asociados (Young et al. 2007).

De todas las actividades presentes en el Desarrollo Integrado de Producto, Proceso y Recursos, el foco de interés de esta tesis se dirige a la planificación de procesos, que es la actividad central en la integración diseño-fabricación (Sormaz, Arumugam y Rajaraman 2004). La planificación de procesos es el elemento clave de la integración de diferentes subsistemas de empresa, especialmente en el caso de la integración de los subsistemas de aplicaciones del dominio del Diseño, conocidas como aplicaciones Product Data Management/Product Life Cicle Managment (Li y Qiu 2006; CIMdata y Grieves 2006), con los subsistemas de Programación y Control de planta, que forman parte de los Manufacturing Execution Systems (Zhong et al. 2013) y con los de Gestión Logística de la Empresa (Kesharwani 2014), entre los que están los de Enterprise Resource Planning (ERP), Supply Chain Management (SCM), etc.

Por lo tanto, son los requisitos establecidos por una Planificación de Procesos, que se realiza en el contexto de un Desarrollo Integrado de Productos, Procesos y Recursos y que mantiene altos niveles de integración con los sistemas MES, ERP, etc., los que han guiado el desarrollo de esta tesis. Un marco de referencia del trabajo se concretó en el proyecto Co-CAPP (Romero, Estruch y Rosado 2009), en el que se establecieron las bases para el desarrollo de un sistema de co-planificación, elemento clave para facilitar un proceso de Desarrollo Integrado y Colaborativo de Productos, Procesos y Recursos ($CIDP^2R$) en empresas OKP virtuales. Se trata de un sistema de co-planificación que requiere, para no comprometer el grado de flexibilidad y agilidad de este tipo de empresas, de la definición de planes de proceso no lineales (con alternativas) que puedan ser ejecutados en estos entornos de fabricación tan dinámicos (Solano et al. 2009). Una circunstancia que obliga a generar modelos que capturen la capacidad de los recursos, que resulta esencial en la planificación de procesos, para posibilitar el funcionamiento de herramientas colaborativas en el ámbito de la planificación de procesos (co-planificación) como ya es habitual en otros ámbitos (co-diseño).

2. Hipótesis de partida

El trabajo desarrollado en esta tesis se basa en las siguientes hipótesis de partida:

- Se puede desarrollar una ontología de proceso-recursos que de soporte al desarrollo concurrente y colaborativo de planes de mecanizado e inspección y su integración con la Programación y Control de la Producción, permitiendo que los agentes implicados cooperen y/o colaboren, de manera síncrona o asíncrona, en la definición incremental de un plan de proceso.
- Es posible establecer una ontología para el dominio global de la planificación del conjunto de actividades involucradas en el desarrollo colaborativo de productos, procesos y recursos, construida sobre una ontología de base que facilite su integración con ontologías específicas de otros dominios.
- La ontología PSL (Process Specification Language) puede servir, a través de extensiones del núcleo, como medio para representar planes de proceso con alternativas que se puedan secuenciar en el momento de su lanzamiento teniendo

en cuenta las capacidades tecnológicas y productivas de los recursos. A su vez, la expresividad de PSL puede dar soporte a una metodología de creación incremental de planes de proceso generados mediante la imposición progresiva de restricciones que van limitando las posibles alternativas existentes.

- La combinación de lenguajes muy extendidos en el ámbito de las aplicaciones distribuidas, como OWL (Ontology Web Language) y SWRL (Semantic Web Rule Language) permite crear ontologías con una expresividad cercana a la obtenida mediante la utilización de lógicas de primer orden y asegurar la finalización de las inferencias lógicas en un tiempo finito.

3. Objetivo de la Tesis

Partiendo de las consideraciones anteriores, el objetivo general de la tesis se centra en la definición de unos modelos ontológicos que den soporte a todas las actividades de planificación de procesos que se realizan en los procesos de Desarrollo de Nuevos Productos y que posean la expresividad requerida por los entornos de desarrollo colaborativo. Un foco de interés que luego se dirige hacia la planificación de procesos de fabricación (mecanizado e inspección) y, en particular hacia las actividades asociadas con la asignación y validación de los recursos, donde se toman decisiones en las que es primordial conocer sus habilidades, tanto en lo referente a los tipos de actividades que pueden realizar como a la cuantificación de dichas habilidades.

Este objetivo general se concreta en los siguientes objetivos particulares:

- Definición de un esquema o marco general para el desarrollo integrado de producto-proceso-recursos y establecimiento de las entidades compartidas entre los diferentes actores que intervienen en el CIDP^{2}R.

- Determinación del tipo de ontología y del enfoque ontológico que resulte más adecuado para representar los conceptos del dominio. Esta determinación incluye tanto aspectos vinculados con el grado de abstracción de la ontología (previsiblemente ontología de base o *foundational ontology)*, su expresividad (previsiblemente *heavy-weight*) y el tipo de lógica (por ejemplo, lógica de primer orden); como los relativos a las alternativas ontológicas (ontología de universales o particulares, descriptiva o prescriptiva, multiplicativa o reduccionista, consideración de los denominados paradigmas 3D y 4D, presencia de entidades concretas y/o abstractas, *endurants* y/o *perdurants*, etc.)

- Definición de una ontología integrada proceso-recurso genérica, que sea compatible con el modelo colaborativo de planificación de procesos en el ámbito de redes de ingeniería y fabricación colaborativa. Esta ontología debe contemplar la integración y la concurrencia de la actividad de planificación de procesos con las actividades de planificación y control de la producción, y ejecución y análisis de la fabricación. Además de lo anterior, la ontología debe dar satisfacción a dos objetivos adicionales: dar soporte a planes de proceso no lineales y permitir la actualización del conocimiento sobre las capacidades de fabricación, necesario en un proceso de auto-aprendizaje orientado a la mejora de los planes generados.

- La especialización de esta ontología para la planificación integrada de los procesos de mecanizado e inspección. Esto incluye tanto la definición de los tipos de actividades y los recursos implicados, como las características de los productos obtenidos que son elementos básicos para la validación del plan.
- Permitir el análisis de la consistencia del trabajo realizado por los diferentes planificadores y facilitar la colaboración en entornos de co-planificación, especialmente para la planificación de procesos de mecanizado e inspección, definiendo una metodología que compatibilice la ontología especializada con este tipo de planificación.
- Implementación y validación de la ontología genérica y de su especialización, a los efectos de valorar su adecuación para los entornos a los que se destinan. Adecuación que se juzgará considerando distintos niveles (agregado, supervisor y operacional), enfoques (variante, generativo y mixto) y estrategias (hacia adelante, hacia atrás y mixta) de planificación.

4. Metodología de la Tesis

En la realización del trabajo se ha seguido una metodología que consta de las siguientes etapas:

- **Establecimiento y especificación del Modelo Colaborativo de Planificación de Procesos**

  En esta primera etapa se define el modelo de planificación de procesos que debe constituir el marco para el proceso de desarrollo de producto, proceso y recursos, y que debe cubrir las necesidades particulares de la planificación de procesos de mecanizado e inspección. Todo ello, en un ámbito distribuido y colaborativo, como el que corresponde a la actuación de la empresa OKP virtual, y otorgando a la planificación de procesos un papel clave como nexo de unión entre las funciones de diseño y las funciones de planificación, programación y control de la fabricación.

- **Estudio del estado del arte**

  A continuación, se aglutinan conceptos y conocimientos de dos campos: el de la planificación de procesos y el de la gestión de la colaboración, contemplando las propuestas más recientes, tanto con enfoques tradicionales como con enfoques ontológicos. Concretamente, se realiza un estudio detallado de la aplicación de PSL como ontología de alto nivel para especificar, compartir y transmitir cualquier proceso, entendido como un conjunto de actividades estructuradas y secuenciadas con alternativas.

- **Establecimiento de los requerimientos de la ontología a desarrollar**

  De acuerdo al tipo de enfoque ontológico adoptado para soportar el desarrollo integrado, concurrente y colaborativo de planes de proceso, se procede a determinar los requerimientos que debe satisfacer dicha propuesta ontológica para dar cumplimiento a su cometido. Unos requerimientos que como mínimo correspondan a los que satisface la ontología PSL.

- **Búsqueda de alternativas de solución, mediante la combinación y ampliación de las propuestas de ontologías de proceso y de recursos existentes**

  En esta etapa se seleccionan tanto las propuestas ontológicas que cubren los dominios de fabricación, planificación de procesos, productos y recursos, como otras de alto nivel que aseguren la consistencia de la solución adoptada, y la interoperabilidad entre ésta y otras ontologías de su mismo nivel, o con ontologías de nivel superior. En ambos casos, se debe seleccionar el lenguaje de representación de la ontología que satisface las necesidades del modelo de planificación de procesos, tomando como punto de partida la semántica de la ontología PSL.

- **Definición de la propuesta ontológica**

  Se trata, en esta etapa, de formalizar una ontología que de soporte al desarrollo integrado y colaborativo de productos, procesos y recursos, y de forma particular al desarrollo y validación de planes integrados de mecanizado convencional e inspección.

- **Implementación y validación de la propuesta ontológica**

  Haciendo uso del lenguaje de representación seleccionado, se realiza la implementación de la propuesta desarrollada. A continuación, se procede a validar la propuesta mediante ejemplos o casos de estudio que permiten verificar el cumplimiento de los requerimientos impuestos a la misma.

- **Documentación del modelo ontológico propuesto**

  Finalmente, en la etapa de documentación, se incluye tanto la descripción de la propuesta ontológica desarrollada, como la recopilación, estructuración y presentación de la información relevante asociada a las etapas precedentes, y que en su conjunto constituye la memoria de la tesis.

5. Organización de la Tesis

Además de este primer capítulo dedicado a la introducción, objetivos, alcance y enfoque, la tesis consta de otros seis capítulos adicionales, cuyos títulos se indican a continuación, acompañados de una breve descripción de su contenido.

- **Capítulo 2. Marco para el desarrollo colaborativo de productos, procesos y recursos**

  En este capítulo se describe el marco en el que se desarrolla la tesis, incluyendo las tendencias, paradigmas y necesidades de las empresas de fabricación actuales. En él se presta especial atención al proceso de Desarrollo de Productos, Procesos y Recursos (DP^2R) en la empresa OKP virtual, así como a su actividad nuclear, la planificación de procesos de fabricación, describiendo la funcionalidad del sistema Co-CAPP. Finalmente, también se exponen algunas iniciativas normativas para el intercambio de información referente al producto, el proceso y los recursos, como STEP o MANDATE.

- **Capítulo 3. Ontologías**

  En el capítulo 3 se reúnen todos aquellos aspectos vinculados con las ontologías que resultan de interés para el desarrollo de la tesis, como son: el concepto y los tipos de ontología, las alternativas ontológicas disponibles para conceptualizar la realidad, y la descripción de herramientas y lenguajes para el desarrollo, implementación y utilización de las ontologías.

- **Capítulo 4. Bases de la propuesta**

  Las bases de la propuesta de la tesis, que se describen en este capítulo, incluyen: la conceptualización de las actividades relacionadas con el proceso DP^2R, y la planificación de las mismas en un entorno social, compartido y colaborativo; la descripción y gestión operacional de estas actividades y de los recursos vinculados a ellas; y la consideración del comportamiento social y agentivo de los recursos, así como el soporte para la caracterización de dicho comportamiento. Estas bases se han recogido bajo la perspectiva de los enfoques: social, funcional, de los procesos y de los recursos.

- **Capítulo 5. Ontología para el desarrollo colaborativo de productos y procesos centrada en capacidades de los recursos (Ontología PPDRC)**

  Tras una breve introducción, en este capítulo se transcribe la versión original del artículo Knowledge Representation for Product and Processes Development Planning in Collaborative Environments, publicado en 2014, donde se describen los antecedentes que justifican la propuesta, junto con los conceptos, entidades y predicados fundamentales de la ontología PPDRC (Product and Processes Development Resource Capabilities). Una ontología que da soporte a cualquier actividad de planificación de procesos inmersa en el proceso de desarrollo colaborativo de nuevos productos. Para certificar esta generalidad, en el artículo se incluye un caso de estudio centrado en la planificación del propio proceso de desarrollo.

- **Capítulo 6. Ontología para la planificación integrada de procesos de mecanizado e inspección centrada en capacidades de los recursos (Ontología MIRC)**

  De igual modo que en el capítulo anterior, en éste, tras una breve introducción, se transcribe la versión original del artículo An Ontology for Integrated Machining and Inspection Process Planning focusing on Resource Capabilities, publicado online en 2015, donde se describen los fundamentos de la ontología MIRC (Manufacturing and Inspection Resource Capability). MIRC es una especialización de la ontología PPDRC para el ámbito de la planificación de los procesos de mecanizado e inspección. Por tanto, hereda de ésta la capacidad para soportar las actividades de la planificación de procesos colaborativa desarrolladas en el contexto de una OKP virtual y para representar el carácter social y agentivo de los recursos requeridos (equipos y útiles de mecanizado e inspección). En el artículo, después de una breve introducción de la ontología PPDRC, se describen el marco conceptual de la ontología MIRC y los dos grupos de actividades que forman parte de un plan de proceso de mecanizado e inspección (actividades de los tipos

Operation y Preparation), para finalizar con unos apuntes sobre su implementación y un caso de estudio con el que se valida la propuesta ontológica.

- **Capítulo 7. Discusión general de los resultados, conclusiones y trabajos futuros**

  En el capítulo 7 se recogen las conclusiones más significativas que se desprenden del trabajo realizado y se listan ciertas aportaciones que caracterizan a las ontologías PPDRC y MIRC, desarrolladas en el marco de la tesis. Para cerrar el capítulo, en el último apartado se plantean algunos trabajos futuros, orientados a otras necesidades del ámbito del desarrollo colaborativo del producto/proceso, que pueden dar continuidad al trabajo realizado en esta tesis.

6. Referencias

Armstrong, S. C. 2001. "Engineering and Product Development Management. The Holistic Approach." Cambridge University Press.

Bernard, A. 2014. "One-of-a-kind production." *Production Planning & Control* 25(16): 1400–1401.

Browning, T. R., E. Fricke y H. Negele. 2006. "Key concepts in modeling product development processes." *Wiley InterScience, Systems Engineering* 9 (2): 104–128.

Buyukozkan, G., T. Dereli y A. Baykasoglu. 2004. "A survey on the methods and tools of concurrent new product development and agile manufacturing." *Journal of Intelligent Manufacturing* 15(6): 731–751.

Buyukozkan, G. y J. Arsenyan. 2012. "Collaborative product development: a literature overview." *Production Planning & Control* 23(1): 47–66.

Calì, A., D. Calvanese, B. Cuenca Grau, G. De Giacomo, D. Lembo, M. Lenzerini, C. Lutz, D. Milano, R. Möller, A. Poggi y U. Sattler. 2005. *State of the art survey Deliverable D01. TONES EU-IST STREP FP6-7603.*

Camarinha-Matos, L. M., H. Afsarmanesh y R. J. Rabelo. 2003. "Infrastructure developments for agile virtual enterprises." *International Journal of Computer Integrated Manufacturing* 16(4–5): 235–254.

Camarinha-Matos, L. M. y H. Afsarmanesh. 2013. "Infrastructures for Virtual Enterprises." Springer Science & Business Media.

Carter, D. E. y B. S. Baker. 1991. "Concurrent Engineering, Product Development for the 90's." Addison Wesley Publishing Company, London.

Cheung, W. M., D. G. Bramall, P. G. Maropoulos, J. X. Gao y H. Aziz. 2006. "Organizational knowledge encapsulation and re-use in collaborative product development." *International Journal of Computer Integrated Manufacturing* 19 (7): 736–750.

CIMdata y M. Grieves. 2006. "Digital Manufacturing in PLM Environments." CIMdata White Paper.

Kesharwani, S. 2014. “Enterprise Resource Planning and Supply Chain Management-Functions, Business Processes and Software for Manufacturing Companies.” *Global Journal of Enterprise Information System* 6(4): 33–35.

Li, W. D. y Z. M. Qiu. 2006. “State-of-the-art Technologies and Methodologies for Collaborative Product Development Systems.” *International Journal of Production Research* 44 (13): 2525–2559.

McGrath, M. E. 2004. “The next generation product development. How to increase productivity, cut costs, and reduce cycle times.” McGraw-Hill.

Mostafei, S., A. Bouras y M. Batouche. 2005. “Effective collaboration in product development via a common sharable ontology.” *International Journal of Computational Intelligence* 2 (4): 206–212.

Paashuis, V. 2013. “The organisation of integrated product development.” Springer Science & Business Media.

Prince, J. y J. M. Kay. 2003. “Combining lean and agile characteristics: Creation of virtual groups by enhanced production flow analysis.” *International Journal of Production Economics* 85 (3): 305–318.

Romero, F., A. Estruch y P. Rosado. 2009. “A Framework for the Development of an IT Platform for Collaborative and Integrated Development of Product, Process and Resources.” Paper presented at the 13th International research/expert conference. Trends in the development of machinery and associated technology, Hammamet, Tunisia, October 16–21.

Rosado, P. y F. Romero. 2009. “A Model for Collaborative Process Planning in a Engineering and Production Network.” Paper presented at the 13th International research/expert conference. Trends in the development of machinery and associated technology, Hammamet, Tunisia, October 16–21.

Solano, L., P. Rosado, F. Romero y F. González. 2009. “Posibilidades de la ontología PSL para representar planes de proceso no lineales.” Paper presented at the 3rd Manufacturing Engineering Society International Conference, Alcoy, Spain, June 17–19.

Sormaz, D. N., J. Arumugam y S. Rajaraman. 2004. “Integrative Process Plan Model and Representation for Intelligent Distributed Manufacturing Planning.” *International Journal of Production Research* 42 (17): 3397–3417.

Wang, C. B., Y. M. Chen, Y. J. Chen y C. Ho. 2005. “Methodology and system framework for knowledge management in allied concurrent engineering.” *International Journal of Computer Integrated Manufacturing* 18(1): 53–72.

Xie, S. Q., Y. L. Tu, R. Y. K. Fung y Z. D. Zhou. 2003. “Rapid one-of-a-kind product development via the Internet: a literature review of the state-of-the-art and a proposed platform.” *International Journal of Production Research* 41 (18): 4257–4298.

Xie, S. S. y Tu, Y. 2011. “Rapid One-of-a-kind Product Development: Strategies, Algorithms and Tools.” Springer Science & Business Media.

Young, R. I. M., A. G. Gunendran, A. F. Cutting-Decelle y M. Gruninger. 2007. "Manufacturing knowledge sharing in PLM: a progression towards the use of heavy weight ontologies." *International Journal of Production Research* 45(7), 1505–1519.

Zhong, R. Y., Q. Y. Dai, T. Qu, G. J. Hu y G. Q. Huang. 2013. "RFID-enabled real-time manufacturing execution system for mass-customization production." *Robotics and Computer-Integrated Manufacturing* 29(2): 283–292.

Capítulo 2. MARCO PARA EL DESARROLLO COLABORATIVO DE PRODUCTOS, PROCESOS Y RECURSOS

1. Introducción

El incremento en el número de variantes, la reducción de costes y de tiempos de entrega, la personalización y la mejora en la calidad de los productos, son algunas de las exigencias impuestas durante las últimas décadas a las empresas de fabricación que compiten en un mercado globalizado dominado por los compradores. En este contexto y como respuesta a estas exigencias, a lo largo del tiempo, los sistemas productivos han adoptado y adaptado diversos paradigmas de fabricación (fabricación ajustada, fabricación ágil, fabricación basada en el tiempo, fabricación masiva personalizada, etc.), para los que es fundamental conseguir mayores niveles de automatización, flexibilidad e integración. Unas metas que hoy son alcanzables gracias a los grandes avances que se han producido en la capacidad de procesamiento de los ordenadores y a las tecnologías Web.

Para Botti y Giret (2002), esta integración significa que cada unidad organizativa tiene acceso a la información relevante para su tarea y conoce el efecto de sus acciones sobre otras funciones, permitiendo la propuesta y selección de alternativas que optimicen los objetivos de la empresa. Un grado de integración que puede establecerse a distintos niveles, por ejemplo, según Chen y Vernadat (2004), el CEN TC310 WG1 ha reconocido tres niveles de integración: (a) integración física o interconexión de dispositivos como máquinas CNC a través de redes de ordenadores; (b) integración de aplicaciones, referida a la interoperabilidad de aplicaciones de software y sistemas de bases de datos en entornos de computación heterogéneos; y (c) integración de las funciones que gestionan, controlan y monitorizan los procesos de la empresa.

Unas posibilidades de integración que han cambiado a la empresa, que hoy se muestra como un conjunto de procesos concurrentes y dinámicos, realizados por personas o por agentes inteligentes dotados de un grado de autonomía creciente. Una forma de trabajo que

muestra el cambio experimentado en las tradicionales estructuras jerarquizadas, que han evolucionado hacia los sistemas técnicos, productivos y organizativos actuales, basados en estructuras con un menor número de niveles organizativos y una gestión por procesos, que ha aportado nuevos enfoques como la ingeniería concurrente[1] al proceso de Desarrollo de Nuevos Productos.

Pero con el tiempo, la presión del mercado y los avances en las tecnologías Web han hecho que este proceso de integración, que en primer lugar se produjo dentro de la empresa, se tuviera que extender a las relaciones entre empresas (integración inter-empresa) en sus diferentes formas. Así, las empresas de fabricación han tendido a adoptar arquitecturas abiertas, que en forma de amplias redes integran sus actividades propias con las de suministradores y clientes, en un claro ejemplo de paradigma de fabricación conducido por la entrega, que les permite competir de forma efectiva en los mercados actuales y que en el campo de la ingeniería del producto ha dado paso a la ingeniería concurrente interempresa o ingeniería colaborativa[2].

Si la integración y la colaboración son fundamentales para el paradigma de la fabricación basada en el tiempo y, por extensión, para el de la fabricación ajustada, no lo es menos para el de la fabricación ágil, que según el enfoque de Goldman y Nagel (1993) a través de Domingo, González y Calvo (2004), asimila en gran medida el rango de las tecnologías de la fabricación flexible, de la gestión de la calidad total (TQM), de la fabricación justo a tiempo (JIT), y de la fabricación ajustada. Para Paolucci (2005) la fabricación ágil extiende el concepto de flexibilidad (reconfigurabilidad y adaptatividad de las estructuras de los sistemas productivos) más allá de las propias fronteras de los sistemas de fabricación, abarcando el entorno y el mercado, donde la gestión de las relaciones con los clientes y la cooperación entre compañías son incluso más importantes. La agilidad requiere una eficiente utilización de los recursos internos y externos para satisfacer las necesidades cambiantes de los clientes de forma flexible, enfatizando la velocidad de respuesta a las nuevas oportunidades del mercado. En este sentido, la fabricación ágil es la evolución lógica de otros paradigmas anteriores, ya caracterizados por centrarse en la capacidad para adaptarse a las necesidades cambiantes de los clientes.

La necesidad de esta agilidad ha originado la aparición de algunos tipos de redes colaborativas, formadas por varias sociedades que se muestran como una empresa única, como es el caso de las empresas virtuales. Esta circunstancia, que forma parte de la estrategia de supervivencia de las SME's (Small and Medium Enterprises), se ha visto favorecida por el desarrollo de las tecnologías de la información, que aportan la infraestructura para permitir una ingeniería y una fabricación distribuida. Pero, para ello también se precisa que las aplicaciones o tecnologías distribuidas conecten adecuadamente con los modelos de fabricación distribuidos. En esta línea, Leitao y Restivo (2000) proponen un marco para aplicaciones de fabricación distribuida basado en arquitecturas

[1] La ingeniería concurrente pretende la integración sistemática del diseño de productos y de los procesos relacionados, incluidos los de planificación, fabricación y soporte, para que sean considerados simultáneamente teniendo en cuenta la información generada durante todo el ciclo de vida del producto. Un objetivo que implica la puesta en práctica de nuevas metodologías, técnicas y herramientas basadas en la ejecución de actividades en paralelo, el trabajo en equipo y el refuerzo de las tareas iniciales del proceso de diseño para facilitar la participación de todos los agentes involucrados en el ciclo de vida del producto.

[2] La ingeniería colaborativa contempla varias formas de colaboración, que van desde la simple visualización, que se centra en el desarrollo de tareas como visualizar, marcar, hacer comentarios, etc., hasta otras manifestaciones basadas en el desarrollo de tareas simultáneas, que se especializan en una función determinada como el co-diseño, la co-planificación, etc.

soportadas por agentes que implementan conceptos de sistemas de fabricación holónicos y biónicos.

A las exigencias del trabajo colaborativo y a las dificultades de interoperabilidad que éste conlleva en el contexto de la empresa extendida también se referían Subrahmanian et al. (2006), que promulgaban la conveniencia de comenzar a hablar de redes de diseño y fabricación. Unas redes de diseño y fabricación, en las que la cadena de valor gira en torno al producto, y que necesitan propuestas que, adaptándose a las particularidades de las empresas virtuales, sirvan para mejorar la colaboración en el desarrollo de productos. En esta línea, Sun (2005) propuso una plataforma para el diseño cooperativo que incluía la integración con los recursos de fabricación. Por su parte, Gialelis et al. (2005) presentan una arquitectura basada en sistemas distribuidos que utiliza tecnologías como workflows, ontologías y servicios Web para hacer frente a los procesos colaborativos actuales, que cubren el rango completo de la infraestructura industrial y toda la cadena de suministro. Unas mejoras que son especialmente necesarias en el caso de las empresas virtuales de tipo OKP. Más recientemente, Milosevic, Todic y Lukic (2012) proponen un marco para un sistema distribuido y colaborativo que facilita la colaboración entre equipos de diseño y expertos en planificación de procesos. Huang et al. (2014) ponen el foco en la planificación del proceso de diseño, presentando una propuesta formal para la planificación del diseño detallado en un entorno colaborativo. Una propuesta que se centra en la determinación de la secuencia de asignación de valores a los parámetros de diseño del producto.

Como ya se ha indicado, la integración requerida exige algo más que la simple transmisión de la información, especialmente en estas redes de diseño y fabricación con relaciones muy complejas, siendo precisa una colaboración basada en modelos para el intercambio de conocimiento. En este sentido, frente a los enfoques basados en compartir datos o bases de datos, los enfoques ontológicos, que están siendo utilizados en diferentes ámbitos relacionados con la fabricación, proporcionan un medio adecuado para aumentar la interoperabilidad semántica, tal y como puede verse en la Figura 2.1, que muestra los diferentes niveles de interoperabilidad propuestos en el Marco de Interoperabilidad Europeo (EIF).

En dicho marco, que ha sido desarrollado conjuntamente por la Comisión Europea (EC) y los estados miembros de la Unión Europea (EU) para hacer frente a las necesidades de intercambio de información de empresas y gobiernos, se establecen tres niveles. Estos niveles, denominados de interoperabilidad técnica, semántica y organizativa, corresponden respectivamente con la transmisión e intercambio de datos, la interpretación consistente de la información, y la funcionalidad de la organización. La interoperabilidad en cada uno de estos niveles está sustentada en el o los niveles inferiores de interoperabilidad (Vernadat 2010).

Los aspectos técnicos de la interoperabilidad en la empresa están relacionados con la comunicación y el intercambio de datos entre los sistemas de información. Estos aspectos se centran en los protocolos de comunicación, el intercambio de datos y la transmisión de mensajes entre aplicaciones, y en la actualidad, todos ellos están evolucionando aprovechando el rápido progreso tecnológico en las TIC. En este sentido, se han desarrollado diversos protocolos (HTTP, SMTP, TCP/IP, SOAP,…), lenguajes y técnicas con un soporte sintáctico para la comunicación y representación de los datos. Pero todo ello no es suficiente para soportar la gestión del conocimiento.

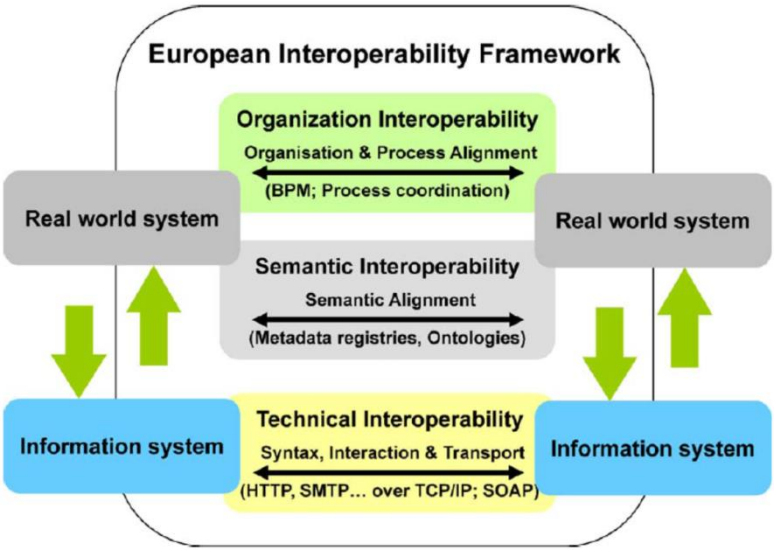


Figura 2.1. Marco de Interoperabilidad Europeo (Vernadat 2010).

Los aspectos semánticos de la interoperabilidad se refieren a la integración de datos/información y a la consistencia requerida para soportar la cooperación, basada necesariamente en el uso compartido de la información y el conocimiento. Pero también puede ser definida como la habilidad para compartir, agregar o sincronizar información y datos entre sistemas de información heterogéneos. En otras palabras, se trata de asegurar que dos sistemas que están comunicándose interpreten la información común o compartida de forma consistente. Para resolver el problema de incompatibilidad entre los conceptos utilizados por diferentes componentes que se quieren interconectar, durante las últimas décadas se han ido estableciendo diferentes estándares. Por ejemplo, STEP (Standard for the Transfer and Exchange of Product Model Data) –ISO 1994–, MANDATE (MANufacturing management DATa Exchange) –ISO 2004a–, etc. que proponen modelos de información y mecanismos para el intercambio de datos con diferentes niveles de implementación, como el uso de ficheros neutros o el acceso compartido a bases de datos mediante interfaces de acceso directo. Unos modelos que en los últimos años han ido ganando en formalidad y que han dado paso al desarrollo de modelos ontológicos, que permiten describir el conocimiento acerca de un determinado dominio de manera formal y declarativa, y que en la actualidad se presenta como el enfoque dominante para conseguir altos niveles de interoperabilidad semántica.

Los aspectos organizativos de la interoperabilidad se centran en la definición de los objetivos de empresa, en la alineación y coordinación de los procesos de empresa, y en la creación de capacidades colaborativas, para organizaciones que desean intercambiar información aun teniendo estructuras y procesos internos diferentes. Por otra parte, el objetivo de la interoperabilidad organizativa es responder a los requerimientos de la comunidad de usuarios poniendo a disposición de éstos unos servicios fácilmente identificables, accesibles y centrados en el usuario. Puede decirse, que al igual que en los otros niveles de interoperabilidad se hace referencia al uso compartido de los datos, la información y el conocimiento, en este nivel se comparten los procesos.

Una vez introducidos ciertos conceptos sobre la integración de la empresa y puesto de manifiesto que ésta es fundamental para conseguir altos niveles de colaboración entre los diferentes agentes que forman parte del sistema empresa, a continuación, en los siguientes apartados, se concretarán algunos de los conceptos desarrollados anteriormente, prestando especial atención a una determinada tipología de empresa, la empresa OKP virtual, y a un proceso, el proceso de Desarrollo de Productos, Procesos y Recursos. Un objetivo que obliga a revisar ciertas propuestas realizadas para estos dominios, desde dos perspectivas: la funcional y la de la información.

Para ello, los siguientes apartados se centrarán en el estudio de propuestas que exponen modelos/arquitecturas que buscan la excelencia en el Desarrollo de Productos, Procesos y Recursos y que se ajustan a las necesidades de las empresas de tipo OKP virtual y, posteriormente, en detallar la funcionalidad del sistema Co-CAPP, que fija el marco en el que se ha desarrollado esta tesis (Romero, Estruch y Rosado 2009). Se trata de una propuesta que fue desarrollada en las fases iniciales del proyecto COAPP que también incluyó una gran parte del trabajo que ahora se muestra. Finalmente, después de realizar esta aproximación funcional, en el último apartado el foco se centrará en la perspectiva informacional, revisándose algunas propuestas y/o iniciativas normativas que han aparecido en los últimos años en el ámbito de la interoperabilidad semántica en los campos del producto, el proceso y los recursos, y que son referencia obligada para este trabajo. Una revisión que se centrará en los modelos de datos/información de estándares que sustentan el intercambio en base a diferentes mecanismos (fichero neutro STEP, SDAI, STEP XML, etc.), mientras que los modelos ontológicos se tratarán el capitulo siguiente.

2. El proceso DP^{2}R en la empresa OKP virtual

En esta sección se presenta la empresa OKP virtual, o empresa virtual con sistema OKP (One-of-a-Kind Production), y el proceso DP^{2}R (Desarrollo de Productos, Procesos y Recursos), que es el proceso clave para empresas de este tipo, y en general para muchas empresas de fabricación. Para ello, en primer lugar se describe la empresa virtual con sistemas OKP, seguido de las exigencias impuestas por el proceso DP^{2}R en el marco de la fabricación distribuida, a las que ésta deberá dar respuesta.

2.1. La empresa OKP virtual

La empresa virtual es el término que se utiliza para definir empresas formadas por un conjunto o red de empresas, que pone de manifiesto su comportamiento como una única empresa, aunque ésta no exista en la realidad. Este comportamiento depende de su nivel de integración: coordinación, cooperación y colaboración. En este tipo de empresas, que están cambiando continuamente, se requiere una consideración especial de su propio ciclo de vida, mucho más corto que el de las empresas tradicionales.

OKP (One-of-a-Kind Production) puede entenderse como una filosofía de producción mediante la cual un producto solicitado por un cliente individual, es decir 'One', y que corresponde a un tipo específico, es decir 'of-a-Kind', debe ser desarrollado y producido 'a la primera', bajo restricciones severas de calidad, coste y plazo de entrega. Las empresas con sistemas OKP se caracterizan por: su trabajo en mercados de productos personalizados, que demandan una continua interacción con el cliente; diseño y fabricación 'a la primera',

lo que implica no disponer de una segunda oportunidad en el proceso de desarrollo y fabricación, ya que se trata productos únicos; óptima utilización de tecnologías y recursos; planificación y control de la producción adaptativos; desarrollo del producto y del proceso con un enfoque evolutivo y concurrente; control distribuido y departamentos de producción autónomos; y estructura de empresa virtual y fabricación deslocalizada (Tu 1997). En resumen, como afirman Romero, Estruch y Rosado (2009), se trata de empresas que deben presentar una gran capacidad para reconfigurarse y adaptar rápidamente su sistema al cambio.

Para responder rápidamente a las presiones del mercado, muchas empresas OKP han adoptado un concepto de fabricación virtual, que en su forma más elemental tiene su antecedente en la subcontratación y la transferencia de tecnología entre una empresa y sus socios. Este concepto de fabricación virtual ayuda a la empresa a satisfacer rápida, y a menudo, económicamente las necesidades de sus clientes gracias a la integración de sus propias capacidades con las de sus socios. Estas empresas virtuales por lo general requieren de apoyo técnico y teórico para gestionar y optimizar su proceso de desarrollo de productos a través de un ciclo de desarrollo conjunto. Este proceso de desarrollo incluye la interpretación de los requisitos del cliente para decidir la forma de abordar sus necesidades a través del diseño del producto, prototipado conceptual y funcional para comprobar la efectividad de los diseños, planificación de los procesos de fabricación (que incluye la planificación de las operaciones a subcontratar) para determinar cómo debe fabricarse el producto, diseño y fabricación de utillajes, fabricación de la preserie, producción, y entrega del producto al cliente o lanzamiento del producto al mercado. Para estas empresas OKP virtuales, que son un caso extremo del paradigma de fabricación basada en el tiempo o fabricación rápida (Xie et al. 2003), resulta particularmente importante satisfacer las necesidades de los clientes 'a la primera' y minimizar los costes en pruebas intermedias y en construcción de prototipos (Tu, Chu y Yang 2000). En este contexto, la competitividad de las empresas OKP virtuales, con una estrategia de mercado basada en 'Engineering To Order' (ETO) está fuertemente condicionada por las características del proceso de desarrollo integrado de producto, proceso y recursos y la eficiencia en su ejecución.

Para conseguir esta reducción en el tiempo y en el coste de Desarrollo del Producto (PD), resulta esencial la combinación e integración de las tecnologías más innovadoras con el desarrollo de nuevas tecnologías para facilitar una mejora progresiva en las habilidades de diseño y fabricación. Tu, Chu y Yang (2000) proponen un sistema de soporte del proceso PD, para un entorno geográficamente distribuido, utilizando internet para facilitar las comunicaciones y el flujo de datos entre diferentes niveles a través del ciclo de vida del producto, integrando nuevas tecnologías de: CAD, diseño colaborativo, comunicaciones en Internet, información, inteligencia artificial, bases de conocimiento, optimización del coste y del tiempo de desarrollo, y ayuda a la toma de decisiones. Especificamente, para empresas de tipo OKP se identifican propuestas concretas como Rapid OKP Product Development (RPD) (Xie y Tu 2006) o Internet-based Rapid OKP Product Development (IRPD) (Xie et al. 2003).

2.2. El proceso de Desarrollo de Productos, Procesos y Recursos. Requisitos

A continuación se presentan los requisitos del proceso DP^{2}R en el marco de la empresa OKP virtual, que deben servir de referencia para el establecimiento de la propuesta de la tesis.

Los entornos e infraestructuras necesarias para realizar el trabajo cooperativo de alto nivel, como es el caso de muchas actividades del proceso DP^{2}R, deben permitir la interacción dinámica y frecuente entre socios, la interoperabilidad de los sistemas de software de los socios, la gestión de la información de los socios y la coordinación de la cooperación. Estos requisitos y la naturaleza dinámica de las EV's han conducido a la necesidad de adoptar nuevas propuestas y direcciones en el diseño de las plataformas de aplicaciones en la Web (Romero, Estruch y Rosado 2009).

En la fabricación OKP es esencial llevar a cabo el desarrollo y fabricación rápida de productos, y esto requiere el establecimiento de altos niveles de integración y colaboración entre los diferentes subsistemas vinculados al diseño y desarrollo de productos, planificación de procesos y planificación, programación y control de la producción. En concreto, es fundamental poder evaluar la fabricabilidad de los diseños y establecer planes de proceso que se adapten a las configuraciones de recursos disponibles y a las asignaciones de recursos factibles, dependiendo del estado de los recursos y de las previsiones de carga de trabajo (flexibilidad). Para ello, se debe diseñar e implementar un sistema de planificación de procesos colaborativa que facilite el aprovechamiento del conocimiento de los expertos en planificación de las distintas empresas, en relación a las posibilidades de sus procesos y sus recursos de fabricación (Romero, Estruch y Rosado 2009). En este sentido, Wang, Jin y Feng (2006) proponen una colaboración efectiva, especialmente en el ámbito de la planificación de procesos, mediante técnicas de gestión de proyectos colaborativa y/o workflows centrados en documentos. Una colaboración que va más allá de la cooperación basada en la división, distribución y coordinación de tareas. En la propuesta de Ying-Ying y Di (2013) se analiza el flujo de trabajo asociado a la planificación de procesos colaborativa y se propone un sistema para controlar y gestionar eficazmente este workflow.

El modelo para el proceso DP^{2}R debe permitir que los miembros de los equipos compartan objetivos y trabajos en las mismas piezas/productos/recursos, ya sea de forma síncrona o asíncrona (Detienne 2006). En otras palabras, el modelo establecido debe soportar el co-diseño, la co-planificación, etc., centrados en la actividad de planificación de procesos e integrados con otras actividades del proceso DP^{2}R (como ingeniería de diseño para fabricación, y configuración y asignación de recursos). Esta colaboración (horizontal) debe promover el vínculo interpersonal en el grupo de trabajo mediante mecanismos que faciliten la interactividad, la negociación y la discusión, y una representación de los planes de procesos de planta que permita alternativas (Plan de Proceso No Lineal o PPNL) y que se construya incrementalmente.

Además, se debe disponer de la riqueza semántica adecuada para expresar las diferentes alternativas de operación y mejorar por tanto la flexibilidad y versatilidad del sistema. Este requerimiento justifica la necesidad de un enfoque ontológico integrado que cubra los ámbitos del producto, los procesos y los recursos. En otras palabras, disponer de una ontología general de producto-procesos-recursos.

Por último, para conseguir la gran flexibilidad y agilidad intrínsecas a la naturaleza dinámica de las OKP virtuales, y para dar soporte de forma eficiente a los procesos cooperativos inter-dominios e inter-empresas entre sus socios, debe existir una alta interoperabilidad entre todos los componentes de los sistemas TIC de los distintos socios. Esta interoperabilidad debe ser efectiva tanto desde la perspectiva de los datos como desde la perspectiva del usuario y de las comunicaciones. Por lo tanto, es necesario alinear los sistemas de información con las infraestructuras TIC para que los servicios den acceso a las funcionalidades a través de interfaces bien definidas. Según Chen et al. (2006), las plataformas desarrolladas en base a 'Service Oriented Architecture' (SOA) e implementadas mediante servicios Web, parecen ser las soluciones más adecuadas por la construcción de sistemas TIC capaces de dar soporte a los procesos de e-colaboración en el marco del proceso DP^2R. En este sentido, también puede considerarse que SOA es un requisito del proceso DP^2R.

Los requisitos anteriores pueden resumirse en: la necesidad de facilitar la interacción dinámica, la coordinación de la cooperación y la gestión de la información entre los socios a través de la interoperabilidad de sus sistemas de software; integración entre los diferentes subsistemas de los socios, incluyendo los componentes de sus sistemas TIC; colaboración horizontal efectiva, con un enfoque ontológico integrado que va más allá de la cooperación basada en la división, distribución y coordinación de tareas; y una planificación de procesos colaborativa, que proporcione planes flexibles construidos incrementalmente, y que facilite la evaluación de la fabricabilidad.

3. La propuesta Co-CAPP

La propuesta Co-CAPP (Romero, Estruch y Rosado 2009), que se describe en este apartado, se articula en torno a la planificación de procesos y constituye el marco en el que se han desarrollado los trabajos de la tesis. Esta propuesta se adapta particularmente bien a las características y exigencias de la planificación de procesos en el ámbito de la OKP virtual, y responde a los requisitos expuestos en el apartado anterior. Pero antes de describirla, se procederá a analizar el significado, objetivos y tareas de la planificación de procesos, prestando una atención preferente a las particularidades que esta actividad presenta cuando se desarrolla en entornos colaborativos y distribuidos.

3.1. Objetivos y tareas de la planificación de procesos

La generación del plan de procesos es el objetivo directo y primordial de la planificación de procesos. En el ámbito de fabricación, la planificación de procesos puede definirse como la determinación sistemática de los procesos y recursos con los cuales un producto debe ser fabricado de forma que se alcancen uno o más objetivos y se satisfagan un conjunto de restricciones, partiendo de un diseño dado en forma de planos de ingeniería, especificaciones, y piezas o listas de materiales. El plan de procesos, que permite transformar los materiales de partida para obtener el producto de acuerdo al diseño y sus especificaciones, se concreta en la selección y secuenciación de los procesos y operaciones, las instrucciones detalladas para los mismos, y sus parámetros, así como las máquinas, herramientas y utillajes necesarios. En resumen, la planificación de procesos es la determinación sistemática de los métodos detallados con los que se ejecutan las actividades

de producción (Alting y Zhang 1989; González 2001; Wu, Fuh y Nee 2002; Kulvatunyou et al. 2004; Shen, Wang y Hao 2006).

3.1.1. Tareas y características generales de la planificación de procesos

Las tareas generales de la planificación de procesos son aquellas que podrían estar presentes en cualquier actividad de planificación de procesos, independientemente de las características organizativas de la empresa o entidad productiva. A estas tareas, que abarcan desde la selección de los procesos y las máquinas hasta la generación del programa de control numérico, se le exigen una serie de requisitos como: facilitar y gestionar eficientemente los datos de entrada de la planificación de procesos (Elmaraghy et al. 1993), por ejemplo, mediante el reconocimiento de elementos característicos (Wu, Fuh y Nee 2002); realizar abstracciones sobre los elementos que intervienen en la representación del proceso (Bonfatti, Monari y Paganelli 1998); mantener la integridad del diseño a lo largo del ciclo de vida del producto (González y Rosado 2003), de modo que pueda asegurarse el cumplimiento de las especificaciones; adaptarse y evolucionar con el entorno, contemplando los cambios en los medios productivos y la incorporación de nuevos procesos (González 2001) o la integración con aplicaciones como CAD, CAM y PPC, que faciliten la inclusión de nuevas bases de datos y de conocimiento (Elmaraghy et al. 1993; González 2001); contemplar la presencia humana, para participar en la toma de algunas decisiones, suministrar el conocimiento heurístico cuando sea necesario y suplementar las habilidades del sistema (Elmaraghy et al. 1993; González 2001); y generar resultados en un formato flexible y de fácil interpretación, por ejemplo, mediante informes y representaciones gráficas proporcionando además la adquisición efectiva de conocimiento y empleando mecanismos y medios para comprobar que el conocimiento adquirido es correcto, completo y consistente (Elmaraghy et al. 1993; Bonfatti, Monari y Paganelli 1998; González 2001). Aunque las referencias anteriores proceden del ámbito del mecanizado, las características mencionadas son extrapolables a la planificación del proceso en otros ámbitos, como por ejemplo, la planificación del proceso de desarrollo del producto.

3.1.2. Tareas y características específicas de la planificación de procesos en entornos colaborativos y distribuidos

En el caso de un entorno distribuido, el desempeño de las tareas generales de la planificación de procesos debe satisfacer las siguientes exigencias: facilitar la transferencia del conocimiento entre los actores que intervienen en un proceso (Dufresne y Martin 2003); representar el proceso a un nivel de detalle adecuado (Bonfatti, Monari y Paganelli 1998); y responder en tiempo real a los cambios del entorno, incluso mediante mecanismos de autorregulación carentes de intervención externa (Leitao y Restivo 2000).

Aunque existe una notable interdependencia entre las tareas que comporta la planificación de procesos en entornos colaborativos y distribuidos, éstas pueden agruparse en tres subgrupos: tareas relacionadas con la gestión de la información; tareas propias de la administración colaborativa y tareas relacionadas con la utilización o explotación del sistema.

Las tareas relacionadas con la gestión de la información deben garantizar la integridad de la información, que se sustancia en asegurar la accesibilidad, uso compartido y transmisión eficiente de la información. En este sentido, y desde el enfoque mediante agentes para el

desarrollo de aplicaciones de fabricación distribuidas basadas en sistemas holónicos y biónicos, se establece que la comunicación requiere una estandarización para que los distintos agentes puedan interactuar (Leitao y Restivo 2000). En la propuesta de Wang, Zhang y Fuh (2014) para desarrollar un sistema de planificación de procesos y programación de la producción en entornos distribuidos, la comunicación y coordinación efectiva entre los agentes implicados se garantiza a través de una arquitectura basada en un modelo de aplicación multi-nivel.

En relación a las tareas propias de la administración colaborativa, en primer lugar conviene establecer las diferencias entre los conceptos de proceso colaborativo y administración colaborativa. Los procesos colaborativos corresponden a las tareas a realizar en el ámbito colaborativo, mientras que la administración colaborativa o la administración de la colaboración se refiere al inicio, delegación, integración, coordinación, supervisión y evaluación de estos procesos (Deek, Tommarello y McHugh 2003). Las tareas relacionadas con la administración colaborativa incluyen: especificar cómo los procesos serán iniciados e identificados durante su ejecución (Dufresne y Martin 2003); coordinar las actividades que forman parte del proceso, distinguir roles y asignarlos a diferentes tareas (Dufresne y Martin 2003); gestionar mecanismos para la delegación y distribución de las tareas entre los distintos socios de fabricación; integrar la información y las comunicaciones (Chen y Liang 2000; Deek, Tommarello y McHugh 2003); seleccionar, negociar y pujar para alcanzar la mejor alternativa en cada situación; supervisar y evaluar los sistemas propuestos; capturar el análisis razonado de las decisiones tomadas en cada nivel enlazando con discusiones en curso (Deek, Tommarello y McHugh 2003); y registrar la descripción de las actividades de un proceso de modo que éste pueda ser realizado en 'sentido inverso' (Dufresne y Martin 2003).

Por último, están las tareas relacionadas con la utilización o explotación del sistema informático que da soporte o que ejecuta las actividades propias de la planificación de procesos en un entorno distribuido y colaborativo. Éstas se centran en: garantizar la gestión segura de los datos (Chen y Liang 2000) y la seguridad del propio sistema (Shen, Wang y Hao 2006); conectar el modelo de proceso de empresa con su fase de ejecución (Bonfatti, Monari y Paganelli 1998); y especificar las características de los procesos de empresa que serían de interés para usuarios externos, como la calidad del servicio y el precio (Dufresne y Martin 2003).

Además, en el contexto de la fabricación distribuida existen ciertas dificultades para la planificación de procesos como son las asociadas a las actividades o situaciones siguientes: creación del marco y la arquitectura en el ámbito de la fabricación distribuida (Leitao y Restivo 2000); compatibilización de los planes de proceso de las distintas empresas consideradas, junto a la especificidad del conocimiento para aplicaciones de fabricación (Zdeblick 1988); transmisión de información entre múltiples aplicaciones, empresas y personas (Pouchard et al. 2004); trasiego de los datos relacionados con la gestión de la fabricación, como datos relativos a intercambios externos, gestión de recursos y flujos de fabricación (Pouchard et al. 2004); necesidad de conjugar una descripción detallada de los procesos que facilite su control, con la reducción de la complejidad de los mismos, para evitar una carga excesiva a las empresas (Bonfatti, Monari y Paganelli 1998); y ausencia de un paradigma de modelado de procesos unificado que proporcione una representación de alto nivel de las operaciones de fabricación para el conjunto de la empresa (Bonfatti, Monari y Paganelli 1998).

3.2. Descripción de Co-CAPP

La propuesta del sistema Co-CAPP está concebida como medio para soportar el proceso de Desarrollo Integrado y Colaborativo de Producto, Procesos y Recursos (CIDP^{2}R) en empresas OKP virtuales con los requisitos indicados anteriormente. Este proceso CIDP^{2}R es la actividad central dentro del Desarrollo de Producto, y se sitúa entre la función de Diseño y la de Planificación y Control de la Producción, con una gran interacción con ambas. El CIDP^{2}R, que gira en torno a la Planificación de Procesos, tiene que alcanzar altas cotas de integración y colaboración, especialmente en empresas OKP. Como se observa en la Figura 2.2, CIDP^{2}R además de la planificación del proceso, incluye todo el desarrollo de ingeniería correspondiente a fabricación y también la configuración y desarrollo de los recursos para la fabricación. Dada la importancia de una ejecución colaborativa/cooperativa, en la propuesta se observa la necesidad de aplicar las tareas propias de la administración colaborativa a la gestión del proceso CIDP^{2}R. De este modo, se asegura un alto grado de integración y colaboración entre los participantes a través de actividades de planificación, seguimiento y control.

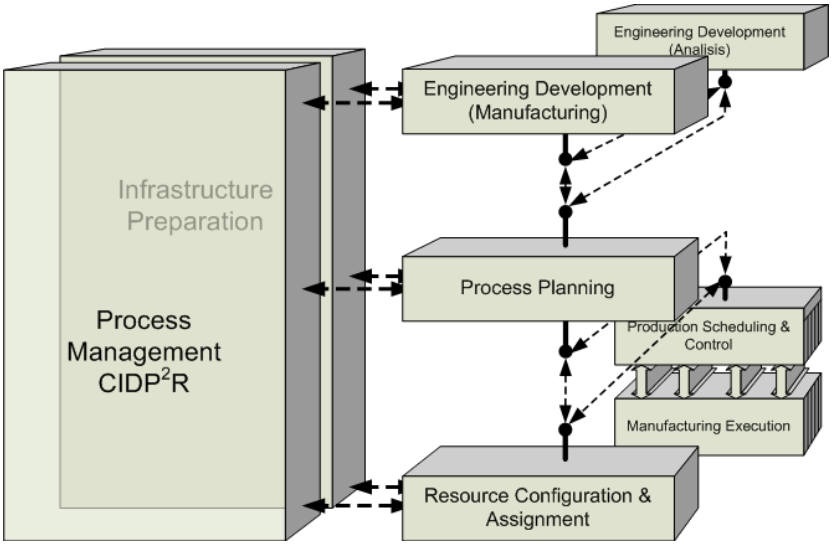


Figura 2.2. Funciones del proceso CIDP^{2}R y proceso de Gestión del CIDP^{2}R (Romero, Estruch y Rosado 2009).

El objetivo de las funciones que se muestran en la Figura 2.2 es generar un conjunto de planes de proceso alternativos como resultado de una acción incremental y cooperativa en la cual los expertos de empresas diferentes pueden incorporar conocimiento de sus procesos y recursos de fabricación. El principal resultado de la aplicación de Co-CAPP es un PPNL que contiene todas las posibles secuencias de operaciones, donde una operación es una ejecución de un proceso destinado a obtener una característica o grupo de características, empleando un recurso determinado o un grupo de recursos. Estos planes de proceso alternativos se describen usando una red cuyos nodos incluyen y relacionan todos los elementos considerados en la planificación, como operaciones de fabricación e inspección, recursos y productos, así como las restricciones que limitan las secuencias de ejecución

posibles. Se trata de una red general, para cualquier ámbito y válida para cualquier nivel de la planificación, donde los nodos o grupos de nodos tienen relaciones de asociación con los nodos de las estructuras que representan la configuración de los recursos asignados para su posible ejecución (Configuración y Asignación de Recursos) y con los nodos de los árboles de características que definen el producto (pieza) desde diferentes perspectivas y niveles de detalle. Estas relaciones corresponden a los diferentes procesos de fabricación implicados y las diferentes fases y etapas del ciclo de diseño en el cual son utilizados (por ejemplo Desarrollo de Ingeniería-Diseño Detallado). Los atributos y las reglas, que pueden ser asignadas a cualquier relación y nodo de la red global, permiten evaluar las distintas secuencias y asegurar la coherencia y completitud de los planes de proceso (Romero, Estruch y Rosado 2009).

El sistema Co-CAPP está formado por tres módulos (Ingeniería, Planificación de Procesos, y Configuración y Asignación de Recursos) que ocupan un rol central en la integración jerárquica entre Diseño Funcional y Desarrollo de Productos, y Planificación y Control de la Producción. Una vez que se ha realizado la actividad de desarrollo a nivel pieza, que puede ser soportada por un sistema de co-diseño basado en elementos característicos, el sistema de co-planificacion Co-CAPP será capaz de soportar la gestión de un proceso de desarrollo integrado de producto, procesos y recursos como CIDP^{2}R. Esto conlleva los pasos siguientes:

- Asegurar la fabricabilidad de la pieza, estableciendo la correspondencia entre los elementos característicos de forma y los elementos característicos funcionales de precisión, y la asociación de éstos con los procesos y recursos de mecanizado/inspección con las capacidades requeridas que están disponibles en el marco temporal de la fabricación de la pieza (Desarrollo de Ingeniería / módulo de Fabricación).

- Especificar la red global de planes de proceso (producto-procesos-recursos), resultante de un procedimiento dinámico para desarrollar gradualmente los detalles del plan en fases y etapas diferentes. Aquí, las acciones que incorporan nodos (características, procesos, recursos) o grupos de nodos imponen relaciones de secuencia; efectúan la valoración económica, la evaluación de la calidad y la determinación del tiempo para las secuencias de ejecución de las diferentes alternativas; y descartan aquellas que no conducen a una solución global flexible o que no están dentro de los límites de la valoración (módulo de Planificación de Procesos).

- Asegurar que la asignación de recursos individuales o de configuraciones de recursos se lleva a cabo con atención a sus capacidades, su disponibilidad, limitaciones logísticas y las políticas de asignación de recursos establecidas en los acuerdos de la OKP virtual. La red de planes establece las secuencias alternativas de operaciones sujetas a las restricciones tecnológicas y de disponibilidad de los recursos establecidos. Esto facilita el Control y Programación de la Producción en tiempo real basado en la incorporación de restricciones de tiempo y/o oportunidad, y en su caso, en los criterios de optimalidad (módulo de Configuración y Asignación de Recursos).

El módulo de Co-CAPP para la planificación de procesos (Figura 2.3) proporciona a cualquier participante en el proceso de co-Planificación (de acuerdo con los permisos

establecidos por el gestor de proyectos) una serie completa de funcionalidades generales (para cualquier ámbito, como el de mecanizado e inspección) que permiten la construcción incremental de la estructura de nodos básica (características, procesos y recursos) de la red global o de una de sus partes (sub-redes), a través de acciones aditivas realizadas empleando las funciones de asignación.

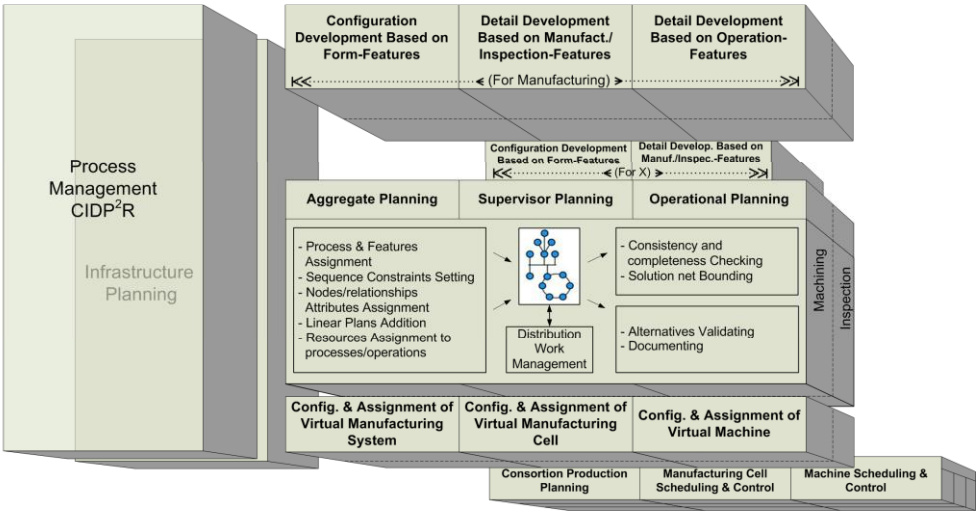


Figura 2.3. Arquitectura funcional del proceso CIDP^{2}R (Rosado y Romero 2009).

La posibilidad de construir sub-redes, independientemente de su posterior integración en la red global, es un mecanismo que facilita la realización del trabajo distribuido coordinado. Además, las restricciones de secuencia deben establecerse sobre las relaciones existentes entre los nodos individuales o de grupo (grupos de nodos del mismo tipo y nivel). Los 'nodos grupo' se construyen para facilitar la expresión de restricciones, como las impuestas por ciertas especificaciones GD&T (Geometric Dimensioning and Tolerancing) y que conducen al planificador a agrupar ciertas características para mecanizarlas en el mismo *setup*. Del mismo modo, los atributos como coste, calidad y tiempo también pueden ser asignados a las relaciones y nodos de la red, los cuales permitirán posteriormente establecer comparaciones entre alternativas con el objetivo de descartar los menos adecuados en relación al criterio de optimalidad elegido. Adicionalmente, el sistema incluye la funcionalidad que permite incorporar en la red global los planes lineales completamente caracterizados y contrastados (completos o parciales). Más tarde, la red de PPNL globales obtenidos a partir de las especificaciones de procesos descritas o seleccionadas mediante el proceso de descartar soluciones, debe ser chequeada para determinar si cumple todas las reglas definidas para asegurar la consistencia tecnológica de las combinaciones característica-proceso-recurso representadas y la existencia de al menos una solución que permite la fabricación completa de la pieza. Finalmente, este módulo también debe tener funcionalidades de: gestión (divide el problema en partes –sub-redes–, distribuye y planea el trabajo requerido para estas partes, coordina su ejecución, gestiona la estructura PPNL, etc.); validación, que actuando sobre algunos de los planes elegidos, permitan comprobar el cumplimiento de ciertos requisitos (coste, tiempos, tolerancias, etc.); y generación de la documentación necesaria (Rosado y Romero 2009).

3.3. Planificación de procesos de mecanizado e inspección a los niveles agregado, supervisor y operacional en el marco de la OKP virtual y de la co-planificación

En función del método para generar el plan y la similitud de éste respecto a planes anteriores, se consideran tres aproximaciones o enfoques a la planificación de procesos: variante, generativa y semigenerativa (Kulvatunyou et al. 2004). Entre las posibles estrategias destacan: la planificación hacia adelante ('forward'), hacia atrás ('backward'), de arriba hacia abajo (del nivel agregado a los desagregados) o de abajo hacia arriba (de los niveles desagregados al nivel agregado), que corresponden respectivamente al desarrollo del plan según: la lógica de su ejecución temporal, en sentido inverso a esta lógica temporal, incrementando el grado de detalle, o reduciendo el grado de detalle.

A continuación se presenta el objetivo y la interacción de la planificación de procesos a los niveles agregado, supervisor y operacional, con las fases equivalentes de la Ingeniería de Desarrollo (para Fabricación) y la Configuración y Asignación de Recursos. Como se muestra en la parte central de la Figura 2.3, el grupo de funcionalidades suministrado por el módulo Co-CAPP centrado en la planificación es común a los tres niveles y ha sido diseñado para ser utilizado en procedimientos de trabajo que implican la cooperación entre personas.

3.3.1. Nivel agregado

La planificación agregada establece un plan general para la fabricación/inspección de la pieza, que representa el primer nivel en la red global de nodos y relaciones y que será detallada en los niveles siguientes. Los nodos básicos, que representan los procesos agregados (meta-procesos), están asociados con los nodos del árbol de elementos característicos a nivel de la pieza, que llevan asociados los atributos de forma general y el grado de tolerancias de la pieza, y con los nodos del árbol de recursos agregados, para formar una red global en este primer nivel. Las restricciones establecidas en las relaciones entre los nodos definen las alternativas del PPNL a nivel de meta-procesos. La actividad de Desarrollo de la Configuración es donde se establece el árbol de los elementos característicos de forma y GD&T. Este árbol describe la geometría de la pieza y su información asociada a un nivel agregado de tal forma que se asegura la factibilidad de la fabricación y la inspección a nivel de meta-proceso (p.e. forja, mecanizado, inspección dimensional, etc.), también realizada como parte de esta actividad. Del mismo modo, en la Asignación y Configuración de la actividad del Sistema de Fabricación Virtual se establece un árbol de recursos agregado, que describe las configuraciones alternativas de las células o grupos virtuales que tienen las capacidades necesarias para realizar uno o más meta-procesos. Estas células o grupos virtuales, que pueden estar formados por recursos pertenecientes a empresas diferentes, están configurados con un proceso de negociación y una toma de decisión consensuada entre las empresas que integran la empresa virtual. Esta última actividad tiene una relación cercana a las actividades de Planificación de la Producción (materiales, recursos y órdenes) a un nivel de la empresa virtual, y a su vez interactúa con las partes correspondientes de cada una de las empresas participantes. La cooperación establecida alrededor de esta actividad permite la configuración alternativa de recursos que pueden ser asignados a los meta-procesos para ser designados para adaptarse a las circunstancias de producción de las empresas en el marco temporal establecido (Rosado y Romero 2009).

3.3.2. Nivel supervisor

En la planificación supervisora los nodos de mecanizado e inspección se descomponen en otros nodos que representan los procesos de estos meta-procesos (p. e. fresado planeado, medición con MMC, etc.) y que están asociados, del mismo modo, con los nodos de los árboles que representan los recursos (máquinas con sus tipos de utillajes y herramientas) y la pieza (elementos característicos de mecanizado/inspección) para detallar este nivel de la red global. Como para el nivel anterior, las restricciones establecidas sobre las relaciones a este nivel determinan las alternativas posibles en la hoja de ruta, describiendo la secuencia de operaciones que deben ser lanzadas y que tienen la información necesaria para la supervisión en un nivel de célula (Rosado y Romero 2009).

Como explican González et al. (2009), este nivel de planificación incluye cinco actividades para la realización de los planes de proceso de mecanizado e inspección: (a) definir la pieza con elementos característicos; (b) asignar procesos alternativos; (c) asignar recursos alternativos a procesos; (d) establecer restricciones; y (e) integrar no linealmente procesos y recursos. Una vez establecida la definición inicial de la pieza con algunos de sus elementos característicos, ya pueden iniciarse las otras actividades que se desarrollarán de forma prácticamente simultánea: la asignación de procesos da paso a la asignación de recursos y al establecimiento de restricciones, y antes de que estas actividades finalicen totalmente se van realizando simultáneamente con ellas la definición del resto de elementos característicos y la integración no lineal de procesos y recursos. Se trata de una metodología que también puede aplicarse a los niveles agregado y operacional de la planificación, y que conduce a la definición de macroprocesos (nivel agregado), operaciones (nivel supervisor) y detalles operativos (nivel operacional).

A continuación se describen brevemente las actividades de esta metodología, aplicada al nivel de la planificación supervisora. Unas actividades cuya secuencia, como se deduce de lo anterior, no es determinante. En la primera actividad, se modela la pieza como un conjunto de elementos característicos de mecanizado e inspección, que son el resultado de las distintas interpretaciones que se pueden dar de una geometría y que orientan el plan hacia distintos procesos de mecanizado e inspección.

En la segunda actividad, los participantes en la colaboración asignan procesos alternativos a cada elemento característico, así como las particularidades generales (grados) de cada proceso. En este contexto, un proceso es un tipo de mecanizado o inspección con una estrategia de trabajo conocida, por ejemplo: planeado frontal o inspección dimensional de la planitud. El grado de cada proceso indica, según el caso, la cantidad de creces a eliminar (subvolumen del elemento característico de mecanizado) o la precisión de la inspección. De este modo, el grado asociado a un proceso condiciona el número de operaciones con las que se mecaniza o inspecciona un elemento. Esta asignación de procesos, que se realiza en base a los datos relativos a las capacidades tecnológicas de los recursos disponibles, conduce a la posibilidad de realizar diferentes secuencias de ejecución para los elementos característicos/procesos.

La asignación de recursos alternativos a cada uno de los posibles procesos de mecanizado e inspección permite vincular recursos a los procesos, de modo que éstos queden disponibles para la ejecución de dichos procesos, y da lugar a un nuevo espacio de soluciones que contiene las posibles secuencias de ejecución a nivel de operaciones. Los distintos tipos de recursos (máquinas, herramientas y utillajes) pueden asignarse de forma simultánea e

independiente, permitiendo, por ejemplo que las herramientas y los utillajes sean independientes de las máquinas. Esta asignación de recursos se reduce a seleccionar entre los pertenecientes a la tipología adecuada a cada proceso-grado, aquéllos que permitan procesar el elemento con la eficacia y eficiencia requerida (aquéllos que reúnan las capacidades adecuadas). Para ello, es preciso considerar características como: fuerza, par, potencia, velocidades, tolerancias, precisión, repetibilidad, dimensiones, carreras útiles o accesibilidad. Además, y también relacionado con las características anteriores, se considera la disposición espacial de los elementos característicos en la pieza para agrupar los que tienen la misma dirección de acceso y seleccionar localizaciones, amarres de pieza y herramientas o instrumentos/sondas de medición. Estas agrupaciones de elementos característicos conducen a agrupaciones de procesos-grados que se pueden ejecutar con la misma combinación de máquina, utillajes y herramientas.

Con el establecimiento de restricciones entre los elementos de la red, a los niveles de elementos característicos, procesos y operaciones, se fijan dependencias y se restringe el espacio de las posibles secuencias de ejecución definido tras las actividades de asignación de procesos y recursos alternativos. Las restricciones sobre el plan de mecanizado e inspección pueden ser de dos tipos: restricciones de secuencia y restricciones de agrupación. Las primeras, expresan la necesidad de respetar un orden determinado en la ejecución de los elementos característicos, procesos y operaciones, y pueden deberse a exigencias tecnológicas o económicas, o a relaciones dimensionales y geométricas entre los propios elementos característicos. Las segundas, restricciones de agrupación, permiten que las entidades agrupadas (elementos característicos u operaciones) sean tratadas a todos los efectos como una entidad única. Por tanto, a estos grupos se les puede asignar recursos y pueden participar en el establecimiento de restricciones de secuencia y agrupación. La definición de dichos grupos obedece: (a) a particularidades técnicas y productivas, como limitaciones de accesibilidad, especificaciones dimensionales y geométricas, reducción del número de amarres y de cambios de herramienta, que obligan o hacen recomendable mecanizar o inspeccionar varios elementos característicos con la misma máquina, sujeción y/o herramienta; o (b) a la necesidad de dividir el problema y distribuir el trabajo de planificación entre los participantes en la tarea colaborativa, estableciendo el alcance de la actuación de cada colaborador o grupo de colaboradores según lo determinado colectivamente o por el coordinador.

La última actividad contemplada en este nivel de planificación, integrar no linealmente procesos y recursos, debe asegurar que el conjunto de todas las alternativas y las restricciones establecidas sobre ellas satisface las exigencias de coherencia y completitud del plan. En otras palabras, el plan integrado y no lineal de mecanizado e inspección resultante de las actividades anteriores debe asegurar la compatibilidad entre los recursos asignados, y de éstos con el proceso, para determinar un plan válido que se traduzca en la ejecución de todos los elementos de la pieza. Adicionalmente, esta actividad también contempla la valoración de las alternativas existentes para facilitar la selección de la secuencia y asignación/agrupación óptimas.

3.3.3. Nivel operacional

En este nivel, el alcance y las consecuencias de la toma de decisiones son muy reducidas, ya que los procesos, con sus operaciones, secuencia de ejecución y recursos han sido establecidos previamente en la planificación a nivel supervisor. Llegados a este punto la

labor de los agentes que participan en la planificación a nivel operacional se centra en completar la especificación de esas operaciones de mecanizado e inspección, junto con los recursos que las realizan. Para ello, se deben fijar los valores de todos los parámetros y características, incluyendo aquéllos que permiten determinar el coste y el tiempo asociado a dichas operaciones: condiciones de ejecución relativas a la máquina, como trayectorias, puntos de posicionamiento intermedios, avances en aproximación y retirada, avances de trabajo, regímenes de giro, etc; configuración de la herramienta o sonda de medición, que incluye la identificación y posición relativa de los distintos elementos que la componen; aspectos relativos a los utillajes, como su configuración y todo lo relativo a la preparación (*setup*) de los recursos, como la forma precisa de establecer los correctores de las herramientas, la calibración de las sondas de medición, o la forma en la que se establece la referencia entre la pieza y el utillaje. Habitualmente, una parte muy significativa de esta información queda recogida en un programa que se obtiene de forma automatizada, semi-automatizada o manual a partir de las especificaciones de la pieza, de sus elementos característicos y/o del resto de información de proceso, operación y recursos resultantes de la planificación a nivel supervisor.

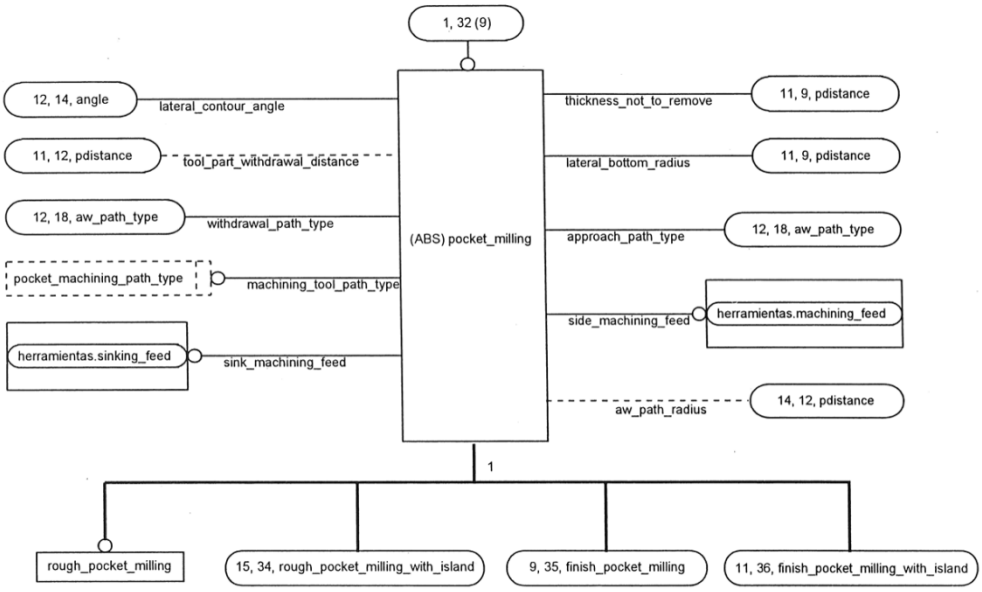


Figura 2.4. Modelo de información para el fresado de una cajera desarrollado en EXPRESS-G (Ríos 1996).

Otra posibilidad, más acorde en el marco de la OKP virtual y la co-planificación consiste en almacenar, utilizar y compartir esta información relativa a las operaciones de mecanizado e inspección de una forma estructurada. Para ello se dispone de STEP-NC, un modelo para el intercambio bidireccional de datos entre los sistemas CAD/CAM y las máquinas CNC, que se definió en la norma ISO 14649 (ISO 2004c) y está en fase de armonización con la norma STEP (ISO 1994) como el protocolo de aplicación AP 238 (ISO 2007). Los *Workingstep* son las entidades de tipo ejecutable pertenecientes al modelo STEP-NC que sirven como bloques básicos para la construcción de un programa de mecanizado o de inspección. Estos

Workingsteps incluyen acciones que dependen de una tecnología de mecanizado como *Machining_workingstep*, que representa un proceso de mecanizado sobre una zona específica de la pieza, y el uso de una herramienta y de unos parámetros tecnológicos; y otras que no dependen de una tecnología de mecanizado, como *Rapid_movement* y *Touch_probing*. A su vez, *Touch_probing* tiene tres subclases: *Workpiece_probing*, que corresponde a la obtención de coordenadas con movimiento en uno de los ejes; *Tool_probing*, que corresponde a las mediciones de longitud y anchura de la herramienta (correctores de la herramienta); y *Workpiece_complete_probing*, que corresponde a un ciclo de medición en el que se obtienen las coordenadas de seis posiciones para determinar la localización y orientación de la pieza. La Figura 2.4 representa un ejemplo de información estructurada para la definición de operaciones de mecanizado anterior al modelo STEP-NC que se recoge en la norma ISO. Igualmente, los trabajos de Barreiro et al. (2003a; 2003b) suponen un antecedente en esta estructuración de información de operaciones, pero aplicado al proceso de inspección y medida.

4. La interoperabilidad basada en Modelos de información

Como ya se apuntó anteriormente, la integración puede realizarse a los niveles de integración física, de aplicaciones y de las funciones. La integración a nivel de aplicación supone la integración tanto de aplicaciones de software como de sistemas de bases de datos, para lo que son precisos modelos de información, que son los que están basados en el significado de la información y los datos. Esta necesidad, ya presente en los primeros trabajos de integración, ha conducido a importantes esfuerzos para el desarrollo y utilización de modelos, en los que la información se estructura y relaciona de acuerdo a unos conceptos con el objetivo de servir de base para aplicaciones basadas en el conocimiento. La evolución en el tratamiento de la información y de los modelos de información incluye diversas iniciativas normativas cuyo objetivo es la integración de la información a través del establecimiento de infraestructuras para su gestión (almacenamiento, transmisión y uso compartido) y la creación de modelos de información basados en la semántica. Unas propuestas y modelos que posteriormente fueron incrementando su formalidad, con la incorporación de axiomas, reglas y otras relaciones, hasta llegar a las propuestas basadas en modelos ontológicos (ver capítulo 3).

Por lo tanto, en este apartado sólo se presentarán los principales estándares no ontológicos pertenecientes al dominio de la Gestión del Ciclo de Vida, conocido por sus siglas en inglés: PLM –Product Lifecycle Management– (Mostafei, Bouras y Batouche 2005). PLM es una estrategia de empresa que tiene por objeto crear y sostener un entorno de conocimiento centrado en el producto que se basa, entre otros aspectos, en conseguir una alta interoperabilidad entre las múltiples aplicaciones informáticas (CAD, CAM, CAPP, Groupware, etc.) que se utilizan a lo largo de las diferentes etapas del ciclo de vida de un producto y que pertenecen a las dimensiones del producto, el proceso y los servicios de empresa. Una parcela de PLM que en las últimas décadas ha sido objeto de importantes iniciativas normativas, tal y como se muestra en la Figura 2.5, algunas de ellas especialmente relevantes para el desarrollo de este trabajo.

En el nivel de servicios de empresa se encuentran EDI (Electronic Data Interchange), ebXML (e-Business XML), PLCS (Product Lifecycle Management Support) y SCOR (Supply-Chain Operations Reference-model) que también se extiende al nivel de procesos

(Figura 2.5). SCOR es un modelo de referencia de procesos para la gestión de la cadena de suministro que permite a sus usuarios dirigir, mejorar y comunicar las prácticas de gestión de la cadena de suministro en el contexto de la empresa extendida. Por su parte, EDI proporciona normas para el intercambio de datos a través de cualquier medio electrónico, y ebXML es un protocolo normalizado para comercio electrónico basado en XML, que está diseñado específicamente para los Servicios Web, tal y como es definido por la W3C (World Wide Web Consortium). A este nivel también pertenece PLCS (PLCS 2015), que establece la estructura para todo el conjunto de información requerida a partir de la entrega del producto, incluyendo su retirada. Una información que se recopila en las etapas de diseño y fabricación y que dado su carácter dinámico es necesario mantener y actualizar durante todo el ciclo de vida del producto.

El resto de iniciativas representadas en la Figura 2.5, SysML (The Systems Engineering Modeling Language), PSL (ISO 2004b) y STEP, se encuadran básicamente en los niveles de proceso y producto. SysML es una extensión de UML que se dirige al dominio de la ingeniería de sistemas. Por tanto, al igual que PLCS, se sitúa en el ámbito de los modelos de información definidos específicamente para dominios particulares que usan un lenguaje genérico para el modelado de la información (Subrahmanian et al. 2006).

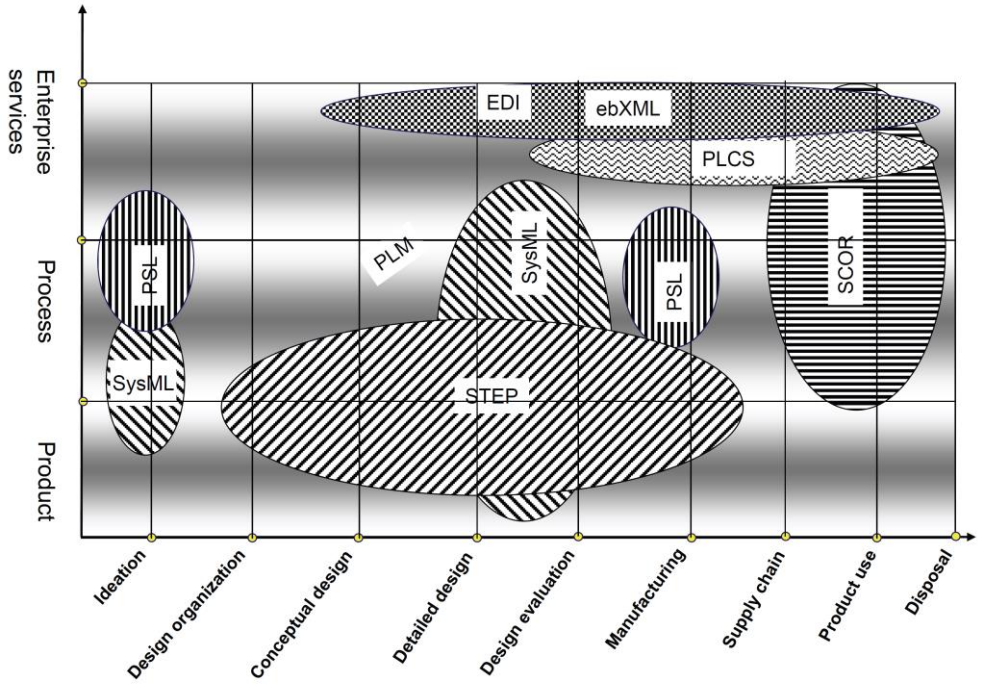


Figura 2.5. Cobertura de diversas iniciativas en relación al espectro de PLM (Subrahmanian et al. 2006).

En los subapartados siguientes se da una breve explicación de STEP, PSL y MANDATE por el interés que presentan para la tesis. Esta última iniciativa, MANDATE (ISO 2004a),

aunque no se ha desarrollado en los comités normativos implicados en el dominio PLM y por ello no se muestra en la Figura 2.5, también cubre aspectos importantes de la dimensión del proceso.

Como puede apreciarse en la Figura 2.5, estas tres iniciativas normativas cubren el ámbito del producto (STEP), los recursos (MANDATE) y los procesos (PSL). Por tanto, desde un nivel de abstracción alto, sus alcances son complementarios. No obstante, sus enfoques son distintos: STEP y MANDATE, incluyen modelos de información para productos y recursos respectivamente, mientras que PSL se centra en la descripción de una ontología de procesos. Por ello, esta propuesta también será tratada en mayor profundidad en el capítulo siguiente.

4.1. STEP

El Comité Técnico ISO TC184 (Industrial Automation Systems and Integration) es el actor principal en la elaboración de normas en el área de modelado de empresas e integración. Entre las iniciativas normativas más relevantes de este Comité Técnico de ISO se encuentran las normas STEP (ISO 1994), MANDATE (ISO 2004a) y PSL (ISO 2004b).

La iniciativa STEP es una de las más importantes en el ámbito del intercambio de información del producto. La familia de normas STEP, que incluye a STEP-NC, define una representación neutra para los datos de los productos a lo largo de todo su ciclo de vida (Pouchard et al. 2004). Los primeros documentos con el rango de norma internacional de la norma STEP se publicaron en 1994. Los trabajos de desarrollo se iniciaron en el Comité Técnico 184 (TC 184), subcomité 4 (SC4) en el año 1984. Los miembros del TC184/SC4 fueron creando documentos de trabajo que pasaron por distintos estados de desarrollo y aprobación, hasta que un conjunto de ellos alcanzó el estado de norma internacional en el año 1994 (Barreiro 2000). El desarrollo de la norma STEP sigue aún vigente.

4.2. MANDATE

La familia de normas MANDATE –ISO 15531– (ISO 2004a) se centra en el modelado de la información relativa a la gestión de fabricación, cubriendo el área de representación de datos relativos a la gestión de los procesos de producción y el intercambio y uso compartido de los datos de gestión en una empresa o entre empresas. MANDATE no estandariza el modelo de los procesos de fabricación sino que proporciona modelos de datos estandarizados para estos tres tipos de datos: datos de producción para intercambios externos; datos de gestión de los recursos de fabricación y datos de gestión del flujo de fabricación. Datos que habitualmente son complejos, con fuerte dependencia temporal y con una estrecha relación entre ellos. El objetivo de MANDATE es facilitar la integración entre numerosas aplicaciones industriales por medio de herramientas estandarizadas comunes, capaces de representar los anteriores tipos de datos que son compartidos e intercambiados durante el todo el ciclo de vida del producto y que están en el núcleo de los procesos de fabricación.

En el capítulo 4 se tratan con más detalle algunos aspectos de MANDATE que son de interés para esta tesis.

4.3. PSL

PSL se desarrolló inicialmente con el objetivo de crear un lenguaje de representación de procesos común para todas las aplicaciones de fabricación, pero posteriormente se propuso para la representación de procesos en general y actualmente ocupa un lugar destacado entre las ontologías de proceso.

La norma ISO 18629:2004 (ISO 2004b) recoge tanto la definición del lenguaje, como la especificación de la ontología establecida con dicho lenguaje. En ella, se apunta a la creación de un lenguaje neutro de alto nivel para la especificación de procesos y a la integración de múltiples aplicaciones relacionadas con los procesos a lo largo del ciclo de vida del producto. Este lenguaje es formal y está basado en una lógica de primer orden y un conjunto de teorías matemáticas. Según Pouchard et al. (2004), su alcance se limita al terreno de los procesos discretos relacionados con la fabricación, incluyendo todos los procesos del ciclo de vida de diseño y fabricación. La metodología de PSL para la construcción de la ontología se basa en la identificación de intuiciones relacionadas con los procesos de fabricación, que se formalizan mediante definiciones y axiomas escritos en una lógica de primer orden que asegura una semántica desarrollada rigurosamente.

5. Referencias

Alting, L. y H. Zhang. 1989. "Computer Aided Process Planning: the State-of-the-art Survey." *International Journal. of Production Research* 27 (4): 553–585.

Barreiro, J. 2000. "Modelo Integrado de Información en Ingeniería Simultánea." PhD diss., Universidad Politécnica de Madrid.

Barreiro, J., J. E. Labarga, A. Vizán y J. Ríos. 2003a. "Information Model for the Integration of Inspection Activity in a Concurrent Engineering Framework." *International Journal of Machine Tools and Manufacture* 43 (8): 797–809.

Barreiro, J., J. E. Labarga, A. Vizán y J. Ríos. 2003b. "Functional Model for the Development of an Inspection Integration Framework." *International Journal of Machine Tools and Manufacture* 43 (15): 1621–1632.

Bonfatti, F., P. D. Monari y P. Paganelli. 1998. "Resource-free and Resource-dependent Aspects of Process Modelling: a Rule-based Conceptual Approach." *International Journal of Computer Integrated Manufacturing* 11 (1): 60–76.

Botti, V. J. y A. Giret. 2002. "Aplicaciones de los Sistemas Multiagente (SMA) en Industria". *Agentes Inteligentes: Sistemas Multiagentes y Aplicaciones* 3: 19–82.

Chen, D. y F. Vernadat. 2004. "Standards on Enterprise Integration and Engineering—State of the Art". *International Journal of Computer Integrated Manufacturing* 7 (3): 235–253.

Chen Q., J. Shen, Y. Dong, J. Dai y W. Xu. 2006. "Building a Collaborative Manufacturing System on an Extensible SOA-based Platform". Proceedings of the 10th International Conference on Computer Supported Cooperative Work in Design. CSCWD'06. IEEE: 1–6.

Chen, Y. M. y M. W. Liang. 2000. "Design and Implementation of a Collaborative Engineering Information System for Allied Concurrent Engineering." *International Journal of Computer Integrated Manufacturing* 13 (1): 11–30.

Deek, F. P., J. D. Tommarello y J. McHugh. 2003. "A Model for Collaborative Technologies in Manufacturing." *International Journal of Computer Integrated Manufacturing* 16 (4–5): 357–371.

Detienne, F. 2006. "Collaborative Design: Managing Task Interdependences and Multiple Perspectives". *Interacting with Computers* 18: 1–20.

Domingo, R., C. González y R. Calvo. 2004. "Análisis del Diseño y Programación de Celdas en Entornos de Fabricación Ágil". *Información Tecnológica* 15 (1): 55–60.

Dufresne, T. y J. Martin. 2003 "Process Modeling for e-Business." Paper presented at INFS 770-methods for information systems engineering: Knowledge management and e-business, George Mason University.

Elmaraghy, H. A., E. Agerman, B. J. Davies, W. H. ElMaraghy, W. Eversheim, D. G. Halevi, I. Ham, H. B. Jasperse, H. J. Kals, A. Y. C. Nee, G., Sohlenius, V. Tipnis, H. K. Tönshoff, C. A. van Luttervelt, H. P. Weindahl y F. Javane. 1993. "Evolution and Future Perspectives of CAPP." *Annals of the CIRP* 42 (2): 739–751.

Gialelis, J. V., A. P. Kalogeras, C. E. Alexakos, M. J. Georgoudakis y G. Papadopoulos. 2005. "Manufacturing Collaborative Process Integration Utilizing State of the Art Technologies". Proceedings of the IEEE ISIE 2005, Dubrovnik, Croatia, June 20–23.

González, F. 2001. "Propuesta Funcional y Representación de Información y Conocimiento para Planificación de Procesos Asistida por Ordenador. Aplicación a la Determinación de Procesos, Operaciones y Máquinas para Piezas Mecanizadas." PhD diss., Universidad Politécnica de Valencia.

González, F., F. Romero, G. Bruscas y S. Gutiérrez. 2009. "Modelado de Actividades para el Desarrollo Integrado de Planes de Mecanizado e Inspección en Entornos Distribuidos y Colaborativos." Proceedings of the Manufacturing Engineering Society International Conference, Seguí, V. J. & Reig, M. J. (Ed.): 264-271, Alcoy, Spain, June 17–19.

González, F. y P. Rosado. 2003. "General and Flexible Methodology and Architecture for CAPP: GF-CAPP System." *International Journal of Production Research* 41 (12): 2643–2662.

Huang, J., Y. Chen, Z. Zhang e Y. Xie. 2014. "A Part Affordance-based Approach for Detailed Design Process Planning in Collaborative Environment." *Concurrent Engineering* 22 (4): 291–308.

ISO (International Organization for Standardization). 1994. *Industrial Automation System and Integration– Standard for the Transfer and Exchange of Product Model Data*. ISO 10303, New York: American National Standards Institute.

ISO (International Organization for Standardization). 2004a. *Industrial Automation Systems and Integration – Industrial Manufacturing Management Data. Part 1, General Overview*. ISO 15531–1. New York: American National Standards Institute.

ISO (International Organization for Standardization). 2004b. *Industrial Automation System and Integration – Process Specification Language. Part 1, Overview and Basic Principles*. ISO 18629–1. New York: American National Standards Institute.

ISO (International Organization for Standardization). 2004c. *Industrial Automation System and Integration – Physical Device Control – Data Model for Computerized Numerical Controllers. Part 10, General Process Data.* ISO 14649–10. New York: American National Standards Institute.

ISO (International Organization for Standardization). 2007. *Industrial Automation System and Integration – Product Data Representation and Exchange. Part 238, Application Protocol: Application Interpreted Model for Computerized Numerical Controllers.* ISO 10303–238. New York: American National Standards Institute.

Kulvatunyou, B., R. A. Wysk, H. Cho y A. Jones. 2004. "Integration Framework of Process Planning based on Resource Independent Operation Summary to Support Collaborative Manufacturing." *International Journal of Computer Integrated Manufacturing* 17 (5): 377–393.

Leitao, P. y F. Restivo. 2000. "A Framework for Distributed Manufacturing Applications". Proceedings of ASI'2000 international conference: 75–80, Bordeaux, France.

Li, W. D. y Z. M. Qiu. 2006. "State-of-the-art Technologies and Methodologies for Collaborative Product Development Systems". *International Journal of Production Research* 44 (13): 2525–2559.

Milosevic, M., V. Todic y D. Lukic. 2012. "Internet-Based Collaborative System for Process Planning." *Journal of Production Engineering* 15 (1): 45–48.

Mostafei, S., A. Bouras y M. Batouche. 2005. "Effective Collaboration in Product Development via a Common Sharable Ontology." *International Journal of Computational Intelligence* 2 (4): 206–212.

Paolucci. 2005. "Agent-Based Manufacturing and Control Systems". CRC, Boca Raton.

PLCS. 2015. Consultada en Julio. http://xml.coverpages.org/productLifeCycle.html

Pouchard, L., A. F. Cutting-Decelle, J. J. Michel, R. I. M. Young y B. Das. 2004. "Utilizing Standard-based Approaches for Information Sharing and Interoperability in Manufacturing Decision Support". Proceedings of the Flexible Automation and Intelligent Manufacturing Conference, 14th International Conference, Toronto, Canada, July 12–14.

Ríos, J. 1996. "Integración de las Funciones de Programación de Máquinas Herramienta de Control Numérico mediante una Aplicación orientada a Objetos basada en un Modelo de Información de Operaciones de Mecanizado." Phd diss., Universidad Politécnica de Madrid.

Romero, F., A. Estruch y P. Rosado. 2009. "A Framework for the Development of an IT Platform for Collaborative and Integrated Development of Product, Process and Resources." Paper presented at the 13th International research/expert conference. Trends in the development of machinery and associated technology, Hammamet, Tunisia, October 16–21.

Rosado, P. y F. Romero. 2009. “A Model for Collaborative Process Planning in a Engineering and Production Network.” Paper presented at the 13th International research/expert conference. Trends in the development of machinery and associated technology, Hammamet, Tunisia, October 16–21.

Shen, W., L. Wang y Q. Hao. 2006. “Agent-Based Distributed Manufacturing Process Planning and Scheduling: A State-of-the Art Survey.” *IEEE Transactions on Systems, Man and Cybernetics* 36 (4): 563–577.

Subrahmanian, E., S. Rachuri, A. Bouras, S. J. Fenves, S. Foufou y R. D. Sriram. 2006. “The Role of Standards in Product Lifecycle Management Support". NIST Internal Report (NISTIR) 7289, National Institute of Standards and Technology.

Sun, L. 2005. “Knowledge-based Cooperative Design Technology of Networked Manufacturing”. In Computer Supported Cooperative Work in Design I. 187–198. Springer Berlin Heidelberg.

Tu, Y. 1997. “Production Planning and Control in a Virtual One-of-a-Kind Production Company.” *Computers in Industry* 34: 271–283.

Tu, Y., X. Chu y W. Yang. 2000. “Computer-aided Process Planning in Virtual One-of-a-Kind Production.” *Computers in Industry* 41(1): 99–110.

Vernadat, F. 2010. “Technical, semantic and organizational issues of enterprise interoperability and networking.” *Annual Reviews in Control* 34: 139–144.

Wang, L., W. Jin y H. Y. Feng. 2006. “Embedding Machining Features in Function Blocks for Distributed ProcessPlanning.” *International Journal of Computer Integrated Manufacturing* 19: 443–452.

Wang, Y. F., Y. F. Zhang y J. Y. H. Fuh. 2014. “An Agent-based Distributed Process Planning and Scheduling System.” *Advances in Systems Science and Applications* 14 (1): 37–51.

Wu, S. H., J. Y. H. Fuh y A. Y. C. Nee. 2002. “Concurrent Process Planning and Scheduling in Distributed Virtual Manufacturing.” *IIE Transactions* 34: 77–89.

Xie, S. Q., Y. L. Tu, R. Y. K. Fung y Z. D. Zhou. 2003. “Rapid One-of-a-Kind Product Development via the Internet: a Literature Review of the State-of-the-art and a proposed Platform.” *International Journal of Production Research* 41 (18): 4257–4298.

Xie, S. Q. y Y. L. Tu. 2006. “Rapid One-of-a-Kind Product Development.” *The International Journal of Advanced Manufacturing Technology* 27 (5–6): 421–430.

Ying-Ying, S. y L. Di. 2013. “Implementation of Workflow Management System for Collaborative Process Planning.” *Information Technology Journal* 12 (14): 2957–2962.

Zdeblick, W. J. 1988. “Process Planning Evolution: The Impact of Artificial Intelligence.” Proceedings of the 19th CIRP International Seminar on Manufacturing Systems: 175–179.

Capítulo 3. ONTOLOGÍAS

1. Introducción

En este capítulo se realiza una aproximación genérica al concepto y a los tipos de ontología, así como a los diferentes enfoques ontológicos que pueden utilizarse para conceptualizar la realidad, ya que la presencia relativamente reciente de las ontologías en el ámbito de la ingeniería, y particularmente en el de la ingeniería de fabricación, hace que éstas puedan no ser suficientemente bien conocidas en dichos ámbitos. Para el objeto de la tesis, resultan de especial interés las denominadas ontologías de base, que pueden servir de sustento a otras ontologías, como las que constituyen la propuesta de esta tesis y que se describen en los capítulos 5 y 6. Por otra parte para facilitar la interpretación del trabajo desarrollado, se describen lenguajes y herramientas que soportan el desarrollo, implementación y utilización de las ontologías.

2. Definición y propósito de las ontologías

Algunos autores coinciden en la extremada dificultad de establecer una definición general para el concepto 'ontología', ya que existen muchas y diversas realidades a las que puede referirse este término, vinculadas normalmente a los diferentes usos de las mismas: la integración de grupos de datos; el uso compartido de bases de conocimiento; la habilitación de la comunicación entre agentes de software; la ayuda en la toma de decisiones; los marcos semánticos para arquitecturas de empresa; la representación de un vocabulario en lenguaje natural; la representación de la semántica para servicios y aplicaciones de software complejas; o la provisión de un marco conceptual para indexar contenidos (Gruninger et al. 2008). Unos usos a los que les corresponden otras tantas formas de entender las ontologías y que configuran un escenario complejo, en el que la iniciativa de ofrecer una única definición pasaría por reunir a muchas comunidades (informáticos, documentalistas, filósofos y expertos en el dominio) que tienen una comprensión diferente de lo que es una ontología. Una consulta al Diccionario de la Real Academia Española proporciona la siguiente definición de ontología: '*parte de la metafísica que trata del ser en general y de sus propiedades trascendentales'* (RAE 2014). En otras palabras, la Ontología pretende, de manera racional, establecer los principios que organizan y orientan el conocimiento del ser, como una parte de la realidad, a través de sus propiedades, principios y causas. Sin embargo, en la literatura científica la mayoría de las definiciones son más concretas y se basan en la utilización de alguno de los siguientes conceptos: especificación de una conceptualización; teoría lógica o formalismo; taxonomía; o conjunto de términos.

Cualquier ontología es en sí misma o necesita de una conceptualización. Esta conceptualización puede entenderse como un conjunto de objetos, conceptos y otras entidades que se asume existen en algún área de interés junto con las relaciones que mantienen entre ellas. Para referirse a todas estas entidades y relaciones se establecen y utilizan unos términos con una interpretación y un significado preciso y compartido. En este sentido Mostefai et al. (2005) establecen que una conceptualización es una vista simplificada y abstracta del mundo que se quiere representar para algún propósito. De forma análoga se pronuncia Gruber (1993) que indica que '*una ontología es una especificación explícita de una conceptualización*', una definición que ha acabado siendo una de las más citadas por los miembros de la comunidad de gestión del conocimiento.

Los formalismos facilitan a humanos y máquinas el uso compartido de cierto conocimiento común de una manera estructurada y precisa, ya que utilizan una sintaxis rigurosa para la representación de entidades y relaciones, asegurando una interpretación inequívoca de su semántica. La lógica utiliza estos formalismos para alcanzar determinadas conclusiones a través de procesos de razonamiento. Algunas de las definiciones de ontología que se basan en teorías lógicas o formalismos se limitan prácticamente a identificar una ontología con un conjunto de términos: *'una ontología es un conjunto de términos, acompañados de la especificación de su significado expresada en un lenguaje formal'* (ISO 2004). Guarino (1995), extiende el formalismo más allá de la especificación de las entidades: *'una ontología es una descripción formal de las entidades dentro de un dominio dado, con las propiedades que tienen, las relaciones en las que participan, las restricciones a las que están sujetas, y los patrones de comportamiento que exhiben'* (Uschold y Gruninger 1996).

Otras definiciones se han basado en los conceptos de taxonomía y clasificación taxonómica, que a partir del establecimiento de criterios y la formación e identificación de categorías, constituyen la herramienta básica y esencial para estructurar el conocimiento en cualquier ámbito. Así, Devedzic (2001) establece que *'desde la perspectiva de la orientación a objetos, las ontologías proporcionan taxonomías jerarquizadas de clases, junto con sus correspondientes relaciones'*. La definición de Roche (2000) responde al concepto de vocabulario, o conjunto de términos: *'la ontología de un campo es un vocabulario de términos cuyos significados están estructurados en un sistema. Definida para un objetivo dado, expresa un punto de vista compartido por una comunidad. Una ontología se define empleando un lenguaje y está basada en una teoría (semántica) garante de las propiedades de la ontología en términos de consenso, coherencia, uso compartido y reutilización'*.

Las ontologías permiten que los conceptos y términos relevantes de un dominio dado sean identificados y definidos de forma no ambigua. Por lo tanto, las ontologías se ven como la clave tecnológica usada para describir la semántica de la información en varios lugares, superando el problema del conocimiento implícito y perdido, y por tanto permitiendo el intercambio de contenidos semánticos (Calì et al. 2005). Otros autores ponen el foco en asegurar la comunicación con independencia de los factores implicados. Éste es el caso de Deshayes, El Beqqali y Bouras (2005), que indican que el principal propósito de una ontología es posibilitar la comunicación entre sistemas informáticos, independizándola de las tecnologías de los sistemas individuales, de las arquitecturas de información y del dominio de aplicación.

Si se aumenta el grado de concreción, se puede decir que una ontología identifica y describe con precisión los conceptos, clases, entidades o términos importantes en su

dominio de interés, así como las relaciones válidas entre ellos, y que proporciona las definiciones formales y axiomas que restringen su interpretación. Para Guarino (1997) el propósito de una ontología es el de caracterizar una conceptualización, limitando las posibles interpretaciones de los símbolos no lógicos del lenguaje lógico, con el objeto de establecer un consenso acerca del conocimiento descrito por ese lenguaje. De hecho, el conjunto de términos y las definiciones de una ontología son compartidos por todos los participantes en el dominio, y por lo tanto constituyen una base para la comunicación acerca de dicho dominio (Chung et al. 2003). Una vez que la ontología ha sido formalizada, esta comunicación puede realizarse tanto entre humanos como entre sistemas de información. En este sentido, una ontología permite una rica variedad de relaciones estructurales y no estructurales, como generalización, herencia, agregación e instanciación, y puede suministrar un modelo de dominio preciso para aplicaciones de software. Por ejemplo, una ontología puede suministrar el esquema de los objetos de un sistema orientado a objetos y la definición de clases para un software convencional (Pouchard et al. 2000).

No obstante, y según lo expuesto hasta ahora, una ontología debe entenderse como algo más que un medio efectivo y eficiente para transmitir la información. Una ontología permite capturar conocimiento en un dominio de interés, y aunque en sus aplicaciones prácticas en ingeniería esto ha supuesto también el almacenamiento y mantenimiento de los datos, las ontologías están situadas en un nivel superior al correspondiente a los sistemas de información, ya que además permiten efectuar razonamientos en base a la información que contienen. Además, este conocimiento capturado puede ser compartido y reutilizado para diversos propósitos y aplicaciones relativas a su dominio. Por ello, las ontologías ocupan un lugar importante entre las técnicas de modelado y representación del conocimiento (Mostefai et al. 2005; Devedzic 2001). Como señalan Chandrasekaran et al. (1999), las ontologías proporcionan la posibilidad de describir y de prescribir nuestro conocimiento acerca de un dominio dado. Las ontologías permiten representar un cuerpo de conocimiento de manera formal y declarativa, y constituyen el fundamento (*foundation*) para construir las bases del conocimiento, representando la terminología propia de un dominio y facilitando la comunicación y la transmisión del conocimiento entre agentes heterogéneos en un sistema basado en el conocimiento (Bräuer 2007).

3. Alternativas de diseño para las ontologías

La conceptualización de un determinado ámbito de la realidad, entendida como una vista simplificada y abstracta de la misma, es la esencia de una ontología para dicho ámbito. Por lo tanto, la forma en la que se definen las entidades y se establecen las relaciones que constituyen esta conceptualización es un aspecto primordial. De ahí que clarificar las motivaciones y restricciones que conducen la conceptualización de la realidad sea de vital importancia en la construcción de una ontología, especialmente en el caso de las *foundational ontologies* descritas en la sección 6 de este capítulo.

Por ello, en la siguiente subsección se presentan los enfoques y alternativas más relevantes que subyacen en la organización de una ontología, y que contemplan distintos aspectos de la conceptualización.

3.1. Principales fundamentos ontológicos

Estos fundamentos ontológicos tienen relación, entre otros, con el carácter de la ontología (descriptiva o prescriptiva, multiplicativa o reduccionista, etc), con las características de sus entidades (particulares o universales, concretas o abstractas, etc.) y con los denominados paradigmas 3D y 4D.

Descriptivo versus prescriptivo

De los posibles enfoques adoptados para la construcción de una ontología destaca la diferenciación entre descriptivo y prescriptivo. En los enfoques descriptivos, el contenido de la ontología se describe mediante una caracterización de las entidades y de las relaciones entre entidades, similar a la que haría un usuario o un experto. Por su parte, en los enfoques prescriptivos el contenido de la ontología establece explícitamente el modo en que esas entidades y sus relaciones son caracterizadas. Habitualmente, los enfoques descriptivos adoptan una noción más flexible de la caracterización, permitiendo objetos arbitrarios en el modelo, que podrían no existir en el mundo real pero que son elementos conceptuales significativos para una comunidad de usuarios determinada. Sin embargo, los enfoques prescriptivos suelen adoptar una noción más estricta de la caracterización, estableciendo que solo los objetos que realmente existen o que representan tipos o categorías de objetos del mundo real puedan estar representados en el contenido del modelo (Gruninger et al. 2008).

Multiplicativo versus reduccionista

Una ontología reduccionista tiene como objetivo describir un gran número de conceptos ontológicos con el menor número de primitivas, mientras que una ontología multiplicativa pone el foco en conseguir la máxima expresividad aunque para ello sea preciso un gran número de conceptos básicos (Masolo et al. 2003). Por ejemplo, en una ontología multiplicativa se admite que diferentes entidades estén co-localizadas en el mismo espacio-tiempo, asumiendo que estas entidades co-localizadas son diferentes porque tienen propiedades esenciales incompatibles. Contrariamente, en una ontología reduccionista se postula que cada ubicación espacio-tiempo contiene como máximo un objeto y que la existencia de propiedades esenciales incompatibles que pueden apreciarse en la misma está ligada a la consideración de diferentes puntos de vista (Oberle et al. 2007). Para explicar la diferencia entre el enfoque multiplicativo y el enfoque reduccionista se puede emplear el ejemplo del jarrón y la arcilla de la que está hecho. Parece normal asumir que un jarrón deja de existir cuando ocurre un cambio radical en su forma, por ejemplo, cuando se rompe. No obstante, la cantidad de arcilla no se ve alterada por este evento. Según el enfoque multiplicativo, estas observaciones muestran que deben existir entidades diferentes que están co-localizadas: el jarrón y la arcilla que lo constituye. No obstante, según el enfoque reduccionista, el jarrón y la arcilla no son entidades diferentes, sino vistas diferentes del mismo objeto espacio-temporal.

Universales versus particulares

La distinción ontológica entre *particular* y *universal* puede caracterizarse mediante la relación de instanciación. Así, considerando que una instanciación es la particularización de una clase o entidad con un individuo concreto perteneciente a la misma, se establece que los *universales (universals)* pueden tener instancias y los *particulares (particulars)* no (Masolo et al. 2003). Es decir, en una ontología concreta, los *particulars* son individuos, o

instancias de determinadas clases, que serán los *universals* de dicha ontología. No obstante, esta distinción puede verse afectada por el punto de vista o por las necesidades concretas de modelado. Así, por ejemplo, 'torno' puede ser un *universal* cuyas instancias sean máquinas concretas, mientras que en otra ontología 'torno' podría ser un *particular* que proviene de la instanciación de la clase 'tipos de máquina'.

Entidades concretas y abstractas

Las entidades abstractas (*abstracts*) no existen en el espacio ni en el tiempo, es decir, no están localizadas. Por otra parte, las entidades concretas (*concretes*) existen al menos en el tiempo. Los objetos matemáticos, como números y conjuntos, son ejemplos de *abstracts*, mientras que los objetos ordinarios, como máquinas y utillajes o eventos (*events*), como la fabricación de una pieza, son ejemplos de *concretes*. Esta caracterización conduce inmediatamente a una aparente contradicción que se expresa mediante la pregunta: ¿cómo es posible que existan los *abstracts* si no existen en ningún instante de tiempo? Dicha contradicción puede soslayarse mediante la afirmación de que los *abstracts* son eternos e inmutables, es decir, que existen en todo momento y no cambian (Masolo et al. 2003).

Paradigma 3D y paradigma 4D

Una de las alternativas ontológicas fundamentales se refiere a la noción del cambio y lo que significa cambiar para una entidad. En general, según el paradigma 3D, los objetos: (a) se extienden en tres dimensiones espaciales; (b) están presentes completamente en cualquier instante de su vida; y (c) son entidades cambiantes, en el sentido de que en instantes de tiempo distintos pueden instanciar propiedades diferentes. Por el contrario, una perspectiva de cuatro dimensiones (paradigma 4D) establece que los objetos: (a) tienen partes temporales que se concatenan entre sí como 'gusanos del espacio-tiempo' conformando la totalidad del objeto; (b) solo están presentes parcialmente en cada instante de tiempo; y (c) son entidades cambiantes, en el sentido de que pueden tener propiedades diferentes en fases diferentes (Masolo et al. 2003). En las dos subsecciones siguientes, se muestran algunos argumentos específicos relacionados con esta cuestión.

Endurants y perdurants

Los *endurants* son entidades que se caracterizan por estar en el tiempo. Los *endurants* están enteramente presentes, con todas sus partes, en cualquier instante de tiempo de su existencia. Por otra parte, los *perdurants* son entidades que suceden en el tiempo. Los *perdurants* se extienden en el tiempo mediante la acumulación de diferentes partes temporales, por tanto, en cualquier instante de tiempo en el que existen solo están presentes algunas de sus partes temporales (Masolo et al. 2003). *Endurants* y *perdurants* pueden caracterizarse de formas diferentes. Una cosa es un *endurant* si existe en más de un instante de tiempo y sus partes pueden ser determinadas solamente en relación a otra cosa, por ejemplo el tiempo. Los *endurants* necesitan una relación de participación indexada en el tiempo, mientras que los *perdurants* no la necesitan (Masolo et al. 2003). Por ejemplo, una afirmación como 'este utillaje forma parte de la máquina' está incompleta, o es imprecisa, a no ser que se especifique el instante de tiempo al que se refiere, ya que el utillaje es un *endurant,* y por tanto puede cambiar en el tiempo; mientras que otra afirmación como 'la colocación del utillaje forma parte de la preparación de la máquina' no precisa de dicha especificación temporal, ya que la colocación del utillaje, por ser un *perdurant*, es independiente del intervalo de tiempo en el que se realice.

Entidades co-localizadas

Es perfectamente normal admitir la existencia de objetos co-localizados temporalmente y de objetos co-localizados espacialmente, mientras que resulta más problemático justificar la existencia de objetos co-localizados espacial y temporalmente (Masolo et al. 2003). Un ejemplo de co-localización temporal sería el de la broca y la pieza durante un proceso de taladrado, mientras que el material a mecanizar (sobrante o creces) y las virutas resultantes del proceso constituyen un ejemplo de co-localización espacial, porque antes de que se produzca el taladrado ambos ocupan una misma posición. La existencia de objetos co-localizados espacial y temporalmente puede encontrar ciertas dificultades, como las derivadas de la conceptualización de las entidades. Por ejemplo, para admitir que la pieza perforada y el agujero son dos objetos co-localizados espacial y temporalmente, sería preciso establecer el significado y las relaciones ontológicas entre un objeto, como la pieza, y sus características o propiedades, como la de estar perforada. No obstante, puede distinguirse la posición de una ontología en relación a la co-localización de entidades. Por ejemplo, distinguiendo entre entidades que están co-localizadas espacialmente con entidades materiales (una persona y su cuerpo) y entidades que son dependientes de otras entidades materiales aunque no están co-localizadas espacialmente con ellas, como agujeros, manchas, sombras, etc. (Masolo et al. 2003).

3.2. Otros enfoques ontológicos

Además de las indicadas previamente, cuando se construye una ontología, deben considerarse otros enfoques ontológicos que condicionarán su capacidad para ajustarse a la realidad a representar y que constituyen opciones de diseño que conducen a conceptualizaciones distintas de la realidad o del mundo:

- **Actualismo** versus **posibilismo**. El *actualismo* afirma que sólo existe lo que es real, mientras que el *posibilismo* admite *possibilia* o posibilidades. Dicho de otro modo, el posibilismo admite situaciones o mundos posibles (Masolo et al. 2003).
- **Presentismo** versus **eternalismo**. El *presentismo* asume que sólo existe lo que está presente, mientras que el *eternalismo* aboga por la existencia del pasado, del presente y del futuro (Masolo et al. 2003).
- **Realismo** versus **constructivismo**. Según el realismo, las entidades no se ven modificadas por el modo en el que son descritas. Sin embargo, para el constructivismo, cualquier *realidad cierta* es relativa, ya que no puede ser modelada sin la carga o sesgo teórico del observador que la describe (DUL 2014). Lambert et al. (2008) presentan la alternativa del realismo frente al nominalismo, donde el significado del realismo es diferente al anterior. Según esta disyuntiva ontológica, los *universals* deben ser considerados como cosas en el mundo (realismo) o las cosas del mundo deben ser consideradas únicamente como *particulars*, o instancias de propiedades y relaciones (nominalismo).
- Formas de representar el **tiempo** y el **carácter modal**. Respecto a este aspecto existen básicamente dos alternativas: el enfoque *includesmodal*, que consiste en incluir en el sistema los operadores modales y temporales desde el principio, como en la expresión ‘es posible que la máquina esté preparada’; y el enfoque *reproducesmodal*, que consiste en razonar con un lenguaje de primer orden

añadiendo parámetros de tiempo y situación a los predicados, como en la expresión 'existe un mundo en el que la máquina está preparada'. El primer enfoque está ligado al *actualismo* y al *presentismo*, mientras que el segundo está ligado al *posibilismo* y al *eternalismo* (Masolo et al. 2003).

- **Analítico** versus **sintético**. El enfoque analítico corresponde a la información orientada hacia la identificación del significado de los términos o información que puede determinarse independientemente del acceso al mundo real. El enfoque sintético también incluye la información orientada hacia el conocimiento relativo al mundo o información, pero que solamente puede determinarse a través del acceso al mundo real (Lambert et al. 2008).

- **Formal** versus **informal**. La distinción entre los enfoques formal e informal se establece a partir de la disyuntiva de expresar los conceptos mediante un lenguaje formal, es decir, asociado con una lógica formal, o mediante un lenguaje informal (Lambert et al. 2008). Se considera que un lenguaje es formal cuando su sintaxis y su semántica pueden ser definidas con precisión. Si la semántica de un lenguaje está definida formalmente, sus expresiones tienen una interpretación única (Mili, Tremblay y Jaoude 2010).

- **Cuantitativo** versus **cualitativo**. Según esta distinción, el mundo puede entenderse cuantitativamente a través de diversas medidas o cualitativamente a través de diversos predicados. Los enfoques cuantitativo y cualitativo no son excluyentes (Lambert et al. 2008).

- **Funcionalismo** versus **no-funcionalismo**. La distinción se refiere a la capacidad de la conceptualización para expresar consideraciones funcionales de las entidades del mundo, como la descripción de las partes funcionales de un sistema y los roles desempeñados por esas partes funcionales (Lambert et al. 2008).

- **Sicológico** versus **no-sicológico**. Esta distinción concierne a la capacidad de la conceptualización para expresar consideraciones sicológicas de las entidades del mundo, como la descripción de estados mentales y cómo se construyen estos estados mentales (Lambert et al. 2008).

- **Social** versus **no-social**. Esta distinción corresponde a la capacidad de la conceptualización para expresar interacciones sociales entre entidades sicológicas del mundo, por ejemplo la descripción de construcciones sociales y el modo en el que ellas se forman (Lambert et al. 2008).

4. Tipos de ontologías y criterios de clasificación

Basándose en la relación entre ontología y lenguaje, Fox y Gruninger (1998) propusieron clasificar las ontologías en informales, semi-formales y formales, al considerar el grado de precisión con el que se establecen tanto su sintaxis como su semántica. En ese contexto, el lenguaje natural sería un ejemplo de ontología informal, los lenguajes lógico-matemáticos serían ontologías formales y a medio camino entre ambos se encuentran las ontologías semi-informales.

En otras clasificaciones se contemplan otros de los múltiples aspectos que caracterizan la gran diversidad de las ontologías existentes, asumiendo la conveniencia de dicha diversidad. Entre estas destaca la de Gruninger et al. (2008) que propone una clasificación en base a dos grupos de dimensiones (dimensiones semánticas y dimensiones pragmáticas), que sirven para identificar las similitudes y diferencias entre diversas ontologías en función de su posición relativa en el espacio definido por estas dimensiones. Las dimensiones semánticas consideradas son: expresividad del lenguaje de representación de la ontología; nivel de formalización y estructuración de las entidades; y nivel de detalle de los contenidos de la ontología. Por su parte, dentro de las dimensiones pragmáticas establecidas en este marco se encuentran: el rol del razonamiento automático, que se establece en función de las características y capacidad del razonamiento empleado; la distinción entre descriptivo y prescriptivo; la metodología de diseño de la ontología, que contempla entre otros los enfoques *bottom-up* y *top-down*; la gobernanza, o cómo se toman las decisiones relativas a la estructura y contenidos de la ontología; y la utilización prevista para la ontología.

Bräuer (2007) establece una clasificación de las ontologías basada en tres criterios: especificidad, propósito y expresividad. A partir de estos criterios se establecen diversas categorías (ontologías de dominio, *core* y genéricas, atendiendo al criterio de especificidad; ontologías de aplicación y de referencia, según el criterio de propósito; y ontologías *light-weight* y *heavy-weight*, respecto al criterio de expresividad) que se muestran en la Figura 3.1. Según estos autores, dichos criterios de clasificación no son independientes, y de este modo, una ontología de aplicación sería también de dominio y *light-weight*, mientras que una ontología de referencia sería genérica y *heavy-weight*.

Las ontologías específicas (de dominio) se utilizan en un dominio concreto, para un propósito particular, y tienen baja exigencia de expresividad sustentada en una representación formal *light-weight* propia del dominio en el que se utiliza, donde los miembros de la comunidad ya conocen de antemano el significado de los términos. Las ontologías genéricas se utilizan en diferentes dominios, para propósitos más generales y tienen una alta exigencia de expresividad, soportada por una representación formal *heavy-weight*, que deriva de la necesidad de establecer acuerdos precisos sobre el significado de los términos entre los dominios para garantizar el intercambio de datos y servicios en comunidades multiculturales y multilingües (Masolo et al. 2003).

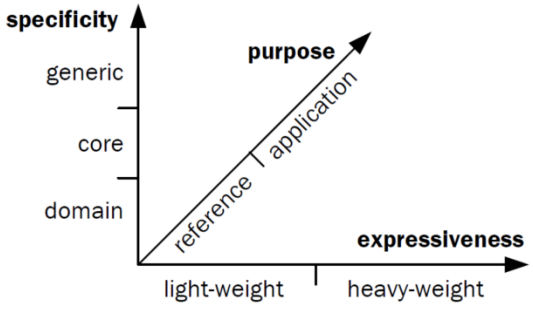


Figura 3.1. Clasificación de las ontologías según Oberle (Bräuer 2007).

A pesar de la coincidencia entre diversos autores, que utilizan varios de los criterios antes expuestos, según Bräuer (2007) puede resultar preferible una clasificación unidimensional

tomando como único criterio el nivel de abstracción (Figura 3.2), que permite clasificar las ontologías atendiendo a su grado de especificidad en: *upper ontologies* (ontologías de base, *foundational ontologies o genéricas*), que definen conceptos aplicables a la mayoría o a todos los dominios; *core ontologies* (ontologías nucleares), que definen conceptos compartidos por varios dominios similares o relacionados; *domain ontologies* (ontologías de dominio), que contienen conceptos específicos a un dominio particular de interés; y *application ontologies* (ontologías de aplicación), que especializan los conceptos de una ontología de dominio con variantes específicas para una aplicación (identificadas por los autores como ontologías de aplicación, aunque esta denominación coincide con uno de los tipos de ontologías según el criterio '*purpose*' de la Figura 3.1). Las ontologías con mayor nivel de abstracción (*upper ontologies*) requieren de mano de obra humana, que puede incorporar el inmenso beneficio de los resultados y metodologías de disciplinas como la filosofía, la lingüística y las ciencias cognitivas. Frente a ellas, las ontologías de más bajo nivel de abstracción y con expresividad *light-weight*, se pueden construir de forma semiautomática, por ejemplo, mediante la explotación de técnicas de aprendizaje automático (Masolo et al. 2003).

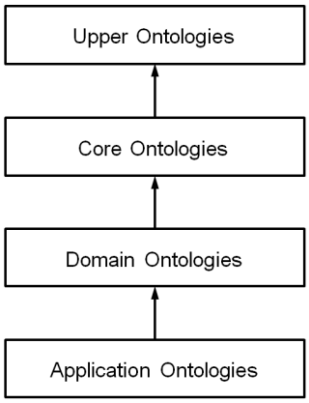


Figura 3.2. Tipos de ontologías según el grado de especificidad (Bräuer 2007).

Tomando como referencia esta última clasificación, los modelos de alto nivel pueden obtenerse a partir de la adaptación de una ontología de base (*upper ontology*) que describe los conceptos básicos (*foundational concepts*), que son suficientemente generales para ser aplicados en un amplio rango de dominios, y está diseñada para que pueda extenderse y reunir los requisitos adicionales introducidos por un dominio específico (Löckelt 2008). En línea con lo expuesto al principio de esta sección, las ontologías de base facilitan la integración de otras ontologías de menor grado de abstracción, ya que permiten compartir vocabulario para describir la semántica de entidades ontológicas sin ambigüedad y de forma procesable por ordenador.

Como hemos visto anteriormente en el punto 1, este aspecto es fundamental a la hora desarrollar ontologías en ingeniería, porque la utilización de diferentes ontologías sin una base común conduce necesariamente a un desentendimiento. Este aspecto, como se verá posteriormente, se ha considerado especialmente importante para el desarrollo de la propuesta.

5. Lenguajes y herramientas para la implementación de las ontologías

En esta sección se abordan distintos aspectos relacionados con la utilización práctica de las ontologías en ingeniería, desde los vinculados con las aplicaciones basadas en la inferencia del conocimiento hasta los relacionados con las herramientas informáticas para la edición, presentación y utilización compartida de las ontologías.

La lógica es la disciplina que estudia los principios del razonamiento. Un razonamiento basado en abstracciones correspondientes a un área de estudio o interés determinado, que permite obtener conclusiones en dicho ámbito. En la siguiente subsección se presentan los principales tipos de lógica que se han utilizado en la implementación de las ontologías.

El World Wide Web Consortium (W3C) ha sido el promotor del desarrollo de la Web semántica, como medio para que las máquinas pudieran realizar análisis más profundos y ayudar a las personas en sus tareas. La consecución de dicho objetivo requiere que la Web semántica tenga determinadas funcionalidades como la capacidad para compartir el conocimiento y conseguir que algunas informaciones fuesen entendibles por las máquinas. Debido a la complejidad inherente a la semántica y al conocimiento, se ha propuesto una estructura jerarquizada para la Web semántica, en la que cada una de sus capas proporciona el soporte operativo y lógico a las capas superiores (ver Figura 3.3), consiguiendo así una mejor utilización de los recursos disponibles en la Web. En la subsección 5.2 se realiza un acercamiento al lenguaje OWL (Ontology Web Language) definido por el W3C.

5.1. Lógica y lenguajes lógicos

Mediante la utilización de lenguajes basados en la lógica es posible conseguir el grado de expresividad que precisan las ontologías, así como realizar una axiomatización de sus entidades para especificar y limitar su significado. Entre las lógicas existentes (ver anexo 1), destacan por su relación con la implementación informática de las ontologías, la lógica proposicional y la lógica de primer orden. La lógica proposicional, o lógica de orden cero, es un sistema formal en el que los elementos más simples son las variables proposicionales, que representan proposiciones y que tienen un valor de verdad definido (verdadero o falso). Además, esta lógica incluye constantes lógicas, o conectivas, que representan operaciones sobre proposiciones, capaces de formar otras proposiciones de mayor complejidad. Por tanto, la lógica proposicional permite realizar inferencias lógicas sobre proposiciones complejas a partir de otras proposiciones simples, pero sin tener en cuenta la estructura interna de estas últimas.

La lógica de primer orden (FOL), también llamada lógica de predicados o cálculo de predicados, es un sistema formal diseñado para estudiar la inferencia en los lenguajes de primer orden, que son lenguajes formales con cuantificadores que alcanzan sólo a variables de individuo, y con predicados que expresan propiedades y relaciones cuyos argumentos son individuos o variables de individuo. A continuación se explican estos conceptos (predicados, propiedades, relaciones, individuos, variables de individuo y cuantificadores). Un predicado es una expresión que puede conectarse con otras expresiones para formar otro predicado de mayor extensión u oración. Por ejemplo, el predicado ‘La torreta es un utillaje’ se compone de los predicados ‘La torreta’ y ‘es un utillaje’; y el predicado ‘La herramienta se fija en la torreta’ está compuesto de los predicados ‘La herramienta’, ‘se fija en’ y ‘la torreta’. El primero de estos predicados compuestos expresa una propiedad,

mientras que el segundo expresa una relación entre dos entidades. En dichos ejemplos, 'torreta', 'utillaje' y 'herramienta' son constantes de individuo, o individuos, ya que son expresiones que hacen referencia a una entidad determinada. La lógica de primer orden también permite hacer uso de variables, o variables de individuo, que hacen referencia a entidades indeterminadas mediante expresiones como 'esto', 'eso' o 'aquello'. Los cuantificadores, como ∀ y ∃, son expresiones que se utilizan para indicar que una condición se cumple para un cierto número de individuos.

Con la lógica de primer orden (FOL), no se puede garantizar la resolución de un problema computacional en un número finito de pasos. Para soslayar esta limitación, se puede restringir el número de variables en los símbolos de los predicados, por ejemplo hasta un máximo de dos, con lo cual se simplifica mucho el trabajo de los razonadores haciendo que la lógica sea resoluble. Las Lógicas Descriptivas (DLs) son un subconjunto de FOL que permite hacer resoluble sus inferencias en intervalos de tiempo asumibles. DL es una familia de formalismos de representación de conocimientos diseñados para representar y razonar sobre ontologías, esquemas de bases de datos, configuraciones, etc. Los sistemas DL se basan en una KB (Knowledge Base) formada por aserciones sobre clases y aserciones sobre inferencias, sobre la que se pueden realizar inferencias (Romero-Llop 2007).

5.2. El lenguaje OWL en el marco de la Web semántica

Un elemento clave para alcanzar los objetivos de la Web semántica son las ontologías, cuya función se centra en la descripción inequívoca y procesable por ordenador de los términos de un vocabulario compartido. Para ello, se precisa una base de conocimiento y unos sistemas de razonamiento compatibles con la utilización de lenguajes formales y con un conocimiento que puede ir creciendo y variando en el tiempo. Lo que conduce a la denominada 'suposición de mundo abierto', según la cual, en la Web semántica no se asume ni garantiza la integridad o disponibilidad constante de la información.

Como puede apreciarse en la Figura 3.3, la primera capa de la Web semántica está formada por Unicode y URIs (Uniform Resource Identifiers). Mediante Unicode se definen los símbolos con los que se construyen las restantes capas, incluida la capa de los URIs, que sirve para hacer referencia a un espacio de objetos o conceptos sobre los que se pretende ofrecer información. En la capa siguiente (capa de intercambio de datos) se encuentran XML, XML Schema y NS (Name Spaces). El intercambio de datos se realiza mediante el lenguaje XML, y la estructura de dichos datos se define con XML Schema (XMLS). Mientras que los espacios de nombres (NS) son URIs que representan conjuntos de nombres dentro de la estructura de datos en la que están definidos.

La capa de aserciones (o aseveraciones) proporciona un modelo de datos básico para expresar aseveraciones acerca de recursos arbitrarios que pueden ser identificados a través de un URI (ver anexo 2). A través de estas aseveraciones, que se realizan mediante RDF (Resource Description Framework), se realiza la descripción de estos recursos. El número de aseveraciones que se pueden realizar para describir un recurso es arbitrario y se puede incrementar para ir completando dicha descripción. Además, el recurso descrito en una aseveración podría ser una propiedad de otra aseveración distinta.

Algunas de las restricciones del lenguaje RDF son: la inexistencia de palabras clave para distinguir clases (entendidas como agrupaciones de objetos), y no distinguir entre propiedades y elementos (en RDF ambos son considerados como recursos). Para solventar esta limitación se disponde de RDFS (RDF Schema), una extensión semántica de RDF que incluye términos como *Class*, *Property*, *type*, *subClassOf*, *range* o *domain*, que añaden soporte a las jerarquías de clase y propiedad, y a las restricciones del valor de las propiedades. No obstante, RDFS carece de restricciones de cardinalidad o existencia, lo que limita la semántica del lenguaje. Por esto, la posibilidad de crear ontologías mediante RDFS se limita a casos muy sencillos basados en relaciones taxonómicas (Bräuer 2007).

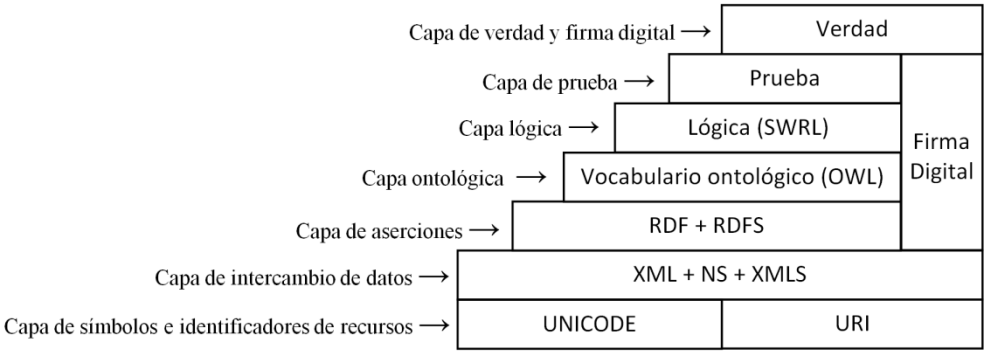


Figura 3.3. Estructura de capas de la Web semántica (ARC 2002).

Las capas que se describen a continuación (capa ontológica y capa lógica) son las más relevantes para el propósito de esta tesis. Sobre ellas se sitúan la capa de prueba (ver subsección 5.4 'Razonadores') y la de verdad y firma digital. En esta última se decide el grado de fiabilidad que se le concede al razonamiento, que dependerá de la confianza de las fuentes de información y la validez de la información original, y de la invariabilidad de dicha información desde que se generó. En este proceso de validación de la información, la firma digital juega un papel clave, ya que garantiza la autenticidad de la autoría y de la fuente de la información (Romero-Llop 2007).

5.2.1. Capa ontológica

En la capa ontológica se enriquece la información asociada al tipo de aseveraciones que se expresan mediante RDF y RDFS. Para ello se utiliza OWL (Ontology Web Language), un lenguaje de marcado desarrollado por el W3C como una extensión del vocabulario de RDF. OWL está diseñado para publicar y compartir datos usando ontologías en la Web, y para ser usado en aplicaciones que procesan el contenido de la información. Para ello, OWL proporciona una semántica formal interpretable por ordenador mediante la definición de vocabulario adicional (Lin 2008). Se trata de un lenguaje de ontologías más rico y mucho más poderoso que RDFS. OWL soporta: la combinación booleana de clases y permite declarar propiedades como transitiva, única o inversa de otra propiedad; los axiomas de OWL, que son una parte esencial de su sintaxis, permiten establecer restricciones de cardinalidad o de propiedades para la especificación de clases; y también se pueden aplicar restricciones a las instancias de las clases y a las propiedades que relacionan a las instancias.

OWL incluye la sintaxis de intercambio RDF/XML y tiene tres sub-lenguajes de diferente expresividad y complejidad: OWL Lite, OWL DL (OWL Description Logic) y OWL Full. OWL-Lite es el más simple sintácticamente. Se destina a situaciones o aplicaciones en las que solamente se necesita una jerarquía de clases simple con restricciones simples, por ejemplo para tesauros, diccionarios o catálogos. OWL Full es el más expresivo y goza de libertad sintáctica para utilizar las construcciones de RDF, pero no ofrece garantías computacionales al razonamiento, ya que esta expresividad se consigue a costa de limitar su capacidad de inferencia. De hecho, el razonamiento automático no es posible en ontologías OWL Full. Por tanto, OWL Full se destina a situaciones en las que una expresividad muy elevada es más importante que la capacidad del lenguaje para completar los razonamientos. OWL DL es una representación intermedia entre las dos anteriores, que enfatiza la completitud computacional y la capacidad de razonamiento reteniendo en la medida de lo posible una expresividad similar a OWL Full. OWL DL utiliza el vocabulario completo correspondiente a Description Logics (DL). No obstante, este vocabulario se aplica con una serie de restricciones, como separación por pares entre clases, *datatypes*, *datatype properties*, *object properties*, etc., que son requeridas para la inferencia y el razonamiento basado en Description Logic. De este modo, OWL DL permite el razonamiento automático, y por tanto, en una ontología conforme a OWL DL es posible computar automáticamente la jerarquía de clases y verificar la consistencia (Jang et al. 2008).

5.2.2. Capa lógica

La capa lógica es la que trata de las relaciones lógicas entre los objetos de estudio, generando una ontología más detallada y completa que proporciona una semántica formal que permite realizar las implicaciones para deducir las definiciones de los términos y de las relaciones (Lin, Harding y Shahbaz 2004).

El Lenguaje de Reglas de la Web Semántica (SWRL) combina OWL Lite y OWL DL con el Lenguaje de Marcado de Reglas (RuleML), proporcionando un lenguaje de reglas compatible con OWL (Nadarajan y Chen-Burger 2006). SWRL permite escribir reglas de tipo *cláusulas de Horn*[3] acerca de una base de conocimiento OWL, lo que facilita razonamientos que no pueden realizarse únicamente con OWL (Bräuer 2007).

5.2.3. OWL versus KIF

Entre los lenguajes para la especificación de ontologías se encuentra KIF (Knowledge Interchange Format), que emplea la lógica FOL y junto a OWL es el formato principal para la representación de ontologías. KIF es un lenguaje lógico-matemático con una notación textual flexible que le permite definir formalmente los conceptos, y ha sido utilizado durante décadas por la comunidad de Inteligencia Artificial (AI). OWL se ha desarrollado recientemente para ayudar al intercambio de conocimiento en Internet (Cochrane et al. 2004), mientras que KIF es un lenguaje declarativo que puede expresar afirmaciones lógicas arbitrarias y reglas, y permite representar el conocimiento y el meta-conocimiento, o conocimiento acerca del conocimiento (Lubell 2003). Por tanto, KIF proporciona el nivel de rigor necesario para definir conceptos de forma no ambigua.

3 Las *cláusulas de Horn* constituyen reglas del tipo 'modus ponendo ponens', es decir, 'si es verdad el antecedente, entonces es verdad el consecuente' (Nadarajan and Chen-Burger 2006). En la tabla 5.3 'Rules of inference written in SWRL' del capítulo 5, se muestran varias reglas escritas en SWRL para la ontología PPDRC.

KIF tiene mayor expresividad semántica que OWL y puede abordar representaciones más complejas, ya que su sintaxis admite relaciones n-arias (ver anexo 3) frente a las relaciones binarias de RDF-OWL. El lenguaje XML en el que se basa OWL es adecuado para la representación de secuencias, jerarquías e instantes de tiempo en la realización de actividades y subactividades, pero es poco adecuado para representar el tipo de restricciones complejas que se precisan para la descripción de procesos. No obstante, las limitaciones expresivas de OWL se ven compensadas en parte por la utilización de reglas mediante SWRL, con las que se pueden introducir axiomas que no pueden expresarse con OWL. Por otra parte, la utilización de un lenguaje ampliamente difundido como OWL DL conlleva ventajas evidentes para la utilización de ontologías en entornos distribuidos, a las que se unen las ventajas ya comentadas de los lenguajes que utilizan DL en relación a sus prestaciones respecto a la inferencia lógica.

5.3. Herramientas de software. Editores de ontologías

Existen muchos entornos para el desarrollo de ontologías o editores de ontologías, en los que se incorporan las metodologías y lenguajes específicos para ontologías. Entre ellos se encuentran: Protégé (Musen 2003), OntoEdit (Sure 2002), WebODE (Corcho 2002) y Hozo (Kozaki 2002; Sunagawa 2003), que cubren un amplio rango del proceso de desarrollo de una ontología (Mizoguchi 2004).

Protégé ofrece grandes prestaciones en la fase de utilización de la ontología. Entre sus características destacan: extensibilidad, que permite a los usuarios redefinir las primitivas representativas; disponer de una herramienta semiautomática para la fusión y el alineamiento de ontologías; capacidad para desarrollar algunas tareas automáticamente y para guíar al usuario en el desarrollo de otras; formato de fichero de salida personalizable capaz de adaptarse a cualquier lenguaje formal; interfaz de usuario personalizable; y potente y sofisticada arquitectura *plug-in* capaz de permitir la integración con otras aplicaciones (Mizoguchi 2004). Protégé, que utiliza entre otras las sintaxis OWL y RDF, permite la construcción de ontologías de dominio, ya que, como se aprecia en la Figura 3.4, con este editor pueden definirse las clases, las jerarquías de clase, las propiedades, las restricciones para el valor de las propiedades, las relaciones entre clases y las propiedades de estas relaciones (Lin, Harding y Shahbaz 2004).

Protégé permite describir conceptos y también proporciona nuevas facilidades o servicios. Cuenta con un rico conjunto de operadores, por ejemplo, intersección, unión y negación. Dado que Protégé se basa en un modelo lógico que permite que los conceptos sean definidos y descritos, los conceptos complejos se pueden construir sobre las definiciones de otros conceptos más simples. Por otra parte, el modelo lógico permite el uso de un razonador que puede comprobar si todas las declaraciones y definiciones de la ontología son coherentes entre sí y también puede reconocer que los conceptos encajan en cada una de las definiciones. El razonador, por tanto, puede ayudar a mantener la jerarquía correctamente. Esto es particularmente útil cuando se trata de casos en los que las clases pueden tener más de una superclase.

El editor de ontologías OntoEdit se basa en CommonKADS, una metodología para sistemas basados en el conocimiento. OntoEdit hace uso de dos herramientas en la fase de captura de la ontología: OntoKick y Mind2Onto. La primera permite construir estructuras relevantes para la descripción de ontologías informales mediante la obtención de preguntas de

competencia que la ontología resultante tendrá que responder. Mientras que Mind2Onto es una herramienta gráfica para la captura de relaciones informales entre conceptos (Mizoguchi 2004).

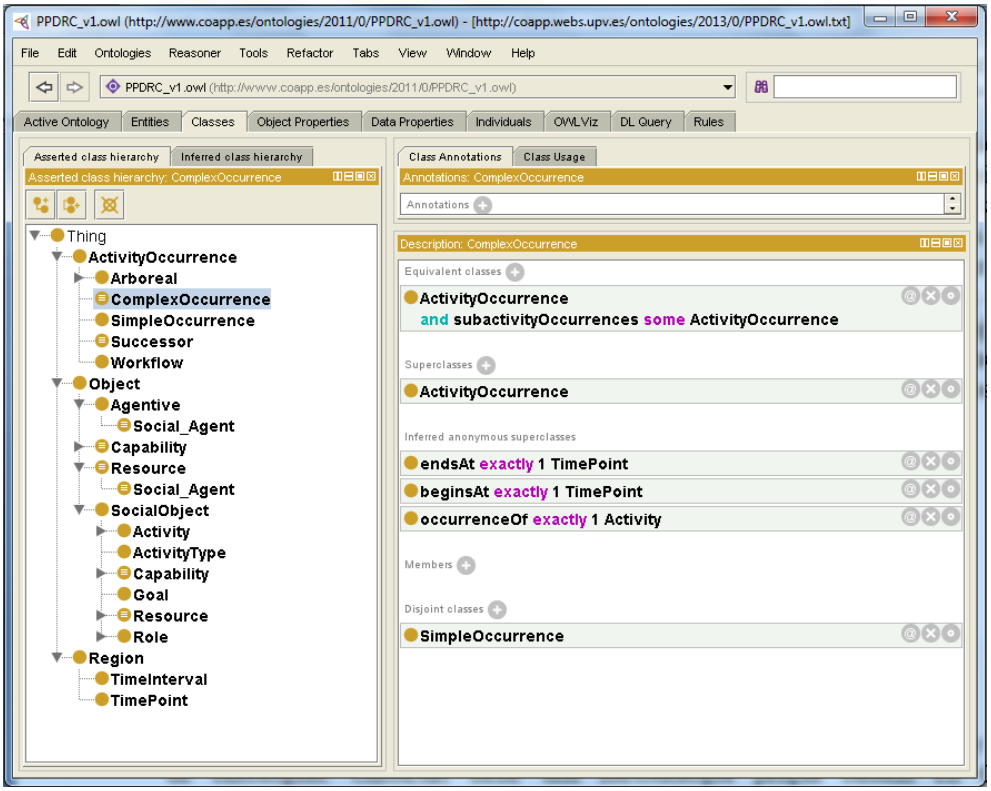


Figura 3.4. Interfaz de usuario del editor de ontologías Protégé, con la taxonomía de clases de la ontología PPDRC y los axiomas de definición de la clase 'ComplexOccurrence', que restringen su semántica (Protégé 2014).

WebODE es un banco de trabajo para la ingeniería ontológica que soporta la construcción de una ontología a nivel de conocimiento, y tiene servicios de importación/exportación con XML, cubriendo la mayoría de los procesos que aparecen en el ciclo de vida de una ontología. Mientras que Protégé y OntoEdit se basan en una arquitectura *plug-in*, WebODE está basado en una arquitectura servidor-cliente que proporciona una gran extensibilidad y amplias posibilidades de uso, a través de la inclusión de nuevos servicios y del uso de los servicios existentes. Las ontologías creadas con WebODE se almacenan en una base de datos SQL (Structured Query Language), lo que permite mejorar las prestaciones de grandes ontologías. Además, WebODE dispone de servicios de traducción a y desde varios lenguajes de especificación de ontologías (Mizoguchi 2004).

Por último, Hozo es un entorno de ingeniería ontológica integrado para la construcción y utilización de ontologías de tarea y ontologías de dominio basado en las teorías ontológicas de base (Kozaki 2002; Sunagawa 2003). Hozo genera código DAML+OIL (DARPA Agent

Markup Language + Ontology Interchange Language) para exportar la ontología y las instancias, mientras que la representación interna del editor de ontología, que está oculta para los usuarios, es XML. Al igual que otros editores, Hozo proporciona una interfaz gráfica para los usuarios que facilita la edición y modificación de las ontologías.

5.4. Razonadores

Además de una interfaz de usuario adecuada y de los mecanismos para facilitar y sistematizar la comunicación entre colaboradores, la principal ayuda para el diseño y mantenimiento de ontologías suministrada por las herramientas actuales es una interfaz con los sistemas de razonamiento o razonadores, la mayoría de los cuales están basados en OWL DL. Los razonadores soportan la capa de prueba de la Web semántica (ver Figura 3.3), y pueden devolver ejemplos de objetos que cumplen unas restricciones definidas o realizar las comprobaciones para determinar la veracidad de una suposición. Según Calì et al. (2005) la funcionalidad principal de los razonadores consiste en clasificar los conceptos de la ontología estableciendo relaciones de tipo subclase y superclase, y en detectar errores lógicos que indiquen fallos de modelado. Es decir, la validación y análisis de las ontologías.

Entre los razonadores que trabajan con ontologías escritas con OWL DL se encuentran Racer (Horridge et al. 2004) y Fact++, que son, junto a Pellet[4] los tres razonadores más utilizados con OWL (Calì et al. 2005). Con ellos, como se ha indicado anteriormente, se chequea la ontología para computar automáticamente la jerarquía de la clasificación, y también para verificar su consistencia lógica. La jerarquía de clases descrita en la ontología se denomina jerarquía aseverada (*asserted hierarchy*), mientras que la jerarquía de clases resultante del proceso de razonamiento se denomina jerarquía inferida (*inferred hierarchy*). Después del chequeo realizado por el razonador, puede visualizarse si alguna clase ha sido reclasificada (es decir, si ha cambiado su superclase) o si es inconsistente.

A pesar de las restricciones de FOL en relación al razonamiento, existen razonadores para FOL como Vampire, un demostrador de teoremas (*theorem prover*) que fue propuesto por el NIST para comprobar la validez de las expresiones de PSL escritas en KIF (PSL 2010). Normalmente, los razonadores para FOL se basan en la búsqueda de incongruencias en una base de conocimiento mediante la utilización de procedimientos no completos, por lo que no se puede garantizar su eficacia en relación a la búsqueda de estas incongruencias.

6. Ontologías de base

Para la mayoría de las aplicaciones prácticas, las ontologías se presentan como estructuras taxonómicas de términos primitivos o compuestos junto con las definiciones correspondientes. En otros casos, cuando la necesidad de establecer acuerdos precisos en cuanto al significado de términos se vuelve crucial, se necesita una axiomatización lógica rigurosa para las relaciones entre los términos y para la estructura formal del dominio a representar, lo que conduce a la utilización de las denominadas ontologías axiomáticas.

4 Pellet ha sido el razonador seleccionado para el trabajo de esta tesis, ya que permite la clasificación y razonamiento con individuos de ontologías OWL/SWRL.

Éstas se expresan en diferentes formas y pueden tener diferentes niveles de generalidad. Entre ellas están las *foundational ontologies* u ontologías de base, que abordan ámbitos muy generales y se dedican en última instancia a facilitar el entendimiento mutuo y la interoperabilidad entre personas y máquinas. A continuación, se presentan algunas de las *foundational ontologies* más representativas, prestando especial atención a DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering) y PSL (Process Specification Language).

Entre las iniciativas más relevantes en torno a las ontologías de base u ontologías de alto nivel se encuentra el proyecto WonderWeb (Masolo et al. 2003), que reúne tres de estas ontologías, con sus opciones ontológicas fundamentales y sus relaciones formales. Los fundamentos y alternativas en que se basan los diferentes módulos (ontologías) de WonderWeb se han identificado de la forma más explícita posible, con el fin de formar una red de ontologías diferentes pero relacionadas sistemáticamente, que pueden dar soporte a cada aplicación a través de los supuestos ontológicos más adecuados en cada caso.

Tabla 3.1. Alternativas ontológicas (Lambert et al. 2008).

	Ontologías						
Alternativas ontológicas	DOLCE	OpenCYC	Mephisto	SUMO	BFO	OCHRE	JC3IEDM
Analítico	X	X	X	X	X	X	X
Sintético		X					
Descriptivo	X	X					X
Prescriptivo			X	X	X	X	
Multiplicativo	X	X		X			X
Reduccionista			X		X	X	
Formal	X	X	X	X	X	X	X
Informal							
Realismo	X	X		X	X		
Nominalismo			X			X	X
Endurantismo	X	X		X	X	X	X
Perdurantismo		X	X	X		X	
Cuantitativo	X	X	X	X	X		X
Cualitativo	X	X	X	X	X	X	
Funcionalismo	X	X	X	X	X	X	X
No-Funcionalismo							
Sicológico	X	X	X	X	X		
No-Sicológico						X	X
Social	X	X	X	X	X		X
No-Social						X	

DOLCE, la primera de las tres ontologías que integra WonderWeb, se describirá más adelante en este capítulo. La ontología OCHRE (Object-Centered High-level Reference Ontology), el segundo módulo de WonderWeb, tiene un enfoque prescriptivo. Ésto constituye su principal diferencia con DOLCE, que tiene un enfoque descriptivo. BFO (Basic Formal Ontology) es el tercer módulo de la ontología Wonder Web, y su objetivo es reconciliar los enfoques *three-dimensionalist* (3D) y *four-dimensionalist* (4D), en lo que podría denominarse una teoría bi-ontológica o una ontología bicategorial (Masolo et al. 2003).

Lambert et al. (2008) analizan las tres ontologías anteriores, junto con otras propuestas ontológicas de alto nivel: OpenCYC (que deriva de 'En-cyc-lopedia'), Mephisto, SUMO (Suggested Upper Merged Ontology), y JC3IEDM (Joint C3 Information Exchange Data Model), con el objetivo de identificar el medio más adecuado para dar una representación significativa, completa, coherente y en soporte informático a una información que procede de fuentes diversas y heterogéneas. El resultado de este análisis se ha sintetizado en la Tabla 3.1, en la que se muestran las alternativas ontológicas adoptadas por cada una de las propuestas consideradas. OpenCYC, la parte libre de CYC, es una base de conocimiento que está dividida en microteorías (como 'movimiento', 'cosas intangibles', 'legislación', etc.), cada una de las cuales contiene conceptos y hechos relativos a un área de interés particular, con cientos de miles de términos y millones de aseveraciones que relacionan los términos. La ontología Mephisto está diseñada para representar aspectos de los dominios militares y de seguridad nacional, destacando el rigor filosófico y el énfasis hacia el desarrollo completo y coherente de teorías formales. Esto hace que sea susceptible de implementación computacional y proporciona claridad de interpretación, pero a costa de alejarse del lenguaje natural, que es más adecuado para un enfoque descriptivo y multiplicativo, en lugar del enfoque prescriptivo y reduccionista de Mephisto. SUMO fue desarrollada a partir de la fusión de diferentes ontologías con el objetivo de facilitar la interoperabilidad de datos, la búsqueda y recuperación de información, la inferencia automática y el procesamiento del lenguaje natural. Entre las ontologías que han servido de base a SUMO se encuentran la ontología de alto nivel de John Sowa y PSL, que por tanto, también pueden considerarse *foundational ontologies*. El modelo de datos JC3IEDM forma parte de la solución propuesta para resolver los problemas de interoperabilidad entre distintos 'National Command and Control Systems', motivo por el cual ha sido considerado como una ontología.

6.1. DOLCE

En esta sección se describen la ontología DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering) y otras ontologías derivadas de ésta. DOLCE se ha adoptado como referencia para las propuestas ontológicas de esta tesis, como se verá en el siguiente capítulo.

DOLCE es una ontología de base cuyo objetivo es dar soporte al diseño de ontologías de dominio. DOLCE se ha utilizado con éxito en proyectos industriales y académicos (Gangemi et al. 2005) y en distintos dominios como leyes, biomedicina y agricultura (Oberle et al. 2007). Como indica su nombre, 'Descriptive Ontology for Linguistic and Cognitive Engineering', DOLCE tiene un claro sesgo cognitivo y pretende captar las categorías ontológicas que subyacen en el lenguaje natural humano y en el sentido común. DOLCE no se centra en un estricto referencialismo metafísico relativo a la naturaleza

intrínseca del mundo, sino que más bien introduce categorías pensadas como objetos o *artifacts* de conocimiento, que en última instancia dependen de la percepción, las huellas culturales y las convenciones sociales humanas. Las categorías de DOLCE son simplemente nociones descriptivas que asisten a la construcción de conceptualizaciones explícitas previamente formadas o existentes (Masolo et al. 2003). En DOLCE se representa el mundo como es percibido por los humanos en lugar de como se ve desde la perspectiva de las teorías científicas (Bräuer 2007). Las categorías de DOLCE no tienen una robustez especial en relación al estado del arte del conocimiento científico, sino que más bien describen las entidades, reflejando más o menos las estructuras superficiales del lenguaje y del conocimiento (Masolo et al. 2003). En DOLCE, entidades diferentes pueden estar co-localizadas en el mismo espacio-tiempo.

DOLCE es una ontología de *particulars* en el sentido de que su dominio de discurso está restringido a *particulars*, o entidades que no pueden tener instancias. No obstante, los *universals*, o entidades que pueden tener instancias, aparecen en una ontología de particulars como DOLCE, en la medida que se utilizan para organizar y caracterizar dicha ontología. En otras palabras, al diseñar esta ontología se establecen las clases de su taxonomía y sus predicados (los predicados son las propiedades de las clases y las relaciones entre ellas), que son *universals*, mientras que la utilización de la ontología se basa en las instancias de estas clases y predicados, que son *particulars* (Masolo et al. 2003).

La clasificación del mundo en categorías ontológicas realizada por DOLCE es muy extensa. Las cuatro categorías de alto nivel de DOLCE son: *endurant*, *perdurant*, *quality* y *abstract*. (Figura 3.5) Los *endurants* son *particulars* en el espacio, que participan al menos en un *perdurant*. Los *endurants* se clasifican en *physical* versus *non-physical*, y en *agentive* versus *non-agentive*. Los *perdurants* son *particulars* en el tiempo, que tienen al menos un participante. Los *qualities* son *particulars* inherentes a otros *endurants* o *perdurants*. Los *abstracts* son entidades que no se extienden en el espacio ni en el tiempo, no tienen ninguna *quality* espacial ni temporal, y no son *qualities* (Masolo et al. 2003).

Las *qualities* pueden verse como las entidades básicas que pueden percibirse o medirse: formas, colores, tamaños, etc. En DOLCE no es posible que dos *particulars* tengan la misma *quality*. Por tanto, cuando se dice que dos entidades tienen la misma *quality*, esto significa en realidad que dichas *qualities*, que son distintas en ambas entidades, tiene la misma posición en una determinada *quality region*. Es decir, que ambas *qualities* tienen el mismo *quale*, que es una entidad abstracta que forma parte de una *quality region* (ver anexo 4).

Dentro de los *non-physicals endurants* de la taxonomía de DOLCE se encuentran los *non-physical objects*, que pueden ser s*ocial objects y mental objects* en función de que sean o no dependientes genéricamente de una comunidad de agentes. A su vez, los *social objects* se dividen en *agentive social object* (como un cargo político, una sociedad o empresa) y *non-agentive social object* (como leyes, normas, etc.).

En DOLCE se definen diversas relaciones como *parthood*, *connectedness*, *localization*, *participation*, *inherence*, *constitution* y *dependence*, cuyo interés radica en que corresponden a diferentes estratos de la realidad. Por ejemplo, según Gangemi et al. (2005) un physical *endurant* solamente puede estar constituido (*constituted)* por otros *physical endurants*; los *non-physical endurants* son siempre dependientes (*dependent)* de al menos un *physical endurant*; el tiempo de un *endurant* es el tiempo de los *perdurants* en los cuales

participa (*participates)*; el espacio de un *perdurant* es el espacio de los *endurants* que participan (*participating)* en él; el tiempo y el espacio de un *non-physical object* son el tiempo y el espacio de los *physical endurants* de los cuales depende (*depends*). En el anexo 4, dedicado a DOLCE, se describen con más detalle las categorías y las relaciones básicas de esta ontología.

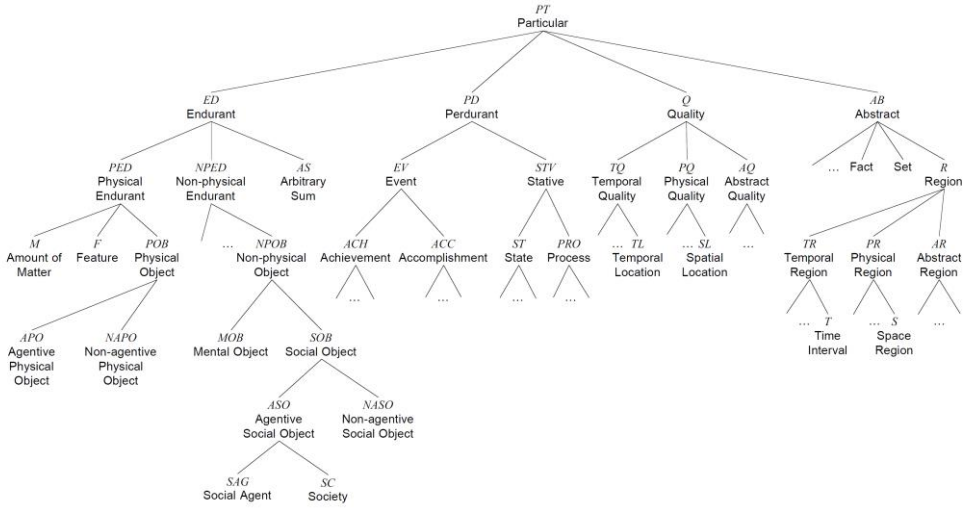


Figura 3.5. Taxonomía de categorías básicas de DOLCE (Masolo et al. 2003).

A partir de DOLCE se han desarrollado otras ontologías que comparten con ésta el propósito de dar soporte al diseño de ontologías de dominio, pero en este caso, a través de una estructuración del conocimiento y una lógica más simples, que facilitan su implementación con las herramientas (lenguajes, razonadores, etc.) disponibles. Entre ellas se encuentran: la ontología de Descripciones y Situaciones –DnS– (Rouquette et al. 2005), DOLCE+DnS Plan Ontology –DDPO– (Gangemi et al. 2005) o DOLCE+DnS Ultralite –DUL– (Scherp et al. 2009), que se comentan brevemente a continuación.

6.1.1. Ontología DnS

La necesidad de crear ontologías de dominio incluyendo los términos de otras ontologías que son de interés en algún área de la primera, ha motivado la aparición de DnS, concebida con el propósito de ayudar a extender el vocabulario de una ontología con los términos de otra (Gangemi et al. 2005). Además, como se expone a continuación, DnS satisface la necesidad de embeber el contenido y la estructura de ciertas entidades utilizando unos recursos expresivos simples. La extensión de una ontología mediante DnS se basa en el mecanismo de reificación (sinónimo de *cosificación:* reducir a la condición de cosa aquello que no lo es), que en la práctica consiste en reducir el orden de las entidades de la ontología. Por ejemplo, el resultado de la reificación de entidades de segundo orden, como clases, relaciones o tuplas serán entidades de primer orden o individuos. De este modo, la

reificación, que se puede aplicar tanto a la *intension* como a la *extension* [5] de una ontología, permite la creación de los llamados contextos embebidos o vocabulario DnS, que al ser añadidos al de otra ontología extiende el vocabulario de la misma.

Tabla 3.2. Ejemplo de entidades de una ontología que son objeto de la reificación, entidades de primer orden (individuos) resultantes de la reificación y sus DnS Types (Gangemi et al. 2005).

Non-reified	**Reified**	***DnS* Type**
Eats(x,y,z)	eats#	Description(x)
Person(x)	person#	Concept(x)
Food(y)	food#	Concept(x)
TimeInterval(z)	timeInterval#	Concept(x)
eats(john#,chocolate#,atNight#)	johnEatingChocolateAtNight#	Situation(x)
extension(Person(x))	personCollection#	Collection(x)
extension(Food(y))	foodCollection#	Collection(x)
extension(TimeInterval(z))	timeIntervalCollection#	Collection(x)
extension(Person(x)^Eats(x,y,z,))	eatingPersonCollection#	Collection(x)
extension(Food(y)^Eats(x,y,z))	eatenFoodCollection#	Collection(x)
extension(TimeInterval(z)^Eats(x,y,z))	eatingTimeIntervalCollection#	Collection(x)
john#	*n/app*	
chocolate#	*n/app*	
atNight#	*n/app*	

Como se muestra en la Tabla 3.2, los individuos resultantes del proceso de reificación deben pertenecer a un tipo determinado de la ontología DnS (DnS Types), que se corresponde con una descripción, una situación, un concepto o una colección. Esto significa que la ontología que se quiere extender con los términos de otra tendrá que incluir estos DnS Types, y en caso contrario, será necesario incorporarlos. Para intentar explicar de forma clara y concisa los dos pasos que requiere el proceso de extensión de una ontología, se propone el ejemplo consistente en embeber la ontología L en la ontología O. Esto es, extender el vocabulario de la ontología O con términos procedentes de la ontología L. Para ello, el primer paso es fusionar las ontologías O y DnS, lo que se representa como:

$$\oplus (O, DnS) \rightarrow O^{+}$$

donde O^{+} es la ontología resultante, que incluye los DnS Types, y $\oplus$ es el operador aumentar, que representa la fusión de ontologías. A continuación, se embebe la ontología L en la ontología O aumentada o fusionada con DnS (ontología O^{+}), lo que se representa como:

$$\hookleftarrow(O^{+}, L) \rightarrow O^{+L}$$

donde O^{+} es la denominada *ground ontology*, O^{+L} es la ontología resultante y $\hookleftarrow$ es el operador embeber. Como resultado de esta operación de embebido, todos los elementos reificados de la ontología L (descripciones, situaciones, conceptos y colecciones) formarán parte de DnS, y por tanto de O^{+L}.

5 La *intension* de una ontología es el conjunto de sus clases y predicados, y la *extension* es el conjunto de los individuos o instancias de dichas clases y predicados.

DnS proporciona un vocabulario y una axiomatización para escribir los nuevos individuos, interrelacionarlos, y enlazarlos a los predicados existentes de la ontología que se extiende, o *ground ontology*. Con DnS se puede describir el comportamiento de unas entidades sociales dadas, como situaciones, sin cambiar el dominio de la ontología extendida, porque los predicados cosificados y las 'tuplas' de la ontología fuente se convierten en instancias de los predicados DnS. La ontología fuente se transforma entonces en un contexto embebido en la ontología extendida (Gangemi et al. 2005).

De lo expuesto hasta ahora, se deduce que la ontología DnS permite modelar contextos mediante un enfoque razonablemente pragmático, que evita los problemas asociados a un planteamiento basado en simples mecanismos de importación. En un modelado heterogéneo abierto, estas importaciones conducirían de forma prácticamente inevitable a incompatibilidades semánticas entre conceptos similares. En la definición de estos contextos es determinante la capacidad de DnS para cosificar entidades (*reificar*), extender (*aumentar*), e incrustar (*embeber*) ontologías. La utilización de DnS es especialmente significativa, ya que facilita notablemente la formalización de la terminología de los conceptos existentes en un determinado ámbito. La incrustación define el marco lógico para asegurar el valor verdadero de las sentencias sobre una ontología extendida. Esto hace que la validez de tales sentencias dependa de la validez del propio contexto embebido, lo que motiva una descripción declarativa explícita del mismo.

DnS permite expresarse ontológicamente acerca de objetos no físicos, especialmente objetos sociales y de conocimiento. Este planteamiento se basa en que las propiedades atribuidas a entidades son también entidades, y pueden ser tratadas como objetos de conocimiento o información. DnS se basa en una distinción fundamental entre descripciones, objetos sociales que representan una conceptualización, y situaciones, vistas consistentes que satisfacen una descripcion en un conjunto de entidades.

6.1.2. Ontología DDPO

La ontología DDPO, también conocida como ontología de planes y tareas, se define a partir de la especialización de conceptos y relaciones de la ontología DOLCE y de algunas de sus extensiones para espacio y tiempo, especialmente las derivadas de DnS (Gangemi et al. 2005). Su objetivo es la descripción de tareas: los tipos de acciones, su secuencia y los controles realizados en ellas. La utilización prevista para DDPO es la especificación de planes sociales o cognitivos a nivel abstracto, con cualquier nivel de detalle, desde cualquier perspectiva, e independientemente de los recursos existentes; y no pretende caracterizar planes ejecutables computacionalmente. Las tres primeras entidades que se describen en su modelo de conocimiento son: *description*, un objeto social non-agentive; *situation,* el escenario, marco o entorno para una serie de entidades; y *time and space of a situation;* el tiempo y el espacio de las entidades en dicho escenario (Gangemi et al. 2005).

DDPO, al igual que DnS tiene un dominio muy extenso que incluye objetos físicos y objetos no físicos, *events*, *estates*, *regions* y *qualities*. Concretamente, DDPO incluye o da soporte a: tareas, que son tratadas como conceptos definidos en los planes que se refieren a las acciones; estructuras de control, como bifurcaciones y bucles, procedentes de la planificación tradicional y de los workflows, que se presentan como tareas de control; ordenación de tareas, que se formaliza utilizando *part relations*, tareas de control y la relación *successor*; una rica jerarquía de subclases en base a los tres ítems anteriores; axiomas para la definición de tipos de planes, tipos de tareas, objetivos, construcciones de

control, etc.; y axiomas para el mapeado automático de la ejecución con la descripción de los planes.

La Figura 3.6 muestra la taxonomía de tareas de DDPO en base a clases OWL, donde los nodos en línea continua representan clases definidas completamente (definidas mediante condiciones de tipo necesario y suficiente). Los nodos en línea punteada representan clases definidas parcialmente (definidas únicamente mediante condiciones de tipo necesario). Las flechas representan relaciones *subclass-of*, o relaciones *instance-of* para tareas de control.

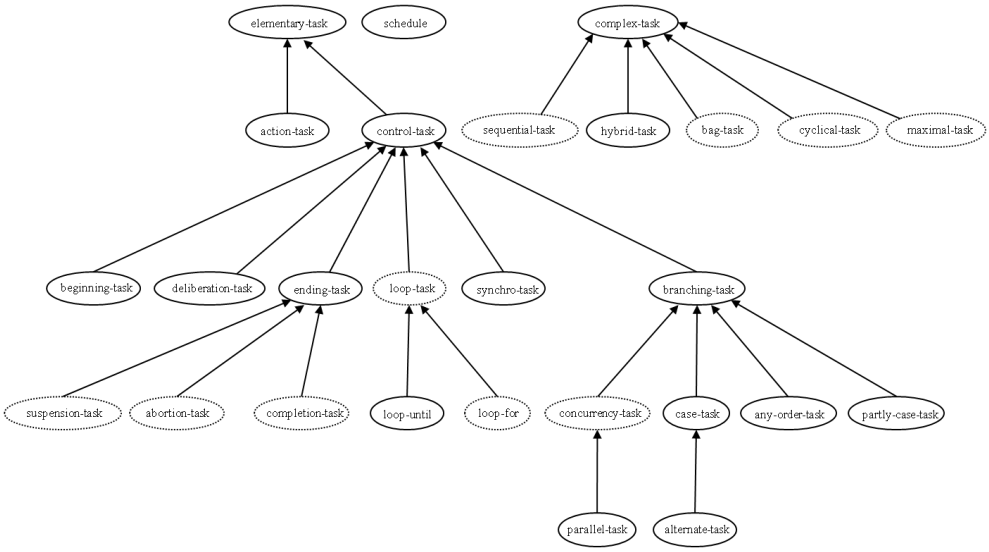


Figura 3.6. Taxonomía de tareas en DDPO (Gangemi et al. 2005).

6.1.3. Ontología DUL

La ontología DUL (DOLCE+DnS Ultralite ontology) ha sido creada como extensión de algunas partes de DOLCE y DnS, y como simplificación de otras partes de estas ontologías (Scherp et al. 2009). El propósito de DUL es proporcionar un conjunto de conceptos de alto nivel que puedan ser la base para facilitar la interoperabilidad entre distintas ontologías de medio y bajo nivel. La arquitectura de la ontología DUL es *pattern-based*, lo que significa que está disponible en varios módulos que pueden utilizarse independiente o conjuntamente (DUL 2014). El resultado final obtenido con DUL es una ontología de base ultraligera y fácil de aplicar para modelar contextos físicos o contextos sociales. La descripción de la entidad de más alto nivel de DUL revela su carácter social: una *entity* es cualquier cosa real, posible o imaginaria, acerca de la cual un modelador desea hablar para algún propósito. Sus entidades de primer nivel son: *abstract*, *informationEntity*, *quality*, *object* y *event* (Figura 3.7). Un *abstract* es cualquier entidad que no puede ser localizada en el espacio-tiempo; un *informationEntity* es un elemento de información; una *quality* es cualquier aspecto de una entidad, distinto de una de sus partes, que no puede existir sin la entidad; un *object* es cualquier sustancia, objeto físico, social o mental y un *event* es cualquier proceso físico, social o mental, evento o estado.

En relación al modelado de contextos, DUL, al igual que DOLCE, representa un compromiso pragmático entre los enfoques excesivamente simplistas como las importaciones ontológicas y los enfoques teóricamente complejos basados en contextos formales (Rouquette et al. 2005).

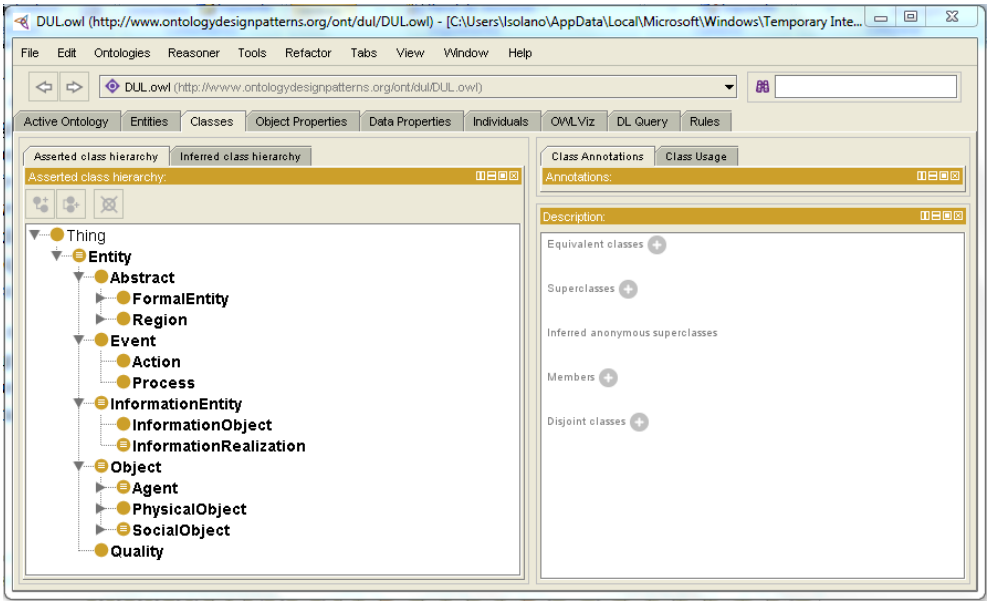


Figura 3.7. Jerarquía de clases de DUL (DUL 2014).

Con DUL se han modelado contextos físicos y contextos sociales, entre los cuales se encuentran extensiones para: objetos de información, sistemas, planes, dominios legales, y dominios léxicos y semióticos (DUL 2014). La primera de ellas, la extensión que modela los objetos de información, ha servido como ontología de base para algunas de las restantes extensiones. La ontología de planes captura la mayoría de patrones de flujos de trabajo mediante la noción de tarea introducida en esta ontología, junto a otros conceptos relevantes para el modelado y evaluación de los planes. No obstante, la representación del plan que proporciona tiene un nivel de agregación alto.

6.2. Ontología PSL

En el análisis de las categorías ontológicas de la sección 3 se ha remarcado la existencia de dos realidades claramente diferenciadas: por una parte, todo aquello que está presente en el tiempo, como los recursos; y por otra, todo aquello que se desarrolla en el tiempo, como los procesos. Los primeros no tienen partes temporales, y pueden participar en la realización de los segundos, que si pueden tener partes temporales. La planificación de procesos, como ya se ha justificado en el capítulo 2, es una función esencial en pro de la consecución de los objetivos a los que se orientan las actividades y procesos planificados. Por tanto, las tareas incluidas en la planificación de procesos deben desarrollarse rigurosa y eficientemente. Entre estas tareas, están las directamente relacionadas con las actividades: la identificación de los tipos de actividades requeridas; la descomposición y estructuración de actividades; la

secuenciación o sincronización entre ellas; la identificación y representación de alternativas, en planes no lineales asociados a la flexibilidad del proceso; etc. Por ello, se requiere una caracterización y conceptualización de las actividades, o procesos, como la que proporciona la ontología PSL.

PSL se desarrolló con el objetivo de crear una representación de proceso común para todas las aplicaciones de fabricación. No obstante, debido a que muchos de los conceptos necesarios para la representación de procesos de fabricación son comunes con otros procesos de empresa, pronto se propuso para la representación de procesos en general, y ha llegado a ocupar un lugar destacado entre las ontologías de proceso. Su metodología se basa en la identificación de intuiciones relacionadas con los procesos de fabricación, que posteriormente se trasladan a elementos de alguna estructura matemática algebraica o combinatoria, y que finalmente se formalizan mediante definiciones y axiomas escritos en una lógica de primer orden que aseguran una semántica desarrollada rigurosamente. La ontología PSL está formada por una serie de módulos interpedendientes construidos a partir de un núcleo que captura los conceptos primitivos de alto nivel inherentes a la especificación de procesos. Cada módulo refina este núcleo capturando conjuntos de conceptos propios de un área concreta relacionada con la especificación de procesos (Bock y Gruninger 2005; PSL 2010).

A continuación se incluye la descripción de la norma en la que se formaliza PSL, junto con otros aspectos generales que ayudan a la contextualización de la ontología: antecedentes, enfoque y justificación inicial, y aplicaciones.

6.2.1. Generalidades

PSL es el resultado de un proyecto iniciado en 1995 por el National Institute for Standards and Technology (NIST) que tenía como objetivo la creación de un lenguaje neutro, normalizado y de alto nivel para la especificación de procesos que permitiera integrar múltiples aplicaciones de procesos relacionados a través del ciclo de vida del producto (Lubell 2003). PSL fue concebido principalmente para la industria de fabricación, y su alcance más genuino corresponde a los procesos de fabricación discreta. La singularidad del proyecto PSL se puede valorar al compararlo con el proyecto METK (Manufacturing Engineering ToolKit), también financiado por el NIST y perteneciente al mismo ámbito, pero que aborda la representación de procesos con un enfoque basado en modelos de información, frente al enfoque de PSL basado en una semántica formal (Lee 1999). En las propuestas de modelado basadas en modelos de información, la conceptualización subyacente que acompaña al modelo no aflora como en el caso de las ontologías, y por tanto, estos modelos de información, que se sustentan fundamentalmente en la sintaxis, carecen de una especificación adecuada de la semántica relativa a la terminología de procesos. Esta situación conduce a una interpretación y utilización de la información inconsistentes, y a problemas de interoperabilidad, especialmente cuando los sistemas que soportan las funciones de las empresas han sido creados independientemente y no comparten la misma semántica para la terminología de sus modelos.

Concretamente, la necesidad de una semántica explícita se hace patente en el marco de modelado de ISO 10303 (STEP). Inicialmente muy centrado en aspectos de diseño, pero que se extiende hacia distintas áreas donde entran en consideración los problemas en el enlace CAD/CAPP/CAM/CNC, y donde se requiere integrar distintas aplicaciones relacionadas con la fabricación que están involucradas en el ciclo de vida del producto, lo

que implica disponer de la relación semántica entre los términos de estas aplicaciones de fabricación. Este es el caso de la norma ISO 14649 (STEP-NC), que sin tener un enfoque ontológico, incluye la descripción de procesos. En contraposición a esto, PSL desarrolla una capa semántica formal, una ontología, que permite una definición clara y explícita de los conceptos intrínsecos a los procesos de fabricación (Deshayes, El Beqqali y Bouras 2005). En otras palabras, PSL define las entidades y relaciones que se necesitan para la representación de procesos (Cochrane 2004). Haciendo referencia a los enfoques ontológicos presentados en el capítulo anterior, puede afirmarse que PSL tiene un enfoque prescriptivo, adoptando una noción más estricta de la caracterización que la correspondiente al enfoque descriptivo propio de lenguajes de modelado de procesos como IDEF3. De este modo, con PSL puede establecerse que únicamente los objetos que realmente existen o que representan tipos o categorías de objetos del mundo real puedan estar representados en el contenido del modelo. Debido a esta precisión, PSL puede capturar sin ambigüedad el significado de los lenguajes que se destinan a la definición de procesos.

Como se ha indicado anteriormente, el enfoque inicial de PSL responde a la necesidad de un lenguaje neutro de intercambio para garantizar la interoperabilidad entre aplicaciones. La clave de PSL radica en las definiciones formales que subyacen en el lenguaje. Gracias a esas definiciones explícitas y no ambiguas, propias de un lenguaje ontológico, el intercambio de información puede realizarse sin estar basado en asunciones ocultas o conversiones subjetivas (Lubell 2003). En cualquier caso, si se considera PSL como un lenguaje, hay que destacar que éste se refiere directamente a las realizaciones de las actividades en tiempo de ejecución. En otras palabras, PSL permite referirse individualmente a cada una de las realizaciones de una actividad, estableciendo restricciones según la evolución temporal que acompaña a estas sucesivas ejecuciones de la actividad (PSL 2010).

Respecto a su consideración como lenguaje de intercambio, hay que destacar que aunque PSL tiene una semántica definida formalmente, no especifica una única sintaxis. PSL ofrece la posibilidad de múltiples sintaxis, dependiendo la elección de factores como la naturaleza de los procesos descritos, y el origen y destino de los datos. Una de estas posibilidades es KIF, que permite una definición muy precisa de los términos que forman el léxico de PSL, una característica necesaria para el intercambio de información de procesos (Lubell 2003). La Figura 3.8 ilustra el mecanismo de traducción de información, basado en la semántica de PSL y la sintaxis de KIF. En el ejemplo se parte de dos aplicaciones diferentes (A y B) y se quiere disponer en B de la información capturada inicialmente en A. Cada una de estas aplicaciones tiene una sintaxis con la terminología propia de su dominio de aplicación. Además, para cada aplicación existe una ontología que explicita y restringe la semántica de los términos, o al menos una conceptualización implícita que incluye el significado de dichos términos. El primer paso del proceso de traducción consiste en determinar si ésta es factible. Para ello, se debe comprobar si los elementos de la aplicación A que constituyen la información a traducir tienen una semántica común con otros elementos de la aplicación B y si las semánticas de ambos conjuntos de elementos están cubiertas por la ontología de PSL (intersección de las zonas rectangulares de la Figura 3.8). A continuación, como se aprecia en la figura, esta traducción se inicia con la traducción sintáctica, entre la aplicación A y KIF, que consiste en expresar los términos del dominio de aplicación de A que son objeto de la traducción según la sintaxis de KIF. Después, se efectúa la traducción semántica entre la aplicación A y PSL, con la cual se obtiene el

contenido de la información a traducir procedente de la aplicación A con la terminología y sintaxis propias de KIF, que expresan la semántica de PSL. El proceso de traducción se completa con otros dos pasos de traducción análogos a los anteriores: la traducción semántica de PSL a la aplicación B, y la traducción sintáctica entre KIF y la aplicación B.

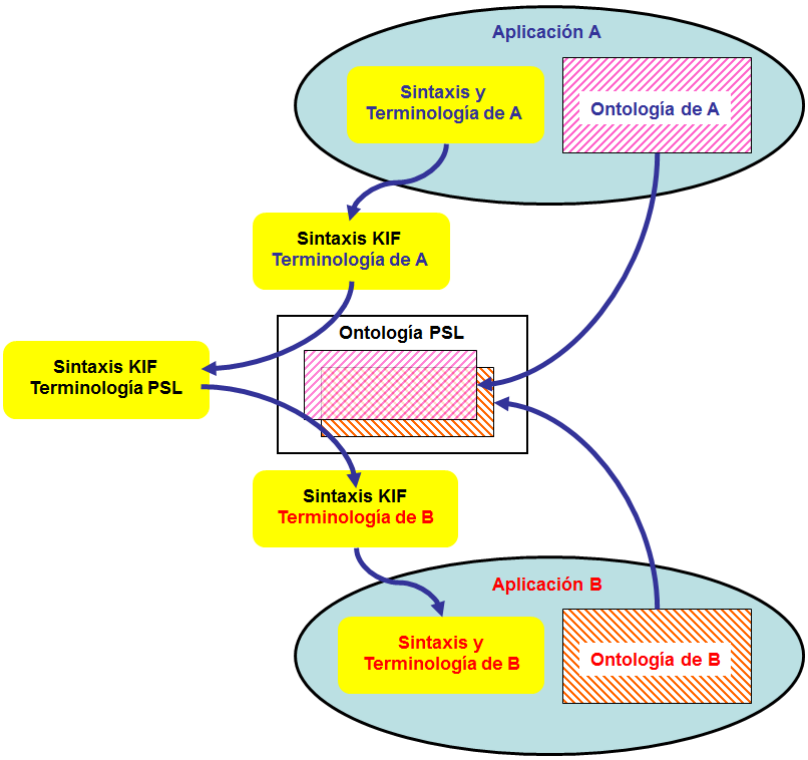


Figura 3.8. Utilización de la sintaxis KIF en el proceso de intercambio de información de procesos mediante la ontología PSL.

PSL se ha convertido en el punto de referencia de las investigaciones relacionadas con las ontologías sobre el conocimiento relativo a procesos (Cochrane 2004) y es probablemente la ontología más desarrollada aplicable a procesos de fabricación (Cochrane et al. 2004). El soporte semántico que proporciona PSL mejora el funcionamiento de la empresa distribuida y la e-colaboración, ya que es una tecnología de intercambio robusta que facilita la colaboración distribuida entre aplicaciones de fabricación (Pouchard et al. 2000). Debido a la capacidad de auto-enriquecimiento del lenguaje PSL a través de su ontología, éste puede ser considerado como una poderosa herramienta para la interoperabilidad (Pouchard et al. 2004). Otras aplicaciones de PSL están relacionadas con: la reutilización de los conocimientos de fabricación durante el diseño (Cochrane et al. 2004); el modelado de procesos de empresa (Chen-Burger, Tate y Robertson 2002); o la integración de empresas mediante sistemas basados en agentes. PSL cubre todos los procesos del ciclo de vida de diseño y fabricación, y mediante extensiones de su ontología puede ser empleado en ámbitos diversos.

6.2.2. La norma ISO 18629

En la norma ISO 18629 (ISO 2004) se recoge tanto la definición del lenguaje, como la especificación de la ontología establecida con dicho lenguaje. El mapa conceptual de la Figura 3.9 proporciona una visión de conjunto de la norma ISO 18629 y de los principios básicos de PSL.

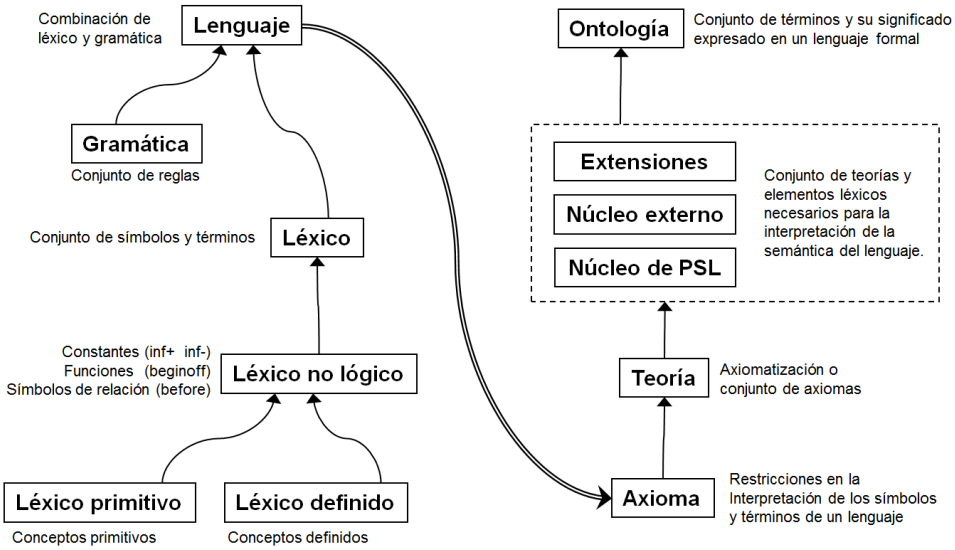


Figura 3.9. Mapa conceptual de la norma ISO 18629 elaborado a partir de (ISO 2004).

El léxico empleado en PSL es un conjunto de símbolos y términos. Entre los símbolos lógicos están los conectores, los cuantificadores y los símbolos *booleanos*. La parte no lógica del léxico consta de expresiones, como constantes, símbolos de función y símbolos de relación, que denotan conceptos propios del dominio de la ontología. Según que estos conceptos sean primitivos o no, el léxico que sirve para identificarlos se denomina respectivamente léxico primitivo o léxico definido. La forma en la que los símbolos lógicos y los términos léxicos pueden combinarse para dar lugar a fórmulas bien construidas queda especificada por un conjunto de reglas o gramática[6]. De modo, que el lenguaje que se define en la norma ISO 18629 es la combinación de un léxico, o conjunto de símbolos y términos, y su gramática, mientras que la ontología PSL establecida con ese lenguaje en ISO 18629 es un conjunto de términos y la especificación de su significado expresada en un lenguaje formal. Las restricciones a la interpretación de los símbolos y términos de la

6 La definición que aparece en la norma ISO 18629 para la gramática ('*specification of how logical symbols and lexical terms can be combined to make well-formed formulae*') se aproxima a la definición de sintaxis de la RAE ('una parte de la gramática que enseña a coordinar y unir las palabras para formar oraciones y expresar conceptos').

Las siguientes definiciones también proceden del diccionario de la RAE: gramática (ciencia que estudia los elementos de una lengua y sus combinaciones); semántica (perteneciente o relativo a la significación de las palabras); léxico (vocabulario, conjunto de palabras de un idioma, o de las que pertenecen al uso de una region, a una actividad determinada, a un campo semántico dado, etc); y terminología (conjunto de términos o vocablos propios de determinada profesión, ciencia o materia).

ontología se denominan axiomas, que son fórmulas correctamente construidas en un lenguaje formal. A su vez, los axiomas se reúnen formando teorías. Así, el concepto de axiomatización puede identificarse como el conjunto de axiomas de una teoría (Figura 3.9).

La ontología PSL está organizada de forma modular, con una serie de extensiones que mantienen dependencias lógicas. Entendiendo que una extensión depende lógicamente de otra cuando alguno de los términos de la primera requiere para su definición de otro término de la segunda. La ontología PSL consta de términos primitivos, definiciones y axiomas para los conceptos de PSL. Los componentes principales de su arquitectura semántica son: *PSL-Core*, el elemento más básico de la ontología; *Outer Core*, un conjunto de extensiones del núcleo de PSL de amplia aplicación; y *Extensions*, otras extensiones que capturan la semántica de la terminología de procesos para diversas aplicaciones (Figura 3.11).

6.2.3. El núcleo de PSL

El propósito del núcleo de PSL (PSL-Core) es transformar en axiomas un conjunto de primitivas semánticas intuitivas que es adecuado para describir procesos básicos. Estas primitivas semánticas intuitivas son: (a) hay cuatro tipos de entidades requeridas para el razonamiento sobre procesos: *activities*, *activity occurrences*, *timepoints* y *objects*; (b) las actividades pueden tener múltiples realizaciones y puede darse el caso de actividades que nunca ocurran; (c) los puntos en el tiempo están ordenados linealmente, adelante hacia el futuro y a la inversa hacia el pasado; y (d) las realizaciones y los objetos están asociados con puntos en el tiempo únicos, que marcan el inicio y el final de la realización o del objeto.

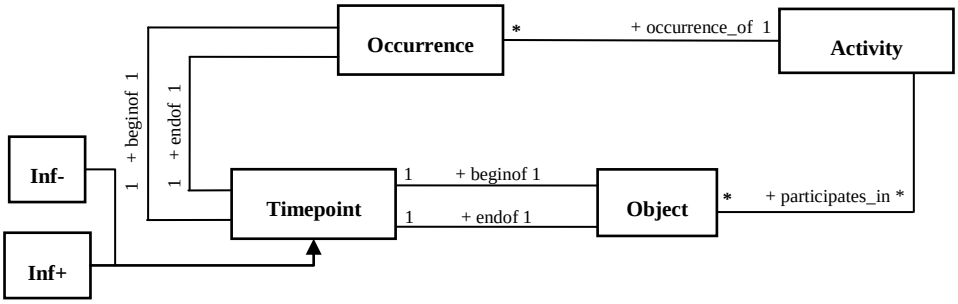


Figura 3.10. Las cuatro entidades básicas del núcleo de PSL y algunas de las funciones y relaciones que se establecen entre ellas (Solano et al. 2009).

El núcleo de PSL es coherente y completo. Es decir, todos sus axiomas son ciertos para el conjunto de la semántica formal de la ontología y cualquier sentencia del lenguaje puede derivarse de dichos axiomas. Para que dos aplicaciones cualesquiera relativas a procesos puedan utilizar PSL (ser *PSL-compliant*) deben compartir al menos el núcleo de PSL, que es relativamente simple, ya que está formado por cuatro clases primitivas: *activity*, *activity_occurrence* (*occurrence*), *timepoint* y *object* (Figura 3.10), cuyos nombres están incluidos entre los símbolos no lógicos del lenguaje. Además, el núcleo de PSL consta de: dos símbolos de funciones (*beginof* y *endof*); dos símbolos de constantes (*inf+* e *inf-*); y

varias relaciones. Algunas de estas relaciones forman parte del léxico primitivo, como *before* o *participates_in*, mientras que otras como *between*, que se define a partir de la relación *before*, forman parte del léxico definido. A pesar de la simplicidad de PSL-Core, el conjunto de sus axiomas escritos en el lenguaje formal de PSL proporciona una semántica primitiva que resulta suficiente para describir los procesos básicos.

6.2.4. Las extensiones del núcleo de PSL

En este apartado se presenta una visión general de las extensiones del núcleo de PSL: *Outer Core* y *Extensions*. Cada una de estas extensiones incluye, entre otros: el léxico no lógico, un conjunto de axiomas, y una gramática para la terminología del léxico no lógico. El *Outer Core* reúne un conjunto extensiones del núcleo de PSL, que son tan genéricas y extendidas en su aplicación que han sido ubicadas aparte. Las extensiones del Outer Core se pueden identificar en la Figura 3.11. Como puede apreciarse en dicha figura, el Outer Core está formado por tres *non-definitional extensions* (las que incluyen al menos una noción que no puede ser definida en términos del núcleo de PSL) y por seis *definitional extensions* (aquellas cuyos elementos lingüísticos pueden ser definidos completamente con términos del núcleo de PSL). La diferenciación entre *definitional* y *non-definitional* se deduce a partir de la posición de las flechas curvadas en la figura, que representan las dependencias lógicas de cada grupo de teorías. De este modo, únicamente PSL-Core y la parte *non-definitional* del Outer Core muestran dependencias lógicas que no proceden de PSL (intuiciones). Mientras que las restantes dependencias lógicas, también representadas por flechas, se establecen entre partes de la ontología PSL.

En el Outer Core, la extensión *Activity-Occurrence* define relaciones que permiten la descripción de cómo las realizaciones de las actividades (*activity_occurrences*) se relacionan entre sí con respecto a los instantes de tiempo en los que empiezan y terminan. En la extensión *Atomic Activity* se introduce la realización de actividades concurrentes. *Complex Activities* especifica la relación entre las realizaciones de las subactividades de una actividad y las realizaciones de la propia actividad. La extensión *Occurrence Tree* introduce una estructura de árbol sobre el conjunto de las posibles realizaciones de actividades; las ramas del árbol corresponden a secuencias diferentes de las realizaciones de actividades primitivas. Esta extensión también introduce la noción de estado y, específicamente, las precondiciones y los efectos de las actividades. La extensión *State* (*Discrete States*) especifica los conceptos básicos de los estados, que cambian como consecuencia de la realización de las actividades, y sus relaciones con estas realizaciones de las actividades. En la extensión *Subactivity* se describe cómo las actividades pueden ser agregadas y descompuestas. En ella también se define el concepto de actividad primitiva: una actividad que no puede ser descompuesta en otras.

Todas las anteriores son *definitional extensions*, y junto con PSL-Core, proporcionan la mayor parte de la infraestructura para especificar las definiciones de la terminología en la norma ISO 18629.

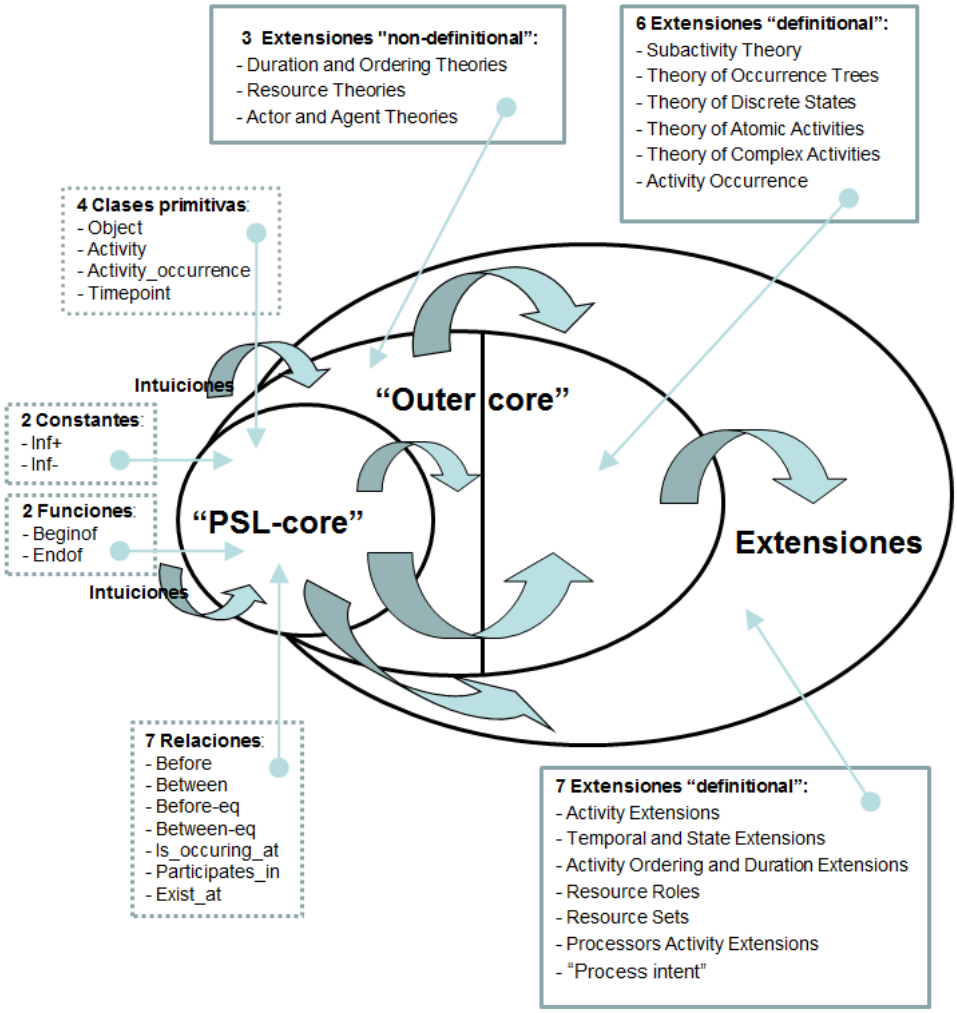


Figura 3.11. Estructuración y contenido de las teorías de PSL.

Entre las *non-definitional extensions* del Outer Core destacan *Duration and Ordering Theories* (Figura 3.11), que incluyen las extensiones *Duration* y *Subactivity Occurrence Ordering*. La primera introduce el concepto de duración como una relación entre instantes de tiempo (*timepoints*). Esto permite la utilización de conceptos cuantitativos relacionados con el tiempo, y también proporciona las bases para la definición de la duración de las actividades, las realizaciones de las actividades y los objetos. Por su parte, la extensión *Subactivity Occurrence Ordering* especifica las relaciones requeridas para representar varios tipos de conjuntos de actividades parcialmente ordenadas, incluyendo secuencias, actividades en paralelo (actividades que se realizan simultáneamente), y nodos AND/OR (*AND splits/junctions* y *OR splits/junctions*).

En los siete grupos de extensiones que conforman las denominadas *Extensions* de PSL ('Extensiones' en la Figura 3.11) se definen entre otros: los tipos de actividades que contienen nodos de separación (*splits*) y nodos de unión (*joints*); las actividades deterministas y no deterministas[7]; los tipos de actividades que tienen la propiedad de ser o no interrumpibles; la realización condicional de las subactividades de una actividad; las relaciones entre la realización de actividades y los intervalos de tiempo en que éstas se realizan, y su utilización prevista para aplicaciones de programación (*scheduling*); los roles de los recursos, como reusable, consumible, y renovable; y la posible incorporación de nuevas extensiones a la ontología.

7. Ontologías en el dominio de la fabricación distribuida

Un modelo de empresa es una representación computacional de la estructura, actividades, procesos, información, recursos, personas, comportamiento, objetivos y restricciones de una empresa (Fox y Gruninger 1998). Entre las llamadas ontologías de empresa, que se centran en la descripción o conceptualización de los modelos de empresa, se encuentra Enterprise Ontology (Uschold et al. 1998). Ésta proporciona una recopilación de términos y definiciones para el modelado de empresas que permiten reducir problemas semánticos como los derivados de manejar varios conceptos con un mismo nombre, un concepto con varios nombres diferentes o dos conceptos similares pero no idénticos. El objetivo del proyecto ALPS (Catron y Ray 1991) fue la creación de un lenguaje de especificación de procesos, que permita la identificación de los modelos de información para facilitar la especificación de procesos y transferir esta información a sus sistemas de control (Deshayes, El Beqqali y Bouras 2005). Los proyectos CPR (Pease y Carrico 1997) y SPAR (Tate 1998) se centraron en el desarrollo de un modelo de soporte para la representación de necesidades de varios sistemas de planificación económica militar, mientras que el ámbito de Enterprise Ontology y TOVE (Toronto Virtual Enterprise) son los procesos de empresa. Estos cuatro últimos proyectos presentan un conjunto de conceptos genéricos comunes, y propios de los procesos de fabricación (Deshayes, El Beqqali y Bouras 2005). Entre ellos destaca TOVE, al que se dedica la siguiente subsección.

7.1. La ontología TOVE

TOVE es una de las primeras propuestas ontológicas relacionadas con la ingeniería de fabricación y ha servido de base para la ontología PSL. De hecho, la concepción general y la estructura de PSL se encuentran representadas en buena medida en su antecesor TOVE. Como su nombre indica, la iniciativa TOVE se ha desarrollado en el marco de la empresa virtual o distribuida, estando sus trabajos iniciales centrados en el desarrollo de ontologías para soportar el razonamiento en entornos afines a la gestión de la cadena de suministro.

El propósito de TOVE se articula en cuatro objetivos generales: (a) crear una representación compartida (ontología) de la empresa, que cada agente de la empresa distribuida pueda entender y usar; (b) definir el significado de cada descripción (semántica); (c) implementar la semántica mediante un conjunto de axiomas que permitan a

7 Una actividad es determinista si todas sus subactividades ocurren al realizarse la actividad, aunque éstas pueden realizarse en distinto orden; mientras que una actividad es no determinista, si es posible que no ocurran todas sus subactividades cuando se realiza la actividad

TOVE deducir automáticamente la respuesta a muchas preguntas de sentido común acerca de la empresa; y (d) definir una simbología para describir un concepto en un contexto gráfico (Fox 1992).

Para TOVE, son claves los conceptos de actividad y de recurso, como lo demuestra el hecho de que sus dos ontologías de base (*foundational ontologies*) son: *Activity Ontology* y *Resource Ontology* (Figura 3.12). En ellas, la mayor parte del esfuerzo se ha dedicado a crear representaciones del comportamiento de la organización (actividad, estado, causalidad y tiempo) y de los objetos que intervienen en las actividades (recursos, inventario, órdenes y productos). Además de estas dos, TOVE incluye varias ontologías (*business ontologies*) que se centran en otros aspectos relacionados con la organización, como los productos y sus requerimientos, la calidad y los costes.

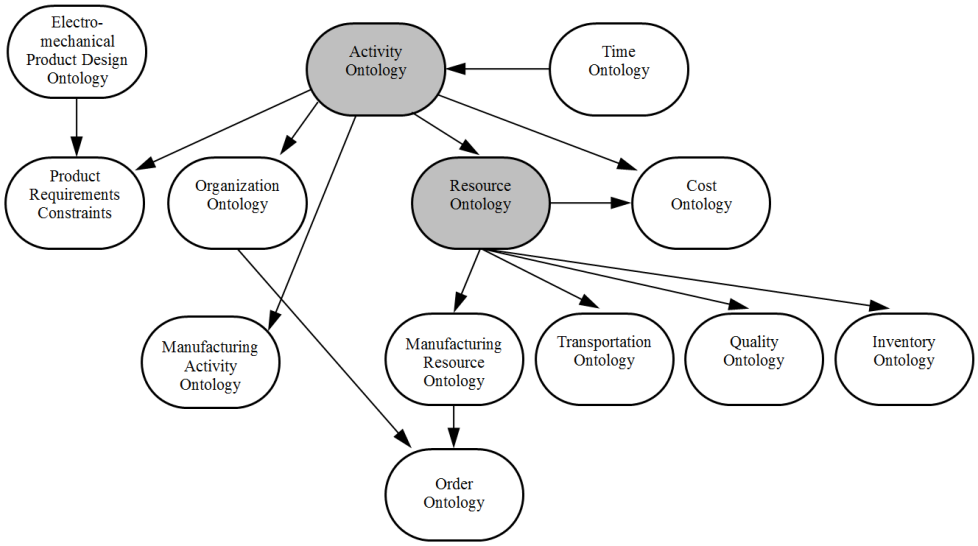


Figura 3.12. Ontologías del Modelo de Empresa Deductivo TOVE (Fox y Gruninger 1998).

TOVE propugna un cambio de orientación desde la inteligencia artificial, caracterizada por el manejo de información y la utilización de reglas, al enfoque ontológico, caracterizado por la gestión del conocimiento. Es lo que en TOVE (2014) se ha denominado 'segunda generación de ingeniería del conocimiento'. En línea con el primero de los objetivos enunciados anteriormente, el enfoque de TOVE vincula la estructura y el comportamiento de modo que los agentes de la organización puedan llevar a cabo acciones que provoquen cambios en la misma. Este vínculo, en el que se asocian los conceptos de autoridad y responsabilidad, es crucial para la unificación de los modelos de empresa y su ejecutabilidad. En este contexto, las demandas de los sistemas de información se han ido incrementando en la medida en que éstos desempeñan un papel más activo en la gestión y en las operaciones de las empresas. Desde su rol tradicional como simples repositorios de datos, se apunta a que los sistemas de información proporcionen una ayuda más sofisticada para la toma de decisiones manual y automatizada. Estos sistemas de información no sólo deben ser capaces de contestar a preguntas referidas a lo que está explícitamente representado en su modelo de empresa (*factual queries*), sino que deben dar respuesta a

preguntas relativas a lo que está implícito en el modelo (*common sense queries*). Éste es otro de los objetivos del proyecto TOVE: la creación de un 'Modelo de Empresa de Sentido Común', entendido como un modelo que tenga la habilidad de deducir las respuestas a preguntas que requieren un conocimiento relativamente superficial del dominio. Esto es, un modelo en el que el sistema de información no posee el conocimiento ni la capacidad de razonamiento necesarios para dar respuesta a las preguntas de mayor complejidad, o *expert queries*.

La metodología propuesta por TOVE para una ontología (ver anexo 5) comienza con la definición de los requisitos de la ontología, que se presentan como las preguntas que una ontología debe ser capaz de responder, denominadas preguntas de competencia. El segundo paso es definir la terminología de la ontología: sus objetos, atributos y relaciones. Finalmente, el tercer paso consiste en especificar, cuando sea posible, las definiciones y restricciones de la terminología mediante axiomas basados en una lógica de primer orden.

7.2. Modelos ontológicos del e-manufacturing.

Para finalizar esta sección dedicada a las ontologías en el dominio de la fabricación distribuida, se hará una breve reseña a los desarrollos ontológicos en los dominios de la *e-manufacturing* y de las cadenas de suministro colaborativas cuyo objetivo se centra en la localización eficiente de recursos y servicios de fabricación. La propuesta de Jang et al. (2008) utiliza OWL para describir la semántica de los servicios de fabricación mediante la definición de perfiles de capacidades de los recursos, y emplea razonadores basados en lógica DL en el proceso de descubrimiento de los servicios que responden a las características demandadas. Ameri y Dutta (2008) introducen un algoritmo de búsqueda (*matchmaking algorithm*) para conectar usuarios y proveedores de servicios de fabricación basado en las similitudes semánticas de dichos servicios. Estas similitudes semánticas se representan formalmente en términos de capacidades de fabricación mediante la ontología MSDL (Manufacturing Service Description Language). El sistema denominado ManuHub (Cai et al. 2010) administra los servicios de fabricación distribuidos que proporcionan distintas empresas virtuales, facilitando la recuperación de los servicios de fabricación requeridos en base a las similitudes semánticas de sus capacidades.

8. Referencias

Ameri, F. y D. Dutta. 2008. "A Matchmaking Methodology for Supply Chain Deployment in Distributed Manufacturing Environments." *Journal of Computing and Information Science in Engineering* 8 (1): 1–9.

ARC Advisory Group. 2002. "Collaborative Manufacturing Management Strategies." *ARC Advisory Group*, November.

Bräuer, M. 2007. "Design of a Semantic Connector Model for Composition of Metamodels in the Context of Software Variability." PhD diss., Technische Universität Dresden.

Bock, C. y M. Gruninger. 2005. "PSL: A Semantic Domain for Flow Models." *Journal of Software and Systems Modeling* 4 (2): 209–231. doi: 10.1007/s10270-004-0066-x.

Cai, M., W. Y. Zhang, G. Chen, K. Zhang y S. T. Li. 2010. "SWMRD: a Semantic Web-based Manufacturing Resource Discovery System for Cross-Enterprise Collaboration."

International Journal of Production Research 48 (12): 3445–3460. doi: 10.1080/00207540902814330.

Calì, A., D. Calvanese, B. Cuenca Grau, G. De Giacomo, D. Lembo, M. Lenzerini, C. Lutz, D. Milano, R. Möller, A. Poggi y U. Sattler. 2005. *State of the art survey Deliverable D01. TONES EU-IST STREP FP6-7603.*

Catron, B. y S. Ray. 1991. "ALPS - A Language for Process Specification." *International Journal of Computer Integrated Manufacturing* 4 (2): 105–113.

Chandrasekaran, B., J. R. Josephson y V. R. Benjamins. 1999. "What Are Ontologies, and Why Do We Need Them?" *IEEE Intelligent Systems* 14 (1): 20–26.

Chen-Burger, Y. H., A. Tate y D. Robertson. 2002. "Enterprise Modelling: A Declarative Approach for FBPML." Paper presented at the European Conference of Artificial Intelligence, Knowledge and Organisational Memories Workshop, Lyon, July 22–26.

Chung, P. W. H., J. Stader, P. Jarvis, J. Moore y A. Macintosh. 2003. "Knowledge-based Process Management an Approach to Handling Adaptive Workflow." *Knowledge-Based Systems* 16 (3): 149–160.

Cochrane, S. 2004. "Knowledge Sharing Between Design and Manufacture." *PhD proposal*, Loughborough University.

Cochrane, S. D., R. I. M. Young, K. Case, J. A. Harding, S. Dani, J. Gao y D. Baxter. 2004. "Manufacturing Knowledge Reuse in Design. Submission for the International Journal of Information Technology and Management." *Special Issue on New_wave Strategic Enterprise Technology*.

Corcho, O., M. Fernández-López, A. Gómez-Pérez y O. Vicente. 2002. "WebODE: An Integrated Workbench for Ontology Representation, Reasoning and Exchange." Proceedings of the 13th International Conference Knowledge Engineering and Knowledge Management: 138–153.

Devedzic, V. 2001. "Knowledge Modeling - State of the Art." *Integrated Computer-Aided Engineering* 8 (3): 257–281.

Deshayes, L. M., O. El Beqqali y A. Bouras. 2005. "The Use of Process Specification Language for Cutting Processes." *International Journal of Product Development* 2 (3): 236–253.

DUL Ontology. 2014. Consultada en Julio.

http://www.loa-cnr.it/ontologies/DUL.owl

Kozaki, K., Y. Kitamura, M. Ikeda y R. Mizoguchi. 2002. "Hozo: An Environment for Building/Using Ontologies Based on a Fundamental Consideration of *Role* and *Relationship*." Proceedings of the 13th International Conference Knowledge Engineering and Knowledge Management: 213–218.

Fox, M. S. 1992. "The TOVE Project: A Common-sense Model of the Enterprise". *Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*. Belli, F. and F. J. Radermacher (Eds.). Lecture Notes in Artificial Intelligence # 604, Berlin, Springer-Verlag, 25–34.

Fox, M. S. y M. Gruninger. 1998. “Enterprise modeling.” *AI Magazine* 19 (3): 109–121.

Gangemi, A., S. Borgo, C. Catenacci y J. Lehmann. 2005. *Task Taxonomies for Knowledge Content. Deliverable D07 of the EU FP6 project Metokis*. Laboratory for Applied Ontology ISTC CNR.

Guarino, N. 1995. “Formal Ontology, Conceptual Analysis and Knowledge Representation.” *International Journal of Human-Computer Studies* 43: 625–640.

Guarino, N. 1997. “Understanding, Building and Using Ontologies.” *International Journal of Human-Computer Studies* 46 (2–3): 293–301.

Gruber, T. R. 1993. “A Translation Approach to Portable Ontologies.” *Knowledge Acquisition* 5 (2): 199–220.

Gruninger, M., O. Bodenreider, F. Olken, L. Obrst y P. Yim. 2008. “Ontology Summit 2007 – Ontology, Taxonomy, Folksonomy: Understanding the Distinctions.” *Applied Ontology* 3 (2008) 191–200.

Horridge, M., H. Knublauch, A. Rector, R. Stevens y C. Wroe. 2004. “A Practical Guide to Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools.” *University of Manchester.*

ISO (International Organization for Standardization). 2004. *Industrial Automation System and Integration – Process Specification Language. Part 1, Overview and Basic Principles.* ISO 18629–1, New York: American National Standards Institute.

Jang, J., B. Jeong, B. Kulvatunyou, J. Chang y H. Cho. 2008. “Discovering and Integrating Distributed Manufacturing Services with Semantic Manufacturing Capability Profiles.” *International Journal of Computer Integrated Manufacturing* 21 (6): 631–646. doi: 10.1080/09511920701350920.

Lambert, D., A. Saulwick, C. Nowak, M. Oxenhan y D. O'Dea. 2008. *An overview of conceptual frameworks.* DSTO-TR-2163. Command, Control, Communications and Intelligence Division, Defence Science and Technology Organisation, Department of Defence, Australian Government.

Lee, Y. T. 1999. “Initial Manufacturing Exchange Specification (IMES): Information Model for the Process Plan – Workstation Level.” *US Department of Commerce, Technology Administration, National Institute of Standards and Technology*.

Lin, Y. 2008. “Semantic Annotation for Process Models: Facilitating Process Knowledge Management via Semantic Interoperability.” PhD diss., Norwegian University of Science and Technology.

Lin, H. K., J. A. Harding y M. Shahbaz. 2004. “Manufacturing System Engineering Ontology for Semantic Interoperability across Extended Project Teams.” *International Journal of Production Research* 42 (24): 5099–5118. doi: 10.1080/00207540412331281999.

Löckelt, M. 2008. “A Flexible and Reusable Framework for Dialogue and Action Management in Multi-Party Discourse.” PhD diss., Universität des Saarlandes.

Lubell, J. 2003. “XML Representation of Process Descriptions.” In *Process Descriptions of Professional XML Meta Data*. Wrox Press.

Masolo, C., S. Borgo, A. Gangemi, N. Guarino y A. Oltramari. 2003. *WonderWeb Deliverable D18. Ontology Library (final)*. Laboratory for Applied Ontology ISTC CNR.

Mili, H., G. Tremblay y G. B. Jaoude. 2010. “Business Process Modeling Languages: Sorting Through the Alphabet Soup.” *ACM Computing Surveys* 43 (1): 4. doi: 10.1145/1824795.1824799.

Mizoguchi, R. 2004. “Tutorial on Ontological Engineering Part 2: Ontology Development, Tools and Languages.” *New Generation Computing* 22 (1): 61–96.

Mostafei, S., A. Bouras y M. Batouche. 2005. “Effective Collaboration in Product Development via a Common Sharable Ontology.” *International Journal of Computational Intelligence* 2 (4): 206–212.

Musen, M. A., R. W. Fergerson, W. E. Grosso, M. Crubezy, H. Eriksson, N. F. Noy y S. W. Tu. 2003. “The Evolution of Protégé: An Environment for Knowledge-Based Systems Development.” *International Journal of Human-Computer Interaction* 58 (1): 89–123.

Nadarajan, G. y J. Chen-Burger. 2006. “Informatics Research Proposal: Mapping Fundamental Business Process Modelling Language to a Semantic Web Based Language.” Paper presented at the 4th Hellenic Conference on Artificial Intelligence (SETN 06), Crete, March 18.

Oberle, D., A. Ankolekar, P. Hitzler, P. Cimiano, M. Sintek, M. Kiesel, B. Mougouie, S. Baumann, S. Vembu, M. Romanelli, P. Buitelaar, R. Engel, D. Sonntag, N. Reithinger, B. Loos, H.-P. Zorn, V. Micelli, R. Porzel, C. Schmidt, M. Weiten, F. Burkhardt y J. Zhou. 2007. “DOLCE ergo SUMO: On Foundational and Domain Models in the SmartWeb Integrated Ontology (SWIntO).” *Web Semantics: Science, Services and Agents on the World Wide Web* 5 (3):156–174.

Pease, R. A. y T. Carrico. 1997. “The JTF ATD Core Plan Representation: A Progress Report.” Proceedings of the AAAI Spring Symposium on Ontological Engineering: 95–100.

Pouchard, L., A. Cutting-Decelle, J. J. Michel, R. I. M. Young y B. Das. 2004. “Utilizing Standard-based Approaches for Information Sharing and Interoperability in Manufacturing Decision Support.” Paper presented at the 14th International Conference on Flexible Automation and Intelligent Manufacturing, Toronto, July 12–14.

Pouchard, L., N. Ivezic y C. Schlenoff. 2000. “Ontology Engineering for Distributed Collaboration in Manufacturing.” Proceedings of the AIS2000 Conference.

Protégé Ontology. 2014. Consultada en Julio.

http://www.ontologydesignpatterns.org/ont/dul/dul2.owl

PSL. 2010. National Institute of Standards and Technology. Consultada en Diciembre.

http://www.mel.nist.gov/psl/

RAE. 2014. Consultada en Noviembre.

http://www.rae.es/

Roche, C. 2000. “Corporate Ontologies and Concurrent Engineering.” *Journal of Material Processing Technology* 107: 187–193.

Romero Llop, R. 2007. “Especificación OWL de una Ontología para Teleeducación en la Web Semántica.” PhD diss., Universitat Politècnica de València.

Rouquette, N. F., G. M. Wasserman y V. D. Carson. 2005. “OWL for Space Mission Systems development at JPL with semantic architecture styles.”

Scherp, A., T. Franz, C. Saathoff y S. Staab. 2009. “F—A Model of Events based on the Foundational Ontology DOLCE+DnS Ultralite.” Proceedings of the fifth International Conference on Knowledge Capture: 137–144.

Solano, L., P. Rosado, F. Romero y F. González. 2009. “Posibilidades de la ontología PSL para representar planes de proceso no lineales.” Paper presented at the 3rd Manufacturing Engineering Society International Conference, Alcoy, Spain, June 17–19.

Sunagawa, E., K. Kozaki, Y. Kitamura y R. Mizoguchi. 2003. “An Environment for Distributed Ontology Development Based on Dependency Management.” Paper presented at the ISWC-2003, Sanibel Island, Florida, 2003.

Sure, Y., S. Staab, M. Erdmann, J. Angele, R. Studer y D. Wenke. 2002. *OntoEdit: Collaborative Ontology Development for the Semantic Web*, 221–235. Springer Berlin Heidelberg.

Tate, A. 1998. “Roots of SPAR – Shared Planning and Activity Representation.” *The Knowledge Engineering Review* 13 (1): 121–128.

TOVE Ontology Project. 2014. Consultada en Julio.

http://www.eil.utoronto.ca/enterprise-modelling/tove/

Uschold, M. y M. Gruninger. 1996. “Ontologies: Principles, Methods y Applications.” *Knowledge Engineering Review* 11 (2): 93–136.

Uschold, M., M. King, S. Moralee y Y. Zorgios. 1998. “The Enterprise Ontology.” *Knowledge Engineering Review* 13 (1): 31–89.

Capítulo 4. BASES DE LA PROPUESTA

1. Introducción

En este capítulo se recogen las bases de la propuesta de la tesis agrupadas bajo la perspectiva de los enfoques social, funcional, de los procesos y de los recursos. Como se ha indicado en el capítulo 3, en los entornos colaborativos propios de la EV, las ontologías de base facilitan la integración a través de un vocabulario común susceptible de ser utilizado conjuntamente en aquellas ontologías que deriven de la misma ontología de base. Pero además, desde la perspectiva social, estas ontologías deben considerar el carácter agentivo propio de ciertos recursos, humanos y no humanos, que intervienen en el proceso DP^2R. Una capacidad de estos recursos que les permite tener iniciativas y tomar decisiones y determina un comportamiento que proviene del paradigma BDI (Belief-Desire-Intention), y está relacionado con la Teoría de Actividades (AT).

Adicionalmente, como se vió en el capítulo 2, la gestión operacional de las actividades y de los recursos humanos y materiales que intervienen en ellas es de vital importancia para el proceso DP^2R. Por ello, en este capítulo también se presenta un modelo funcional centrado en la monitorización y control de los procesos industriales (ISO/CD 1997), que constituye otra de las bases de la propuesta.

Finalmente, se muestra la ontología PSL, que aporta una gran expresividad semántica en la descripción de los procesos, y que constituye el núcleo de la propuesta desde la perspectiva de los procesos. No obstante, aunque PSL cubre bien la caracterización de las *capacities* de los recursos, no contempla adecuadamente sus *capabilities*. Por ello, la perspectiva de los recursos de la propuesta se completará con aportaciones procedentes del modelo de recursos de MANDATE (ISO 2005).

2. Perspectiva social

Más allá de las propuestas puramente mecanicistas que modelan las actividades como simples procesos o transformaciones, en esta sección se pretende dar una perspectiva de las mismas basada en la consideración de los aspectos sociales y sicológicos ligados a las personas que intervienen en su realización. En la investigación sobre Human-Computer Interaction (HCI) se manejan algunas consideraciones de interés en el marco de la tesis: la visión de los seres humanos como actores activos y no solo como colecciones de atributos propios de procesadores cognitivos; la utilización del término 'actores humanos', que enfatiza el concepto de persona, entendido como agente autónomo que tiene la capacidad de regular y coordinar su comportamiento, en lugar de ser un simple elemento pasivo en un

sistema hombre-máquina; y el reconocimiento de que la cooperación, comunicación y coordinación son, a menudo, vitales para el desarrollo exitoso de las tareas (Bannon 1991).

En los dos subapartados siguientes se introducen la Teoría de Actividades y el carácter agentivo, con los que se explican en buena medida los aspectos sociales y sicológicos ligados a las actividades. Particularmente, en el segundo subapartado se abordan una serie de consideraciones que tienen validez tanto para agentes humanos como para agentes no humanos.

2.1. Teoría de Actividades

La Teoría de Actividades (AT) es un marco filosófico e inter-disciplinar para estudiar formas diferentes de prácticas humanas como los *development processes*, tanto a nivel individual como social, considerando las interrelaciones entre dichos niveles. En este marco de la AT, las actividades son unidades básicas de análisis, donde ellas y sus elementos están en continuo cambio y desarrollo, y donde la correcta comprensión de la situación actual requiere la consideración de su contexto y el conocimiento de la evolución de las actividades (Kuutti 1995).

Una acción se sitúa siempre en un contexto, y es imposible comprenderla sin dicho contexto. Según la AT, para las acciones individuales se debe incluir un mínimo contexto significativo en la unidad básica de análisis, que se denomina actividad. En otras palabras, una actividad proporciona el mínimo contexto significativo para comprender una acción individual.

Una actividad puede estar compuesta de diferentes acciones, y una acción puede pertenecer a actividades diferentes, en cuyo caso, las motivaciones de estas actividades obligarán a que esta acción tenga un sentido distinto en cada actividad. Por otra parte, las actividades no son unidades estáticas, sino que ellas y sus elementos están sometidos a continuos cambios y desarrollo, siendo este desarrollo irregular y discontinuo. Según esto, la comprensión completa de una actividad puede requerir el conocimiento y análisis de las fases de dicha actividad a lo largo del tiempo en que la actividad ha estado realizándose.

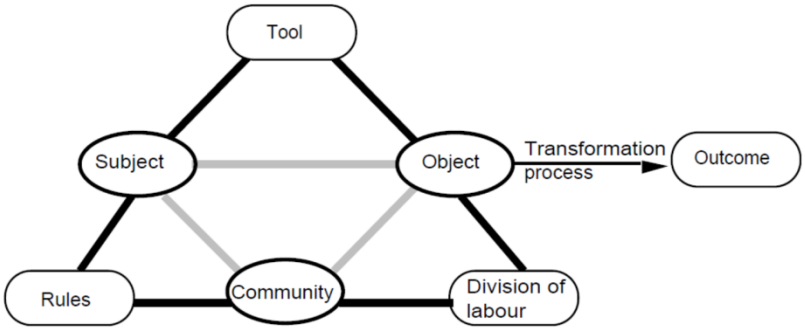


Figura 4.1. Estructura básica de una actividad (Kuutti 1995).

La estructura básica de una actividad en la AT queda establecida, como se muestra en la Figura 4.1, por el triángulo básico, que está constituido por el *sujeto*, el *objeto* y la *comunidad*. Las actividades transforman objetos, que puede ser materiales o inmateriales,

en salidas (*outcome*). Para ello, se requiere de un sujeto o actor, cuya intervención sobre el objeto no es directa, sino que es mediada por una *herramienta*. Además, la representación de todos los aspectos de las relaciones existentes entre un individuo y su entorno en el marco del desarrollo de una actividad precisa de otro componente fundamental de la AT (la *comunidad*), y dos nuevas relaciones (*sujeto-comunidad* y *comunidad-objeto*) que, como se aprecia en la Figura 4.1, son mediadas respectivamente por *reglas* y por la *división del trabajo* (Kuutti 1995).

La mediación es otro de los aspectos claves de la AT, además del contexto y los antecedentes que se explicaron anteriormente. Así, por ejemplo, en el papel de mediación de las herramientas, hay que considerar dos aspectos: la capacidad de facilitar el desarrollo de la actividad y los límites que establecen sobre el desarrollo de la actividad en función de sus propias características. Una *herramienta* puede ser cualquier cosa que se utiliza en el proceso de transformación de la actividad, incluyendo las herramientas materiales (como máquinas e instrumentos) y las cognitivas/intelectuales (como leyes y procedimientos). A su vez, estos elementos de mediación han sido creados y transformados durante el desarrollo de una actividad en la que ellos desempeñaban el papel de *objetos* (*Objects*). El concepto de r*egla* abarca normas explícitas e implícitas, convenciones, y relaciones sociales en una comunidad. La *división del trabajo* se refiere a la organización explícita e implícita de una comunidad en relación al proceso de transformación del *objeto* en salida (*outcome*). Cada uno de estos elementos de mediación se ha conformado a largo del tiempo, conjuntamente con la actividad, y está abierto a modificaciones y desarrollos futuros.

En relación a las tecnologías de la información (TICs), cabe destacar que éstas proporcionan soporte a la AT al menos de dos modos. Por una parte, facilitan la conexión o comunicación necesaria entre varios participantes en una actividad, y por otra, hacen posible que una actividad disponga de un *objeto* que de otro modo sería imposible de capturar, como por ejemplo un objeto de información que incluye un gran volumen de datos con relaciones complejas entre los mismos.

Como se ha indicado previamente, las *actividades* constan de *acciones*, que a su vez constan de *operaciones*[8], existiendo un paralelismo entre esta jerarquía y la jerarquía de *motivación*, *objetivo* y *condición* (Figura 4.2). Además de la relación de inclusión o jerarquía que existe entre ellas, las actividades, las acciones y las operaciones se diferencian por el modo y condicionantes de su ejecución: las actividades se desarrollan como acciones individuales y cooperativas con la misma motivación global (*Motive*); las acciones se desarrollan con control consciente y tienen un objetivo (*Goal*) inmediato; y las operaciones, son rutinas bien definidas que se ejecutan con control inconsciente como respuesta a las condiciones (*Conditions*) que se manifiestan durante el desarrollo de la acción. No obstante, los límites entre actividad y acción, y entre acción y operación son difusos. Por ejemplo, la consideración de acción es relativa y puede modificarse en un sentido o en otro. Como se verá a continuación, una actividad puede perder su motivación y transformarse en una acción, y una acción puede convertirse en una operación cuando cambia el objetivo o cuando se produce una rutinización de la misma (Figura 4.2).

8 Tanto esta terminología (actividad, acción y operación), como las relaciones que se establecen entre actividades, acciones y operaciones, son propias de la Teoría de Actividades y no son necesariamente compartidas por las propuestas de esta tesis ni por las propuestas de otros autores incluidas en este documento.

Según la AT, antes de que una acción se realice en el mundo real, dicha acción es planeada en el ámbito mental o cognitivo mediante un modelo. Cuanto mejor sea este modelo, más exitosa podrá resultar la acción. Esta fase previa al desarrollo de la acción, en la que se crea el modelo, se denomina orientación (*orientation*). Los modelos creados no son descripciones rígidas y precisas de los pasos de la ejecución, sino modelos incompletos y tentativos. Inicialmente, cada operación es una acción consciente, que incluye las fases de orientación y ejecución. Pero cuando el modelo correspondiente es suficientemente bueno y la acción ha sido practicada durante el tiempo suficiente, la fase de orientación se desvanece y la acción se transforma (colapsa) en una operación, que se desarrolla de forma mucho más fluida que la acción (inconscientemente). A la vez que desaparece esta acción, que ha colapsado dando lugar a una operación, se crea una nueva acción que tendrá un alcance más amplio que la anterior y que contendrá la nueva operación recién formada como una subparte de ella. De forma análoga, cuando cambian las condiciones, una operación puede desplegarse (*unfold*) y alcanzar al nivel de acción, del mismo modo que al desplegarse una acción se obtiene una actividad.

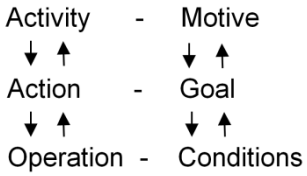


Figura 4.2. Niveles jerárquicos de una actividad (Kuutti 1995).

Otra característica relevante de la AT deriva de la relación recíproca existente entre el sujeto y el objeto de la actividad. Al tiempo que el sujeto está transformando al objeto, las características del sujeto se ven afectadas por la naturaleza, características y comportamiento del objeto durante el desarrollo de la actividad. Por otra parte, es importante recordar que la AT considera que las actividades están inmersas en un proceso continuo de cambio y desarrollo a todos los niveles (operaciones, acciones y actividades), y que en dicho proceso, las modificaciones en uno de estos niveles implican cambios en los restantes. De hecho, las actividades son como los nodos de redes jerárquicas entrelazadas, que se ven influenciadas por otras actividades y por el entorno. Los factores externos de este entorno ocasionan cambios en determinados elementos de las actividades que a su vez, provocan desequilibrios, o desajustes, que pueden impulsar el desarrollo de nuevas actividades.

La Teoría de Actividades y el concepto mismo de actividad parecen particularmente adecuados y ricos para ser utilizados como base del estudio de las interacciones entre personas y, entre personas y máquinas. Particularmente en entornos sociales, donde el carácter dinámico, la consideración de los antecedentes y la mediación, y la interacción con todos los elementos del contexto son aspectos determinantes. Por otra parte, el carácter integrado de la AT posibilita la discusión acerca de asuntos pertenecientes a diferentes niveles, siempre que éstos puedan situarse en un marco común. Por ejemplo, el marco del ciclo de desarrollo del producto, donde cada individuo o equipo de trabajo es un *sujeto*, el conjunto de los individuos o equipos de trabajo implicados en el proceso constituyen la *comunidad* y el *objeto* es el trabajo desarrollado.

En resumen, tres son los aspectos destacables de la AT como fundamento de la presente tesis: su carácter multinivel; la posibilidad de estudiar interacciones embebidas en contextos sociales; y la capacidad para tratar con situaciones dinámicas y en desarrollo.

2.2. Carácter agentivo en DOLCE

Los conceptos de motivación y control consciente propios de la AT están relacionados con la existencia de agentes o actores implicados en la realización de las actividades. Según Kethers (2000), un *actor* es una entidad activa que lleva a cabo acciones para alcanzar objetivos (*goals*) mediante el ejercicio de su saber hacer. Para Ferrario y Oltramari (2004), los agentes son entidades que tienen creencias, deseos e intenciones (*Beliefs*, *Desires* e *Intentions*), características que se reúnen bajo la denominación de carácter BDI o carácter agentivo, y que sirven para caracterizar la naturaleza y comportamiento de dichos agentes.

Como se expuso en el capítulo anterior, DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering) y algunas de sus extensiones han sido utilizadas con éxito para dar soporte a distintas ontologías de dominio. DOLCE tiene un claro sesgo cognitivo y pretende captar las categorías ontológicas que subyacen en el lenguaje natural humano y en el sentido común, introduciendo categorías pensadas como objetos de conocimiento que dependen de la percepción, las huellas culturales y las convenciones sociales humanas. Estas categorías son nociones descriptivas que asisten a la construcción de conceptualizaciones explícitas previamente formadas o existentes (Masolo et al. 2003). En este sentido, la extensión de DOLCE *Computational Ontology of Mind* (COM) obedece a la necesidad de conceptualizar, desde un punto de vista ontológico, el conjunto de las entidades que juegan un rol en la tecnología de agentes para los sistemas de información y establecer las conexiones adecuadas entre dichas conceptualizaciones. Las entidades a considerar son: agentes con intenciones (*intentional agents*) y entidades mentales, como objetos mentales (*mental objects*) y estados mentales (*mental states*) (Ferrario y Oltramari 2004).

Los tres estados mentales (especializaciones de la clase *Mental State*) de la ontología COM, corresponden a las categorías *Belief*, *Desire* e *Intention* (Figura 4.3). Las creencias (*Beliefs*) representan el conocimiento que un agente tiene acerca del mundo. Los deseos (*Desires*) representan los estados que un agente quiere alcanzar, y por tanto, en cierto sentido, sus objetivos, mientras que las intenciones (*Intentions*) representan los deseos que el agente se ha comprometido a alcanzar. Estas últimas, las intenciones desempeñan un rol muy importante, ya que restringen el comportamiento y las acciones del agente, determinando si éste continúa persiguiendo un objetivo determinado o lo abandona en favor de otro.

Por su parte, un *intentional agent* se define como aquel que siempre actúa en base a sus deseos, creencias e intenciones, y posee la habilidad de adaptarse al entorno, actualizando y revisando estos desesos, creencias e intenciones. La caracterización de los *intentional agents* obvia su apariencia física y conduce a una representación próxima a un ser humano, un robot o un software. En el caso de agentes no humanos, incluso si existe una explicación física basada en su funcionamiento, es más simple, a nivel descriptivo, dar una representación en términos de intencionalidad, lo que equivale a equiparar un estado interno de la máquina (robot o hardware) con las creencias o intenciones humanas. Los *intentional agents* pueden considerarse dirigidos por objetivos en dos sentidos: en general, porque sus acciones terminan con la consecución de un objetivo; y más específicamente, porque

construyen una representación mental del objetivo, o de la acción necesaria para su consecución, y de las consecuencias resultantes. Otra característica nuclear de los *intentional agents* es que pueden estar, y habitualmente están, ubicados en un determinado contexto. Por tanto, poseen capacidades sociales que se traducen básicamente en su capacidad de interactuar. Esta interacción, que puede ser directa o mediada a través de su entorno, significa que los *intentional agents* pueden ser influenciados por otros *intentional agents* en sus actividades mentales, en el cumplimiento de sus objetivos o en la ejecución de sus tareas. Para propiciar esta interacción, debe existir algún tipo de comunicación entre los agentes, no necesariamente verbal, y de forma específica, los agentes deben ser capaces de representarse a sí mismos y de comunicarse acerca de ellos mismos. Según Wooldridge (2000), las propiedades que debe poseer un *intentional agent* son: autonomía, que significa que el agente no está influenciado directamente por otros agentes; reactividad, o capacidad para reaccionar adecuadamente ante estímulos externos, de modo que cualquier nueva percepción debe determinar una revisión de sus estados mentales, aunque éstos continúen inalterados después de ser reconsiderados; proactividad, o capacidad para tomar la iniciativa; y habilidad social, o capacidad para interactuar con otros agentes, que incluso pueden ser fuente de nuevos estados mentales.

Siguiendo con la explicación de la ontología COM, cabe indicar que un objeto mental es simplemente la representación de algo. En DOLCE, los objetos mentales se han caracterizado como un subtipo de la clase *Non-physical Objects*. En la taxonomía de DOLCE (Figura 4.3) se aprecia que estos *Non-physical Objects* se dividen en *Social Objects* y *Mental Objects*, según que éstos dependan o no de una comunidad de agentes. A su vez, la clase *Mental Object* se divide en *Computed Object* y *Percept*. Las percepciones (*percepts*) son independientes de cualquier otro objeto mental, mientras que los objetos computados (*computed objects*) dependen al menos de otro objeto mental. Los objetos mentales pertenecientes a la categoría de *Computed Objects* son el resultado de los procesos computacionales que ocurren cada vez que una entrada (externa o interna al *intentional agent*) es procesada. Estos *computed objects* pueden ser de tres tipos: *computed beliefs*, *computed desires* y *computed intentions*, en función de que la entidad procesada sea respectivamente, una creencia, un deseo o una intención.

A continuación se presenta un ejemplo relacionado con la planificación de procesos, que facilita la interpretación de algunas de las entidades de la ontología COM expuestas en esta subsección. Sea un planificador (*intencional agent* o *Agentive Physical Object* en la taxonomía de la Figura 4.3) encargado de la planificación de un determinado proceso de fabricación. Cuando este planificador, en el marco del desarrollo de su tarea de selección de recursos, haya evaluado las capacidades de un recurso ejecutando una actividad, tendrá una representación de dichas capacidades (un *Mental Object*). Esta representación será de tipo *Computed Object* ya que es el resultado de un proceso cognitivo del planificador consistente en el procesado de la información relativa a las *capabilities* del recurso. Concretamente, este objeto computado será de tipo *Computed Belief*, ya que corresponde a la representación del conocimiento de las capacidades del recurso. En ese instante, el planificador tiene un estado mental (*Mental Attitude*) de tipo *Belief*. Es decir, conoce (*belief*) algo relativo a dicho recurso ya que tiene una representación mental (*Computed Belief*) de sus capacidades. A partir de ese momento, el planificador puede procesar ésta y otras informaciones relacionadas con la asignación de los recursos, que le lleve a una representación mental de la decisión que va tomar (*Computed Intention*) en relación a la asignación del recurso. Esta representación mental de su decisión se traduce en un estado

mental de tipo *Intention*. Es decir, el planificador ha tomado su decisión y está en disposición de realizar la acción correspondiente para alcanzar un determinado objetivo o deseo (*Desire*), que también es un estado mental (*Mental Attitude*). A su vez, dicho objetivo (*Desire*) está asociado a un objeto mental computado (*Computed Desire*) que ha sido el resultado de un proceso mental (cognitivo) por el cual, el planificador es informado o decide acerca del objetivo a alcanzar.

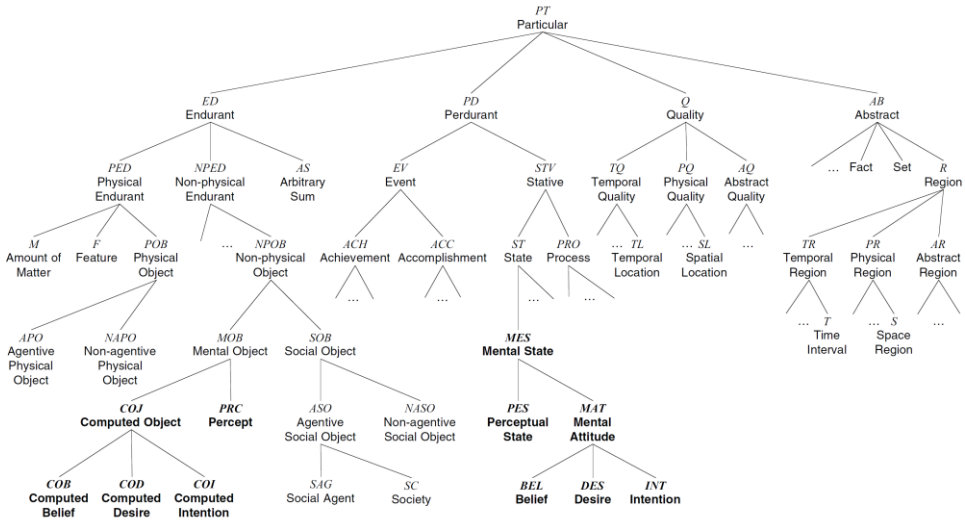


Figura 4.3. Taxonomía que integra las categorías básicas de las ontologías DOLCE y COM (Ferrario y Oltramari 2004).

3. Perspectiva funcional

Como se ha indicado en la introducción de este capítulo, la gestión operacional de las actividades y de los recursos humanos y materiales que intervienen en el proceso de Desarrollo de Productos, Procesos y Recursos (DP^2R) es de vital importancia para este trabajo. Además, en el marco de esta tesis era importante considerar la aproximación que bajo la perspectiva funcional de los procesos realiza la norma ISO 15531 (ISO 2004).

La iniciativa ISO 15531 –Industrial Manufacturing Management Data–, también conocida como MANDATE, especifica las características para una representación de la información relativa a la gestión de la fabricación en el ámbito de una empresa de fabricación discreta, e incluye un modelo común normalizado para la información relativa a la gestión de los recursos de fabricación (ISO 2004; ISO 2005). En esta norma se incluye el modelo conceptual para la monitorización y control de procesos industriales, una especie de modelo funcional que puede relacionarse con IDEF0 (NIST 1993). Es decir, un modelo centrado en la descripción de las funciones a través de las relaciones entre los objetos que intervienen, con distintos roles, en las actividades del modelo.

Las normas de la serie ISO 15531-4x, que mantienen estrechos vínculos con los datos de gestión de uso de los recursos, soportan el control y monitorización de los flujos de

fabricación o de procesos industriales en el entorno de fabricación discreta (ISO/CD 1997). En ellas se definen los datos que describen las *capacities* de los recursos de fabricación; se proporciona un modelo de tiempo, un modelo conceptual para la monitorización y control del flujo; y un conjunto relacionado de bloques de construcción (*building blocks*) con los cuales se especifican los modelos y las representaciones normalizadas de los datos para la planificación, programación, control y monitorización del flujo de materiales.

Los tres tipos de datos descritos en el conjunto de las normas ISO 15531 (información relativa a: los intercambios externos, los recursos empleados en la fabricación y los flujos de fabricación) se representan mediante EXPRESS y son consistentes con STEP (ISO 10303). Esto facilita la integración entre las numerosas aplicaciones industriales en las que dichos conjuntos de datos forman el núcleo del proceso de fabricación, se comparten y son intercambiados durante el ciclo de vida del producto (ISO 2004).

3.1. El modelo conceptual para la monitorización y control de procesos industriales

Este modelo se basa en: (a) la representación del proceso y de las actividades necesarias para monitorizar y controlar dicho proceso; y (b) la existencia de interfaces que faciliten la comunicación entre estas dos representaciones, especialmente para la transmisión de órdenes de control a los procesos industriales y para la retroalimentación de la información al sistema de información de control. A partir de estas bases, la propuesta de modelo conceptual para la monitorización y control de procesos industriales se articula en torno a la modelización de: (a) los elementos y procesos que serán monitorizados y controlados; y (b) los elementos (datos) y procesos requeridos para realizar las actividades de monitorización y control propias de la gestión industrial. Para capturar la semántica de los elementos asociados con la monitorización y control de procesos discretos, se han definido una serie de bloques de construcción (ver Tabla 4.1), que soportan la representación gráfica de los principales aspectos del modelo y con los que se pretende disponer de una rápida interpretación y comprensión del mismo.

En el modelo, una operación está definida como una transformación de elementos específicos de un estado a otro. Es decir, una transformación de al menos un elemento de entrada en uno o más elementos de salida. Un elemento en el modelo es una representación estática de cualquier cosa que pueda ser caracterizada por su comportamiento y sus atributos, considerando que tanto estos atributos como los estados dependen del contexto y pueden cambiar a lo largo del tiempo. En cualquier caso, en el modelo solamente es preciso especificar aquellos elementos que requieren decisiones y que se representan mediante un pequeño círculo en color negro (*element-in-state*).

Aunque cada elemento tiene sus propias características que lo diferencian del resto de elementos, para determinadas actividades de monitorización y control no es necesario referenciar o identificar individualmente a cada elemento, sino a categorías que están definidas como grupos de elementos con características comunes, que pueden variar con el contexto de aplicación. Las jerarquías de *element-in-state-categories* representan a grupos de elementos con un estado común para los cuales están pendientes decisiones comunes. La descripción de las categorías viene determinada por la situación en la que deben tomarse las decisiones y se visualiza en una representación gráfica mediante el símbolo '△' (*element-in-state-category-node*). Debido a la dinámica de los procesos industriales, la asignación de elementos a las *element-in-state-categories* puede cambiar a lo largo del tiempo. Así, se

considera que un elemento es miembro de una *element-in-state-category* si tiene las características de estado que definen a esa categoría. En cuyo caso, dicho elemento será considerado idéntico al resto de los elementos de la categoría. Por otra parte, también cabe la posibilidad de limitar el número máximo o mínimo de elementos asociados a una *element-in-state-category*. Igualmente, puede crearse una *operation-category* a partir de un número de operaciones u *operation-categories*. Las *operation-categories* pueden ser declaradas como *operation-categories-nodes*, cuya representación gráfica se realiza mediante una figura rectangular (□).

Tabla 4.1. Bloques de construcción del modelo para la monitorización y control de procesos industriales (ISO/CD 1997).

Concepto	Bloque de construcción	Representación gráfica
Elemento en un estado para el cual está pendiente una decisión.	element-in-state	•
Conjunto de elementos en un mismo estado para los cuales están pendientes decisiones comunes.	element-in-state-category	(sin símbolo)
Conjunto seleccionado de elementos en un mismo estado para los cuales están pendientes decisiones comunes.	element-in-state-category-node = selected element-in-state-category	△
Transición de estado.	operation	(sin símbolo)
Conjunto de operaciones con características comunes para las cuales están pendientes decisiones comunes.	operation-category	(sin símbolo)
Conjunto seleccionado de operaciones con características comunes para las cuales están pendientes decisiones comunes.	operation-category-node = selected operation-category	□
Flujo de elementos.	element path	————
Intercambio de información de control.	control information path	- - - - - -
Relaciones de intercambio de datos.	data for data exchange path	- · - · - · -

Para el modelado del comportamiento dinámico en un contexto determinado, se requiere una definición de la medida del tiempo en dicho contexto basada en la secuencia de eventos repetibles, donde un instante de tiempo es un evento, y dos instantes de tiempo definen la unidad de duración del tiempo. Así, un dominio de tiempo puede definirse como:

$(T, \leq)$ donde T es un conjunto de puntos en el tiempo y $\leq$ es una relación de orden total en T.

Los dominios de tiempo formulados pueden ser continuos o discretos:

time (R+) o time $(R+, \leq)$ es un dominio continuo

time (N_o) o time $(N_o, \leq)$ es un dominio discreto

Para observar y comparar eventos relevantes en el sistema a modelar, debe especificarse un dominio de tiempo, que será elegido dependiendo del objetivo del modelo y del tipo de comportamiento dinámico de los elementos y operaciones a modelar. En cualquier caso, resulta recomendable la definición de un dominio de tiempo general, que sirva de base para especificar los dominios de tiempo particulares correspondientes a diferentes sistemas. De este modo, en caso de ser necesarias, se simplificarían las reglas de transformación entre los diferentes dominios de tiempo.

Las categorías de operaciones (*operation-categories*) representan transiciones de estado que transforman elementos de entrada con su estado, en elementos de salida con otro estado. El establecimiento de enlaces (links) entre *category-nodes*, que se representa gráficamente mediante líneas de trazo continuo entre los *category-nodes* correspondientes, permite la descripción de un posible flujo de elementos desde una categoría a otra (flujo, entendido como la asignación de elementos a categorías, y no como movimiento físico de los elementos). El modelo resultante es un grafo de *element-in-state-categories* conectadas y *operation-categories* decribiendo la transformación de elementos de entrada en elementos de salida (Figura 4.4). En este gráfico, un *element-in-state-category-node* deberá ir precedido de una *operation-category-node* y viceversa.

El comportamiento del sistema está representado por el flujo de elementos en el grafo. A la entrada de una operación se consumen elementos, y al final de la operación se generan nuevos *element-in-states*. A lo largo de las rutas (*paths*) definidas, los elementos son asignados por las actividades de monitorización y control local a las siguientes *element-in-state-categories*.

En este modelo, un elemento puede desempeñar un único rol en una operación, que podrá ser como objeto de operación o como medio de operación. Los elementos que intervienen como objetos de operación cambian sustancialmente durante la operación. Los medios de operación constituyen una ayuda para realizar la operación y no forman parte de la salida demandada. Normalmente, estos medios de operación están diseñados para ser utilizados un número indeterminado de veces, una vez en el peor de los casos, recuperando su estado al inicio de la operación cuando ésta termina. Es decir, que los medios de operación vuelven al mismo *element-in-state-category* tras finalizar la operación en la que han intervenido.

A su vez, los medios de operación pueden ser de dos tipos: medios físicos de operación e información. Estos últimos pueden ser abstractos, usualmente son fácilmente duplicables y representan la información de entrada requerida para una operación, y por lo tanto, su disponibilidad necesita ser programada.

Un *element path* (ver Tabla 4.1) que simboliza el flujo de objetos de operación entre un *element-in-state-category-node* y un *operation-category-node* se representa gráficamente mediante un línea horizontal. Mientras que si se trata de representar el flujo de los medios de operación, la línea será vertical, ya que como se muestra en la Figura 4.4, los elementos de tipo information y de tipo medio físico de operación, se sitúan encima y debajo respectivamente de las operaciones.

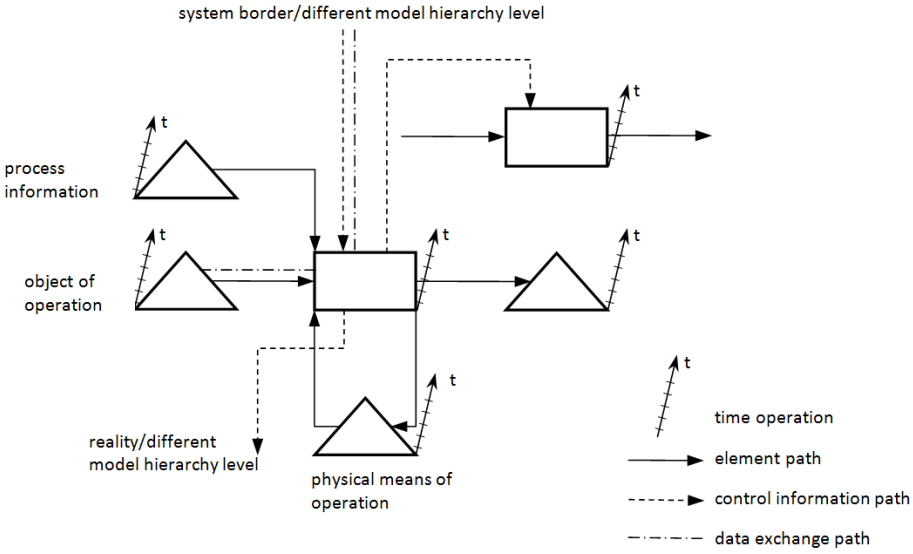


Figura 4.4. Representación de la conceptualización del modelo de información para la monitorización y control de procesos industriales (ISO/CD 1997).

Además del flujo de elementos, para la ejecución de las actividades de monitorización y control en procesos locales, a menudo, se requiere un flujo de información entre las categorías. Este tipo de información incluye restricciones adicionales relevantes para las actividades de monitorización y control que no están disponibles localmente. Para marcar el posible flujo de información de control, se utiliza una línea de trazo discontinuo. En este caso, al contrario que con los *element paths*, no existe restricción en la secuencia de nodos. Finalmente, cuando es preciso intercambiar los datos de control y monitorización del flujo de materiales con el modelo, se necesita una nueva relación de intercambio, que se representa gráficamente mediante una línea discontinua de punto y trazo (Figura 4.4).

4. Perspectiva de los procesos

En esta sección se presentan, analizan y ejemplifican los elementos y recursos expresivos de PSL que se identifican claramente como bases de la propuesta de esta tesis. Dichos recursos se han agrupado en tres subsecciones. En las dos primeras se reúnen aquéllos correspondientes a PSL-Core y Outer Core (Figura 3.11), y en la tercera se encuentran los recursos expresivos relacionados con la representación de la flexibilidad del plan de procesos. Algunos de los ejemplos y expresiones KIF que aparecen a continuación están relacionados con el plan de proceso para la fabricación de la pieza identificada como ‘casquillo AB’, que incluye varias operaciones de torneado agrupadas en dos subfases (Figura 4.5).

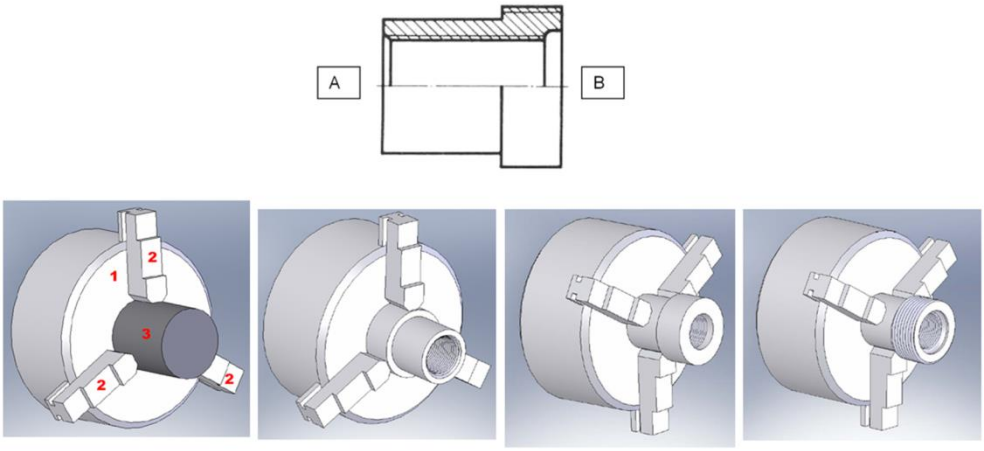


Figura 4.5. Casquillo AB y representación gráfica de su proceso de fabricación.

4.1. Recursos expresivos de PSL-Core

La consideración del tiempo es un aspecto fundamental en la especificación de los procesos. Una de las entidades básicas del núcleo de PSL es *timepoint*. Los instantes de tiempo (*timepoints*) constituyen un conjunto ordenado linealmente cuyos extremos se sitúan en el infinito (Gruninger y Menzel 2003). El significado de las constantes de PSL-Core *inf+* e *inf-* se asocia a sendos instantes de tiempo que son posterior y anterior respectivamente a cualquier otro instante de tiempo considerado. Así, por ejemplo, la expresión KIF (= ?t inf+) es verdadera si y solo si ?t es el único instante de tiempo posterior a todos los demás instantes de tiempo. El hecho de que todos los instantes de tiempo distintos de inf+ son anteriores a inf+ se expresa formalmente con la ayuda de la relación *before*, que se establece entre dos instantes de tiempo:

```
(forall (?t)
        (if (and (timepoint ?t)
                 (not (= ?t inf+)))
                 (before ?t inf+)))
```

El significado de esta expresión KIF en lenguaje natural sería: para cualquier ?t, si ?t es un instante de tiempo y ?t no es igual a inf+, entonces ?t es anterior a inf+.

Las actividades y sus realizaciones son los conceptos más importantes de PSL. En torno a ellas se articulan prácticamente todas las entidades y relaciones de la ontología, y el tratamiento que reciben es un buen ejemplo de la gran expresividad de PSL. Una *activity*, o actividad, es un comportamiento que puede repetirse un número indeterminado de veces, o que puede no realizarse en ninguna ocasión. Mientras una *activity_occurrence*, o realización, es cada una de las realizaciones concretas de una actividad. Conviene reflexionar sobre la naturaleza de las actividades y sus realizaciones para clarificar las diferencias entre ambas. La entidad *activity* de PSL tiene una dimensión social como 'objeto de la comunicación'. Es decir, sobre ella tiene que establecerse un entendimiento mutuo entre los agentes o funciones. Por ejemplo, los agentes que intervienen en la

planificación de los procesos de fabricación, para desarrollar sus tareas deben compartir el mismo concepto de las actividades que planifican, entendidas éstas como comportamientos que pueden repetirse. En este sentido, una *activity* es un *Social Object* de la ontología DOLCE, y no tienen partes temporales, mientras que una *activity_occurrence*, es una entidad que sucede en el tiempo, y que puede tener partes temporales. Por ejemplo, la relación *is_occurring_at* sitúa la realización de una actividad en un determinado instante de tiempo comprendido entre los instantes de tiempo que marcan el inicial y final de dicha realización.

Conviene igualmente insistir en la idea de que una *activity_occurrence* no es una instancia de una *activity*. Ambas son entidades que pueden tener sus instancias, entre las cuales pueden existir relaciones como la relación *occurrence_of*, que se establece entre una *activity_occurrence*, u *occurrence*, y la *activity* a la que corresponde. Por ejemplo, en la expresión siguiente se indica que `?occMecanizarCasquilloAB` es una realización de la actividad `?mecanizarCasquilloAB`. Para ello, previamente, se ha declarado que `?mecanizarCasquilloAB` es una actividad, y que `?occMecanizarCasquilloAB` es la realización de una actividad.

```
(activity ?mecanizarCasquilloAB)
(activity_occurence ?occMecanizarCasquilloAB)
(occurrence_of ?occMecanizarCasquilloAB ?mecanizarCasquilloAB)
```

El inicio y final de las realizaciones de las actividades, y el intervalo en el cual existen los objetos viene determinado por *timepoints* o instantes de tiempo. Por ejemplo, las funciones *beginof* y *endof* de la expresión siguiente devuelven respectivamente el instante de tiempo en el que se inicia y finaliza la realización, o ejecución, de una actividad consistente en mecanizar una pieza del tipo que se ilustra en la Figura 4.5.

```
( = ?t1 (beginof ?occMecanizarCasquilloAB))
( = ?t2 (endof ?occMecanizarCasquilloAB))
```

Para representar la participación de un objecto en la realización de una actividad se utiliza la relación *participates_in*. Así, la expresión (`participates_in ?x ?occ ?t`) significa que el objeto `?x` desempeña algún rol en la realización `?occ` en el instante de tiempo `?t`. Obviamente, un objeto solamente puede participar en la realización de una actividad en los instantes de tiempo en los que existe y la actividad se está realizando.

La ontología PSL da soporte al concepto del tiempo transcurrido entre dos *timepoints* cualesquiera mediante la función *duration*, perteneciente al Outer Core. Por ejemplo, *duration* puede proporcionar la duración del intervalo de tiempo correspondiente a la realización de una actividad. Del mismo modo, la función *duration* puede servir para especificar la duración de un tiempo de espera o tiempo transcurrido entre las realizaciones de dos actividades, que a priori, pueden considerarse consecutivas en el tiempo. Por ejemplo, el tiempo transcurrido desde que se mecaniza una pieza hasta que se desembrida del utillaje:

```
( = ?t1 (endof ?occMecanizarCasquilloAB))
( = ?t2 (beginof ?occDesembridar))
( = ?t3 (duration ?t1 ?t2))
```

En el caso de la expresión anterior, el valor ?t3 devuelto por la función *duration* se obtiene como la diferencia de ?t2 menos ?t1.

4.2. Recursos expresivos del Outer Core

Uno de los conceptos básicos de PSL, incluido en la extensión denominada *Subactivity Theory* (Outer Core), se sustancia en la posibilidad de que las actividades pueden descomponerse de forma recurrente en otras actividades. Así que, una actividad es susceptible de ser considerada como subactividad de otra de la cual forma parte, y a su vez, una subactividad puede estar dividida en otras subactividades. Este tipo de actividades, las que pueden tener subactividades, se denominan complejas, mientras que las que no pueden tener subactividades, se denominan actividades primitivas. Del mismo modo, las 'realizaciones primitivas' son las realizaciones de actividades primitivas, y las 'realizaciones complejas' son las realizaciones de actividades complejas. No obstante, esta clasificación depende del nivel de detalle de la aplicación. Por ejemplo, desde un punto de vista general, puede considerarse que `fabric` (fabricación) es una actividad compleja compuesta por actividades primitivas como `prepararHtas` (preparar herramientas). Mientras que desde otro punto de vista, la actividad `prepararHtas` podría estar subdividida en actividades, siendo por tanto una actividad compleja.

Relacionando lo anterior con el ejemplo de la Figura 4.5, la actividad correspondiente al mecanizado de la subfase B del casquillo AB (`subfaseBCasquilloAB`), sería una actividad compleja formada por tres subactividades primitivas (`mandrinadoB`, `refrentadoB` y `roscadoInterior`), y la expresión siguiente sirve para representar la relación jerárquica entre la actividad y sus subactividades, mediante la relación `subactivity`.

```
(activity ?subfaseBCasquilloAB)
(subactivity ?mandrinadoB ?subfaseBCasquilloAB)
(subactivity ?refrentadoB ?subfaseBCasquilloAB)
(subactivity ?roscadoInterior ?subfaseBCasquilloAB)
```

La misma relación jerárquica puede expresarse entre realizaciones de actividades. Así, en la expresión siguiente se representa la realización de una subactividad (`?occMandrinadoAB`) como parte de la realización de una actividad compleja (`?occSubfaseCasquilloAB`):

```
(occurrence_of ?occSubfaseBCasquilloAB ?subfaseBCasquilloAB)
(occurrence_of ?occMandrinadoB ?madrinadoB)
(subactivity_occurrence ?occMandrinadoB ?occSubfaseBCasquilloAB)
```

Además de las actividades primitivas y complejas, en PSL se contemplan las llamadas actividades atómicas. Una actividad es atómica si y solo si es primitiva o si es la superposición concurrente de un conjunto de actividades primitivas. En PSL las actividades que participan en agregaciones concurrentes se pueden considerar como subactividades de actividades atómicas (Bock 2006), sin embargo, y en cualquier caso, una actividad atómica es diferente de una actividad compleja (PSL 2010).

La relación *conc*, que aparece en la teoría de actividades atómicas (*Theory of Atomic Activities*), se aplica sobre un conjunto de actividades atómicas, dando como resultado otra

actividad atómica que es su composición concurrente. La superposición concurrente de actividades proporciona un resultado que no coincide necesariamente con el resultado obtenido a partir de cualquiera de las actividades concurrentes por separado, o a partir de cualquier otra combinación de dichas actividades primitivas.

Las actividades, o sus realizaciones se agrupan para definir un proceso en el cual se van sucediendo según una pauta determinada total o parcialmente. Para facilitar la comprensión de la semántica de PSL en relación a esta idea, conviene analizar previamente los conceptos de *occurrence tree* y *activity tree*, que pueden traducirse literalmente como 'árbol de realizaciones' y 'árbol de actividades', pasando a continuación al tratamiento de las relaciones que permiten definir el inicio y el final de una actividad, la secuencia de las actividades, y otros aspectos relacionados con el modelado del proceso.

Según Bock y Gruninger (2004a), uno de los fundamentos de PSL es *situation calculus*, que representa una situación inicial a partir de la cual se define una estructura de tipo árbol que muestra todas las posibles formas en las que se pueden desplegar los acontecimientos o realizaciones en el ámbito considerado (Fox y Gruninger 1998). Como la estructura de las *situations* es de tipo árbol, dos secuencias de acciones diferentes conducen a situaciones diferentes. Así, cada rama que comienza en una situación inicial puede ser entendida como un futuro hipotético. La estructura de árbol de *situation calculus* muestra todos los caminos posibles en los que se pueden desplegar las realizaciones de las actividades. Por tanto, cualquier secuencia arbitraria de acciones identifica una rama en el árbol de situaciones (Gruninger y Fox 1995).

Situation calculus es coherente con el concepto de *occurrence tree* que se maneja en la ontología PSL. El *occurrence tree* de una actividad, según ilustra la Figura 4.6, representa todo lo que puede suceder durante la ejecución de dicha actividad, incluyendo realizaciones que no forman parte de esa actividad y que pueden ocurrir entre las realizaciones de las subactividades declaradas explícitamente como parte de la actividad (Bock y Gruninger 2005). En este sentido, la semántica de PSL permite reflejar situaciones similares a las que suceden en la realidad cuando se ejecuta una actividad, en la que además de las realizaciones de las subactividades especificadas pueden suceder otras no especificadas, que dependiendo de los casos podrían calificarse como previsibles, posibles, e incluso físicamente imposibles.

En el caso de la pieza y proceso mostrados en la Figura 4.5, una realización previsible, aunque no especificada en la planificación sería la inspección visual o con la ayuda de otros medios, realizada tras alguna de las operaciones de mecanizado para comprobar la adecuación del resultado de la misma; una realización posible, sería la rotura de la herramienta durante alguna de las operaciones de mecanizado; y una realización físicamente imposible sería recuperar la forma previa de la pieza tras una de las operaciones de mecanizado.

En tiempo de ejecución, cada realización del *occurrence tree* tiene exactamente un sucesor, indicando la posibilidad de que dicha actividad pueda realizarse a continuación. Por tanto, las ramas del *occurrence tree* representan las trazas de las posibles ejecuciones (Bock y Gruninger 2005). Para representar este concepto se utiliza la relación *successor*, incluida en la *Theory of Occurrence Trees* (Outer Core), y que sirve para identificar la actividad atómica que se ejecuta tras la realización de otra actividad atómica (ver Figura 4.6). La expresión siguiente muestra la utilización de la relación *successor*. Su significado es que

`?occ2` es la realización de la actividad `?a` que viene a continuación de `?occ` en el *occurrence tree*. O dicho de otro modo, `?occ` es el antecendente de `?occ2`.

```
(= (successor ?a ?occ) ?occ2)
```

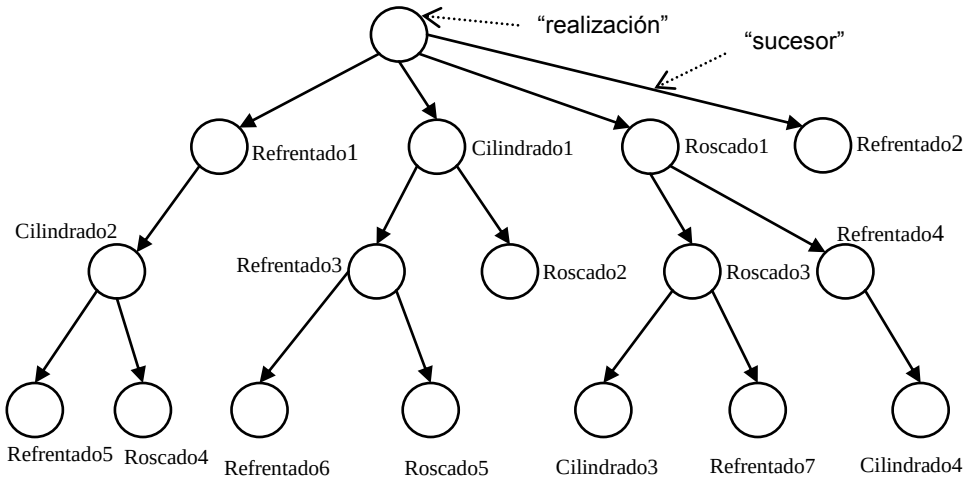


Figura 4.6. Vista parcial del *occurrence tree* de una actividad de mecanizado.

El interés de la relación *successor* radica en que se establece entre realizaciones de actividades para las cuales no existe ninguna restricción más allá del hecho de ser actividades atómicas. Y esto último puede soslayarse fácilmente, ya que cualquier actividad es susceptible de ser representada como atómica a un determinado nivel. Como se verá más adelante, la ontología PSL prevé distintos modos de establecer la secuencia entre las realizaciones de las subactividades de una actividad. Sin embargo, en este caso, al no existir este tipo de restricción, con *successor* es posible secuenciar la realización de actividades cuya única relación es la que deriva de la secuencia temporal en la que se desarrollan.

La Figura 4.6 muestra una vista parcial del *occurrence tree* de la actividad 'mecanizarCasquilloAB', donde aparecen las realizaciones de algunas de las actividades atómicas posibles bajo esta actividad compleja. Los elementos del *occurrence tree* son las realizaciones atómicas, que se representan mediante círculos, mientras que las flechas del diagrama identifican las relaciones de tipo *successor*. En esta figura, a partir de una realización cualquiera (la que está identificada como 'realización') se muestran ejemplos de lo que podría acontecer en todas las realizaciones posibles de esta actividad. Así por ejemplo, hay ramas en las que cambia la secuencia de las operaciones y otras en las que se repiten algunas de las operaciones. Como también puede apreciarse, no todas las ramas tienen el mismo número de operaciones. Cada elemento del *occurrence tree* siempre tiene un único antecedente, mientras que podría tener distintos sucesores (*successors*) si se consideran diferentes realizaciones complejas a partir de los elementos del *occurrence tree*.

No obstante, en el contexto de una realización compleja en concreto, cada realización atómica tendrá un único sucesor.

Por otra parte, las realizaciones de una actividad compleja que cubren todas las posibilidades de una ejecución de la misma, se denominan colectivamente *activity tree* (Bock y Gruninger 2005). El *activity tree* de la Figura 4.7 representa las tres realizaciones posibles de una actividad compleja determinada, bajo el supuesto de que la realización correspondiente a la primera subactividad sea el refrentado. A diferencia del *occurrence tree*, cuyos elementos son las realizaciones de actividades atómicas, los elementos del *activity tree* son realizaciones de actividades complejas. Para el caso de la Figura 4.7, todos los círculos que pueden unirse mediante una trayectoria recorrida en el sentido de las flechas (relación *successor*), constituyen la realización de una actividad compleja. De modo que este *activity tree*, cuya raíz es una operación de refrentado ('Refrentado1'), presenta tres posibles realizaciones complejas, que se corresponden con los conjuntos de las realizaciones atómicas encerrados por las líneas de trazo discontinuo.

En la Tabla 4.2 se exponen de forma sintética las diferencias y relaciones entre el *occurrence tree* y el *activity tree*.

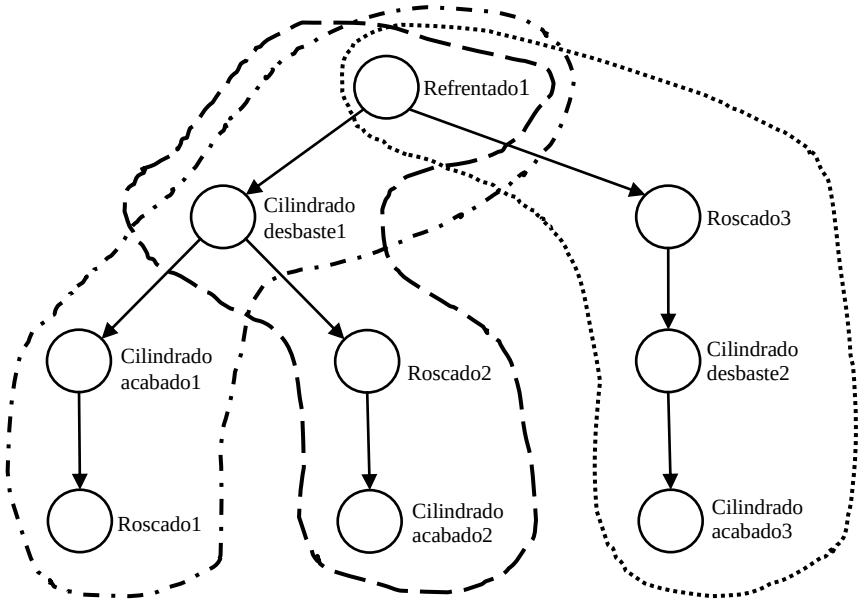


Figura 4.7. *Activity tree* para la subfase A del casquillo AB.

Es sobradamente conocida la importancia de establecer una secuencia, o al menos unas relaciones de precedencia entre las subactividades o tareas que componen cualquier actividad o proceso. Sin embargo, en ocasiones, debido al carácter de las subactividades y la compleja relación que existe entre ellas, incluyendo el solapamiento de las mismas, no puede atribuírseles una relación trivial de secuencia. Ahora bien, el cumplimiento de determinados hitos que marquen el inicio y el final de cada una de estas subactividades

permitiría establecer una relación de precedencia entre las mismas. De este modo, aunque varias de ellas puedan desarrollarse de forma simultánea o coexistir durante un periodo de tiempo determinado, existiría una forma de encadenarlas y de asociar la secuencia de las mismas. Para ello, se pueden utilizar las relaciones *root* y *leaf* del Outer Core de PSL (*Theory of Complex Activities*), que identifican el inicio y final de una actividad. Así, la expresión KIF (`root ?s ?a`) significa que la subactividad `?s` es el inicio o raíz del *activity tree* para la actividad `?a`. Mientras que (`leaf ?s ?a`) significa que la subactividad `?s` es el final, u hoja, del *activity tree* para la actividad `?a`.

Tabla 4.2. Diferencias y relaciones entre el occurrence tree y el activity tree.

	Occurrence tree	**Activity tree**
Teoría	*Theory of occurrence trees.*	*Theory of complex activities.*
Tipo de teoría	Extensión 'definitional' del núcleo externo de PSL.	Extensión 'definitional' del núcleo externo de PSL.
Significado	Todo lo que puede suceder durante todas las realizaciones posibles de una actividad.	Todo lo que puede suceder en una realización concreta de una actividad compleja.
Elementos	Realizaciones de actividades atómicas.	Realizaciones de actividades complejas.
Relación entre ambos	Los elementos del *activity tree* no son elementos legales del *occurrence tree*, pero el *activity tree* es un subconjunto del *occurrence tree*. Los *activity trees* son subárboles del *occurrence tree*, y las *branchs* (ramas) de un *activity tree* son las realizaciones de actividades complejas. En ese sentido, los *activity trees* son un microcosmos del *occurrence tree* en el cual consideramos que se despliega dicho *occurrence tree*.	

El hecho de que una subactividad `?s` sea inicio (*root*) de una actividad `?a` se traduce en la imposibilidad de que exista alguna subactividad *legal* en el contexto de la actividad `?a` que sea anterior a la subactividad `?s`. Análogamente, si una subactividad `?s` es final (*leaf*) de una actividad `?a`, entonces será imposible que exista alguna subactividad *legal* en el contexto de la actividad `?a` que sea posterior a la subactividad `?s`. En PSL se considera que una actividad es *legal* bajo determinadas restricciones cuando puede formar parte de las actividades presentes en el *occurrence tree* que cumplen dichas restricciones. Por ejemplo, si en el plan de proceso para la pieza 'casquillo AB', se establece que las operaciones primera y última sean respectivamente refrentado y roscado exterior, cada vez que se fabrique una de estas piezas deberá respetarse la secuencia de operaciones prevista que se inicia con el refrentado y finaliza con el mecanizado de la rosca exterior. En la primera y segunda línea de la expresión siguiente se asigna al refrentado y al roscado exterior el carácter de subactividades de la actividad 'mecanizar casquillo AB', mientras que en las dos líneas siguientes se establece que estas subactividades constituyen respectivamente el inicio y final de la actividad.

```
(subactivity ?refrentarCasquilloAB ?mecanizarCasquilloAB)
(subactivity ?roscarExtCasquilloAB ?mecanizarCasquilloAB)
(root ?refrentarCasquilloAB ?mecanizarCasquilloAB)
(leaf ?roscarExtCasquilloAB ?mecanizarCasquilloAB)
```

De forma análoga, haciendo uso de las relaciones *root_occ* y *leaf_occ* se puede establecer respectivamente el inicio y el final de una realización compleja:

```
(root_occ ?occ1RefrentarCasquilloAB ?occ1MecanizarCasquilloAB)
(leaf_occ ?occ1RoscarExtCasquilloAB ?occ1MecanizarCasquilloAB)
```

Aunque no se hayan definido el inicio y/o el final para una actividad, si es posible definir el inicio y/o el final para una o varias de las realizaciones de dicha actividad. Esto se traduce en la posibilidad de que las realizaciones de una misma actividad no tengan la misma estructura entre ellas, ni tampoco la misma estructura de la actividad a la que corresponden. Expresándolo en modo de restricciones, se diría que determinadas relaciones como *root* y *leaf* sirven para establecer restricciones a nivel de actividad, que deberán ser cumplidas por todas las realizaciones de dicha actividad. Mientras que existen otras relaciones como *root_occ* y *leaf_occ* para establecer restricciones aplicables a nivel de las realizaciones de la actividad, que serán diferentes a las restricciones de la propia actividad, aunque en cualquier caso deberán ser compatibles con ellas.

Para definir ordenación y precedencia entre las realizaciones de una misma realización compleja, se utilizan las relaciones *next_subocc* y *min_precedes* (*Theory of Complex Activities*). En concreto, la relación *next_subocc* permite expresar la secuencia precisa u ordenación total entre las realizaciones de actividades primitivas de una realización compleja. Por su parte, *min_precedes* permite expresar relaciones de precedencia u ordenación parcial (Bock y Gruninger 2005) entre las realizaciones de actividades primitivas de una realización compleja. La semántica de PSL impide que las relaciones *next_subocc* y *min_precedes* se puedan establecer entre las realizaciones de actividades que en un determinado contexto sean consideradas como actividades complejas.

La expresión siguiente, especifica la relación de precedencia entre las operaciones de desbaste y acabado que corresponden a las realizaciones de sendas subactividades de la actividad 'mecanizar casquillo AB':

```
(min_precedes ?occCilindrarDesbasteCasquilloAB
      ?occCilindrarAcabadoCasquilloAB ?mecanizarCasquilloAB)
```

Otra posibilidad sería la de realizar el acabado inmediatamente después de la operación de desbaste. Lo que se expresaría como:

```
(next_subocc ?occCilindrarDesbasteCasquilloAB
      ?occCilindrarAcabadoCasquilloAB mecanizarCasquilloAB)
```

Se puede comprobar que *next_subocc* implica *min_precedes*, pero no es cierta la implicación en sentido contrario: *min_precedes* no *implica next_subocc*.

También es posible establecer relaciones de precedencia entre realizaciones atómicas que forman parte de distintas realizaciones complejas. Por ejemplo, mediante las relaciones *earlier* y *precedes* (*Theory of Occurrence Trees*), que sirven para definir la secuencia entre los elementos del *occurrence tree*. La relación (`precedes ?occ1 ?occ2`) significa que las dos realizaciones de actividades (`?occ1` y `?occ2`) se encuentran en una rama del *occurrence tree* en la que no hay bifurcaciones, y por tanto, la realización de `?occ1`

conduce necesariamente a `?occ2`. La relación `(earlier ?occ1 ?occ2)` expresa una restricción menos severa, ya que solamente implica que `?occ1` es anterior en el tiempo a `?occ2`. Por tanto, representadas en un gráfico de tipo árbol, ambas realizaciones podrían estar en una rama del árbol en la que hay bifurcaciones, y si `?occ1` se sitúa aguas arriba de la bifurcación y `?occ2` se encuentra aguas abajo de la bifurcación, entonces `?occ1` no conduce necesariamente a `?occ2`.

En el *occurrence tree* de la Figura 4.8, las secuencias encerradas mediante una línea de trazo discontinuo cumplen la relación (`earlier ?refrentar ?roscar`), ya que en las tres secuencias 'Refrentar' está más próximo a la raíz del *occurrence tree* que 'Roscar'. Para dos de las secuencias también se cumple la relación (`precedes ?refrentar ?roscar`), ya que en ambos casos, las secuencias corresponden a ramas del *occurrence tree* en las que no hay bifurcaciones. Sin embargo, en la tercera secuencia, la situación de 'Refrentar' y 'Roscar' respecto a la bifurcación no garantiza el cumplimiento de la relación (`precedes ?refrentar ?roscar`). Con esto, se constata que *precedes* implica *earlier*, pero *earlier* no implica *precedes*.

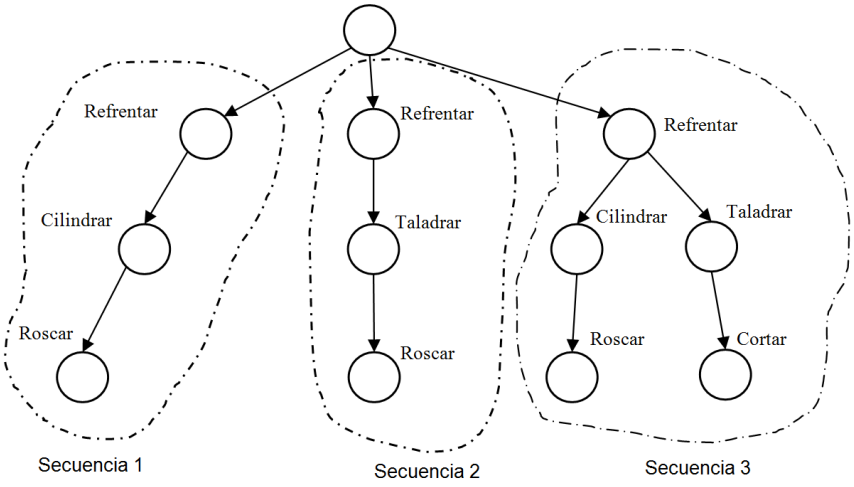


Figura 4.8. *Occurrence tree* con ejemplos de grupos de realizaciones que cumplen las relaciones earlier y/o precedes.

Después de analizar las actividades, sus realizaciones, y ciertos aspectos relacionados con la secuencia de las mismas, se revisarán algunas cuestiones relativas a los objetos, que son otra entidad fundamental en la descripción de procesos. Las intuiciones en las que se basa el núcleo de PSL conducen a una definición de los objetos por exclusión: los objetos son todas las entidades de PSL distintas de las actividades, las realizaciones de las actividades y los instantes de tiempo o *timepoints*. Por tanto, puede preverse una gran heterogeneidad entre los elementos de este tipo de entidades. Para PSL, los objetos son todo aquello que interviene o que está relacionado con el desarrollo de las actividades y que tiene un periodo de existencia o validez comprendido entre dos instantes de tiempo. Entre los objetos de PSL se encuentran los recursos que intervienen en una actividad (normalmente objetos

físicos), y los estados (objetos virtuales). En PSL, un recurso es cualquier objeto que es requerido por alguna actividad (Pouchard et al. 2005), y se define formalmente como:

```
(defrelation resource (?r) :=
(and  (object ?r)
(exists (?a)
(requires ?a ?r))))
```

Una definición que se concreta y restringe con varios axiomas de la extensión *Resource Theories* (Outer Core), como el que liga los recursos a las actividades atómicas, y según el cual una actividad `?a` requiere un recurso `?r (requires ?a ?r)` si y solo si existe una subactividad atómica que requiere dicho recurso. Otros axiomas ponen de manifiesto que la caracterización de los recursos en PSL se centra en sus *capacities* (o capacidad productiva relacionada con la programación o *scheduling*). Por ejemplo, la axiomatización del concepto de recurso en PSL establece que un recurso `?r` está disponible para una actividad `?a` (`available ?r ?a`) si y solo si la cantidad del recurso `?q1` para la actividad (*the quantity of the resource point*) es superior a la suma de la demanda para el recurso por parte de esta actividad `?q2` y la demanda total resultante del resto de actividades para el recurso `?q3`. Lo que se representa mediante la expresión `(greater ?q1 (plus ?q2 ?q3))`.

La Figura 4.9 facilita la comprensión de las relaciones *prior* y *holds* (*Theory of Discrete States*), que se establecen entre las realizaciones de las actividades y los estados (*states*), que son una especialización de la entidad *object*. Las relaciones *prior* y *holds*, que no forman parte de PSL-Core, permiten especificar respectivamente los estados anterior y posterior a la realización (*occurrence*, o *activity_occurrence*) de una actividad.

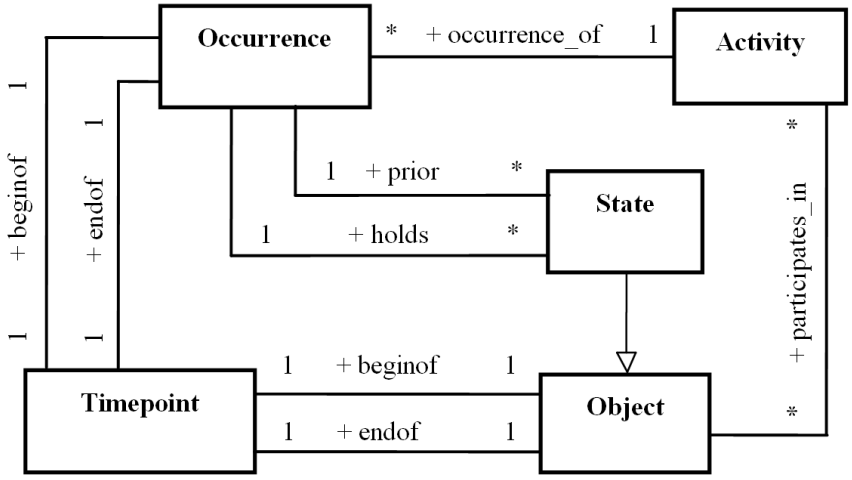


Figura 4.9. Representación de algunas relaciones y funciones básicas de PSL (Solano, Rosado y Romero 2009).

Los estados representan las propiedades y las relaciones del dominio que pueden cambiar como resultado de las realizaciones de las actividades (PSL 2010). Como los estados son objetos, pueden participar en un proceso o actividad, y la participación de un estado en un

proceso puede ser interpretada en el sentido de que el proceso tiene conocimiento, o al menos tiene la intuición (*believing*) del estado que presenta (*holds*) el mundo o entorno en el que se desarrolla el proceso (Bock y Gruninger 2004b). La utilización de las relaciones *prior* y *holds* se muestra en las expresiones: (`holds ?f ?occ`), que significa que el estado `?f` es cierto tras la realización de la actividad `?occ`; y `(prior ?f ?occ)`, que significa que el estado `?f` es cierto antes de la realización de la actividad `?occ` (PSL 2010). Por ejemplo, si el mecanizado de una pieza determinada se efectúa en una máquina de control numérico, una de las actividades de la planificación del proceso consistirá en la generación de dicho programa, y la existencia de un programa de control numérico para la fabricación de la pieza es un 'estado del mundo' previo (*prior*) a la realización del mecanizado de la pieza.

4.3. Recursos expresivos para representar la flexibilidad del plan de procesos

Anteriormente, se han dado ejemplos de algunas de las capacidades expresivas de PSL a través del estudio de diversos casos y situaciones relacionadas con el plan de proceso para la fabricación de una pieza concreta. Un plan de proceso, entendido como el conjunto de las posibles secuencias de las actividades a realizar, está representado por un *activity tree*, donde cada uno de sus elementos corresponde a una variante o alternativa del plan. A continuación se analizará la flexibilidad en el plan de proceso y algunas de las formas de representarla con PSL.

La expresión siguiente, propone distintas alternativas para fabricar la pieza 'casquillo AB' mediante una combinación de procesos de fabricación que se presentan como subactividades de la actividad 'fabricar AB'.

```
(activity ?fabricarAB)
       (subactivity ?forjarAB ?fabricarAB)
       (subactivity ?moldearAB ?fabricarAB)
       (subactivity ?mecanizarAB ?fabricarAB)
```

Estas subactividades, según la semántica de PSL, podrán o no ser realizadas en el desarrollo de la actividad correspondiente. Con ellas, aunque se restringe el espacio de soluciones posibles, se traslada una cierta indefinición o ambigüedad en la representación del plan de proceso. A continuación, se muestra un ejemplo sencillo en el que utilizando algunas de las relaciones de PSL expuestas previamente, se consigue eliminar esta ambigüedad y representar un plan de proceso perfectamente definido a este nivel de detalle. En él, la fabricación se inicia con la obtención de una preforma moldeada, que posteriormente se forja, y finalmente es mecanizada:

```
(forall (?occFabricarAB)
  (implies
  (occurrence_of ?occFabricarAB ?fabricarAB)
        (and (root_occ ?occMoldearAB ?occFabricarAB)
             (next_subocc ?occMoldearAB ?occForjarAB fabricarAB)
              (next_subocc ?occForjarAB ?occMecanizarAB fabricarAB)
              (leaf_occ ?occMecanizarAB ?occFabricarAB))))
```

Según Benjaafar y Ramakrishnan (1996), uno de los posibles tipos de flexibilidad en la fabricación, obedece a la existencia de restricciones que condicionan la posibilidad de que se puedan llevar a cabo las diferentes alternativas, y en su caso, el orden en el que dichas alternativas pueden realizarse. En ocasiones, el plan de proceso que debe representarse está restringido hasta el punto de no contemplar ninguna alternativa, pudiendo decirse en este caso que se trata de un plan de proceso lineal. Para ello, PSL dispone de las ya conocidas relaciones *next_subocc*, *root_occ* y *leaf_occ* que establecen una secuencia precisa entre las realizaciones de las subactividades de una actividad. Así, la expresión siguiente corresponde a un plan de proceso lineal.

```
(root_occ ?occAmarre_AB ?occTornearAB)
(next_subocc ?occAmarre_AB ?occMandrinar tornearAB)
(next_subocc ?occMandrinar ?occRefrentar tornearAB)
(next_subocc ?occRefrentar ?occRoscar tornearAB)
(next_subocc ?occRoscar ?occDesamarre_AB tornearAB)
(leaf_occ ?occDesamarre_AB ?occTornearAB)
```

En esta expresión, la definición del inicio y del final de la subfase es aparentemente redundante. Por ejemplo, respecto al final de la subfase, en la antepenúltima línea de la expresión se indica que después de la realización del refrentado se realizará el roscado. Si se suprime la penúltima línea, en la que se indica que tras el roscado se desembrida la pieza, el significado de la expresión parece no verse alterado, ya que en la última línea se indica que el desamarre será la operación que cierre la realización de `?occTornearAB`, y por tanto, el desamarre será posterior a la operación de roscado interior. No obstante, para cerrar el plan de proceso (evitar cualquier modificación en el mismo) es necesaria la penúltima línea de la expresión, ya que ésta impide que pueda realizarse alguna subactividad (operación) no declarada, pero posible en el dominio del proceso de torneado entre las operaciones de roscado y desamarre. Dicho de otro modo, sin la presencia de esa penúltima línea, sería legal realizar una actividad posterior al roscado, siempre que ésta se desarrollara con anterioridad al desamarre, y por tanto, el plan de proceso descrito admitiría alternativas que aunque no declaradas explícitamente serían factibles y conducirían a un plan de proceso que no sería lineal en sentido estricto.

La flexibilidad de un plan de proceso también se puede manifestar a través de la selección de recursos entre un conjunto de posibles opciones, sin afectar a la secuencia ni a las operaciones del proceso (Benjaafar y Ramakrishnan 1996). En el ejemplo (expresión) siguiente, se muestra la posibilidad de expresar la flexibilidad en los recursos sin necesidad de recurrir a la teoría del Outer Core *Resource Theories*, cuyo alcance se centra en la definición y axiomatización de los recursos y en el tratamiento global de conjuntos de recursos. En dicha expresión se define una operación (*activity*) de roscado en la que interviene el recurso máquina `?m` y el recurso herramienta `?h`. Siendo que para ambos existe la posibilidad de seleccionar entre dos alternativas: máquina_1 o máquina_2, y herramienta_1 o herramienta_2 respectivamente.

```
forall (?a ?m ?h)
       (implies (= ?a roscado(?m ?h))
              (and   (activity ?a)
                     (or (maquina1 ?m)(maquina2 ?m))
                     (or (herramienta1 ?h)(herramienta2 ?h)))))
```

Tal vez, la opción más acertada para gestionar la planificación del proceso sea representar el plan conforme se ve durante la planificación: como una secuencia de operaciones, preferiblemente asociadas a elementos característicos de mecanizado. Para ello, además de establecer las operaciones se deben definir las restricciones que afectan a cada operación individual y que permiten establecer la secuencia y en su caso, la posterior agrupación en subfases y fases. Partiendo de este planteamiento, se podrían definir operaciones de cambio de máquina y de cambio de embridaje, al mismo nivel de las operaciones de mecanizado y de las operaciones complementarias (preparación, inspección, etc.), que cuando se producen marcan el inicio de nuevas fases y subfases respectivamente. De este modo, una vez establecidas las restricciones para cada actividad, bastaría identificar las actividades ya realizadas y/o los estados alcanzados, para determinar la factibilidad de cada una de las operaciones. Tras la realización de una actividad se produce un cambio en el estado de uno o más objetos implicados en dicha actividad. En el ámbito de PSL se considera que los estados son discretos, lo que significa que el cambio en los mismos solamente puede realizarse a la finalización de la realización de una actividad y nunca durante el desarrollo de la misma. Esto se traduce en la existencia de una relación entre estados y realizaciones de actividades. Por ejemplo, después de realizar la operación de refrentado, la pieza presenta el estado de 'pieza refrentada' y este estado solo se alcanza como consecuencia de la operación de refrentado. No obstante, hay una característica de las actividades (y de los estados) que condiciona la relación entre ellos. En un determinado dominio de interpretación, existen actividades cuyos resultados o estados son permanentes o no reversibles, mientras que los estados que se alcanzan con otras actividades pueden modificarse o revertirse. Utilizando el ejemplo anterior, podría afirmarse que el estado de 'pieza refrentada' es permanente, o al menos que no existe modo de revertirlo haciendo que la pieza adquiera el mismo estado que tenía antes de ser refrentada. Sin embargo, hay otras actividades como el amarre o desamarre de la pieza, que conducen a 'estados reversibles', o estados que pueden ser modificados por la realización de actividades posteriores.

En este contexto, se plantea la posibilidad de utilizar bien la realización de actividades, o bien los estados como antecedentes o precondiciones exigidas para que una actividad pueda desarrollarse, pero asumiendo que la realización de una actividad será equivalente al estado que se alcanza con ella solamente en los casos en que dicha actividad (o dicho estado) sean 'no reversibles'. Por ejemplo, si una precondición para refrentar la pieza por un extremo (el extremo A) es que ésta esté sujeta por el otro extremo (extremo B), no basta con asegurar que antes del refrentado se haya realizado el amarre, ya que esta operación es 'reversible' y después de ella, podría haberse procedido al desamarre; y tampoco será siempre viable o conveniente exigir que inmediatamente antes de refrentar se proceda a amarrar la pieza. Pero si bastaría con establecer la precondición de 'pieza amarrada' antes de refrentar. En el otro supuesto, para actividades 'no reversibles', ambas precondiciones serían equivalentes. Por ejemplo, para la operación de acabado es equivalente afirmar que antes de dicha operación debe haberse realizado el desbaste o comprobar que se cumple el estado de 'pieza desbastada' con anterioridad a realizar el acabado.

El caso anterior, es decir, la restricción de precedencia entre las operaciones de desbaste y acabado se puede intentar representar mediante otros recursos expresivos de PSL, como la relación *poss* incluida en la *Theory of Occurrence Trees*, mediante la cual se indica la posibilidad de realizar una actividad tras la realización de otra actividad. Según esto, la expresión (`poss ?a ?occ`) significa que la actividad `?a` podrá realizarse (será posible)

una vez que se haya llevado a cabo la realización concreta de otra actividad representada por `?occ`.

Finalmente, cabe destacar la capacidad de ciertos lenguajes de especificación del comportamiento, incluido PSL, para soportar decisiones en curso de ejecución que definen cómo se desarrollará una tarea (Bock y Gruninger 2005). Esta flexibilidad en la ejecución del plan de proceso viene determinada por un tipo de objetos, los estados, que se utilizan en la selección de alternativas. Normalmente, los estados solo serán conocidos en curso de ejecución, por lo tanto, las variaciones en el proceso que se derivan de dichas alternativas solamente pueden elegirse durante la ejecución del proceso. Por ejemplo, la relación *holds* de PSL permite especificar que un estado es cierto tras la realización de una actividad y puede determinar la realización de la actividad siguiente. En la expresión siguiente, se utiliza *holds* para representar el estado que presenta la pieza después de la operación de desbaste que habilita la realización de actividades posteriores como el acabado.

```
(state ?piezaDesbastada)
(activity ?desbastar)
(occurrence_of ?occDesbastar ?desbastar)
(holds ?piezaDesbastada ?occDesbastar)
```

5. Perspectiva de los recursos

La sección anterior se ha centrado en el análisis de los procesos entendidos como entidades que suceden en el tiempo. Para lo cual la ontología PSL se ha manifestado como una herramienta sumamente eficaz. No obstante, todos los procesos necesitan la intervención o participación de recursos. Concretamente, el uso eficiente de los recursos de fabricación es una de las metas principales de la gestión industrial y de la gestión de costes en particular, y para ello se requiere una información completa, precisa y actualizada de los mismos. Por otra parte, muchas funciones de la empresa, y por consiguiente, también diferentes sistemas de tecnología de la información, tratan con los recursos de fabricación. De hecho, una de las tareas primordiales de la planificación de procesos es la definición, configuración y asignación de recursos. Recursos que pertenecen a la clase primitiva *objects* de PSL. No obstante, la descripción y estructuración de los objetos no está incluida en PSL. Para PSL, los objetos simplemente existen, asumiendo que su descripción se realiza mediante métodos ajenos a su propio ámbito (Lubell 2003). Por tanto, se requieren otras herramientas, como las incluidas en la iniciativa MANDATE, capaces de modelar y capturar (conceptualizar) todos los aspectos de los recursos que son necesarios para las tareas de representación y planificación de procesos.

El modelo de información de recursos descrito en MANDATE se ha estructurado de forma modular, con la entidad *resource* como elemento central. Cualquier descripción, clasificación o detalle de las características de un recurso está relacionado con dicha entidad. La Figura 4.10 muestra como se estructuran y relacionan el resto de entidades del modelo, que se agrupan en unidades lógicas correspondientes a los siguientes aspectos de los recursos: jerarquía, estructura de las características, estado, definición de las vistas, definición de las características, y configuración. A partir de este modelo conceptual se define el modelo formal mediante EXPRESS (Figura 4.11), algunas de cuyas entidades y relaciones se describen y/o comentan a continuación.

La entidad *resource* tiene cuatro especializaciones: *resource_group*, *generic_resource*, *specific_resource* e *individual_resource*. Un *resource_group* es un tipo de recurso que incluye otros recursos o grupos específicos de recursos. A su vez, estos grupos de recursos (*resource_groups*) pueden estar formados por otros grupos de recursos. Es decir, como se aprecia en la Figura 4.11, cualquier grupo de recursos puede tener una relación de tipo *super_grupo* con un conjunto de grupos de recursos. El *super_grupo* de un recurso genérico (*generic_resource*) también es un *resource_group*. Los recursos genéricos son tipos de recursos caracterizados por una definición completa de todos los atributos relacionados, pero sin el requisito de vinculación a los valores específicos de la empresa real. Según el modelo (Figura 4.11), todo *specific_resource* pertenece a un recurso genérico, y cada recurso específico se obtiene a partir de un recurso genérico mediante la asignación de valores reales a todos los atributos, excepto aquellos relacionados con las tareas de programación (scheduling), para las cuales se precisa información dinámica. Finalmente, la entidad *individual_resource*, en cuya representación se establece una relación con la información dinámica, permite la descripción de los recursos de fabricación físicos de un sistema de producción. Estos recursos individuales heredan todos los valores de los atributos correspondientes al recurso específico al que pertenecen.

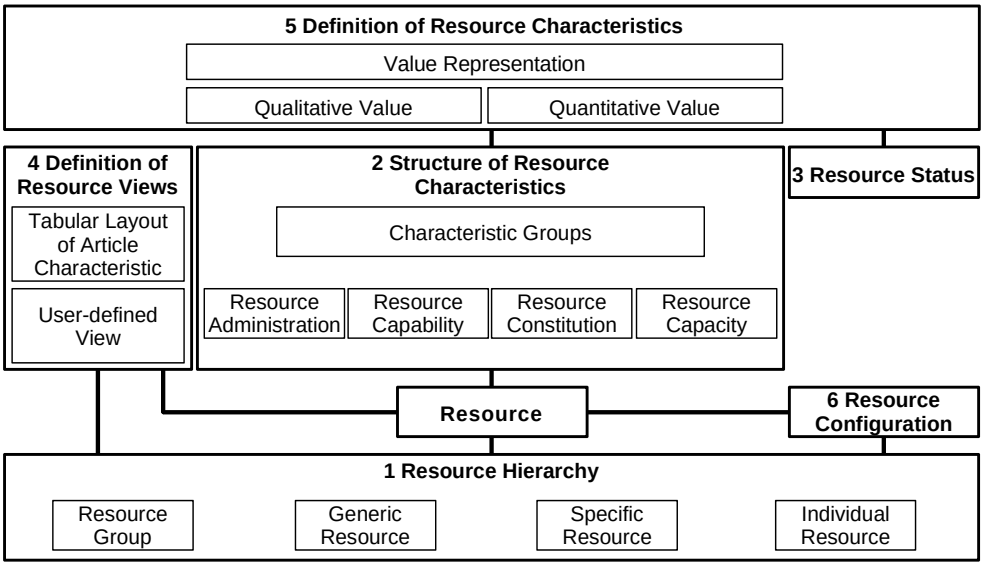


Figura 4.10. Módulos lógicos del modelo de información de recursos (ISO 2005).

La representación de una jerarquía de recursos se puede obtener a partir de las instancias de la entidad *resource_group*. El modelo permite representar una jerarquía de recursos compuesta de múltiples capas, a través de la utilización recursiva de una relación de tipo *super_grupo* sobre la entidad *resource_group*. Cada recurso genérico está incluido en un *resource_group* y dichos recursos genéricos deben ser definidos de acuerdo a las restricciones establecidas por los usuarios o las necesidades de generalidad/especificidad relativas al propósito de su reutilización. Si se pretende disponer de una representación más general o reducir el número de niveles en la jerarquía de recursos, la red de recursos puede estar basada en estos recursos genéricos. Por otra parte, como se ha indicado anteriormente,

a partir de los recursos genéricos se obtienen los recursos específicos, y a partir de éstos se obtienen los recursos individuales.

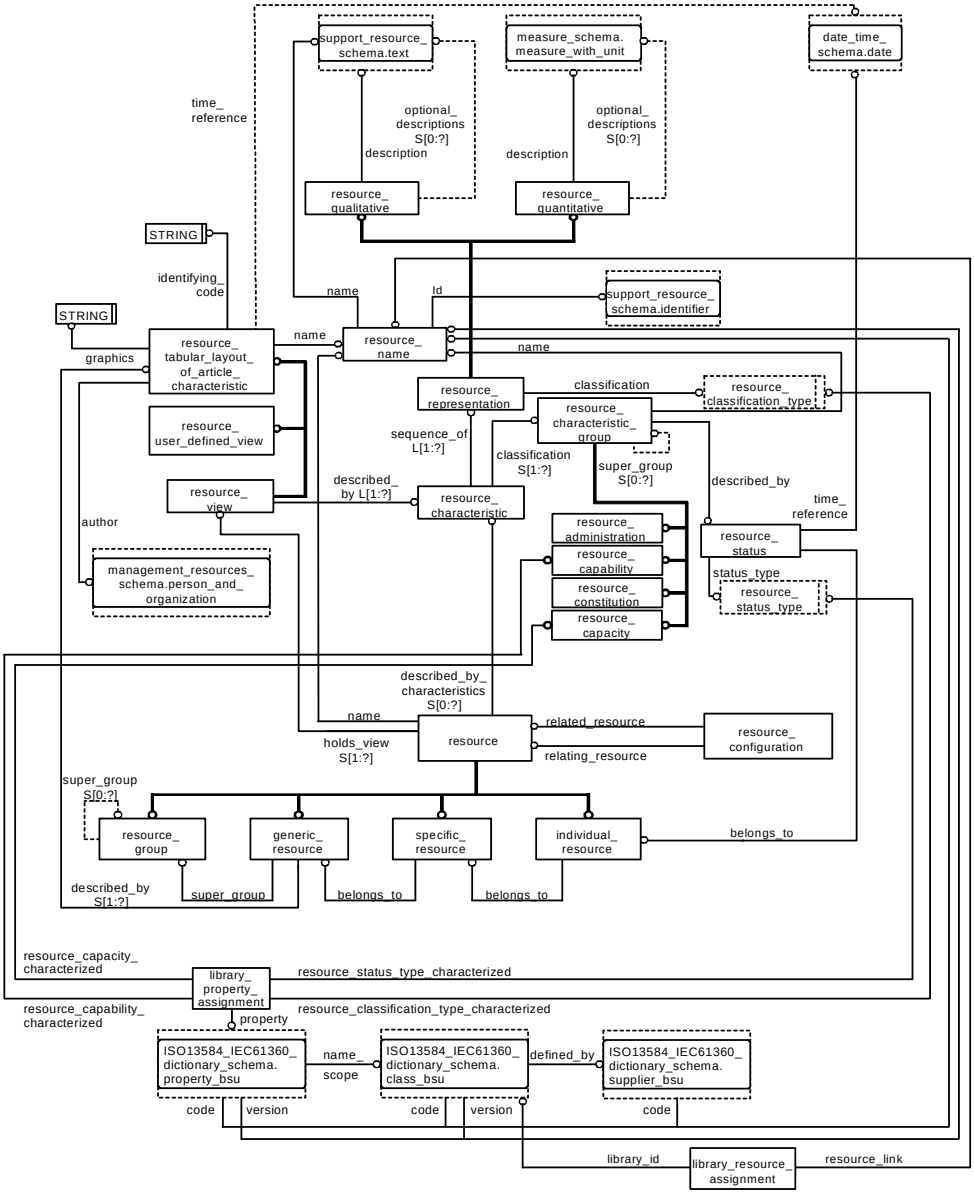


Figura 4.11. Resource_usage_management schema – EXPRESS-G diagram (ISO 2005).

Las características del recurso (*resource_characteristics*) incluyen información acerca del estado real de un recurso (*resource_status*) susceptible de ser utilizada para la gestión de su utilización. La clasificación de atributos del modelo establece los siguientes grupos de

características de recursos (*resource_characteristic_group)*: *resource_administration*, *resource_capability*, *resource_constitution*, y *resource_capacity* (Figura 4.10 y Figura 4.11). La entidad *resource_administration* representa características que describen información administrativa acerca de los recursos de fabricación. Por ejemplo, toda la información que se utiliza para la gestión de los recursos a nivel de empresa: características de disposición (*disposition characteristics*), como la aprobación por parte de una persona autorizada que se exige previamente a la utilización de una máquina compleja o potencialmente peligrosa; la carga de trabajo real; o las características económicas y los costes de producción. La entidad *resource_capability* define características que especifican aspectos funcionales de los recursos de fabricación. Concretamente, incluye la especificación de las tareas de la actividad que un recurso de fabricación puede ejecutar. La entidad *resource_constitution* representa características que describen el estado real de los recursos de fabricación. Por ejemplo, las características de los recursos relativas a su geometría, tolerancia, material y orientación de sus superficies. Por último, la entidad *resource_capacity* describe la capacidad productiva (*capacity*) de los recursos de fabricación. Es decir, las características relacionadas con las tareas de programación (*scheduling*). Por ejemplo, el tiempo máximo para completar una tarea o la potencia disponible de una máquina herramienta son una medida de su *capacity*.

La entidad *resource_status* proporciona un mecanismo para el seguimiento de la información relativa a un recurso individual, en un instante de tiempo en el que dicho recurso interviene en un proceso, y en relación a dicho proceso. Los estados de la información acerca del recurso pueden corresponder a uno de los tipos siguientes: *requested*, o petición de información realizada; *registered*, si la información está almacenada pero no ha sido analizada; *analysed*, cuando la información está preparada para desarrollar medidas para la optimización del proceso; *dispatched*, si la información ha sido entregada/enviada; y *noticed*, en el caso de que las medidas de optimización hayan sido realizadas.

Una vista del recurso (*resource_view)* es una lista abierta de características del recurso. Las *resource_views* pueden ser de dos tipos: *resource_user_defined_view* y *resource_tabular_layout_of_article_characteristic*. La primera es una combinación de características de recursos en un contexto orientado a una aplicación. Por ejemplo, en el contexto de la selección de recursos de fabricación para un área específica de la planta, la vista *resource_user_defined_view* podría incluir entre otras características, la localización de los recursos y los productos en cuya fabricación interviene cada recurso. Mientras que la segunda es una vista que se utiliza para combinar y seleccionar objetos físicos y abstractos con características similares (ver *Tabular Layout of Article Characteristic –TLAC–* en Figura 4.12).

Los valores físicos, cualitativos o cuantitativos de las características de los recursos de fabricación se representan mediante la entidad *resource_representation* (Figura 4.11). Una característica de un recurso de fabricación puede tener valores cualitativos, como por ejemplo, su habilidad para alimentar las piezas automáticamente, o valores cuantitativos, como las dimensiones de las piezas que el recurso puede mecanizar.

La entidad *resource_configuration* describe la configuración de un recurso para una tarea de fabricación específica, donde dicha configuración limita la utilización del recurso. Por ejemplo, un centro de mecanizado puede incluir un grupo de herramientas específicas para

el tipo de máquina en cuestión. Por una parte, la máquina estará limitada por el tipo de herramientas que puede utilizar, y por otra parte, las herramientas se pueden utilizar únicamente en determinados tipos de máquinas, lo que también limita el uso de éstas.

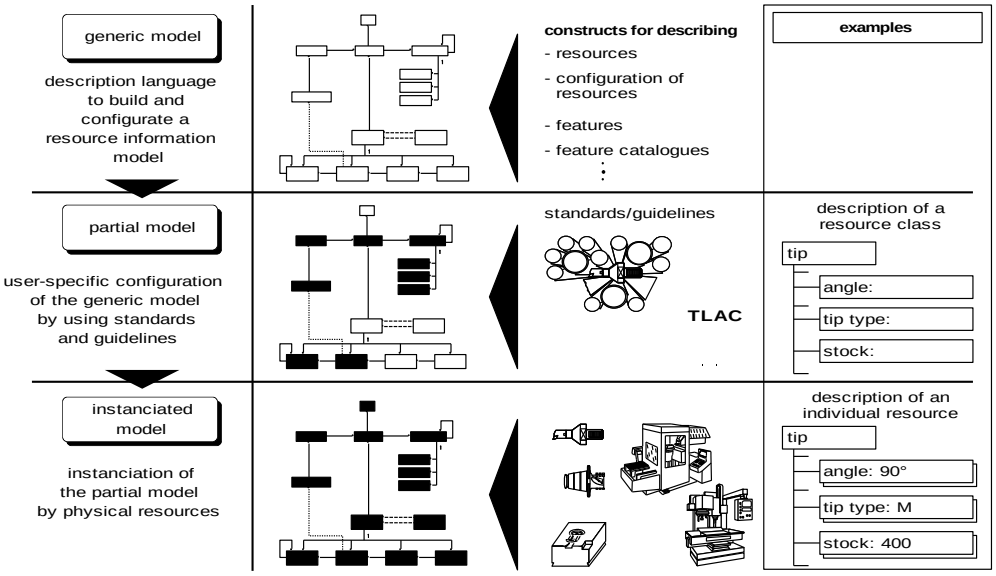


Figura 4.12. Niveles de instanciación del modelo de información conceptual (ISO 2005).

La representación de la jerarquía de recursos y la definición de una *resource_view*, de las que se habló anteriormente, constituyen el primer nivel de instanciación de un modelo genérico, del cual deriva el modelo parcial que da respuesta a las necesidades específicas del usuario. Para ello, se realiza la instanciación de los *resource_groups* y *generic_resources*, incluyendo la definición de *resource_views*. Posteriormente, en base a este modelo parcial, se efectúa la instanciación efectiva del modelo con datos de los recursos físicos (*resource_representation*). En la Figura 4.12 se muestran estos tres niveles de instanciación. En ella, el contenido de una TLAC (*Tabular Layout of Article Characteristic*), que se sitúa en el segundo nivel de instanciación, o modelo parcial, podría presentarse como una tabla en la que se recogen las características de los recursos.

A pesar del incremento experimentado en la automatización del trabajo, las personas continúan vinculadas a, y son esenciales en la mayoría de los sistemas productivos y de negocio. Particularmente, su presencia sigue siendo necesaria en el diseño y gestión de sistemas hombre-máquina. De hecho, es habitual considerar una relación de interdependencia entre el hombre y la máquina, siendo inviable que el trabajo se realice de forma efectiva o continuada sin la participación de ambos. En relación a esto, debe considerarse que las personas, al igual que otros recursos físicos (p.e. máquinas) pueden ser formalizadas según el modelo de información de recursos descrito en esta sección, contemplando sus habilidades, capacidades y limitaciones.

6. Referencias

Benjaafar, S. y R. Ramakrishnan. 1996. "Modeling, Measurement and Evaluation of Sequencing Flexibility in Manufacturing Systems." *International Journal of Production Research* 34(5): 1195–1220.

Bannon, L. J. 1991. "From Human Factors to Human Actors: The Role of Psychology and Human-Computer Interaction Studies in System Design." *In Design at Work: Cooperative Design of Computer Systems*, edited by J. a. M. K. Greenbaum, 25–44. Hillsdale: Lawrence Erlbaum.

Bock, C. 2006. Interprocess Communication in the Process Specification Language. NIST Internal Report (NISTIR) 7348. National Institute of Standards and Technology.

Bock, C. y M. Gruninger. 2004a. Inputs and Outputs in the Process Specification Language. NIST Internal Report (NISTIR) 7152. National Institute of Standards and Technology.

Bock, C. y M. Gruninger. 2004b. Messaging in the Process Specification Language. NIST Internal Report (NISTIR) 7258. National Institute of Standards and Technology.

Bock, C. y M. Gruninger. 2005. "PSL: A Semantic Domain for Flow Models." *Journal of Software and Systems Modeling* 4 (2): 209–231. doi: 10.1007/s10270-004-0066-x.

Ferrario, R. y A. Oltramari. 2004. "Towards a Computational Ontology of Mind." Paper presented at the 3rd International Conference FOIS (Formal Ontology in Information Systems), Turin, November 4–6.

Fox, M. S. y M. Gruninger. 1998. "Enterprise modeling." *AI Magazine* 19 (3): 109–121.

Gruninger, M. y C. Menzel. 2003. "The Process Specification Language (PSL) Theory and Applications." *AI Magazine* 24 (3): 63–74.

Gruninger, M. y M. S. Fox. 1995. "Methodology for the Design and Evaluation of Ontologies." Paper presented at the Workshop on Basic Ontological Issues in Knowledge Sharing, Montreal, August 19–20.

ISO (International Organization for Standardization). 2004. *Industrial Automation Systems and Integration – Industrial Manufacturing Management Data. Part 1, General Overview*. ISO 15531–1. New York: American National Standards Institute.

ISO (International Organization for Standardization). 2005. *Industrial Automation System and Integration – Industrial Manufacturing Management Data: Resources Usage Management. Part 32, Conceptual Model for Resources Usage Management Data*. ISO 15531–32. New York: American National Standards Institute.

ISO/CD (International Organization for Standardization). 1997. *Industrial Automation System and Integration – Industrial Manufacturing Management Data: Manufacturing Flow Management Data. Part 41, Overview and Fundamental Principles*. ISO 15531–41. New York: American National Standards Institute.

Kethers, S. 2000. "Multi-Perspective Modeling and Analysis of Cooperation Processes." PhD diss., RWTH Aachen.

Kuutti, K. 1995. "Activity Theory as a Potencial Framework for Human-Computer Interaction Research." In *Context and Consciousness: Activity and Human Computer Interaction*, edited by B. Nardi, 17–44. Cambridge: MIT Press.

Leontjev, A. N. 1989. "The Problem of Activity in the History of Soviet Psychology." *Soviet Psychology* 27 (1): 22–39.

Lubell, J. 2003. "XML Representation of Process Descriptions." In *Process Descriptions of Professional XML Meta Data*. Wrox Press.

Masolo, C., S. Borgo, A. Gangemi, N. Guarino y A. Oltramari. 2003. *WonderWeb Deliverable D18. Ontology Library (final)*. Laboratory for Applied Ontology ISTC CNR.

Pouchard, L. C., A. F. Cutting-Decelle, J. J. Michel y M. Gruninger. 2005. "ISO 18629 PSL: A Standardised Language for Specifying and Exchanging Process Information." Proceedings of the 16th International Federation of Automatic Control (IFAC) World Congress: 4–8.

PSL. 2010. National Institute of Standards and Technology. Consultada en Diciembre.

http://www.mel.nist.gov/psl/

Solano, L., P. Rosado y F. Romero. 2009. "PSL Ontology as a Link between Manufacturing Planning and Execution." Paper presented at the 20th International DAAAM Symposium, Vienna, November, 25–28.

Wooldridge, M. 2000. "Reasoning about Rational Agents." Cambridge: MIT Press.

Capítulo 5. ONTOLOGÍA PARA EL DESARROLLO COLABORATIVO DE PRODUCTOS Y PROCESOS CENTRADA EN CAPACIDADES DE LOS RECURSOS (ONTOLOGÍA PPDRC)

1. Introducción

En este capítulo se incluye la versión original del artículo *Knowledge Representation for Product and Processes Development Planning in Collaborative Environments*, publicado en 2014 por la revista *International Journal of Computer Integrated Manufacturing*, cuya versión *online* está disponible desde el 10 de septiembre de 2013 (doi: 10.1080/0951192X.2013.834480).

En el artículo se describen los antecedentes que justifican la propuesta, junto con los conceptos, entidades y predicados fundamentales de la ontología PPDRC (ontology for Product and Processes collaborative Development focused on Resource Capabilities), que se presentan más exhaustiva y detalladamente en el anexo 6. La ontología PPDRC se ha diseñado para soportar cualquier actividad de planificación de procesos involucrada en el proceso de desarrollo integrado de nuevos productos en entornos colaborativos desplegados en el contexto de una empresa OKP virtual, dónde la comunicación y cooperación entre diferentes actores es posible gracias a la alta conectividad que ofrecen las tecnologías Web. Como se ha visto en capítulos anteriores, la implementación de propuestas ontológicas mediante aplicaciones basadas en la Web es una solución ampliamente aceptada para dar respuesta a las necesidades de representación, transmisión y uso compartido de información y conocimiento. En particular, para el desarrollo de la ontología PPDRC se han utilizado los lenguajes OWL (Ontology Web Language) y SWRL (Semantic Web Rule Language), definidos por el consorcio W3C (World Wide Web Consortium), y la herramienta de edición *Protégé*. En el caso de las actividades de planificación de procesos, las necesidades expuestas son particularmente importantes, tanto por su complejidad, ya que están sujetas a realimentaciones y adaptaciones continuas, como por la naturaleza colaborativa del proceso y el carácter agentivo de los actores involucrados.

La ontología PPDRC contempla la representación de todas las situaciones posibles y de todos los entes que forman parte de un escenario de co-planificación de procesos asistida por ordenador (Co-CAPP), los cuales se pueden agrupar en tres categorías. A la primera pertenecen los elementos que intervienen en el proceso de planificación, descrito en el capítulo 2: (a1) los procesos mentales, intelectuales y cognitivos, junto a cualquiera de sus realizaciones; (b1) los agentes, que pueden participar de forma concurrente y simultánea en dichos procesos; (c1) los datos empleados en la toma de decisiones; y (d1) los conceptos, como las aptitudes y habilidades que caracterizan a las entidades y que influyen en la toma de decisiones. A la segunda categoría pertenecen todos los entes que constituyen y/o intervienen en la ejecución del plan generado: (a2) las actividades físicas y cualquiera de sus realizaciones; (b2) los medios materiales, considerando su estado, morfología, ubicación espacial, y sus relaciones de asociación, agregación, dependencia, modificación y reconfiguración; y (c2) los datos que facilitan la caracterización de las entidades. Finalmente, en la tercera se encuentran aquellos entes que dan soporte a las dos categorías anteriores: (a3) el almacenamiento y recuperación de los datos, incluyendo su evolución temporal y relación con las entidades y circunstancias en las que se originan sus valores; (b3) la información y el soporte físico que la sustenta; (c3) la dimensión social de las entidades, que permite que éstas sean percibidas homogéneamente por todos los miembros de un colectivo, junto con la visión individual y sesgada que pueden presentar dichas entidades al ser consideradas aisladamente; y (d3) las entidades en las que coexiste lo material con lo inmaterial o que pueden verse bajo ambos puntos de vista.

Como ya se ha indicado en el capítulo anterior, en el desarrollo de PPDRC se incorporan los conceptos ontológicos de otras propuestas específicas de los ámbitos del proceso y de los recursos, que se integran en la ontología de base DOLCE+DnS Ultralite (DUL). Entre ellos, los procedentes de PSL y los que emanan de MANDATE. La utilización de la ontología DUL facilita la interoperabilidad semántica de PPDRC con ontologías de otros ámbitos. De hecho, las tres entidades de primer nivel de PPDRC (*Object*, *ActivityOccurrence* y *Region*) pueden considerarse especializaciones de *Endurant*, *Perdurant* y *Quality*, tres de las cuatro entidades de primer nivel de DUL. Análogamente, la interoperabilidad de PPDRC con otras ontologías del dominio de los procesos se facilita con la incorporación directa de las entidades *Activity* y *ActivityOccurrence* de PSL, y con la incorporación de ciertas propiedades de las entidades *Object* y *Timepoint* de PSL en las entidades Resource y Region de PPDRC, respectivamente.

Como la propia denominación deja entrever, la ontología PPDRC centra su foco en una gestión de los recursos basada en sus capacidades (*capabilities*), una aportación original que no ha sido encontrada previamente en la bibliografía. Para PPDRC, el recurso es una entidad que participa en la realización de las actividades y que está caracterizada por sus capacidades ejecutando un tipo de actividad y por un rol, que manifiesta dicha capacidad y que está presente en la realización de actividades de este tipo. Por lo tanto, la capacidad de un recurso establece el tipo de actividades que éste puede realizar y, en los casos en que está cuantificada, expresa un determinado nivel de desempeño en la realización de la actividad. Además, como se ha indicado antes, también incluye otros aspectos relativos a las actividades que configuran el plan y a su ejecución. Entre éstos son de particular interés los vinculados con la representación de alternativas, a través de planes no lineales establecidos bajo distintos enfoques (generativo o variante) y estrategias (hacia adelante o hacia atrás), y a diferentes niveles de agregación. En este sentido, la ontología PPDRC está en la línea de las ontologías integradas de productos, procesos y recursos que definen una

semántica compartida para establecer el nexo entre las actividades de diseño y desarrollo del producto y la programación y control de la producción en el ámbito de la planificación de procesos colaborativa.

Otra característica destacable de la ontología PPDRC es que se sitúa en el dominio de la planificación de procesos de cualquier tipo, como puede ser el proceso de fabricación de un producto o un proceso mucho más general como es el propio desarrollo de un nuevo producto, que es el caso desarrollado en el artículo. Además, este carácter general de PPDRC posibilita su especialización en ontologías particulares orientadas a los tipos de proceso a considerar. Éste es el caso de la ontología MIRC, objeto del siguiente capítulo, que resulta de la particularización de PPDRC para el ámbito de los procesos de mecanizado e inspección.

Para mostrar la generalidad de PPDRC y para su validación, en el artículo se incluye el estudio del caso correspondiente a la planificación distribuida del proceso de desarrollo de un nuevo producto en una OKP virtual, estableciendo las tareas necesarias y asignando los recursos requeridos a partir de sus capacidades. El estudio demuestra que PPDRC, poniendo énfasis en los recursos y sus capacidades, sirve de apoyo al proceso de desarrollo de un producto, facilitando las actividades de planificación, programación, coordinación y control del proyecto. Todas las actividades implicadas en este ejemplo (de gestión, técnicas y de soporte) pueden ser realizadas a diferentes niveles de agregación, identificándose las capacidades requeridas por ellas, y consultando las capacidades de los recursos disponibles, que posteriormente serán asignados a la ejecución de dichas actividades. Para ello, se formulan preguntas (*queries*) a la ontología mediante el lenguaje SPARQL y a través de una aplicación desarrollada para este propósito, algunas de las cuales, junto a sus resultados, se presentan en forma de tabla en el ejemplo.

Por último, en la publicación se presentan las conclusiones y se plantean unas líneas de trabajos futuros, que el lector puede encontrar en el capítulo 7 de este documento.

2. Contenido del artículo “Knowledge representation for product and processes development planning in collaborative environments.”

International Journal of Computer Integrated Manufacturing, 2014
Vol 27, No. 8, 787-801, http://dx.doi.org/10.1080/0951192X.2013.834480

Received: 14 January 2013
Accepted: 26 July 2013
Published on line: 10 September 2013

Knowledge representation for product and processes development planning in collaborative environments

Lorenzo Solano [1]*, Pedro Rosado [2], Fernando Romero [3]

[1, 2] *Departamento de Ingeniería Mecánica y de Materiales, Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain.*

[3] *Departamento de Ingeniería de Sistemas Industriales y Diseño, Universitat Jaume I, Avda. Vicente Sos Baynat, 12071 Castellón, Spain*

* Corresponding author at: Departamento de Ingeniería Mecánica y de Materiales, Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain. Tel.: +34963877000 Ext. Int.: 76273; fax: +34963877629.
E-mail address: lsolano@mcm.upv.es

[2] E-mail address: prosado@mcm.upv.es Tel.: +34963877000 Ext. Int.: 76221; fax: +34963877629

[3] E-mail address: fromero@esid.uji.es Tel.: +34964728209; fax: +34964728170

Knowledge representation for product and processes development planning in collaborative environments

Abstract

Efficiency in the management of integrated product and processes development is a basic requisite to guarantee competitiveness and success for manufacturing companies. This means that operational management of activities, and human and material resources is extremely important, especially in virtual OKP (One-of-a-Kind Production) systems, and must cover related aspects of their capabilities and social character as well as assignment criteria. In this context, and to facilitate collaborative resources management, an ontology focused on resources and capabilities is proposed in this work. This ontology supports the necessary knowledge for generic and collaborative process planning, providing a shared common semantic for all the members of the virtual company. This work differs from other proposed ontologies in the area of process planning in that the resources considered are all those elements that participate in the execution of the different activity types involved in this wide and complex process. The ontology directly covers the shared, social nature of the resources, the agentive behavior of many of them and a characterization of their capabilities, thus providing specific solutions to the needs of the Collaborative Integrated Development of Products, Processes and Resources ($CIDP^2R$) process.

Keywords: Resource Capabilities, Product and Process Development, virtual OKP, Engineering Ontologies

1. Introduction

The survival of companies in the current business panorama, characterized by globalization, individualization and personalization of demand, and strong competition, depends largely on establishing flexible, adaptable, and agile organization, focusing on customer satisfaction, cooperation, learning and knowledge management and a culture of change (Sherehiy, Karwowski, and Layer 2007) which is particularly appropriate for the OKP systems, at which this work is directed. The OKP systems include practices aimed at successful product development and manufacture in one go (Li, Xie, and Xu 2011), and involve increased agility through distributed control and the use of extremely dynamic organizational structures designed for virtual companies, and which have led to the emergence of what is known as virtual OKP companies (Xie and Tu 2011). A key element for success and for the intended agility of these company networks is the collaboration between members which must take place in an environment of exchange and sharing of knowledge and information which allows them to work in harmony and coordinate their efforts to achieve their shared objectives, adopting a process approach.

With this new approach, the company has a wide variety of process (core, support and management) going on concurrently (Mili, Tremblay, and Jaoude 2010), which need to be planned, establishing the structure of work, the activities and operations sequence and the types of resources allocated. The resources, often shared, include people, systems and information, among others. The management, assignment and configuration of these resources for each of the necessary activities will establish, to a great degree, the effectiveness of each and every one of the processes taking place.

One of the most relevant core processes in manufacturing companies in the field of engineering is New Product Development (NPD), which includes manufacturing processes. In the context of a virtual OKP company, NPD must involve a high degree of collaboration. This is what various authors have identified as Concurrent New Product Development (Büyüközkan, Dereli, and Baykasoglu 2004), Rapid Product Development process (Romero, Estruch, and Rosado 2009; Xie and Tu 2011) or Collaborative Product Development (Büyüközkan and Arsenyan 2012). These proposals, in accordance with Allied Concurrent Engineering (Chen, Shir, and Shen 2002), agree on the need to establish a process-oriented cycle of product development, involving a high degree of integration and collaboration, enabled by a management model based on projects and supported by ICTs (Information and Communication Technologies). In recent years, to facilitate collaboration and integration in distributed environments companies have been increasingly using ontologies that allow the integration of knowledge and semantic interoperability (Poli, Healy, and Kameas 2010; Lin et al. 2011). Along these lines, both the more general proposals for explicit foundational ontologies and those defined for the different domains of product development can be found (Cheung et al. 2006; Baxter et al. 2007). It is also important to mention recent developments in enterprise ontologies which have come from a wide range of sources (O'Leary 2010). From among these it is important to mention TOVE (TOronto Virtual Enterprise project) and EO (Enterprise Ontology), as ontologies of multiple domain, or PSL (Process Specification Language) which specializes in process sub-domain. However, none of these deals directly with resource capabilities, because they have usually been centered directly on temporal aspects of the resources in a particular process. Other recent works (Rosado and Romero 2009) have shown the need to integrate the different ontologies from the areas of product, processes and resources as a basis for collaborative process planning of the Product and Processes Development process.

In response to these current needs, in this work an ontological model to represent the Resource Capabilities in the Product and Processes Development process (PPDRC) is presented. This specific ontology is one of the basic pillars of the integrated ontology that supports the Collaborative Integrated Development of Products, Processes and Resources ($CIDP^2R$) process and provides the necessary knowledge to carry out dynamic and optimum assignment of resources for the activities of this process (design, manufacturing, process planning, scheduling, execution coordination and control, execution, etc.), in a Service Oriented Architecture (SOA) and for any level of aggregation.

At this point it is important to note that process planning activities do not only include those defining the manufacturing process but also those defining the Product and Processes Development process itself. To achieve this goal, in the first place, a framework in which to place the key concepts of the ontology is established. Later, the entities and the predicates that constitute the core of a generic ontology to represent capabilities for the Product and Processes Development process are identified, so that all needs can be met. Finally, the implementation of the ontology is carried out in order to prove the validity of the proposal.

2. State of the art

Currently, in the domain of design and manufacturing engineering, there is a great deal of interest in knowledge management and the integration of people and systems, and ontologies are being used to develop these areas (Lin et al. 2011).

The definition of ontology given by Guarino in Uschold and Gruninger (1996) states that "an ontology is a formal description of the entities within a given domain, the properties they possess, the relationships they participate in, the constraints they are subject to, and the patterns of behavior they exhibit". According to Borgo and Leitao (2008), ontologies can be very general (foundational ontologies) or domain dependent (core ontologies). Bräuer (2007) states that core ontologies, domain ontologies and application ontologies can be distinguished among the domain dependent ontologies. Foundational ontologies, such as DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering), create a framework for the development of ontologies, which allow a more careful definition of parts of the world, an interaction between data in different formats (Shadbolt, Hall, and Berners-Lee 2006) and the mutual understanding and interoperability between people and machines (Poli, Healy, and Kameas 2010).

An important characteristic of any ontology is its capacity to conceptualize the reality that is being presented. This ability depends on the ontological foundations adopted. Thus, *endurantism* treats entities as objects that pass through time and are totally present at any given moment, whereas *perdurantism* assumes that entities are extended in time and space (Oberle et al. 2007). Moreover, an ontology adopts a *revisionist* position when it aims to capture the intrinsic nature of the world, while a *descriptive* approach covers the concepts based on human cognition, natural language and common sense (Bräuer 2007). Another basic ontological foundation is related to the ontology's ability to represent the entities social character, while the *functionalism* of the conceptualization indicates the capacity to express the functional considerations of the world's entities (Lambert et al. 2008). Finally, from the range of the ontological choices, *actualism* only deals with what is real (such as the execution of the activities and resource participation in the activity occurrences over time), in contrast to *possibilism*, which considers what is possible (Masolo et al. 2003), such as the assessment of resources according to their ability to participate in the activities.

The previous ontological foundations must be studied in order to consider the concepts that must be present in the PPDRC ontology. However, to establish these concepts it is also necessary to take a close look at the subject matter, which in our case is the process planning of particularly complex processes, such as a CIDP^{2}R process.

IDEF3 was one of the first methodologies used to model processes, offering a language for its graphic representation and formalizing the necessary concepts and relationships to describe processes. Many of these concepts and relationships were later included in the PSL (Process Specification Language) ontology (ISO 2004b). In PSL, as with IDEF3, a process is a collection of activities with the restrictions that govern their relationships. The PSL ontology provides four basic concepts to describe processes and their execution (*activity*, *activity_occurrence*, *timepoint* and *object*). In PSL, an *activity* is a behavior or action that can be repeated, while an *activity_occurrence* is each of the occurrences or executions of an *activity*. The *timepoints* represent the moments in time usually linked to the beginning and end of the execution of an activity. Finally, an *object* is any entity that does not fall into any of the previous three categories (Gruninger 2009). Some ontologies for specific processes have been developed as extensions of PSL (Deshayes, El Beqqali, and Bouras 2005; Lohse et al. 2005). PSL has also served as a foundational layer for the ontological proposal of Chungoora and Young (2011).

Activity Theory (AT) is another contribution to the area of processes which covers some considerations not included in PSL, especially those related to the social character of activities. According to AT, *activities* are made up of *actions*, which in turn are made up of *operations*, whose objectives are respectively: *motive*, *goal* and *condition* (Karanasios et al. 2011). Participation in a motivated activity (with motive) means the development of an action with conscious control and an immediate objective. The concept of conscious control in AT is coherent with the BDI paradigm, in which the participants have Beliefs, Desires and Intentions (Ferrario and Oltramari 2004, Tamma et al. 2005).

PSL, in a similar way to the Resource ontology of TOVE, introduces the concept of resource as any object that is needed during the execution of an activity. Another approach to the concept of resource, from the area of manufacturing planning and execution, is the one introduced in MANDATE (ISO 2004a) which considers a resource to be any device, tool and means (including human) at the disposal of the enterprise to produce goods or services. MANDATE focuses on resource management, and considers the two roles that the various elements can play in operations: operation objects and operation means. Moreover, MANDATE incorporates two key concepts for the characterization and use of resources: a) capability, the quality of being able to execute an activity, e.g., a group of characteristics that describes functional aspects of manufacturing resources and b) capacity, the capability of a system (or sub-system) or resource to perform its expected function, in particular in terms of amount of production.

In the literature, other ontologies that have introduced the resource concept can be found. ADACOR (Borgo and Leitao 2008) is a core ontology covering manufacturing scheduling and control operations, and making use of the resource concept as any entity that can execute a range of operations as long as its capacity is not exceeded. Similar considerations for resources are found in the work of Lin, Harding, and Shahbaz (2004) intended to provide a manufacturing system engineering ontology for extended environments. Other ontological developments described in Tamma et al. (2005) and Rajsiri et al. (2008) reveal the importance of resources for collaboration. In these works, resources are understood to be the social use of objects in coordination and collaboration environments.

Finally, in the e-manufacturing and collaborative supply chains domains, there are some works which demonstrate the need to discover manufacturing resources and services in an efficient way to improve the agility of the manufacturing process (Jang et al. 2008; Ameri and Dutta 2008; Cai et al. 2010). With this objective in mind, ontological models base on OWL are used. However, here the proposals are focused on certain types of resources, and which describe their capabilities without considering the agentive nature that many of them possess, as is the case with people or other intelligent agents, which are fundamental in knowledge intensive collaborative processes.

3. Ontology objectives and requirements

One of the first steps in the development of an ontology is to clearly define its scope and domain. This scope and domain enable the establishment of the ontology requirements, and help to determine the ontological foundations necessary to represent the concepts and the property and data relations to be dealt with. Following this, the requirements imposed on the ontology are addressed according to their objectives.

Process planning can include recursively other process planning operations. For instance, process planning of product development includes the process planning of: project management; technical processes and support processes. At the same time, process planning of project management includes the planning of the following project activities: project planning; project scheduling and project execution coordination and control. In turn, process planning of technical processes includes the planning of technical activities such as the following: product specifications; product development and product manufacturing. Therefore, the overall scope of PPDRC ontology is much greater than those corresponding to manufacturing process planning.

The basic competence of the PPDRC ontology is to model and make explicitly clear the knowledge necessary for the assignment of resources (or types of resources) to activities (or types of activities) involved in process planning in a collaborative environment. This includes the structuring of the activities required, the allocation of resources to each of them based on their capabilities and taking into account the collaborative environment, where resources may have a behavior influenced by their social character and guided by their interests. For this reason, the social character consideration of the participants in the activities and their occurrences is needed. It is also necessary to consider the intrinsic agentive nature of certain elements that establish the control and development of the execution of activities, fixing their start and finish, orientation, strategy and objective (Lin, Harding, and Shahbaz 2004; Gangemi et al. 2005; Izhar et al. 2013). This agentive character is maintained on a cognitive level requiring a BDI approach, where all the resources that take part in activities and guide their development are agentive, and are associated with one or more objectives that set their intentions.

The required level of detail for modeling of manufacturing resource capabilities may be greater than for modeling of other resources capabilities involved in the Product and Processes Development process. For these reasons, it is necessary that the ontology to be developed provides responses to questions with a different degree of generality, referring to specific aspects or to generic questions in the domain such as: a) What are the capabilities of the resources? b) How is the process/activity defined and structured? c) What is the implication of the resources in the process/activity? d) What is the objective of the process/activity?. The responses to these questions must incorporate a certain level of knowledge beyond the simple recovery of information that has been previously stored. Rather, it is enhanced know-how obtained from relating and/or processing this information. Another requirement for the PPDRC ontology is to represent all the entities of interest for the assignment of resources to activities in a collaborative and distributed context with different levels of detail. The activities involved are notably complex due to their intrinsic nature and includes activities that are developed on a physical or material level, such as manufacturing operations; and others that are developed on a mental, intellectual or cognitive level, such as a scheduling or execution, coordination and control tasks (Ferrario and Oltramari 2004; Tamma et al. 2005).

Finally, the ontology must consider complete traceability, temporal evolution and the relationship between information and data and the entities and circumstances in which they originate.

4. A resource capability ontology for product and process development

The PPDRC ontology is based on the following fundamental concepts: activities, activity_occurrences, workflows, objects, resources, capabilities, agentives, roles and objectives, which are needed in the Product and Processes Development process. This proposal satisfies the requirements expressed in the previous section and has come as a result of studying the state of the art.

4.1. Fundamental concepts of the domain

Activities represent anything that can be carried out (Kethers 2000; Lin, Harding, and Shahbaz 2004) and constitute behavior that can be repeated. This abstract concept of an activity, as a repeatable pattern, is materialized whenever an activity is executed in time and is denominated occurrence. Activity occurrences or occurrences use and produce or transform objects. In PPDRC ontology, workflows are a type of complex occurrence with a common objective shared by one or more agents, which coordinate in order to achieve this objective (Tamma et al. 2005), and which are coherent with the use of PSL for activity flow modeling as proposed by Bock and Gruninger (2005). Contrary to the approach found in PSL, in which objects are defined by exclusion with respect to the other three first level entities of the ontology (*activity*, *activity occurrence* and *timepoint)* (Qiao, Kao, and Zhang 2011), in the PPDRC ontology, an object is a tangible or intangible entity that does exist (Solano, Romero and Rosado 2010). Among these objects are social objects, or shared descriptions which allow mutual understanding between the members of a community (Ferrario and Oltramari 2004), as is the case with the activity dealt with previously.

A resource is the description of how an object participates in the execution of an activity. As established in MANDATE, resources are characterized by the degree of qualitative or quantitative performance referred to in the concept of capability. The resource can participate in an activity occurrence according to different roles which can be grouped into four types: control; input; output and mechanism (NIST 1993). If these concepts are placed in the more general framework of a foundational ontology such as DOLCE, whose high level ontological categories are: endurant; perdurant; quality and abstract (Gangemi et al. 2005), the object and the occurrence of PPDRC ontology are covered by the concepts of endurant and perdurant respectively, which along with the concept of region (which serves to express qualities of any concept) form the basic semantics of the PPDRC ontology.

4.2. Entities and predicates of the PPDRC ontology

Taking into account the concepts mentioned previously, the first level of the PPDRC ontology is structured with three types of mutually exclusive entities: *Object*; *ActivityOccurrence* and *Region* (Figure 5.1). The predicate *formedBy* (entity, entity) can be applied to any of these entities to express relationships of composition between entities of the same type: between objects; between occurrences and between regions. The predicate f*ormedBy* provides the mechanism to structure the entities of the ontology at different levels of detail or aggregation, and expresses transitive relationships that are limited to identifying the entities that constitute another entity, without specifying the type of relationship between them.

The next three subsections present the main entities of Object, ActivityOccurrence and Region type, and some of the predicates that are established between these entities.

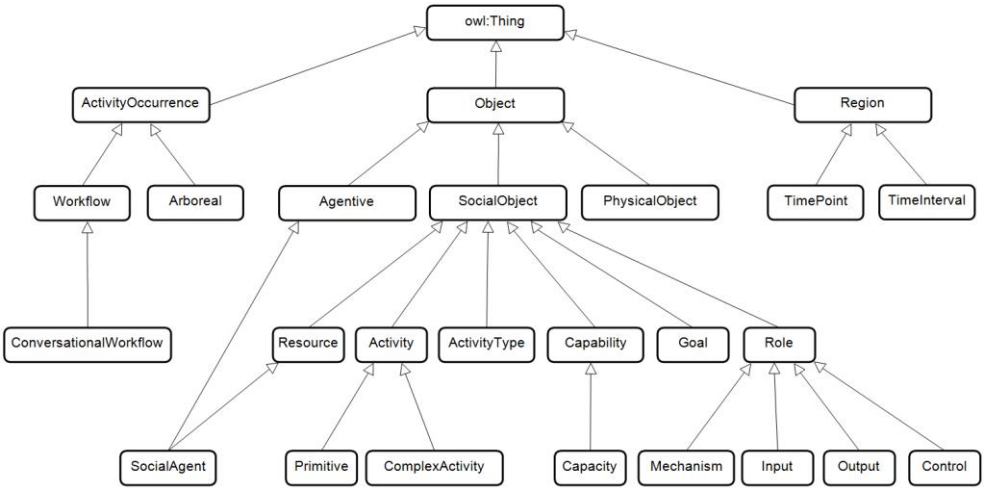


Figure 5.1.Taxonomy of the PPDRC ontology.

4.2.1. Objects and their predicates

The entity *Object* has three specializations: *PhysicalObject*, *SocialObject* and *Agentive*. The entity *PhysicalObject* represents things that have a physical existence, such as machines, IT systems, people, etc. The entity *SocialObject* corresponds to the concept of social object mentioned previously. Finally, the entity *Agentive* corresponds to those objects, such as people, who have beliefs, desires and intentions (BDI) and whose behavior is driven by motivation. An object can belong to several of the three last categories as these are not exclusive. In this sense, it is important to highlight the entity *SocialAgent* which belongs simultaneously to the entities *Agentive* and *SocialObject*. For example, a person is a *SocialAgent*, that is to say, a person is a social object who has an agentive character.

In the context of the PPDRC ontology, the social objects considered are: *Activity*; *ActivityType*; *Capability*; *Resource*; *Role* and *Goal*, which are all mutually exclusive, but which do not use up all the possibilities of the social objects. An *Activity* is the entity that describes the activity concept defined previously, while an *ActivityType* is the social object that represents the abstraction of a group or type of activities. *Capability* is the shared definition that characterizes the use of an object in a type of activity, expressing certain competences to execute activities of this type and, when necessary, the level of performance achieved in their execution. For example, *developing the mechanical assembly of a gear pump* is an activity type, while the competence to develop mechanical assemblies is a capability. Capabilities are related to resources through the predicate *characterizedBy (Resource, Capability)*, while the predicate *executing (Capability, ActivityType*) fulfills its meaning as an entity necessarily linked to the execution of a type of activity. *Capacity* is a *Capability* expressed in terms of amount of production. For example, the capacity of a resource, such as a grinding machine, can express its rate of production (in mm^3 / min) machining flat surfaces with a determined level of roughness.

Any object that has a relationship with an individual of the entity *Capability* belongs to the entity *Resource*. Therefore, all the individuals of the entity *Resource* have the competence or ability to realize at least one type of activity. The *Capability* of a resource refers to the

carrying out of a certain type of activity. For example, a person who can carry out the activity *molded piece design* is a resource. In short, any of the three categories of objects shown above (*Agentive*, *SocialObject* and *PhysicalObject*) can be a *Resource*.

The entity *Role* is a social object that is identified by the manifestation of certain capabilities in the occurrence of an activity (*ActivityOccurrence*). Resources are the only objects that can have roles. While a capability is linked to a type of activity, an individual of the entity *Role* establishes the type of participation of a resource in a concrete activity occurrence through the predicate *behaves* (*Role, Capability*). To complete its specification, the predicates *hasRole* (*Resource, Role)* and *isPresent (Role, Arboreal)* show the participation of a resource in the occurrence of a primitive activity, an occurrence that belongs to the entity *Arboreal (*an *ActivityOccurrence* type*).* A resource can participate in the same activity with different roles according to the capabilities that it exhibits on each occasion. For example, a resource that has the ability to direct, perhaps does not participate as director in the occurrence of an activity. The entity *Role* has four specializations (*Control*, *Input*, *Output* and *Mechanism)* whose individuals can only relate through *isPresent* relationships with occurrences of *Arboreal* type.

Finally, the description of the entity *Goal* represents the definition of the objective. The entities of the type *Agentive* are the only ones who can have objectives, which is expressed through the predicate *hasGoal* (*Agentive, Goal).* The fulfillment of their goals is the guide to the initiatives of the agentives during the execution of activities.

4.2.2. ActivityOccurrences and their predicates

The second entity of the first level of the PPDRC ontology is *ActivityOccurrence,* which, as can be seen in Figure 5.2, is associated with other entities through different predicates. The relationships of their individuals establish the characteristics of an execution, for example, the object that carries out an activity, the time points in which it takes place, its restrictions and relationships with other occurrences, start and sequencing mechanisms, partial milestones and the state achieved after it is carried out (Tamma et al. 2005). The predicate *occurrenceOf* (*ActivityOccurrence, Activity*) establishes the relationship between the occurrence of an activity and the activity itself. The predicate identified in Figure 5.2 as *PSL_relations* represents the set of predicates that are similar to those of the PSL ontology (*after*, *before*, *earlier*, *precedes*, *hasLeaf_occ*, *hasRoot_occ*, *min_precedes*, *next_subocc*, *root*, *root_occ*, *leaf*, *leaf_occ*, *poss*, *sucessor*, etc.) which specify the relationships of sequence and precedence between the occurrences of the activities that form a process. As with PSL, the PPDRC ontology distinguishes between primitive *Activities* (*Primitives*) and complex activities (*ComplexActivities*). *ActivityOccurrences* of the *Arboreal* type are those *ActivityOccurrences* of primitive activities from which precedence relationships are established.

The *Workflow* is a type of complex *ActivityOccurrence* in which it is necessary to show its configuration. In other words, in the occurrence of a *Workflow* type activity the structure and sequence of its temporal parts must be specified. For Kethers (2000), the concepts of *action workflow* and *declaration workflow* support respectively the bilateral conversations for the development of an activity between the customer and the supplier of a process, and the bilateral conversations necessary for the development of a negotiation (Figure 5.3). The entity *ConversationalWorkflow* of the PPDRC ontology groups together these two types of *workflow*. *ConversationalWorkflow* is a type of execution of a complex activity, in which

each of its stages can contain another *workflow*. *ConversationalWorkflow* is a specialization of *Workflow* whose execution can only be initiated by an individual of the entity *SocialAgent* through the predicate *triggers*. In fact, *SocialAgent* is an entity characterized by the *Capability* to trigger and manage *ConversationalWorkflows.*

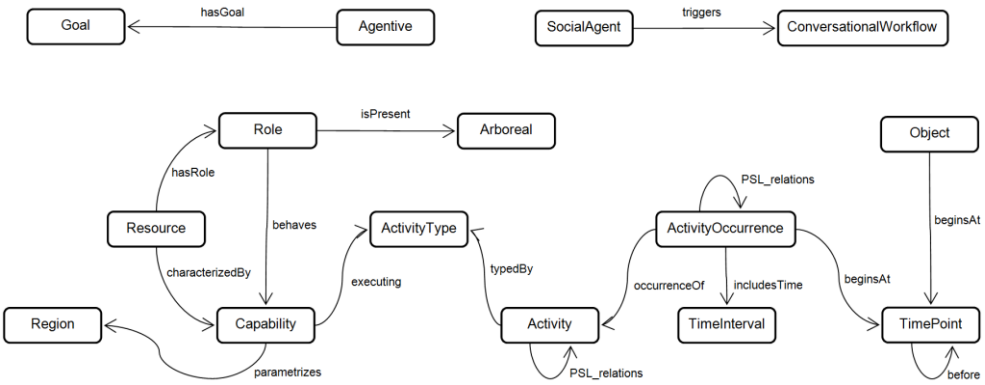


Figure 5.2. Predicates of the PPDRC ontology.

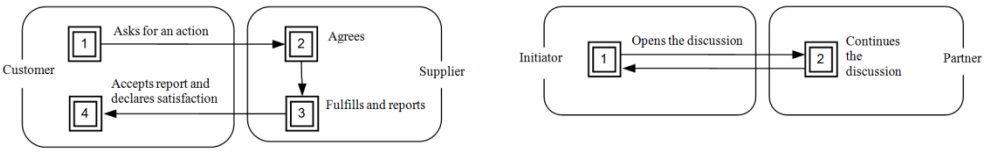


Figure 5.3. Action workflow and declaration workflow.

4.2.3. Regions and other predicates

The third of the top level entities of the ontology is *Region*, which represents the space in which the values (*quales*) corresponding to a quality are found. A *quale* is a particular value from the allowed values for quality. *TimeInterval* and *TimePoint* are regions that represent respectively intervals and points in time, and are inherent in the activity occurrences and in the existence of the objects. The predicate *parametrizes* (*Capability*, *Region*) expresses the relationship between a capability and its *quales*, which identifies the values or attributes of the capabilities. For example: *0,15 µm* is the value of the characteristic *average roughness (Ra)* for the capability to obtain flat surfaces; *high complexity* is the value of the characteristic *part complexity* associated with the capability to plan the manufacturing process. All types of capabilities can be modeled with any degree of detail, and can even be quantified. However, the high degree of detail or the quantification normally required in capabilities of production means, such as *obtain flat surfaces*, is not necessarily relevant in other capabilities such as *design parts* or *plan the manufacturing process*.

As well as the predicates already discussed, the ontology has others which serve to express the relationships between the objects (endurants) or activity occurrences (perdurants) and time: an endurant exists in time and can genuinely change over time, while a perdurant

exists in time but cannot change over time. The PPDRC ontology is able to represent both the temporal space in which the occurrence of an activity is developed or in which an object exists, as well as the changes in the object over time. The first is expressed through the predicates *beginsAt* and *endsAt (Object or ActivityOccurrence, TimePoint)*, which establish the relationship between the occurrence of an activity or the existence of an object and its *quales* of temporal region. For the second of these cases, the predicate *version (Object, Object)* is used, which in a similar way to MANDATE (ISO 2005), associates the different versions of the same object. The predicate *version* has various specializations, such as *upgrade* and *planned*. *Upgrade* allows reference to an object stating its new version, while *planned* associates it with its planned future state. For example, the capabilities of a resource can be updated through *upgrade*.

4.3. Implementation

In response to the requirement of supporting a Web-based OKP system, it was decided to model the ontology using a standardized and widespread language. Therefore, the modeling of the PPDRC ontology was carried out through OWL/SWRL, which are widely used within globally extended manufacturing teams, because they enhance the semantic interoperability and reuse of knowledge resources (Cai et al. 2010; Lin et al. 2011). SWRL (Semantic Web Rule Language) improves the semantic expressivity of OWL (Ontology Web Language) as it allows the expression of knowledge that cannot be directly defined with the OWL axioms (Lin 2008).

The PPDRC ontology has been developed and edited using the ontology editor Protégé and can be consulted at (PPDRC 2013). Additionally, to check the consistency between predicates and definitions of the ontology developed, maintain the hierarchy of the ontology and allow consultation, the reasoner Pellet was used. Pellet is an open code reasoner which allows the classification and reasoning with individuals in OWL/SWRL ontologies.

In order to validate the proposal, a Java application has been implemented. The application incorporates Java libraries to operate with OWL files (OWL2 API) and to reason and query with Pellet (Pellet OWL2 API). This Application Programme Interface (API) incorporates a query engine that provides SPARQL support. Additionally, the application incorporates a front-end interaction interface that translates the user query into SPARQL and transforms the SPARQL query results into a more user readable format.

5. An illustrative example

This case study shows the process planning of a Collaborative New Product Development process in a virtual OKP Enterprise, emphasizing resources and their capabilities. All involved activities (management, technical and support) can be carried out at different levels of aggregation, defining and analyzing the activities, identifying the capabilities required for these, consulting the capabilities of the resources available and assigning the resources to concrete occurrences. For the sake of simplicity, the example only shows some of the activities necessary, and does not aim to describe an optimum management of the project. Its objective is to show how the PPDRC ontology allows the management, in a distributed form and with different levels of aggregation, of the information on resources that is necessary in New Product Development process at different levels. The use of

PPDRC ontology is of special interest in process planning, scheduling and coordination and execution control corresponding to the project management.

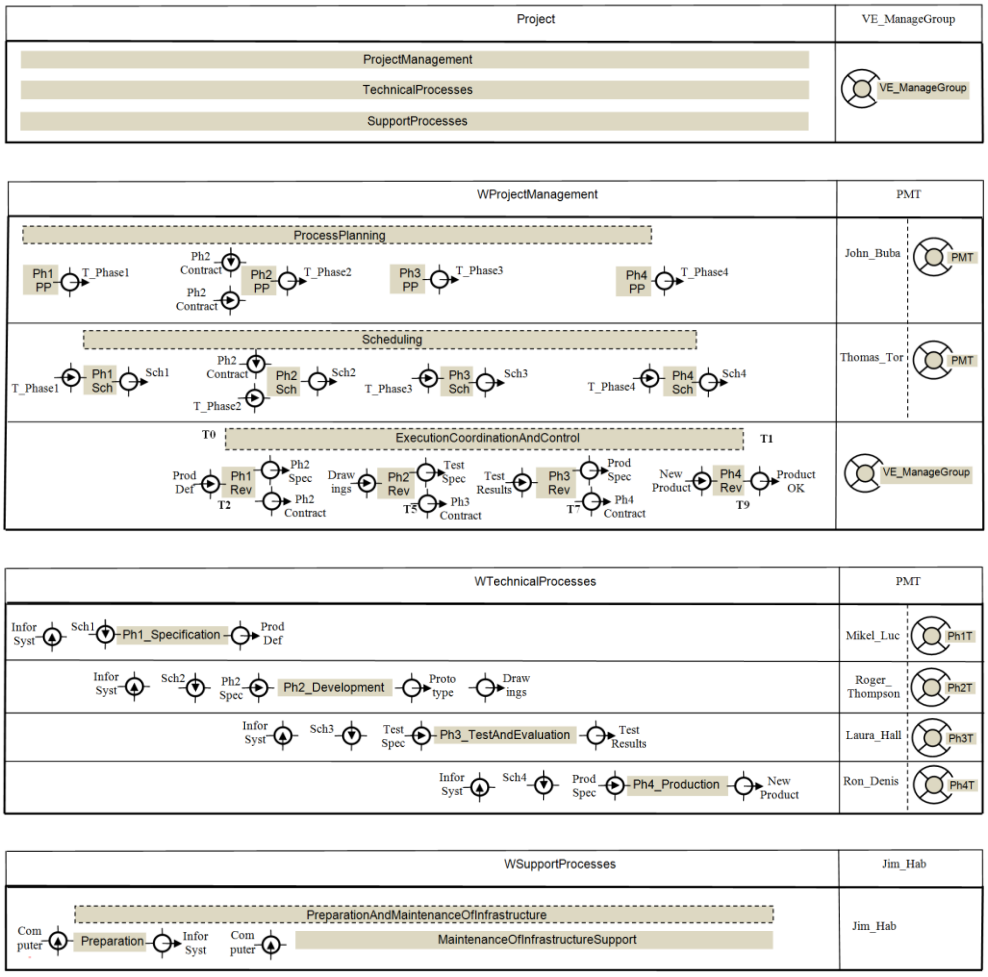


Figure 5.4. Execution of the complex activity Project.

An overview of the process at the Project level is given in Figure 5.4, showing the activities of the technical processes (specification, development, test and evaluation and production phases) and management and operative support activities directly linked to the technical activities. In the illustration, the occurrences that make up the global occurrence Project are represented with shaded horizontal bands. If they are complex, the band is outlined with a discontinuous line and without outline if they are primitive. The column on the right identifies the people or organizational units, who participate in occurrences with a mechanism role. Some of the teams included in the figure are: VE_ManageGroup (Virtual_Enterprise_Manage_Group), PMT (Project_Manager_Team), Ph1T (Phase1_Team), etc. As well as the human resources with a mechanism role, there are other

resources that are necessary to carry out the activities which are represented together with the occurrences in which they participate using the icons in Figure 5.5.

Figure 5.5. Symbols used to represent the roles of resources.

5.1. Project level

The occurrence Project is broken down into three occurrences: ProjectManagement, TechnicalProcesses and SupportProcesses (Figure 5.4). These occurrences establish the responsibilities and conditions under which the action workflows (WProjectManagement, WTechnicalProcesses and WSupportProcesses) are triggered. In this case, the VE_ManageGroup delegates the execution of the first two workflows to the PMT team, and the third to Jim_Hab, maintaining overall responsibility for the coordination and execution control at the project level following the project management models proposed by McGrath (2004**)**. In turn, the PMT team delegates to John_Buba the responsibility for planning the technical process and to Thomas_Tor the project scheduling. Finally, the management of technical phases is delegated to other members of the PMT, who are supported by multi-disciplinary groups created for each phase.

Table 5.1. Query 1.

Query 1		
Query in SPARQL	*Natural language interpretation*	*Answer*
SELECT ?Capability WHERE {	What capabilities	Capability | ManageWorkflow | C_Preliminary_Develop | C_Detailed_Develop | C_Product_Assembly_Develop | C_Product_Elect_Develop | C_Develop | C_Product_Mech_Develop | ManageWorkflow | C_Specify | ManageWorkflow | C_Produce | ManageWorkflow | C_Evaluate |
?Capability <http://www.coapp.es/ontologies/2011/0/PPDRC_v1.owl#capabilityRequiredBy> ?Occurrence .	are required to execute the occurrences that	
?Occurrence a (<http://www.coapp.es/ontologies/2011/0/PPDRC_v1.owl#subactivityOccurrenceOf> value <http://www.coapp.es/ontologies/2011/3/Ejemplo10.owl#WTechnicalProcesses>) . }	are subactivity_occurrences of WTechnicalProcesses?	

The use of PPDRC ontology during the course of the project begins with the process planning stage at Project Management, as an aid in defining: technical activities (phases) and support activities; the structure of deliverables (input and output resources); the roles of human resources and work teams; and the time and resource dependencies. Later, in the scheduling occurrences of Project Management, the ontology is used to assign the resources that participate according to previous roles established in the process planning, and to set other restrictions relative to execution times. The ontology must also respond to the needs of those who participate in the coordination and execution control of phases as well as their execution (WTechnicalProcesses) and support for the phases (WSupportProcesses). This

last group provides support for the rest of those involved and facilitates inter-department and/or inter-company collaborative work.

The queries in this section were formulated in an application developed for this purpose, which uses the SPARQL language to interrogate the ontology. In the first query, Table 5.1 shows the query in SPARQL, its interpretation in natural language and the result. In the rest of the queries only the interpretation and the results are shown.

5.1.1. Process planning scenario

In this scenario (Figure 5.6), the ontology must support the planner in the definition of the Project occurrences, which is carried out by refining or modifying Project model templates, defining, at different levels of aggregation, a structure of activities (WBS –Work Break Structure–).

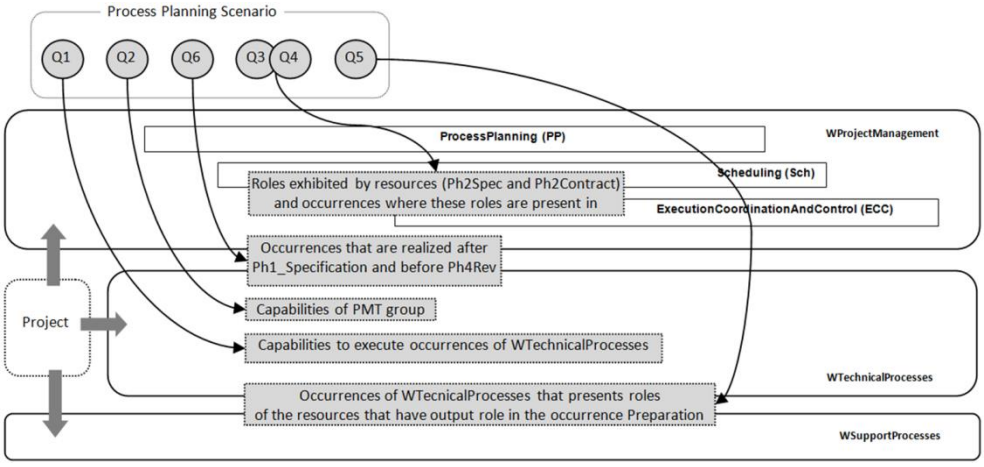


Figure 5.6. Examples of queries in the process planning scenario corresponding to the planning of phases of the project.

According to adopted template, the planner could consult the required capabilities for the types of activities present in WTechnicalProcesses (Table 5.1). Later, in order to check the suitability of the PMT group to which the WTechnicalProcesses occurrence is assigned, the planner can consult the combined capabilities of its members (Table 5.2).

Results of queries can be derived from facts or assertions, or can come as the result of a process of inference. For example, query 1 results show an inferred relationship between some occurrences and the necessary capabilities by means of rule 19 (Table 5.3). Similarly, rule 12 allows inference that a resource is characterized by some capability, if that capability has appeared in any role played by the resource.

In this same planning scenario, queries can also be raised about the deliverables. For example, the planner raises queries 3 and 4 referring to the roles fulfilled by the deliverables Ph2Spec and Ph2Contract (Figure 5.6).

The planner may be interested in other types of resources. This is the case of Query 5, which raises a question about the WTechnicalProcesses occurrences which use the IT resources obtained in the occurrence Preparation (Figure 5.6).

Table 5.2. Query 2.

Query 2	
Natural language interpretation	*Answer*
Which capabilities characterize the resources that form the PMT group?	Capability | C_Execute | C_Evaluate | C_Specify | C_Preliminary_Develop | C_Schedule | C_Coordinate | C_Control | C_Coordinate_Execution | C_Develop | C_Plan | C_Test |

Table 5.3. Rules of inference written in SWRL.

	Rules
1	TimePoint(?x) , TimePoint(?y) , quale(?x, ?u) , quale(?y, ?v) , lessThan(?u, ?v) , differentFrom(?x, ?y) -> before(?x, ?y)
2	earlier(?x, ?y) -> differentFrom(?x, ?y)
3	after(?x, ?y) , before(?x, ?y) -> sameAs(?x, ?y)
4	Atomic(?act1) , Atomic(?act2) , Input(?i1) , Input(?i2) , beginsAt(?act1, ?t1) , beginsAt(?act2, ?t2) , hasRole(?r, ?i1) , hasRole(?r, ?i2) , isPresent(?i1, ?act1) , isPresent(?i2, ?act2) -> sameAs(?t1, ?t2)
5	subactivities(?x, ?y) , subactivities(?y, ?x) -> sameAs(?x, ?y
6	behaves(?r, ?cap) , isPresent(?r, ?occ) , occurrenceOf(?occ, ?a) , typedBy(?a, ?at) -> executing(?cap, ?at)
7	beginsAt(?x, ?in) , endsAt(?x, ?fin) -> before(?in, ?fin)
8	beginsAt(?sub, ?t) , root_occ(?sub, ?oc) -> beginsAt(?oc, ?t)
9	endsAt(?sub, ?t) , leaf_occ(?sub, ?oc) -> endsAt(?oc, ?t)
10	beginsAt(?y, ?ini) , earlier(?x, ?y) , endsAt(?x, ?fin) , differentFrom(?fin, ?ini) -> before(?fin, ?ini)
11	beginsAt(?x, ?in) , endsAt(?x, ?fin) -> differentFrom(?in, ?fin)
12	Resource(?x) , behaves(?y, ?z) , hasRole(?x, ?y) -> characterizedBy(?x, ?z)
13	Atomic(?a1) , Atomic(?a2) , Input(?rol) , Output(?sal) , hasRole(?r, ?rol) , hasRole(?r, ?sal) , isPresent(?rol, ?act2) , isPresent(?sal, ?act1) , occurrenceOf(?act1, ?a1) , occurrenceOf(?act2, ?a2) -> earlier(?act1, ?act2)
14	Atomic(?a1) , Atomic(?a2) , Input(?i) , Mechanism(?m) , hasRole(?r, ?i) , hasRole(?r, ?m) , isPresent(?i, ?act1) , isPresent(?m, ?act2) , occurrenceOf(?act1, ?a1) , occurrenceOf(?act2, ?a2) -> earlier(?act2, ?act1)
15	Control(?c) , Input(?i) , beginsAt(?act1, ?t1) , beginsAt(?act2, ?t2) , hasRole(?r, ?c) , hasRole(?r, ?i) , isPresent(?c, ?act2) , isPresent(?i, ?act1) -> before(?t2, ?t1)
16	Control(?c) , Output(?o) , endsAt(?act1, ?t1) , endsAt(?act2, ?t2) , hasRole(?r, ?c) , hasRole(?r, ?o) , isPresent(?c, ?act2) , isPresent(?o, ?act1) -> before(?t1, ?t2)
17	Atomic(?a1) , Atomic(?a2) , Mechanism(?mec) , Output(?o) , hasRole(?r, ?mec) , hasRole(?r, ?o) , isPresent(?mec, ?act2) , isPresent(?o, ?act1) , occurrenceOf(?act1, ?a1) , occurrenceOf(?act2, ?a2) -> earlier(?act1, ?act2)
18	Atomic(?x) , Atomic(?y) , subactivityOf(?x, ?y) , differentFrom(?x, ?y) -> ConcurrentSuperpositionAtomic(?y)
19	executing(?cap, ?at) , occurrenceOf(?occ, ?a) , typedBy(?a, ?at) -> requiresCapability(?occ, ?cap)

As well as the queries related to resources, roles and capabilities, the planner could be interested in the sequence restrictions. This is the case of Query 6 (Table 5.4), which involves checking the occurrences to be carried out between Ph1_Specification and Ph4Rev. The result of this query, as well as the sequence restrictions declared by the process planner, incorporates facts arising from rules 13 to 17 of Table 5.3. These rules introduce sequence dependencies among the primitive occurrences arising from the roles present in them. For example, as the output generated in Ph1_Specification (ProdDef) is the

input required in Ph1Rev, it is inferred that the beginning of Ph1Rev will take place after the completion of Ph1_Specification (rule 13). This knowledge inference has special relevance in process planning in collaborative environments, where the planner role may be shared or distributed, meaning that different planners can consult or enter facts to the ontology.

Table 5.4. Query 6.

Query 6	
Natural language interpretation	*Answer*
Which occurrences are realized after Ph1_Specification and before Ph4Rev?	Occurrence | Ph3Rev | Ph3Sch | Ph2_Development | Ph2Sch | Ph3PP | Ph3_TestAndEvaluation | Ph2PP | Ph4PP | Ph2Rev | Ph4Sch | Ph1Rev | Ph4_Production |

5.1.2. Scheduling scenario

In the scheduling scenario (Figure 5.7), the scheduler carries out the assignment of resources with a mechanism role and establishes the values of the *timepoints* which define the beginning or end of certain occurrences and may create additional precedence relationships to those fixed previously in the process planning scenario. As with the planner role, the scheduler role can also be shared and/or distributed. In order to assign resources the scheduler can query (Table 5.5) the ontology about the people who bring together the necessary capabilities for any occurrence such as Development (Query 7) or about the resources that have a particular capability, such as C_Product_Elect_Develop (Query 8). Other similar queries are: Queries 9, 10 and 11 (Figure 5.7).

In the PPDRC ontology, the resources can only be assigned to primitive occurrences, as the roles only admit isPresent relationships with occurrences of the *Arboreal* type. However, queries may be formulated about the resources participating in complex occurrences. For example, in Query 12 (Figure 5.7) the ontology is consulted about resources with an input role that participate in the complex occurrence ExecutionCoordinationAndControl.

As was shown previously, the scheduler, as part of his work, may require information on the start and finish timepoints of a complex occurrence such as ExecutionCoordinationAndControl, and for this, he will raise Query 13 (Table 5.6). In this case, the result of the query is obtained from the beginning timepoint of the occurrence Ph1Rev (T2) and end timepoint of the occurrence Ph4Rev (T9) and by inferring that the beginning and end timepoints of the occurrence ExecutionCoordinationAndControl (T0 and T1) match with T2 and T9 respectively, as can be seen in Figure 5.4.

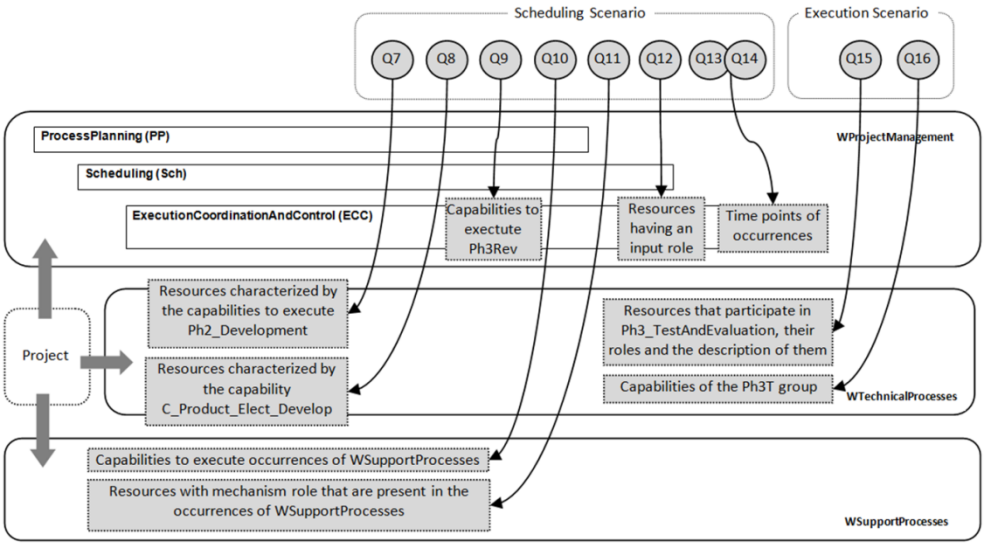


Figure 5.7. Examples of queries in the scheduling and execution scenarios during the scheduling of the project and during the execution and control of the project.

Table 5.5. Queries 7 and 8.

Query 7	
Natural language interpretation	*Answer*
Which resources are characterized by the capabilities required to execute the occurrence Ph2_Development?	Resource | Capability | Mikel_Luc | C_Preliminary_Develop | Stp1T | C_Preliminary_Develop | Tom_Lee | C_Detailed_Develop | Stp2T | C_Detailed_Develop | Frank_Cisco | C_Product_Assembly_Develop | Tsk3T | C_Product_Assembly_Develop | VE_ManageGroup | ManageWorkflow | Tsk2T | C_Product_Elect_Develop | Ana_Lupfer | C_Product_Elect_Develop | Roger_Thompson | C_Develop | Ph2T | C_Develop | Nick_Nolte | C_Product_Mech_Develop | Tsk1T | C_Product_Mech_Develop |
Query 8	
Which resources are characterized by the capability C_Product_Elect_Develop?	Resource | Tsk2T | Ana_Lupfer |

Table 5.6. Query 13.

Query 13	
Natural language interpretation	*Answer*
At what time point does the occurrence ExecutionCoordinationAndControl begin, and at what time point does the occurrence ExecutionCoordinationAndControl end?	Beginning | End | 20 | 430 |

As with the timepoints T2 and T9, T5 and T7 were fixed by the planner as the timepoints for completion of the occurrences Ph2Rev and Ph3Rev respectively. From this information (Query 14 in Table 5.7), the scheduler can check the duration established for phase Ph3_TestAndEvaluation with the time interval between T5 and T7. After checking this information, the scheduler can consider different options, such as: simultaneously develop phase Ph3_TestAndEvaluation with other phases, increasing the resources assigned to this phase or modifying the time points T5 or T7 to avoid overlapping of occurrences.

Table 5.7. Query 14.

Query 14	
Natural language interpretation	*Answer*
At what time point does the occurrence Ph2_Rev end and at what time point does the occurrence Ph3Rev end?	EndPh2Rev | EndPh3Rev | 220 | 260 |

5.1.3. Execution scenario

During the execution of an activity, any of the people involved in it can raise queries related to the needs of the execution of these activities (Figure 5.7). For example, Laura_Hall, who is responsible for the execution of Ph3_TestAndEvaluation can raise queries such as those in 15 and 16 (Table 5.8), in which the ontology is consulted about a range of aspects related to the execution of the occurrence.

5.2. Phase and step level

The workflows triggered at the project level produce the launch of other workflows at a phase level in which the primitive occurrences that make them up, the resources involved, or other related factors are shown. For example, Development is the occurrence of a primitive activity (Figure 5.4), for which the person responsible (Roger_Thompson) triggers a complex workflow which brings about the execution of two occurrences: Preliminary and Detailed (Figure 5.8). In this workflow, Roger_Thompson assigns the occurrence Phase2_WDevelopment to the team Ph2T, who delegate the process planning to Rose_Hall and the scheduling to Pitt_Moor, leaving the coordination and execution control under the responsibility of Roger_Thompson himself. The management of the occurrences Preliminary and Detailed is assigned to members of Ph2T who are supported by work teams set up for these occurrences. The rest of the resources that participate in Phase2_WDevelopment have also been represented in Figure 5.8, together with the occurrence in which they participate and with an indication of their mechanism, control, input and output roles. Any of the people involved in the corresponding workflow for a phase can raise queries similar to those shown throughout this section. In a similar way, as workflow decomposition is recursive, each step of a workflow can give rise to a new workflow. In the new workflow, participants can also consult the ontology according to their responsibilities and the scenarios that correspond to their level, which are similar to the scenarios of superior levels.

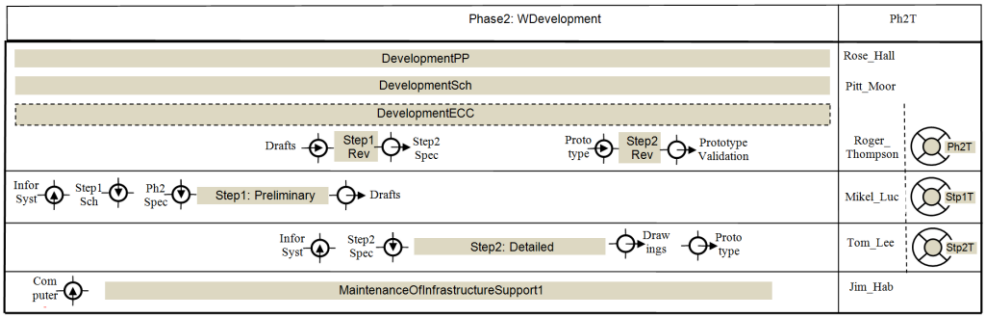


Figure 5.8. Workflow WDevelopment.

Table 5.8. Queries 15 and 16.

Query 15	
Natural language interpretation	*Answer*
Which resources participate in the occurrence Ph3_TestAndEvaluation, what roles do those resources have and what is the description of those roles?	Resource | Role | Description | TestResults | Out16 | Product evaluation results | Sch3 | Ctrl8 | Schedule for Phase3 | InforSyst | Mech1 | Execution support | TestSpec | Inp10 | Product evaluation specification | Laura_Hall | Mech6 | Person responsible for evaluation of product | Ph3T | Mech7 | Execution of product evaluation |
Query 16	
Which capabilities do the members of Ph3T group have?	Capability | C_Evaluate | C_Coordinate | C_Test | C_Coordinate | C_Produce |

6. Conclusions and future lines of research

To achieve a successful result in enterprise core processes, it is vital to share information and knowledge, both in engineering activities and operational management activities. This requirement takes on a special relevance in structuring, resources assignment, scheduling, control and monitoring of all engineering activities involved in Product and Processes Development process and when these are executed in distributed and collaborative environments. In order to satisfy, this, and taking into account the diversity and complexity of these activities (co-design, co-development and co-manufacturing), the presented ontology (PPDRC) integrates concepts belonging to different ontological theories from the domain of the company and its processes (as is the case of the theory of "situation calculus", which provides part of the semantics to PSL and TOVE, or Activity Theory) and general frameworks for the development of ontologies (DOLCE). Particular attention was paid to the incorporation of the concepts necessary to represent the social and agentive character of the resources, a fundamental aspect in collaborative processes.

In order to demonstrate the validity and general applicability of the proposal, a tool was developed that allows consultation of the ontology with queries about aspects related to planning, scheduling and control of engineering processes. In the example, the tool was applied to a New Product Development process in a virtual OKP environment, which

allowed verification of whether it was able to respond satisfactorily according to the roles of the different participants (planner, scheduler, etc.).

Currently the particularization of capabilities for machining resources according to the requirements established by the CIDP^{2}R process is being accomplished. Future lines of research are concerned with developing collaborative and expert computer aided tools that interact with the ontology through web services and represent this ontology in a foundational ontology, for later integration with other application domains.

Acknowledgments

This work has been possible thanks to funding received from the Ministry of Science and Education through the COAPP Research Project - reference DPI2007-66871-C02-01/02.

References

Ameri, F., and D. Dutta. 2008. "A Matchmaking Methodology for Supply Chain Deployment in Distributed Manufacturing Environments." *Journal of Computing and Information Science in Engineering* 8 (1): 1–9.

Baxter, D., J. Gao, K. Case, J. Harding, B. Young, S. Cochrane, and S. Dani. 2007. "An Engineering Design Knowledge Reuse Methodology Using Process Modeling." *Research in Engineering Design* 18 (1): 37–48. doi: 10.1007/s00163-007-0028-8.

Bock, C., and M. Gruninger. 2005. "PSL: A Semantic Domain for Flow Models." *Journal of Software and Systems Modeling* 4 (2): 209–231. doi: 10.1007/s10270-004-0066-x.

Borgo, S., and P. Leitao. 2008. "Foundations for a Core Ontology of Manufacturing." In *Ontologies: a Handbook of Principles, Concepts and Applications in Information Systems*, edited by Sharman, R., R. Kishore, and R. Ramesh, 751–775. Springer.

Bräuer, M. 2007. "Design of a Semantic Connector Model for Composition of Metamodels in the Context of Software Variability." PhD diss., Technische Universität Dresden.

Büyüközkan, G., T. Dereli, and A. Baykasoglu. 2004. "A Survey on the Methods and Tools of Concurrent New Product Development and Agile Manufacturing." *Journal of Intelligent Manufacturing* 15 (6): 731–751.

Büyüközkan, G., and J. Arsenyan. 2012. "Collaborative Product Development: a Literature Overview." *Production Planning & Control* 23 (1): 47–66. doi: 10.1080/09537287.2010.543169.

Cai, M., W. Y. Zhang, G. Chen, K. Zhang, and S. T. Li. 2010. "SWMRD: a Semantic Web-based Manufacturing Resource Discovery System for Cross-Enterprise Collaboration." *International Journal of Production Research* 48 (12): 3445–3460. doi: 10.1080/00207540902814330.

Chen, Y. M., W. S. Shir, and C. Y. Shen. 2002. "Distributed Engineering Change Management for Allied Concurrent Engineering." *International Journal of Computer Integrated Manufacturing* 15 (2): 127–151. doi: 10.1080/09511920110047181.

Cheung, W. M., D. G. Bramall, P. G. Maropoulos, J. X. Gao, and H. Aziz. 2006. "Organizational Knowledge Encapsulation and Re-Use in Collaborative Product

Development." *International Journal of Computer Integrated Manufacturing* 19 (7): 736–750. doi: 10.1080/09511920500504479.

Chungoora, N., and R. I. M. Young. 2011. "The Configuration of Design and Manufacture Knowledge Models from a Heavyweight Ontological Foundation." *International Journal of Production Research* 49 (15): 4701–4725. doi: 10.1080/00207543.2010.504754.

Deshayes, L. M., O. El Beqqali, and A. Bouras. 2005. "The Use of Process Specification Language for Cutting Processes." *International Journal of Product Development* 2 (3): 236–253.

Ferrario, R., and A. Oltramari. 2004. "Towards a Computational Ontology of Mind." Paper presented at the 3rd International Conference FOIS (Formal Ontology in Information Systems), Turin, November 4–6.

Gangemi, A., S. Borgo, C. Catenacci, and J. Lehmann. 2005. *Task Taxonomies for Knowledge Content*. *Deliverable D07 of the EU FP6 project Metokis.* Laboratory for Applied Ontology ISTC CNR.

Gruninger, M. 2009. "Using the PSL Ontology." In *Handbook on Ontologies*, edited by Staab, S., and R. Studer, 423–443. Springer Berlin Heidelberg.

ISO (International Organization for Standardization). 2004a. *Industrial Automation Systems and Integration – Industrial Manufacturing Management Data. Part 1, General Overview*. ISO 15531–1. New York: American National Standards Institute.

ISO (International Organization for Standardization). 2004b. *Industrial Automation System and Integration – Process Specification Language. Part 1, Overview and Basic Principles*. ISO 18629–1. New York: American National Standards Institute.

ISO (International Organization for Standardization). 2005. *Industrial Automation System and Integration – Industrial Manufacturing Management Data: Resources Usage Management. Part 32, Conceptual Model for Resources Usage Management Data*. ISO 15531–32. New York: American National Standards Institute.

Izhar, T. A. T., T. Torabi, M. I. Bhatti, and F. Liu. 2013. "Recent Developments in the Organization Goals Conformance Using Ontology." *Expert Systems with Applications* 40 (10): 4252–4267. doi: 10.1016/j.eswa.2013.01.025.

Jang, J., B. Jeong, B. Kulvatunyou, J. Chang, and H. Cho. 2008. "Discovering and Integrating Distributed Manufacturing Services with Semantic Manufacturing Capability Profiles." *International Journal of Computer Integrated Manufacturing* 21 (6): 631–646. doi: 10.1080/09511920701350920.

Karanasios, S., J. Mishra, D. Allen, A. Norman, D. Thakker, and L. Lau. 2011. "Capturing Real-World Activity: A Socio-Technical Approach." Paper presented at the eChallenges e-2011 Conference.

Kethers, S. 2000. "Multi-Perspective Modeling and Analysis of Cooperation Processes." PhD diss., RWTH Aachen.

Lambert, D., A. Saulwick, C. Nowak, M. Oxenhan, and D. O'Dea. 2008. *An Overview of Conceptual Frameworks. DSTO-TR-2163*. Command, Control, Communications and

Intelligence Division, Defence Science and Technology Organisation, Department of Defence, Australian Government.

Li, B. M., S. Q. Xie, and X. Xu. 2011. "Recent Development of Knowledge-Based Systems, Methods and Tools for One-of-a-Kind Production. *Knowledge-Based Systems* 24 (7): 1108–1119. doi: 10.1016/j.knosys.2011.05.005.

Lin, Y. 2008. "Semantic Annotation for Process Models: Facilitating Process Knowledge Management via Semantic Interoperability." PhD diss., Norwegian University of Science and Technology.

Lin, H. K., J. A. Harding, and M. Shahbaz. 2004. "Manufacturing System Engineering Ontology for Semantic Interoperability Across Extended Project Teams." *International Journal of Production Research* 42 (24): 5099–5118. doi: 10.1080/00207540412331281999.

Lin, L. F., W. Y. Zhang, Y. C. Lou, C. Y. Chu, and M. Cai. 2011. "Developing Manufacturing Ontologies for Knowledge Reuse in Distributed Manufacturing Environment." *International Journal of Production Research* 49 (2): 343–359. doi: 10.1080/00207540903349021.

Lohse, N., H. Hirani, S. Ratchev, and M. Turitto. 2005. "An Ontology for the Definition and Validation of Assembly Processes for Evolvable Assembly Systems." Paper presented at the 6th IEEE International Symposium on Assembly and Task Planning: From Nano to Macro Assembly and Manufacturing, Montreal, July 19–21.

Masolo, C., S. Borgo, A. Gangemi, N. Guarino, and A. Oltramari. 2003. *WonderWeb Deliverable D18. Ontology Library (final)*. Laboratory for Applied Ontology ISTC CNR.

McGrath, M. E. 2004. The Next Generation Product Development. How to Increase Productivity, Cut Costs, and Reduce Cycle Times. McGraw-Hill.

Mili, H., G. Tremblay, and G. B. Jaoude. 2010. "Business Process Modeling Languages: Sorting Through the Alphabet Soup." *ACM Computing Surveys* 43 (1): 4. doi: 10.1145/1824795.1824799.

NIST (National Institute for Standards and Technology). 1993. Draft Federal Information Processing Standards Publication 183. Announcing the Standard for Integration Definition for Function Modeling (IDEF0).

Oberle, D., A. Ankolekar, P. Hitzler, P. Cimiano, M. Sintek, M. Kiesel, B. Mougouie, et al. 2007. "DOLCE ergo SUMO: On Foundational and Domain Models in the SmartWeb Integrated Ontology (SWIntO)." *Web Semantics: Science, Services and Agents on the World Wide Web* 5 (3): 156–174.

O'Leary, D. E. 2010. "Enterprise Ontologies: Review and an Activity Theory Approach." *International Journal of Accounting Information Systems* 11 (4): 336–352. doi:10.1016/j.accinf.2010.09.006.

Poli, R., M. Healy, and A. Kameas, eds. 2010. Theory and Applications of Ontology: Computer Applications. Springer.

PPDRC ontology. 2013. Accessed June http://www.coapp.es/ontologies/2013/0/PPDRC_v1.owl.

Qiao, L., S. Kao, and Y. Zhang. 2011. "Manufacturing Process Modelling Using Process Specification Language." *International Journal of Advanced Manufacturing Technology* 55 (5–8): 549-563. doi: 10.1007/s00170-010-3115-3.

Rajsiri, V., J. P. Lorré, F. Bénaben, and H. Pingaud. 2008. "Collaborative Process Definition Using an Ontology-Based Approach." In *Pervasive Collaborative Networks*, edited by Camarinha-Matos, L. M., and W. Picard, 205–212. Boston: Springer.

Romero, F., A. Estruch, and P. Rosado. 2009. "A Framework for the Development of an IT Platform for Collaborative and Integrated Development of Product, Process and Resources." Paper presented at the 13th International Research/Expert Conference. Trends in the Development of Machinery and Associated Technology, Hammamet, Tunisia, October 16–21.

Rosado, P., and F. Romero. 2009. "A Model for Collaborative Process Planning in a Engineering and Production Network." Paper presented at the 13th International Research/Expert Conference. Trends in the Development of Machinery and Associated Technology, Hammamet, Tunisia, October 16–21.

Shadbolt, N., W. Hall, and T. Berners-Lee. 2006. "The Semantic Web Revisited." *IEEE Intelligent Systems* 21 (3): 96–101.

Sherehiy, B., W. Karwowski, and J. K. Layer. 2007. "A Review of Enterprise Agility: Concepts, Frameworks, and Attributes." *International Journal of Industrial Ergonomics* 37 (5): 445–460. doi: 10.1016/j.ergon.2007.01.007.

Solano, L., F. Romero, and P. Rosado. 2010. "An Ontological Approach for Manufacturing Resources Modeling." Paper presented at the 21st International DAAAM Symposium, Zadar, Croatia, October 20–23.

Tamma, V., C. Aart, T. Moyaux, S. Paurobally, B. Lithgow-Smith, and M. Wooldridge. 2005. "An Ontological Framework for Dynamic Coordination." Paper presented at the 4th International Semantic Web Conference, Galway, Ireland, November 6–10.

Uschold, M., and M. Gruninger. 1996. "Ontologies: Principles, Methods, and Applications." *Knowledge Engineering Review* 11 (2): 93–136.

Xie, S., and Y. Tu. 2011. Rapid One-of-a Kind Product Development. Strategies, Algorithms and Tools. Springer.

Capítulo 6. ONTOLOGÍA PARA LA PLANIFICACIÓN INTEGRADA DE PROCESOS DE MECANIZADO E INSPECCIÓN CENTRADA EN CAPACIDADES DE LOS RECURSOS (ONTOLOGÍA MIRC)

1. Introducción

En este capítulo se incluye la versión original del artículo *An Ontology for Integrated Machining and Inspection Process Planning focusing on Resource Capabilities*, publicado online por la revista *International Journal of Computer Integrated Manufacturing* el 14 de enero de 2015, con DOI 10.1080/0951192X.2014.1003149 y que estará disponible en el siguiente enlace permanente: http://dx.doi.org/10.1080/0951192X.2014.1003149.

En él se describen los fundamentos de la ontología MIRC (Manufacturing and Inspection Resource Capability ontology), presentándose en el anexo 7 una información más completa de su alcance y posibilidades, así como la descripción de su taxonomía y de sus predicados. MIRC es una especialización de la ontología PPDRC, descrita en el capítulo anterior. Por tanto, hereda de ésta la capacidad para soportar las actividades de la planificación colaborativa –en este caso vinculadas con la planificación de determinados procesos de fabricación (mecanizado e inspección)– desarrolladas en el contexto de una OKP virtual y para representar el carácter social y agentivo de los recursos requeridos –en este caso equipos y útiles de mecanizado e inspección–. La especialización de los conceptos de PPDRC proporciona a la ontología MIRC la interoperabilidad semántica requerida para su integración con otras ontologías de los dominios de la fabricación y el desarrollo de producto.

Por ello, el artículo, después de una breve introducción, presenta una sección que resume los conceptos de la ontología PPDRC. Se trata de una sección que el lector puede omitir, en el caso de que haya leído el capítulo anterior. A continuación, en la siguiente sección, se describe la ontología MIRC, comenzando con la definición del marco conceptual y continuando con dos subsecciones centradas en los dos grandes grupos de actividades que

forman parte de un plan de proceso de mecanizado e inspección. Una sección que finaliza con unos breves apuntes sobre la implementación de la ontología MIRC. Posteriormente se presenta otra sección dedicada al análisis de un caso de estudio, para finalizar con las conclusiones y trabajos futuros que el lector puede encontrar en el capítulo 7 de este documento.

Pero antes de mostrar literalmente el artículo, se van a presentar algunos conceptos y contenidos que pueden ayudar y/o complementar la lectura del mismo. La ontología MIRC permite representar tanto los recursos de mecanizado e inspección y sus capacidades dimensionales y geométricas como las actividades involucradas en un plan de mecanizado e inspección para una pieza. Esto es así porque para asegurar que una pieza cumple con las exigencias de calidad establecidas, expresadas mediante los requisitos dimensionales y geométricos correspondientes, no es suficiente con asignar a cada una de las actividades unos de los recursos compatibles con el grado de desempeño requerido, sino que esta asignación deben realizarse en el contexto de un plan, que establece la estructura (fase, subfase y operación) y dependencias entre estas actividades. La selección y asignación de estos recursos en el contexto de un plan de mecanizado e inspección es la que determina la eficiencia del mismo, especialmente en entornos distribuidos y colaborativos, como el que corresponde al proceso de desarrollo integrado y colaborativo de productos en una OKP virtual.

En la ontología MIRC se especializan cuatro entidades de la ontología PPDRC: *Resource*, *ActivityType*, *Capability* y *Region*, que representan los conceptos esenciales utilizados en la definición de un plan de proceso (recursos y tipos de actividades) y aquellos que son necesarios para la asignación de recursos a las actividades en la creación y validación del plan de proceso (capacidades y su cuantificación). De igual manera que PPRC, la ontología MIRC se ha desarrollado con el editor de ontologías *Protégé*, y su implementación ha sido realizada con OWL (Ontology Web Language) y SWRL (Semantic Web Rule Language).

Dado que una planificación de procesos de mecanizado e inspección contempla tanto las propias actividades de mecanizado e inspección como otras relacionadas con la preparación de los medios empleados para su ejecución, en la ontología MIRC se consideran dos tipos de actividades: *Operation activities* y *Preparation activities.* En las primeras se incluyen aquellas actividades que se realizan sobre la pieza modificando sus características, mientras que en las de tipo *Preparation* se recogen las actividades que se realizan sobre los recursos con el objeto de modificar sus características.

En la ontología, un recurso es un objeto que puede ejecutar alguna actividad exhibiendo un comportamiento activo. Sin embargo, éste puede tener un comportamiento pasivo cuando estando involucrado en la actividad no tiene responsabilidad en su ejecución. El comportamiento activo corresponde a un rol de participación en la actividad de tipo mecanismo (*Mechanism*), mientras que los roles de tipo entrada, salida y control (*Input*, *Output* y *Control*) corresponden a un comportamiento pasivo. En el primero de estos comportamientos (activo) el recurso transmite los valores que cuantifican sus capacidades a las características del objeto resultante. Esa transmisión de características en la ejecución de la actividad se materializa a través de una interfaz entre el recurso y el objeto procesado, que introduce una dispersión adicional condicionada por el tipo de control bajo el que se ejecuta la actividad. En el segundo comportamiento (pasivo), el recurso es el objeto que

recibe la actividad, como ocurre en las actividades de tipo *Preparation* orientadas a la configuración del recurso.

Por otra parte, los recursos se consideran a diferentes niveles de agregación, pudiendo ser simples o complejos. Estos últimos se configuran mediante actividades de tipo *Preparation*, que modifican sus capacidades. Unas capacidades que, tanto en los recursos simples como en los complejos, también varían en el tiempo por el desempeño de sus funciones. En este sentido, las actividades de preparación (*loading* y *setup)* de los recursos de mecanizado e inspección son esenciales para determinar la adecuación de los recursos asignados según criterios tecnológicos relacionados con sus capacidades dimensionales y geométricas. Por tanto, puede afirmarse que la ontología MIRC soporta todo el conocimiento necesario para la toma de decisiones concernientes a la preparación y asignación de los recursos durante el desarrollo de un plan de proceso integrado de mecanizado e inspección, y es capaz de dar soporte a la evaluación y validación de cualquier plan.

Para esta evaluación y validación del plan se establece una metodología que facilita la comprensión del marco conceptual de la ontología MIRC. Una metodología que está soportada en unas representaciones gráficas que permiten visualizar tanto el plan en su conjunto, mostrando la secuencia de actividades de procesado y de configuración de los recursos, como el detalle de cada una de las etapas que lo componen. Además, en las representaciones gráficas de detalle se muestra el tipo y la cuantificación de las características asociadas a los recursos y objetos que intervienen en la realización de las actividades. Específicamente, estos grafos sirven para mostrar el efecto de las capacidades de los recursos y de la ejecución de la actividad (interfaz) sobre las características de los objetos o recursos resultantes.

Para mostrar la validez de la propuesta se incluye un caso de uso, en el que se puede ver cómo el planificador de procesos puede consultar la ontología, mediante *queries*, con el propósito de obtener el conocimiento necesario para la toma de decisiones, y cómo puede utilizar las representaciones gráficas para la asignación y validación de los recursos. Un ejemplo de uso que pone de relieve la flexibilidad aportada por la ontología y la metodología propuestas para el desarrollo de planes de proceso con distintos enfoques (variante y generativo) y estrategias (hacia adelante y hacia atrás) en cualquier nivel de la planificación de procesos (agregado, supervisor y operacional).

2. Contenido del artículo “An ontology for integrated machining and inspection process planning focusing on resource capabilities.”

International Journal of Computer Integrated Manufacturing, 2015
http://dx.doi.org/10.1080/0951192X.2014.1003149

Received: 26 February 2014
Accepted: 26 November 2014
Published on line: 14 January 2015

An ontology for integrated machining and inspection process planning focusing on resource capabilities

Lorenzo Solano [1]*, Fernando Romero [2], Pedro Rosado [3]

[1,3] *Departamento de Ingeniería Mecánica y de Materiales, Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain.*

[2] *Departamento de Ingeniería de Sistemas Industriales y Diseño, Universitat Jaume I, Avda. Vicente Sos Baynat, 12071 Castellón, Spain*

* Corresponding autor at: Departamento de Ingeniería Mecánica y de Materiales, Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain. Tel.: +34963877000 Ext. Int.: 76273; fax: +34963877629.
E-mail address: lsolano@mcm.upv.es

[2] E-mail address: fromero@esid.uji.es Tel.: +34964728209; fax: +34964728170

[3] E-mail address: prosado@mcm.upv.es Tel.: +34963877000 Ext. Int.: 76221; fax: +34963877629

An ontology for integrated machining and inspection process planning focusing on resource capabilities

Abstract

The search for and assignment of resources is extremely important for the efficient planning of any process in a distributed environment, such as the Collaborative Product Integrated Development process. These environments require a degree of semantic interoperability, which currently can only be provided by ontological models. However, the ontological proposals centered on Resources for Machining and Inspection Process Planning have a limited reach, do not adopt a unified view of machining and inspection, and fail to express knowledge in the manner required by some of the planning tasks, as is the case with those concerned with resource assignment and plan validation. With the aim of providing a solution to these shortcomings the MIRC (Manufacturing and Inspection Resource Capabilities) ontology has been developed, as a specialist offshoot of the Product and Processes Development Resources Capabilities ontology. This ontology considers resource capabilities to be a characteristic of the resource executing any activity present in an Integrated Process Plan. Special attention is given to resource preparation activities, due to their influence on the quality of the final product. After describing the MIRC ontology, a case study demonstrates how the ontology supports the process planning for any level, approach or plan strategy.

Keywords: resource capabilities ontology; inspection and machining resources; integrated process planning; resource assignation; process plan validation

1. Introduction

As is widely recognized, ontologies allow distributed knowledge to be used and shared, while guaranteeing semantic interoperability and integration between different applications and agents (Cai, M., W. Y. Zhang, and K. Zhang 2011), and thereby facilitating collaboration between all the parties involved. However, the proposals made up to now for extended and complex environments, characterized by inter-functional interaction, are partial and only consider some of the outlooks needed to achieve effective integration, in environments defined by a holistic management of the company (Kantola 2009). This is the case with the Collaborative Product Integrated Development processes, in which integration of the ontologies focusing on product, processes and resources is absolutely necessary (Zdravkovic and Trajanovic 2009; Ramos 2010; Honggen et al. 2012).

One of the most proven ways to integrate ontologies is based on the use of foundational ontologies (Oberle et al. 2007). Particularly, Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE), described by Masolo et al. (2003), has served as the basis for a number of ontologies used in the domain of design and manufacturing, such as ADACOR (ADAptive holonic COntrol aRchitecture) (Borgo and Leitao 2007) or the PPDRC (Product and Processes Development Resources Capabilities) ontological model

(Solano, Rosado, and Romero 2013), which represents Resource Capabilities in the Product and Processes Development process.

As with the proposal by Newman and Nassehi (2009), in the PPDRC ontology a resource is an entity characterized by the set of its capabilities. However, an important difference is that in the PPDRC ontology the capabilities of a resource are linked to the execution of activities, adopting concepts from DOLCE, Activity Theory (Kuutti 1995) and Process Specification Language – PSL – (ISO 2004b). From the first two, it takes the social and agentive character of its entities, while from PSL it takes the semantic richness necessary to support any activity planning and execution that falls within the collaborative framework proposed by Rosado and Romero (2009).

Upon these foundations, PPDRC reformulates the concepts and axioms linked to the resources and their capabilities that are present in generic proposals, such as MANDATE (ISO 2004a), ISA-95 (ISA 2000), ADACOR and TOVE (Fadel, Fox, and Gruninger 1994), or that belong to specific proposals for the service-oriented manufacturing domain (Ning, Tian-guo, and Wen-jian 2010; Ameri et al. 2011). With the exception of MANDATE, they all characterize the resource from only a partial perspective, focusing on operational capabilities (such as production capacity, throughput time, cost per unit, etc.) with the objective of supporting production planning and control activities, but ignoring the technological capacities, which are fundamental for decision-making in process planning. These representations, which see capabilities as static and individual properties inherent in the resource and independent of state and conditions of use, have been improved in the PPDRC. For this purpose, it considers that: a) resources can participate in the execution of an activity with different roles; b) the capability associated to a resource executing an activity depends on the type of activity; c) capability is conditioned by the input objects' fulfilling certain requirements; d) capabilities change as a consequence of the resource preparation activities and other uncontrolled causes that may occur throughout its lifetime; e) the resource and its capabilities can be considered at different levels of aggregation (section, cell, machine, etc.) in order to cover the needs of the different levels of process planning (aggregated, supervisor or operational); and f) a complex resource shows behavior which goes beyond the sum of its individual parts. Thus, the PPDRC differs from others proposals by allowing the reasoning and the integration of planning, programming and control of intelligent systems. Furthermore, it can support web manufacturing services, like the validation of resource selection in a process plan, which are not covered by other ontologies centered only on resource assignment.

In the machining and inspection process planning domain, and particularly at the supervisor level (setup selection and sequencing, assignment of machines and tools, process plan validation, etc.) few proposals can be found which deal with the technical characteristics of resources in sufficient detail. Vichare et al. (2009) define a resource model concerned with setups and the dimensional and geometric validation of solutions, but it does not include the precision of the kinematic characteristics of resources and so it does not establish relationships with manufacturing process and product models. Newman and Nassehi (2009) refer to the status of devices over time, and define a resource capability profile as an aggregation of the individual profiles corresponding to the tool, fixture, machine axes and part family production policies; likewise, they also propose monitoring and prediction strategies to update the current status of the resource capability profile. In the area of

inspection, research contributions are scarce (Martínez-Pellitero et al. 2011) and do not deal with the representation of measurement capabilities of resources.

From the review of publications that was carried out, it is clear that: a) the approach adopted by the generic proposals, including the PPDRC, does not respond to the specific needs of integrated, collaborative planning focusing on the supervisor level; and b) the specific proposals for machining and inspection have a limited reach and do not represent an integral approach. To solve these shortcomings, the MIRC ontology (Manufacturing and Inspection Resource Capability), which is described in this paper, specializes the concepts and predicates of the PPDRC ontology in order to represent the capabilities of resources in machining and inspection operations and in the preparation activities involved therein. To achieve this objective it is important to take into account the difficulty inherent to the activities involved in machining and inspection integrated process planning in order to establish a set of machining and inspection operations and their sequence for satisfying all the quality requirements of the machined part. This process will be influenced by a number of diverse factors, among which it is worth highlighting the integrated assignment and configuration of manufacturing and inspection resources. This is especially important in Virtual companies in which different agents participate in the definition of the process plan, using and exchanging shared knowledge about geometric deviations of the process and manufacturing resources, which constitute one of the most critical considerations in the successful definition of integrated machining and inspection plans.

Another relevant aspect is the integration of process planning and execution-controlling activities, which allows the final quality to be improved by taking advantage of the inspection data (in-process and post-process) in order to adapt the process plans to the actual resource capabilities. To do so, information on configuration and traceability of resources is essential (González et al. 2009).

The design of the MIRC ontology must ensure support to several queries related to collaborative and distributed process planning referring to types of machining and inspection operation; capabilities needed for an operation; resource configuration, their participation in occurrences, etc. These queries, which are known as *competency questions*, can be seen in the case study (section 4) centered on the planner's task of calculating or estimating the resource capability values bearing in mind the influence factors. Previously, in sections 2 and 3, the proposals adopted (PPDRC ontology and MIRC ontology) are shown.

2. PPDRC ontology

On the first level of the PPDRC ontology, three types of mutually exclusive entities can be found: *Object*, *ActivityOccurrence* and *Region* (Figure 6.1). These entities are specializations of *Endurant*, *Perdurant* and *Quality* respectively, which together with *Abstract* are the four entities at the first level of DOLCE. An *Object* is a tangible or intangible entity with existence (Solano, Romero, and Rosado 2010). *ActivityOccurrences* are executions of activities and they use, produce and transform objects. Finally, the entity *Region* quantifies, via a field and a value, the qualities of the other entities.

In the PPDRC ontology, social objects are of great importance and consist of shared descriptions which allow mutual understanding between members of a community (Ferrario and Oltramari 2004). *SocialObjects* include: *Activities*, *ActivityTypes*, *Resources*, *Roles* and

Characteristics. *Activities* are the basic entities that make up the process plan and, when carried out (*ActivityOccurrences*), represent the execution of the plan. The entity *ActivityType* represents the types to which *Activities* belong. *Resources* are objects which have the competence or ability (capability) to carry out an activity, and reach a particular level of performance in this execution. A resource is a social object that has capabilities and is linked to a physical object. Role is description of how a physical object participates in the execution of an activity, and may be: *Mechanism*, *Input*, *Output* or *Control*. *Characteristic* is an entity whose individuals express the qualities of other individuals. Any object can have a relationship with *Characteristic* via the predicate *characterizedBy* (*Object*, *Characterisitic*), while the predicate *parametrizes* (*Characteristic*, *Region*) expresses the relationship existing between a characteristic and the regions that quantify it (Figure 6.2).

The entity *Capability* is a specialization of *Characteristic* that characterizes the use of a resource executing a particular type of activity. A capability is the ability to carry out a type of activity with a level of performance that is quantifiable via regions. The relationship that exists between a capability and its regions is expressed through the predicate *parametrizes* (*Capability*, *Region*), which has two specializations: *parametrizes_Occ* and *parametrizes_Object*. In the first type, the regions of the resource capabilities (production rates, power, time, dimensions, etc.) are not transmitted to the object on which the activity is carried out, while the regions linked to *paramatrizes_Object* are transmitted.

To consider the influence of the characteristics of the object that is transformed in the activity, the ontology uses the predicate *requires* (*Capability*, *Region*), which expresses the resource's capability requirements to the initial object characteristics. Therefore, the predicate *requires* implies the existence of restrictions on the object with *Input* role so that the capability of the resource can be considered at its specified value. A particularly interesting object role in the PPDRC ontology is the *Mechanism*, because these objects support execution of the activity and influence the quality of the result (output). Furthermore, a physical object can fulfill various roles in the same occurrence, for example, input and mechanism. The object that has an *Output* role in the execution of an activity obtains a characteristic quantified by the capability regions of the resource (regions linked to *parametrizes_Object*). Finally, the object with the *Control* role specifies the execution conditions required to produce the desired results.

In general, the changes that are produced in a physical object when an activity is executed are shown by the regions of its characteristics. However, on occasions these changes are not reflected in the physical characteristics of the object, as they only affect its social dimension. (as the resulting characteristics of a part verification activity). Similarly, changes in a resource are shown by the regions of its capabilities, and may be due to logical changes, changes in data or changes to location of the resource. These modifications to objects (or resources) that are not accompanied by physical changes lead to versions of the object (or resource) which are related through the transitive predicate *hasVersion* (*Object*, *Object*). The versions of an object may correspond, for example, to its current situation, compared to its future planned state, or the optimum capability provided by the manufacturer compared to the real capability, which becomes known after the verification of the results obtained.

This predicate allows the management of the historical information and traceability associated to resource changes.

Finally, another relevant aspect in the PPDRC ontology refers to complex resource configurations management. The predicate *relatedTo* (*Object*, *Object*) shows the relationships that exist between physical resources that are physically connected. Those connections between elemental resources are represented by interfaces, which are associated to certain types of activity (assembly, fixing, setup, etc.) and their characteristics are quantified by the resource capability executing that activity.

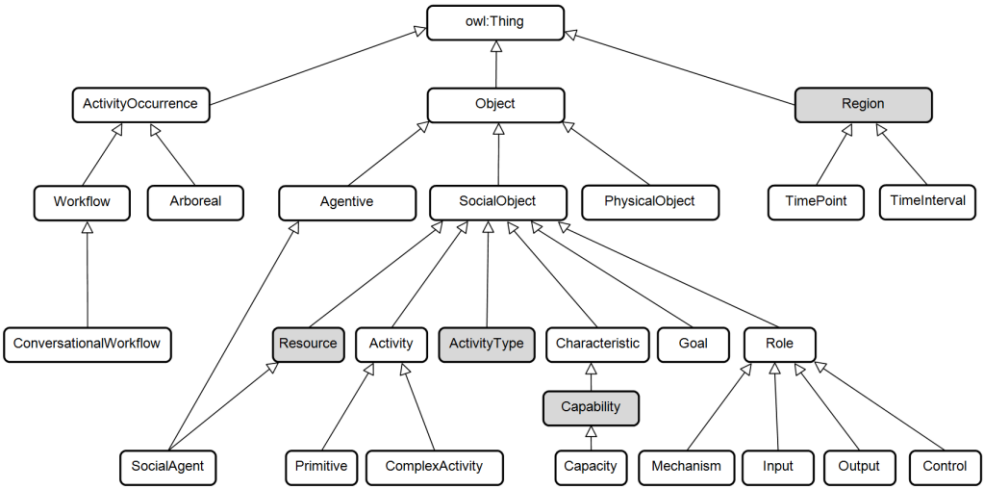


Figure 6.1. Taxonomy of the PPDRC ontology.

3. Description of the Manufacturing and Inspection Resource Capabilities ontology

As mentioned earlier, the PPDRC ontology can be tailored to diverse application domains, such as that of technological planning. This is the case of the MIRC ontology, which can tailor activities and capabilities of manufacturing resources with the aim of supporting the knowledge required to take the decisions concerning the selection, assignment and preparation of resources during the development of a machining and inspection integrated process plan. In the PPDRC, a process plan is seen as a collection of planned activities to be executed with resources that have a mechanism role, with the purpose of producing a part (*Object*) with particular characteristics.

Among the various activity types, one can find the following (Figure 6.2): a) transform the physical characteristics of the part (such as *Machining Operation*); b) obtain information on the physical characteristics of the part (such as *Inspection Operation*); c) modify the location, state or storage conditions of the part (such as *Transport* and *Storage*); and d) create or modify the characteristics of the resources that participate in any activity of the plan (such as *Preparation*).

To reach this goal, the *Resource*, *Capability*, *Region* and *ActivityType* entities are specialized (Figure 6.2). The following subsection is dedicated to the description of the first three entities, and to the conceptualization and representation of the activity entity. In the remaining subsections, the specific aspects of activity types *Operation* and *Preparation* that are relevant in the manufacturing and inspection process planning are described.

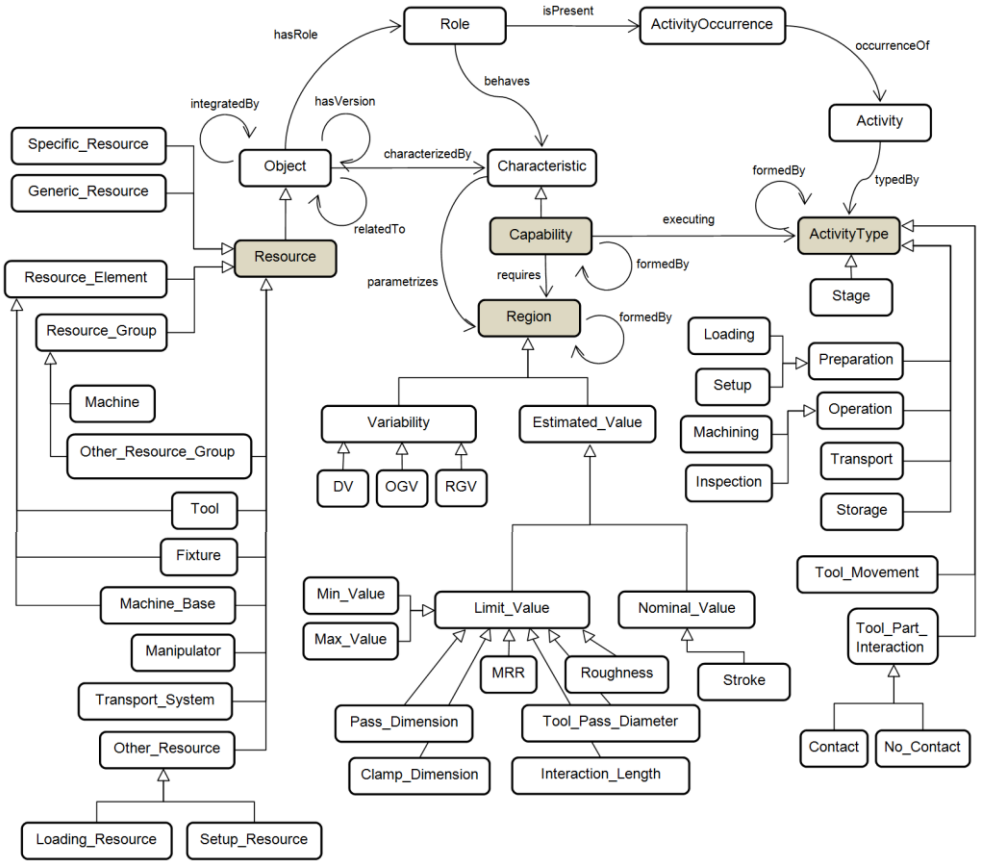


Figure 6.2. Predicates of the PPDRC ontology, and taxonomy of the entities *Resource*, *Regio*n and *ActivityType* in the MIRC ontology.

3.1. Framework of the proposal

3.1.1. Activity. Conceptualization and representations

An essential element in the MIRC ontology is the conceptualization of *Operation* and *Preparation* activities, which is based on the following principles: a) the characteristics of the output object are obtained through the composition of the resource capabilities and through the characteristics of the execution of the activity itself (interface), which can be viewed as a link in the activity chain defining the characteristic composition; b) the capabilities of a resource are conditioned by the fulfillment of certain characteristics of the input object, which therefore participate indirectly in the activity; and c) the characteristics of the interface are regulated by the characteristics that control the activity, which also participate indirectly.

A link in that activity chain, associated to the execution of the activity, is represented in Figure 6.3 (a). It shows how the characteristics of the output object, sketched by the double solid arc, depend on the characteristics of the objects that take part in the execution of an activity with *Input*, *Mechanism* and *Control* roles, sketched by the dotted arc, solid arc and

solid/dotted arc respectively, and on the characteristics of the interface of the activity, which acts as a hinge (interface) on an axis representing the execution of the activity. A three-dimensional graph, which is replaced by a plane representation (Figure 6.3 (b)) in the rest of the paper, arising from the abatement of a combination of *Input* (I), *Mechanism* (M), *Control* (C) and *Output* (O) planes. In that same figure, the arc of the input object's characteristic connects with the capabilities of the resource and, similarly, that of the control object connects with that of the characteristics of the activity interface. Additionally, Figure 6.3 (c) shows how object characteristics with *Input* and *Mechanism* roles are the object characteristics with an output role in the execution of the aforementioned activities, thus allowing the definition of the chains that represent the plan (e.g. Figure 6.5 and Figure 6.6).

These activity representations show that in order to guarantee some particular characteristics of the output part, it is necessary to restrict certain characteristics of the participating objects and activity interfaces. This is relevant in process planning, in which a backward strategy is usually adopted, meaning that the output object characteristics from the last activity of the plan will decide the characteristics of all the objects involved (input, mechanism, control) and activity interfaces present in the plan.

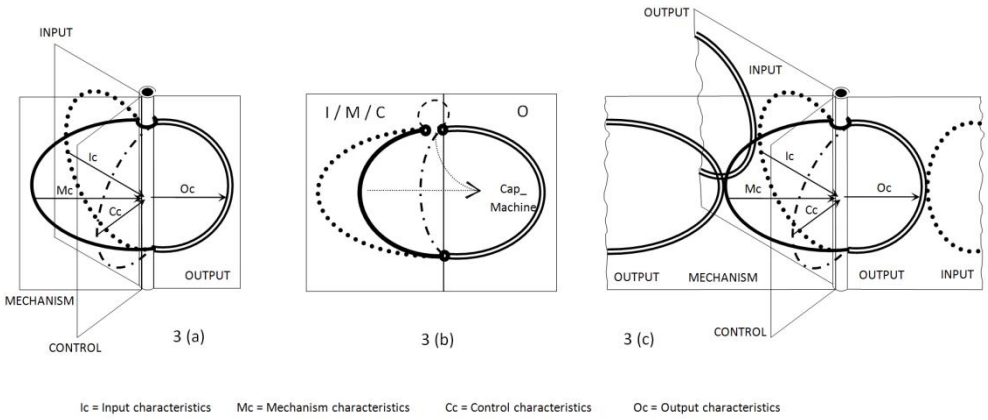


Figure 6.3. Spatial and plane representation of the characteristics associated to the execution of an activity.

Similarly, a projected view shows arrows representing the traces of the I, M and O planes and the direction of the progress of the plan. In this representation, the objects that participate in the execution of an activity with a particular role can be seen, along with a chain of activities. This allows easy visualization of the structure of the plan's activities. These projected views complement the previous ones by making it possible to see some details about the activity chain, such as: a) that in an atomic (indivisible) activity, only one resource (simple or complex) can participate with a *Mechanism* role, as the resource characteristic is that which is transmitted to the output object; and b) that a capability of a resource executing an activity may require various characteristics of one or more input objects. Figure 6.11 shows these details for a chain of preparation activities.

Although in the previous representations all the activities are atomic, the plan activities, which are understood to be a pattern of repeatable actions, can also be complex, as is the case with *Stage*. The individuals of the type *Stage* are groups of *Activities*, made up of all the machining and inspection operations that share the same resource, like: a) *Machine_Base*; b) *Machine_Base* and *Fixture*; or c) *Machine_Base*, *Fixture* and *Tool*. *Activities* that make up a complex activity, such as those of the type *Stage* can be executed sequentially and/or simultaneously. However, atomic activities can only be decomposed into other activities which are executed concurrently, as is the case for an activity of the type *Operation*, which involves the concurrent execution of atomic activities of the types *Tool_Movement* and *Tool_Part_Interaction* (Figure 6.2). In process planning, the selection of resources is usually carried out at the level of these atomic activities, with the later possibility of grouping them into complex activities, which share, in a spatial and temporal sense, the same resources (plan stage) with the aim of reducing preparation time. With this in mind, the assignment of mechanism resources is only carried out at the level of atomic activities of the type *Operation* (*Machining* and *Inspection*) and *Preparation*. Thus, it is guaranteed that any change in part characteristics is only due to the resource characteristics executing a type of activity.

When Geometric and Dimensional (GD) characteristics are the focus of analysis, it must be considered that interface characteristics are established between the active geometries during the activity – both those of the mechanism object and those of the output object. These interfaces represent the physical interaction of the object with the resource (mechanical, electromagnetic, etc.) during activity execution. For example, in the case of a *Machining* operation, both the relationship part-fixture and the relationship tool-part are considered.

3.1.2. Resource selection. Groups and capabilities

As the selection of a specific resource capable of executing an atomic activity is usually carried out from groups of resources that have similar capabilities, the MIRC ontology considers the type *Generic_Resource* formed by all *Specific_Resources* related through the predicate *includes* (*Resource*, *Resource*). The capability that characterizes a generic resource concerns both the types of activities that can be executed as well as the levels of achievement reached. Following on from this, a generic resource is understood to be an abstract resource whose participation in an activity implies the participation of one of the specific resources included within it.

The *Resource* has been presented as if it were an elemental entity, although it is usually integrated by other elemental resources (*Resource_Element*) or complex resources (*Resource_Group*) which act together in the execution of the atomic activity. This relationship is established through the predicate *integratedBy* (*Resource*, *Resource*), inherited from PPDRC objects, which expresses the grouping of the social objects that make up the social object *Resource_Group*.

The specializations related to the PPDRC predicate *relatedTo* (*relatedToPart* and *relatedToTool*) express the connections between the resources that make up a resource group and the direction of the connection. This allows management of the particular resource configurations. However, it is worth noting that the characteristics of the resulting object, in this case the capabilities of a resulting resource group, depend on: a) the capabilities of the resource elements or resource groups which form them; and b) the

characteristics of the resources (capabilities) and interfaces corresponding to the activities of the type *Preparation* (*Loading* or *Loading* plus *Setup*).

3.1.3. Object characterization. Regions

Looking a little deeper into the quantification of capabilities through regions, it is important to remember that the values can be obtained from: a) historical generic data, coming from catalogues, manuals and reports, which consider a range of execution methods and conditions; or b) data considering specific execution conditions. In both cases, the data can correspond to different levels of aggregation of the resource and may have been obtained directly by measurement at this level or as a collection of results from lower levels. This latter case highlights the importance of having data available from elemental resources. The predicate *hasVersion* allows relationships to be established between resources that are differentiated by the values (regions) of their characteristics.

Another aspect worth highlighting is the way to quantify a characteristic. To do so, the MIRC ontology establishes three types of regions that consider the way to compose the values: *Variability*, *Nominal_Value* and *Limit_Value* (Figure 6.2). Quadratic composition rules (or similar) are applied for regions of type *Variability*, whereas in the regions of the type *Limit_Value*, the maximum or minimum values of all the regions considered are taken. Finally, in the regions of type *Nominal_Value*, the composition is carried out using the algebraic sum.

To consider dimensional and geometrical aspects, three specializations for *Variability* regions have been considered (Figure 6.2): *Dimensional_Variability* (*DV*), *Own_Geometric_Variability* (*OGV*), and *Reference_Geometric_Variability* (*RGV*). *DV* regions express variability in lengths and angles. *OGV* regions express intrinsic geometric variability, for example flatness or roundness. Finally, *RGV* regions express orientation and position variability, such as parallelism or perpendicularity. Another region needed to complement dimensional and geometrical specifications is *Roughness*, which is a *Limit_Value* region.

3.2. Activities of type Operation

After the most relevant aspects of the conceptual framework of the proposal have been outlined, the current section focuses on activity of the type *Operation*. An *Operation* is a concurrent grouping of activities of types *Tool_Part_Interaction* and *Tool_Movement*. *Tool_Part_Interaction* represents the interaction between tool and part, which involves removing material or inspecting the part, and *Tool_Movement* represents the relative movement between the tool/probe and the part. A *Tool_Movement* is the combination of movements (linear or rotatory) that are defined by the machining and inspection operation strategies. In turn, the individuals of the type *Tool_Part_Interaction* can be of two types: *Contact* and *No_Contact*, depending on the characteristics of the interface between tool/probe and part (Figure 6.2).

The *Resource_Group* that participates with a *Mechanism* role in the *Operations* is called *Machine* (Figure 6.2) and comprises a *Machine_Base* individual and one or more *Tool* and/or *Fixture* individuals. The *Machine_Base* is characterized by the capabilities to execute *Tool_Movement* activities, while *Tool* and *Fixture* are characterized by capabilities to execute *Tool_Part_Interaction* and locating/fixing activities respectively.

As was shown in the previous section, in the GD area, final characteristics of parts are determined directly by the GD capabilities of the machine that executes the operation and the GD characteristics of the operation interface, considering tool-part interaction (Figure 6.4). An interface quantifies the discrepancies that exist between machine capabilities in real execution conditions and in those used in the operations that served for the quantification of machine capabilities (test and historical). This is because the estimated capabilities are always closely linked with the actual loading level and the specific type of control.

As shown in Fig 4(b), various GD characteristics of the part can be generated in a machining operation. This circumstance, which is present in the majority of the activities, leads to multi-characteristic graphs with several loops, which show the existence of different active geometrical interfaces, both in the tool-part relationship and in the part-fixture relationship (Figure 6.4). Different resource capabilities are used in each loop.

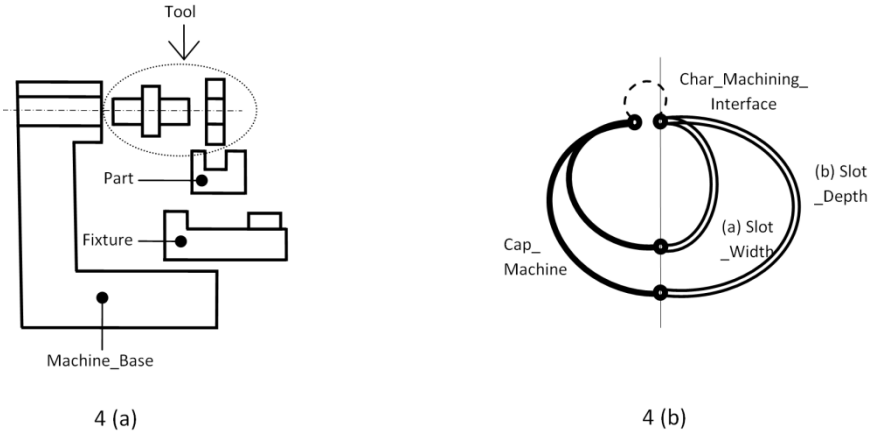


Figure 6.4. Physical representation of the resource participating in the machining of the slot (a) and graph with the characteristics of the machined slot (b).

In Figure 6.4 (b) the ends of the solid arcs represent active geometries (machined or datums) or part and resource references involved in the operation. One end of the arc representing a part characteristic is linked to the machining interface (Machining Feature), and the other is linked to a reference or active geometry of the resource.

In the case of feature's intrinsic characteristics, like slot width (Figure 6.4 (b)), one end of their representing arc is the machined geometry and the other one is a feature self-reference that will be made to coincide with a *Machine_Base* reference. Nevertheless, in the case of feature extrinsic characteristics, like slot depth (Figure 6.4 (b)), the second link corresponds to a characteristic datum, which will coincide with a local machine reference. In the next section and in the case study, it will be seen that the arc representing resource capability is composed of several arcs corresponding to individual capabilities of resources involved and interface characteristics between them.

3.3. Activities of type Preparation

As is well known, machining and inspection resource capabilities are established by a set of *Preparation* activities, needed for the formation of a *Resource_Group* and/or its modification and characterization. Hence, the *Preparation* entity has two specializations (*Loading* and *Setup*) which are not mutually exclusive (Figure 6.2), thereby allowing the existence of *Preparation* activities made up of activities of both types.

The result of a *Loading* activity is a resource that is different from those participating with an *Input* role. The output resource will have some characteristics of the input objects and will also acquire new characteristics. For example, as can be seen in Figure 6.5, the resulting resource capability (Cap_MB+F) of the activity *Fixture_Loading* will depend on the characteristics of its execution (Char_Fixture_Loading_Interface), on the capabilities of the *Loading_Resource* executing the activity (Cap_LR1), and on certain characteristics of those input objects that compose the output resource (Cap_MB and Cap_F). In *Loading* activities, the input objects act with the roles of both mechanism and input, which is a significant difference from operation activities. This particularity can be seen in Figure 6.5, where the capabilities of all participating mechanisms (Cap_MB, Cap_LR1, and Cap_F in Fixture_Loading) are transmitted to the output capability (Cap_MB+F). Furthermore, in Figure 6.5 it can be seen how the result of the *Fixture_Loading* activity participates as a mechanism in the execution of the *Tool_Loading* activity, whose result, in turn, participates in the execution of the *Part_Loading* activity, from which the resource that finally executes the machining operation is obtained. The thick horizontal broken lines, which appear in Figure 6.5 and in the following figures, represent the extension of a point on the graph. In other words, these thick broken lines do not represent characteristics.

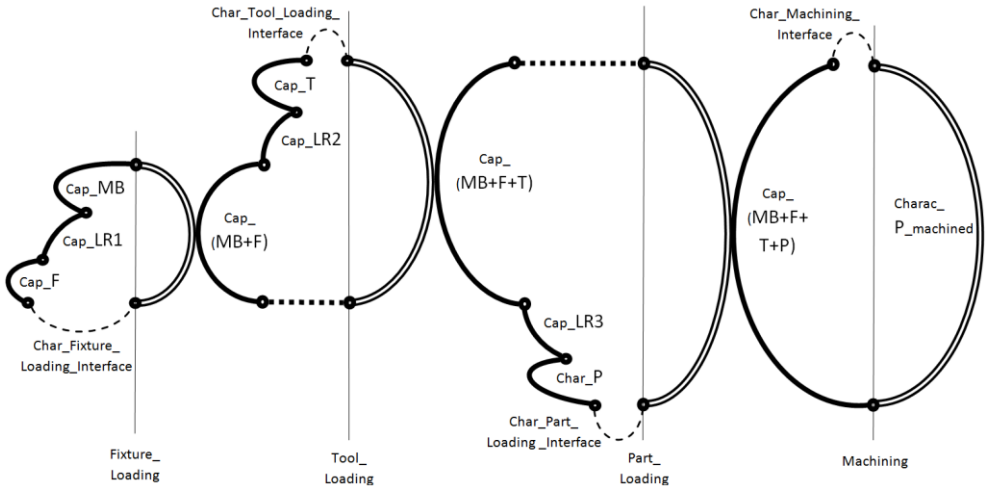


Figure 6.5. Plane view in which the execution of the three *Loading* activities are linked with the execution of a *Machining* activity.

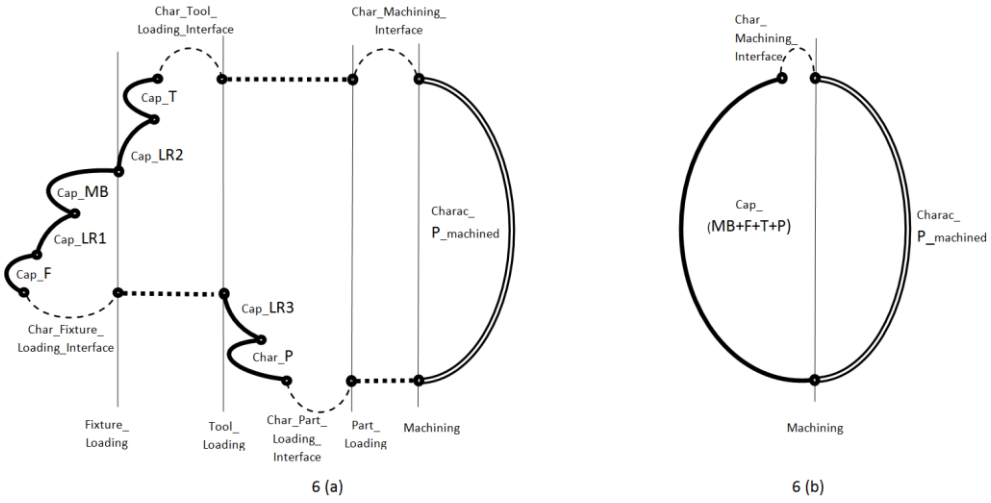


Figure 6.6. Compact view of detailed *Loading* activities to configure a resource for machining (a). Plane view of the *Machining* operation using the complex resource as an elemental resource (b).

A more compact way to represent the activity chain is shown in Figure 6.6. In the graph on the left, the arcs linking the executions of the activities have been deleted, and in the one on the right all the arcs representing mechanism capabilities are drawn in a single arc (composition).

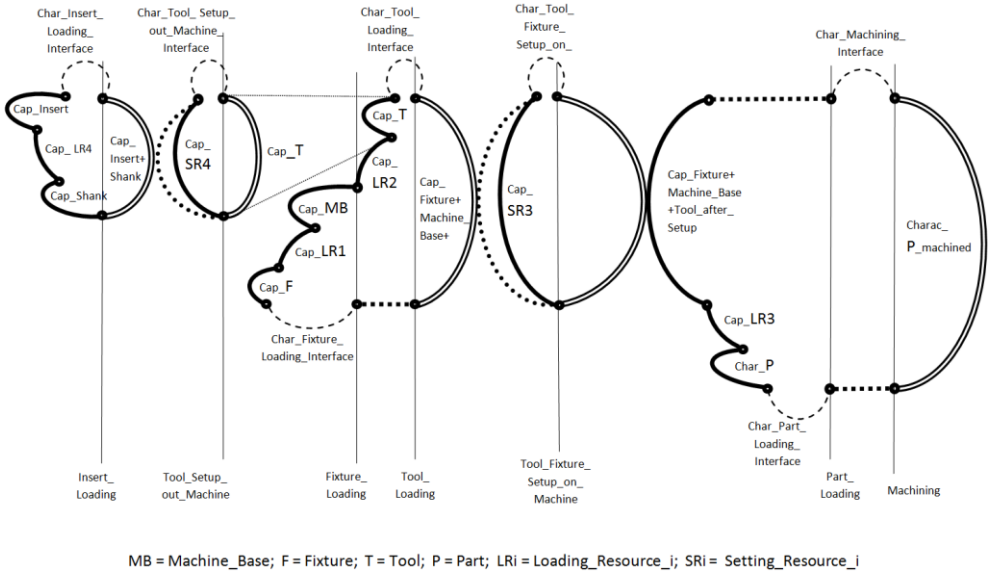


Figure 6.7. Chain of *Loading* and *Setup* operations to configure a resource for machining.

In setup activities, which include a measure and/or correction, the capabilities of a resource group are quantified directly, as if a single entity were being dealt with. As can be seen in Figure 6.7, in *Setup* activities, the input resources do not play a mechanism role and only connect with previous activities. This means that they do not transmit their capabilities to the output resource (prepared resource), which is only influenced by the *Setting_Resource* (SR3 or SR4). As is generally recognized, the inclusion of this type of activity improves the capabilities of the resources.

In the MIRC ontology (Figure 6.8) various *Setup* specializations have been considered: *Setup_on_Machine* and *Setup_out_Machine*. Figure 6.7 shows a *Setup_on_Machine* (*Tool_Fixture_Setup_on_Machine*) and a *Setup_out_Machine* (*Tool_Setup_out_Machine*). In the first type, a *Machine_Base* is always involved, which is not necessary in the *Setup_out_Machine*.

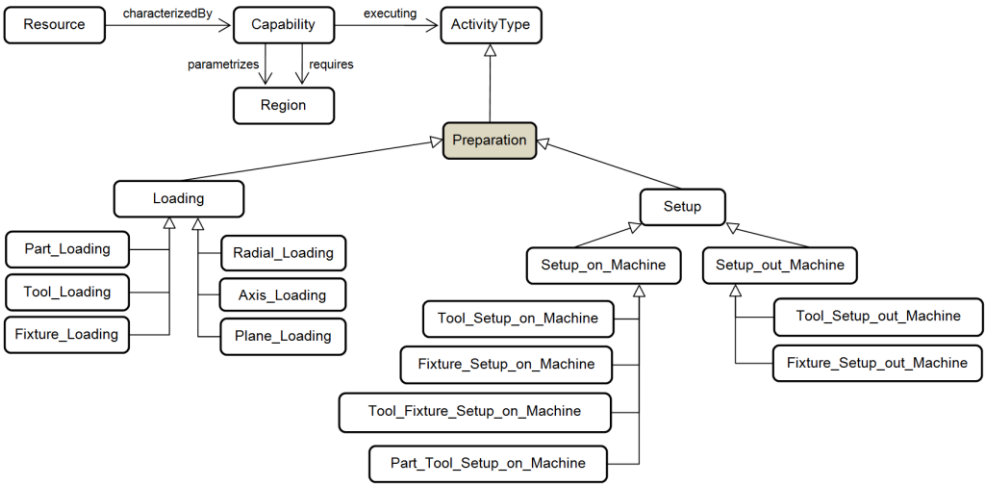


Figure 6.8. *Preparation* activities in the MIRC ontology.

3.4. Implementation of the ontology

In response to the requirement of supporting a Web-based OKP system, it was decided the ontology should be modeled using a standardized and widespread language. Therefore, the modeling of the MIRC ontology was carried out through OWL (Ontology Web Language) and SWRL (Semantic Web Rule Language), adopted by World Wide Web Consortium (Cai et al. 2010; Lin et al. 2011). SWRL improves the semantic expressivity of OWL, thus allowing it to express knowledge that cannot be directly defined with the OWL axioms (Lin 2008).

The MIRC ontology has been developed and edited using the ontology editor Protégé and can be consulted at MIRC (2014). Additionally, to check the consistency of the ontology between predicates and definitions, to maintain the ontology hierarchy, and to allow consultations, the Pellet reasoner was used. Pellet is an open-source reasoner which allows classification and reasoning with individuals in the OWL/SWRL ontologies.

In order to validate the proposal, a Java application was implemented. The application incorporates Java libraries to operate with OWL files and to reason and query with Pellet, which includes query support using the SPARQL language (Mariot et al. 2007). In that way, the application is provided with a front-end interaction interface that translates the user query into SPARQL and transforms the SPARQL query results into a more user-readable format.

4. Case study

The use of the MIRC ontology can be shown with the description of the process planning of Part_1 (Figure 6.9), which is manufactured in a virtual OKP context. This case study concentrates on decision-making tasks based on capability knowledge, as in the case of the selection and validation of machining and inspection resources. In this way, the aim is to show how the MIRC ontology facilitates distributed process planning, regardless of the level at which it is being carried out (aggregated, supervisor or operational), the approach adopted (variant or generative) and the strategy used (forward or backward).

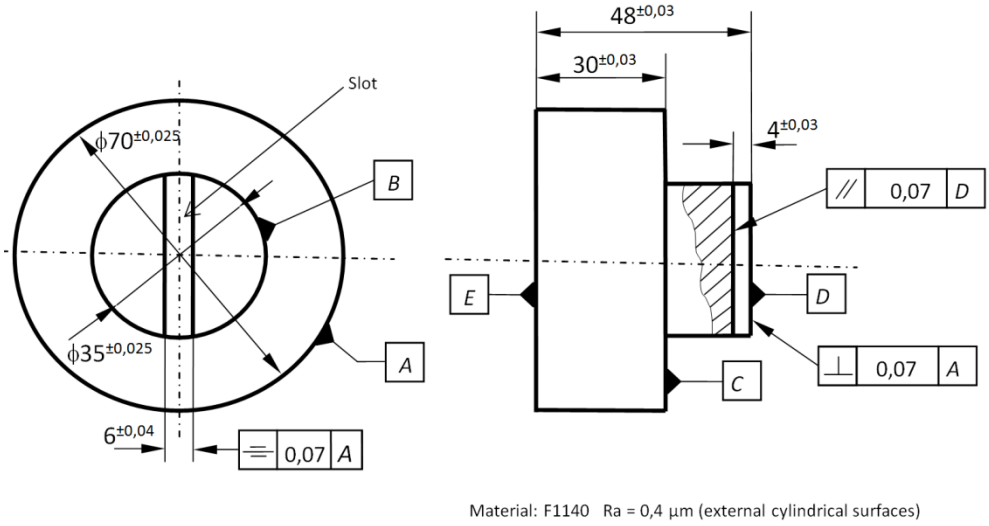


Figure 6.9. Dimensional and geometric specifications for Part_1.

To do this, it will be shown how the different planners or software agents involved can consult knowledge and data in the MIRC ontology The framework within which these planners carry out their work takes into account the relationship between process planning, resource planning and scheduling (Sormaz, Arumugan, and Rajaraman 2004) and is presented schematically presented in Figure 6.10. The companies that make up the virtual OKP (C1, C2, C3, etc.) have the capabilities to carry out the activities of the Integrated Development of Products, Processes and Resources. These companies include resources with capabilities to execute activities of *Operation* and *Preparation* types. The data relating to these generic or specific resources and to their capabilities, together with the data on the characteristics of the part machining features, are represented in the corresponding knowledge base (KB).

In the aggregated planning and adopting a backward strategy, once Part_1 has been identified as a part of type Revolution_ with_ No_Revolution_Features with a minimum value of 0.05 mm for its critical regions, the planner can define the alternative machining macro-plans that are valid for it. To this end, the planner will query the KB about the resources of the companies in the virtual OKP with capabilities to execute the required operations (Table 6.1). In this query (Query 1), the KB is asked about the generic resources that can execute the manufacturing activities for Part_1 (machining flat surfaces, cylindrical surfaces and slots) with the necessary capabilities and the blank requirements for these resource capabilities. The queries are formulated using the application developed for this purpose. In this first case, Table 6.1 shows the query in SPARQL, its interpretation in natural language and the response. In the rest of the queries only the interpretation and the results are shown. The two type of routes established from the results of Query 1 ("Answer") are: a) conventional machining with in-process inspection (type_A), and b) conventional machining, grinding and post-process inspection (type_B). In both routes, the tolerance for the raw material must be less than 0.5 mm.

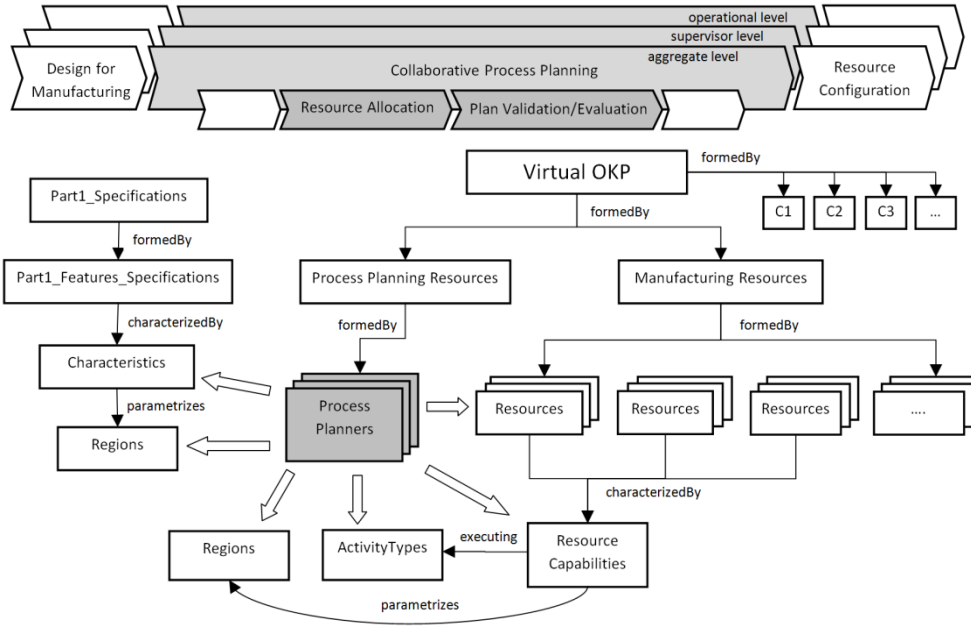


Figure 6.10. Schematic representation of the framework for process planning.

Table 6.1. Query 1.

Query in SPARKQL	

	SELECT DISTINCT ?Generic_Resource ?Valor_Required_Machining WHERE {
	?Generic_Resource a <http://www.coapp.es/ontologies/2012/10/MRO_V3.owl#Generic_Resource> .
	?Generic_Resource <http://www.coapp.es/ontologies/2011/0/CDPP_v4.owl#characterizedBy>
	?Cap_Plane_Surface .
	?Cap_Plane_Surface a (<http://www.coapp.es/ontologies/2011/0/CDPP_v4.owl#executing> some
	<http://www.coapp.es/ontologies/2012/11/Example_2_v1.owl#AT_Plane_Obtaining>) .
	?Generic_Resource <http://www.coapp.es/ontologies/2011/0/CDPP_v4.owl#characterizedBy>
	?Cap_Cyl_Surface .
	?Cap_Cyl_Surface a (<http://www.coapp.es/ontologies/2011/0/CDPP_v4.owl#executing> some
	<http://www.coapp.es/ontologies/2012/11/Example_2_v1.owl#AT_Cyl_Surface_Obtaining>) .
	?Generic_Resource <http://www.coapp.es/ontologies/2011/0/CDPP_v4.owl#characterizedBy> ?Cap_Slot .
	?Cap_Slot a (<http://www.coapp.es/ontologies/2011/0/CDPP_v4.owl#executing> some
	<http://www.coapp.es/ontologies/2012/11/Example_2_v1.owl#AT_Slot_Obtaining>) .
	?Cap_Plane_Surface <http://www.coapp.es/ontologies/2011/0/CDPP_v4.owl#parametrizes> ?Region_Plane .
	?Cap_Cyl_Surface <http://www.coapp.es/ontologies/2011/0/CDPP_v4.owl#parametrizes> ?Region_Cyl .
	?Cap_Slot <http://www.coapp.es/ontologies/2011/0/CDPP_v4.owl#parametrizes> ?Region_Slot .
	?Region_Plane <http://www.coapp.es/ontologies/2012/11/Example_2_v1.owl#valor> ?Valor_R_Plane .
	?Region_Cyl <http://www.coapp.es/ontologies/2012/11/Example_2_v1.owl#valor> ?Valor_R_Cyl .
	?Region_Slot <http://www.coapp.es/ontologies/2012/11/Example_2_v1.owl#valor> ?Valor_R_Slot .
	?Generic_Resource <http://www.coapp.es/ontologies/2011/0/CDPP_v4.owl#characterizedBy>
	?Cap_Machining .
	?Cap_Machining a (<http://www.coapp.es/ontologies/2011/0/CDPP_v4.owl#executing> some
	<http://www.coapp.es/ontologies/2012/11/Example_2_v1.owl#AT_Machining>) .
	?Cap_Machining <http://www.coapp.es/ontologies/2012/10/MRO_V3.owl#requires> ?R_Req_Machining .
	?R_Req_Machining <http://www.coapp.es/ontologies/2012/11/Example_2_v1.owl#valor>
	?Valor_Required_Machining .
	FILTER (?Valor_R_Plane < 0.05)
	FILTER (?Valor_R_Cyl < 0.05)
	FILTER (?Valor_R_Slot < 0.05)
	}
	ORDER BY DESC (?Generic_Resource)

Natural language interpretation	
	What generic resources are characterized by capabilities that parametrize regions with a value lower than 0.05 executing types of activities to obtain flat surfaces, cylindrical surfaces and slots, and what are the values of the regions required for each generic resource to execute machining activities?
Answer	
	Generic_Resource | Valor_Required_Machining | LatheCenter_A_C2 | 0.45 | LatheCenter_A_C1 | 0.5 | GrindingMachine_B_C3 | 0.1 | GrindingMachine_B_C2 | 0.15 |

In the case that a type_A route is selected, the planners must define the resources needed at supervisor level. Assuming that machining of the Slot_Feature (Figure 6.9) is the planners' focus, they can ask about resources of type *Machine* which on some occasion have executed a *Slot_Operation* with a *DV* lower than 0.06 mm. The result of this query, corresponding to a variant approach, shows the two capable machine resources (Machine_32GT and Machine_32GT_Part_1_RM) and their components (Table 6.2).

If Machine_32GT_Part_1_RM (which includes the part) is the selected resource, the planner would need a deep knowledge about the preparation activities that were used to configure this complex resource. For this purpose, a query (Table 6.3) can be launched, with the goal of obtaining the information needed to define and validate a complete plan of the preparation stage, particularly the activity sequence and their input and output objects. Later queries will allow identification of the resources that act as a mechanism in these preparation activities, thus allowing representation of the chain (Figure 6.11). Finally, to validate the selection, additional queries (Table 6.4) must still be addressed with the aim of representing the operation multi-characteristic graph (Figure 6.12) necessary for this task.

Table 6.2. Query 2.

Natural language interpretation	*Answer*
What resources of Machine type, included in the generic resource LatheCenter_A_C1 or in the generic resource LatheCenter_A_C2, have participated in an activity of type Slotting_In_LatheCenter executed using a capability which parametrizes a region of type DV with a value lower than 0.06 mm, and what are the resources that integrate these Machine type resources?	Machine | Resource | Machine_32GT | Generic_Turning_Tool_A2 | Machine_32GT | Machine_32 | Machine_32GT | LatheCenter_3 | Machine_32GT | Chuck_2 | Machine_32GT_Part_1_RM | Generic_Turning_Tool_A2 | Machine_32GT_Part_1_RM | Machine_32 | Machine_32GT_Part_1_RM | Part_1_RoughMaterial | Machine_32GT_Part_1_RM | LatheCenter_3 | Machine_32GT_Part_1_RM | Machine_32GT | Machine_32GT_Part_1_RM | Chuck_2 | Machine_32GT_Part_1_RM | Part_Axis_A | Machine_32GT_Part_1_RM | Plane_E |

Table 6.3. Query 3.

Natural language interpretation	*Answer*
What activities of type Preparation generate a resource that forms the resource Machine_32GT_Part_1_RM and what are the inputs and outputs of these activities?	Preparation_Activity | Input_Resource | Output_Resource | Act_Chuck_2_Loading | LatheCenter_3 | Machine_32 | Act_Chuck_2_Loading | Chuck_2 | Machine_32 | Act_Generic_Turning_Tool_A2_Loading | Generic_Turning_Tool_A2 | Machine_32GT | Act_Generic_Turning_Tool_A2_Loading | Machine_32 | Machine_32GT | Act_Part_1_RM_Loading | Part_1_RoughMaterial | Machine_32GT_Part_1_RM | Act_Part_1_RM_Loading | Machine_32GT | Machine_32GT_Part_1_RM |

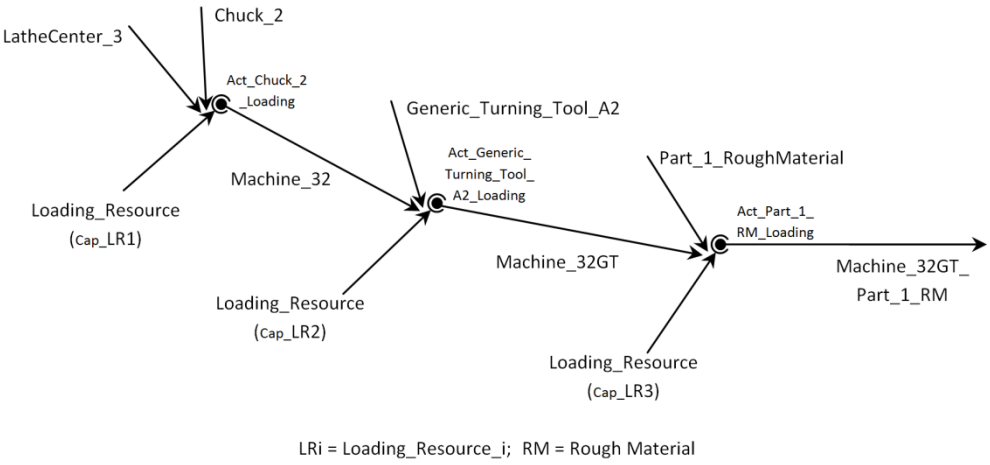


Figure 6.11. Projected view with the *Preparation* activities needed to obtain the complex resource Machine_32GT_Part_1_RM.

As can be seen in the graph in Figure 6.12, resource capabilities, operation interface and blank characteristics (left-hand part of the graph) participate in the final characteristics of

the part (right-hand part of the graph). This can be clearly seen in the loop corresponding to the Slot_Bottom_Position characteristic, in which different interface characteristics participate, particularly the one due to part loading. This Char_Part_Loading_Interface is a characteristic between the Plane_E feature and the feature of the fixture Chuck_2 with which it is matched.

Table 6.4. Queries 4 and 5.

Natural language interpretation	*Answer*
Query 4 What are the resources and features of the raw part that make up the resource Machine_32GT_Part_1_RM_Setup1, and how are these resources and features linked?	Object | Object_2 | Generic_Turning_Tool_A2 | LatheCenter_3 | LatheCenter_3 | Chuck_2 | Chuck_2 | Part_1_RoughMaterial | Chuck_2 | Part_Axis_A | Chuck_2 | Plane_E |
Query 5 What features make up the part after the machining of the slot (Part_1_Final), and what are the characteristics of these features?	Object | Characteristic | Slot_Bottom_Plane | Slot_Parallelism_to_D | Plane_D | Slot_Parallelism_to_D | Slot_Bottom_Plane | Solt_Depth | Plane_D | Solt_Depth | Slot_Middle_Plane | Slot_Symmetry_to_A | Part_Axis_A | Slot_Symmetry_to_A | Plane_E | Distance_PlaneE_PlaneD | Plane_D | Distance_PlaneE_PlaneD

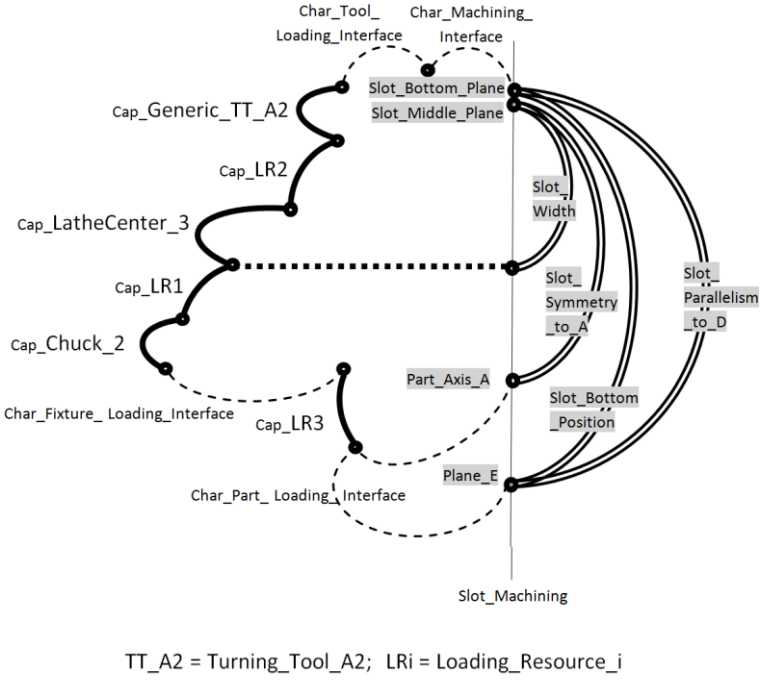


Figure 6.12. Recovered graph of a validated process plan for a *Slot Machining* operation.

Following a variant approach, the planner must adapt the recovered plan when a characteristic differs from the one in the case study. As the equivalent characteristic Slot_Depth (Figure 6.13 (a)) differs from Slot_Bottom_Position (Fig 12), in order to close the loop a new characteristic Distance_PlaneE_PlaneD is introduced, which relates the feature Slot with Plane_E. This new characteristic, resulting from stacking all the characteristics present in the loop, must be compared with the capability of the complex resource in order to validate the new process stage.

If this evaluation is positive, the stage plan thus recovered can be used directly. However, if the evaluation is negative, the planner must propose chain modifications to adjust the value of the capability of the resource to an acceptable value. To do so, the planner can follow two alternative methods: a) substitute one of the resources for another which has better capabilities; or b) incorporate *Setup* activities, as was indicated in section 3.3, allowing substitution of a loop portion by a single link.

In this case the second option is chosen, incorporating the *Part_Tool_Setup_on_Machine* activity into the plan. This alternative, which can be seen in Figure 6.13 (b), will only be valid if the measuring capability (uncertainty) of the resource that executes the *Setup* activity is less than 0.03 mm. This is a condition that is fulfilled if the setting is carried out using the on-machine measuring capabilities of the resource Machine_32GT.

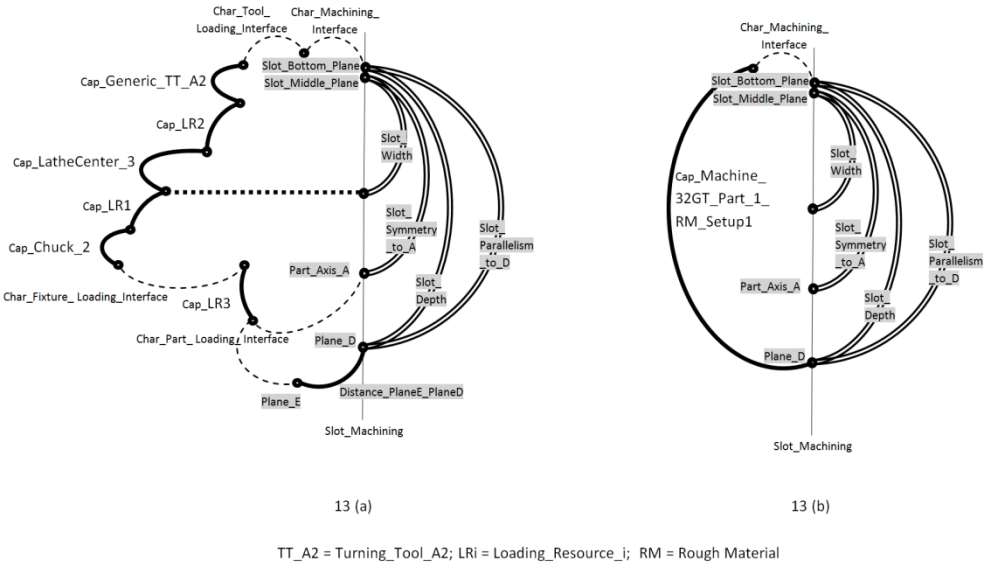


Figure 6.13. Operation multi-characteristics graph of: (a) initially adapted process plan for Part_1; (b) the modified process plan including an additional *Setup* operation.

Once the configuration of the resource that allows the critical slot characteristic (Slot_Depth) to be obtained has been defined, the validation will be extended to the rest of the critical characteristics of other features to be obtained in the same stage.

The case study has been centered on resource assignment and validation tasks at aggregated and supervisor levels. The MIRC ontology also supports knowledge representation relating to the sequencing and grouping of operations, a function that it inherits from the PPDRC ontology.

5. Conclusions and future work

The MIRC ontology is a specialization of the PPDRC ontology which details the activities and capabilities of manufacturing resources with the aim of supporting the activities that are executed during machining and inspection process planning. The fact that the PPDRC ontology is based on the foundational ontologies, PSL and DOLCE, gives the MIRC ontology the semantic inter-operability required for its integration with other specific ontologies considered within the collaborative integrated framework.

Thus, the most relevant contribution of this proposal is its capacity to represent the social and agentive character of the manufacturing resources, aiming to achieve integration of the process planning of both the product definition process and the integrated machining and inspection process.

Other relevant aspects that characterize the MIRC ontology are: a) the homogeneous treatment given to the different activities (machining and inspection operations and preparations), to the objects that participate in the execution of the activities (raw material and product in process, machines, tools, people in charge of preparation, etc.) and to the characteristics (capabilities of resources, characteristics of the part, etc.); and b) the way in which the characteristics of an output object are quantified, which means adopting values for the capabilities of the resource that executes the activity and the setting of specific values for the input and control objects.

To check the validity of the proposal a case study has been presented showing how the process planner can consult a KB, supported by the MIRC ontology, to obtain the knowledge necessary for effective decision-making concerning resource assignment tasks and validation of the solutions used. Moreover, it is obvious that the query formulation is extremely flexible, thus allowing the planner to work using different approaches (variant and generative) and strategies (forward and backward) at each level of process planning (aggregated, supervisor and operational).

To support the planner's tasks, an original graphical representation has been developed, valid for different process plan approaches and levels of detail. This representation helps visualize how GD capabilities and characteristics data from the MIRC participate and are transmitted along the execution of plan activities. This representation is the essential component of a methodology based on new concepts and relations about part characteristics, resource capabilities and interfaces, which represent the activity execution characteristics governed by the control characteristics.

Future lines of research include: a) extending the scope of the MIRC ontology to cover complex resources with a level of aggregation higher than those of type *Machine*, such as workstations, workcells and virtual workcells; b) looking more deeply into the nature and typologies of the objects with a *Control* role and their influence on the quantification of activity execution; and c) broadening the characteristics studied, which in this first part of the research has been limited to the domain of dimensional and geometric quality, to

include other resource capabilities (power, range of advances, cutting speed, material removal rate, etc.) related to different plan efficiency indicators.

Acknowledgments

This work has been possible thanks to funding received from the Spanish Ministry of Science and Education through the COAPP Research Project - reference DPI2007-66871-C02-01/02.

References

Ameri, F., Ch. McArthur, B. Asiabanpour, and M. Hayasi. 2011. "A Web-based Framework for Semantic Supplier Discovery for Discrete Part Manufacturing." Paper presented at the 39th SME/NAMRC (North American Manufacturing Research Conference), Corvallis, Oregon, June 13–17.

Borgo, S., and P. Leitao. 2007. "Foundations for a Core Ontology of Manufacturing." In *Ontologies: A Handbook of Principles, Concepts and Applications in Information Systems*, edited by R. Sharman, R. Kishore, and R. Ramesh, 751–775. New York: Springer.

Cai, M., W. Y. Zhang, G. Chen, K. Zhang, and S. T. Li. 2010. "SWMRD: a Semantic Web-based Manufacturing Resource Discovery System for Cross-Enterprise Collaboration." *International Journal of Production Research* 48 (12): 3445–3460.

Cai, M., W. Y. Zhang and K. Zhang. 2011. "ManuHub: a Semantic Web System for Ontology-based Service Management in Distributed Manufacturing Environments." *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans* 41 (3): 574–582. doi: 10.1109/TSMCA.2010.2076395.

Fadel, F. G., M. S. Fox, and M. Gruninger. 1994. "A Generic Enterprise Resource Ontology." Paper presented at the 3rd IEEE Workshop on Enabling Technologies. Infraestructure for Collaborative Enterprises, Morgantown, April 17–19.

Ferrario, R., and A. Oltramari. 2004. "Towards a Computational Ontology of Mind." Paper presented at the 3rd International Conference FOIS (Formal Ontology in Information Systems), Turin, November 4–6.

González, F., F. Romero, G. Bruscas, and S. Gutiérrez. 2009. "Modelado de Actividades para el Desarrollo Integrado de Planes de Mecanizado e Inspección en Entornos Distribuidos y Colaborativos." Paper presented at the 3rd Manufacturing Engineering Society International Conference, Alcoy, Spain, June 17–19.

Honggen, Z., J. Xuwen, P. Baojun, and Z. Xiaojun. 2012. "Ontology Theroy Based Process Planning and Supporting System under E-manufacturing." *Przegląd Elektrotechniczny* 88 (1b): 107–110.

ISA (Instrument Society of America). 2000. *Enterprise-Control System Integration. Part1: Models and Terminology*. ANSI/ISA–S95.00.01, North Carolina: American National Standards Institute.

ISO (International Organization for Standardization). 2004a. *Industrial Automation Systems and Integration – Industrial Manufacturing Management Data. Part 1, General Overview*. ISO 15531–1, New York: American National Standards Institute.

ISO (International Organization for Standardization). 2004b. *Industrial Automation System and Integration – Process Specification Language. Part 1, Overview and Basic Principles*. ISO 18629–1, New York: American National Standards Institute.

Kantola, J. I. 2009. "Ontology-based Resource Management." *Human Factors and Ergonomics in Manufacturing* 19 (6): 515–527. doi: 10.1002/hfm.20181.

Kuutti, K. 1995. "Activity Theory as a Potencial Framework for Human-computer Interaction Research." In *Context and Consciousness: Activity and Human Computer Interaction*, edited by B. Nardi, 17–44. Cambridge: MIT Press.

Lin, Y. 2008. "Semantic Annotation for Process Models: Facilitating Process Knowledge Management via Semantic Interoperability." PhD diss., Norwegian University of Science and Technology.

Lin, L. F., W. Y. Zhang, Y. C. Lou, C. Y. Chu, and M. Cai. 2011. "Developing Manufacturing Ontologies for Knowledge Reuse in Distributed Manufacturing Environment." *International Journal of Production Research* 49 (2): 343–359.

Mariot, P., J. P. Cotton, C. Golbreich, A. Berger, and F. Vexler. 2007. "Querying Multiple Sources with OWL Ontologies: an Exploratory Study in an Automotive Company." Paper presented at the 3rd International Workshop OWL: Experiences and Directions (OWLED), Innsbruck, Austria, June 6–7.

Martínez-Pellitero, S., J. Barreiro, E. Cuesta, and B. J. Álvarez. 2011. "A New Process-based Ontology for KBE System Implementation: Application to Inspection Process Planning." *International Journal of Advanced Manufacturing Technology* 57 (1–4): 325–339. doi: 10.1007/s00170-011-3285-7.

Masolo, C., S. Borgo, A. Gangemi, N. Guarino, and A. Oltramari. 2003. *WonderWeb Deliverable D18. Ontology Library (final).* Trento: Laboratory for Applied Ontology ISTC CNR.

MIRC ontology. 2014. Accessed February. http://coapp.webs.upv.es/ontologies/2014/0/MIRC_v1.owl.txt

Newman, S. T., and A. Nassehi. 2009. "Machine Tool Capability Profile for Intelligent Process Planning." *CIRP Annals - Manufacturing Technology* 58 (1): 421–424.

Ning, G., J. Tian-guo, and L. Wen-jian. 2010. "Research on Manufacturing Resource Domain Ontology Integration based on OWL." Paper presented at the 2nd International Workshop on Database Technology and Applications (DBTA), Wuhan, China, November 27–28.

Oberle, D., A. Ankolekar, P. Hitzler, P. Cimiano, M. Sintek, M. Kiesel, B. Mougouie, S. Baumann, S. Vembu, M. Romanelli, P. Buitelaar, R. Engel, D. Sonntag, N. Reithinger, B. Loos, H.-P. Zorn, V. Micelli, R. Porzel, C. Schmidt, M. Weiten, F. Burkhardt, and J. Zhou. 2007. "DOLCE Ergo SUMO: On Foundational and Domain Models in the SmartWeb Integrated Ontology (SWIntO)." *Web Semantics: Science, Services and Agents on the World Wide Web* 5 (3): 156–174.

Ramos, L. 2010. “Toward Dynamic Ontologies for the Industrial Manufacturing Domain.” Paper presented at the 4th International Workshop on Ontology Dynamics, Shanghai, China, November 8.

Rosado, P., and F. Romero. 2009. “A Model for Collaborative Process Planning in a Engineering and Production Network.” Paper presented at the 13th International Research/expert Conference. Trends in the Development of Machinery and Associated Technology, Hammamet, October 16–21.

Solano, L., F. Romero, and P. Rosado. 2010. “An Ontological Approach for Manufacturing Resources Modeling.” Paper presented at the 21st International DAAAM Symposium, Zadar, Croatia, October 20–23.

Solano, L., P. Rosado, and F. Romero. 2013. “Knowledge Representation for Product and Processes Development Planning in Collaborative Environments.” *International Journal of Computer Integrated Manufacturing*. Available online: 10 Sep 2013. doi: 10.1080/0951192X.2013.834480.

Sormaz, D. N., J. Arumugan, and S. Rajaraman. 2004. “Integrative Process Plan Model and Representation for Intelligent Distributed Manufacturing Planning.” *International Journal of Production Research* 42 (17): 3397–3417.

Vichare, P., A. Nassehi, S. Kumar, and S. T. Newman. 2009. “A Unified Manufacturing Resource Model for Representing CNC Machining Systems.” *Robotics and Computer-Integrated Manufacturing* 25 (6): 999–1007.

Zdravkovic, M., and M. Trajanovic. 2009. “Integrated Product Ontologies for Inter-organizational Networks.” *Computer Science and Information Systems* 6 (2): 29–46. doi: 10.2298/csis0902029Z.

Capítulo 7. DISCUSIÓN GENERAL DE LOS RESULTADOS, CONCLUSIONES Y TRABAJOS FUTUROS

1. Conclusiones

En este trabajo se ha desarrollado una ontología integrada de procesos y recursos para el desarrollo colaborativo de planes de proceso, que cumple con los objetivos planteados inicialmente para la tesis. Cabe recordar que el objetivo general que se estableció fue la mejora en el desarrollo colaborativo del producto, a través de la definición de un modelo ontológico que posibilitara una interpretación compartida de los datos/información utilizados en el trabajo en equipo, y garantizara la consistencia y precisión de los mismos. Además, se planteó que esta propuesta ontológica pudiera particularizarse para ámbitos más concretos como el de la planificación del proceso de fabricación y, en concreto, para los de mecanizado e inspección. Para ello, el modelo ontológico debería integrar los dominios del producto y del proceso, considerando de forma especial tanto el conjunto de actividades que se realizan en el proceso como el de recursos implicados en su ejecución.

Además, con el desarrollo de la ontología, se ha podido corroborar la validez de las hipótesis de partida formuladas inicialmente. En concreto dos que a nuestro entender han sido fundamentales para el desarrollo del trabajo: (a) la definición de una ontología integrada de dominio con la riqueza semántica de PSL; y (b) su implementación mediante los lenguajes OWL y SWRL.

En este sentido, se ha validado que la ontología PPDRC proporciona una respuesta eficiente a las necesidades derivadas del desarrollo de productos en entornos distribuidos y colaborativos. En las pruebas de validación se ha constatado que la semántica de la ontología permite la estructuración, programación, control y seguimiento de las tareas implicadas en este proceso de desarrollo.

También se ha constatado la adecuación de la propuesta ontológica global (ontología PPDRC) para la utilización, transmisión y uso compartido de la información y el conocimiento necesarios en el desarrollo de nuevos productos. La ontología da soporte y proporciona un tratamiento homogéneo a las actividades de gestión del proyecto (planificación, programación, y coordinación y control de la ejecución), las actividades técnicas de desarrollo del producto y el resto de tareas relacionadas con los procesos de soporte.

Por otra parte, la ontología, construida con DOLCE como ontología de base (*foundational ontology*), reúne conceptos procedentes de otras ontologías y de otras propuestas/iniciativas no ontológicas: desde la consideración del carácter social propio de la ontología DOLCE y la Teoría de Actividades, hasta el tratamiento de recursos y roles que proviene de la iniciativa MANDATE, pasando por la descripción de actividades y de planes no lineales que deriva de la ontología PSL, de la que también aprovecha su gran riqueza semántica.

Se ha constatado que la ontología PPDRC, implementada mediante OWL y SWRL para su funcionamiento en red, permite el trabajo concurrente y deslocalizado de varios planificadores sobre un plan de proceso único. Este tipo de implementación y ejecución posibilita la introducción de datos (modificación de la *extension* de la ontología) procedentes de distintos equipos de trabajo/planificadores sobre una única estructura de clases y predicados (la *intension* de la ontología).

Además, la ontología PPDRC se puede aplicar a cualquier tipo de planificación, ya que sus elementos son de carácter general. Por lo tanto se pueden aplicar tanto a la planificación de procesos de cualquier tipo como a la planificación de la producción. En otras palabras, PPDRC es aplicable a cualquier proceso donde hay actividades que se desarrollan según un flujo determinado y que precisan de unos recursos con ciertas capacidades (*capabilities* y *capacities*). Estos dos términos (*capabilities* y *capacities*) hacen referencia respectivamente a la capacidad tecnológica de los recursos, lo que saben hacer los recursos, y a su capacidad productiva vinculada al tiempo y a la productividad, que están asociadas a la cuantificación de los resultados obtenidos.

Otra característica reseñable es que, debido a que la ontología PPDRC realiza consideraciones que incluyen el tiempo y caracteriza los recursos mediante sus *capacities*, esta propuesta facilita su conexión con modelos y ontologías focalizadas en la planificación y programación de la producción (*scheduling*). La vinculación de esta función con unos planes de proceso no lineales generados en la planificación de procesos permitiría explotar eficientemente la flexibilidad del sistema productivo en función de su estado.

En línea con lo anteriormente indicado, la propuesta contempla la representación del carácter dinámico del sistema productivo. Para ello, PPDRC introduce el concepto/entidad *version* con el que se pueden representar las diferentes versiones de los individuos de la ontología. Estas *versions* caracterizan los cambios en los individuos a lo largo del tiempo y en particular aquellos que afectan a los recursos, como son sus configuraciones, y por tanto sus *capabilities* y *capacities*. Esto permite la actualización de estas *capabilities*/*capacities* y permite la actualización del estado con los resultados obtenidos en cada instante. De este modo, la *extension* de la ontología crea un conocimiento más concreto y particularizado al ámbito en el que se utiliza.

En línea con lo expuesto al comienzo de este apartado, también se ha constatado la adecuación de la ontología para la utilización, transmisión y uso compartido de la información y el conocimiento en actividades de planificación de los procesos de mecanizado e inspección. Concretamente en las pruebas realizadas sobre la ontología MIRC, que especializa la ontología PPDRC en dicho ámbito, se ha corroborado cómo mediante diversas *queries*, el planificador puede obtener información sobre la estructura del plan, la configuración actual del recurso, y la composición/descomposición de sus *capabilities* y de todos los elementos que intervienen en la operación, etc.

Finalmente, se ha validado que las ontologías PPDRC, en el ámbito del desarrollo integrado y colaborativo de producto, proceso y recursos, y MIRC, en el ámbito de la planificación de procesos de mecanizado e inspección, proporcionan un soporte suficientemente flexible que permite la planificación con diferentes enfoques (variante, generativo y mixto) y estrategias (hacia adelante, hacia atrás y mixta) para cualquier nivel de planificación (agregado, supervisor y operacional).

2. Aportaciones

Una vez expuestas las principales conclusiones derivadas del trabajo desarrollado para la elaboración de esta tesis, a continuación, se detallan las aportaciones específicas más significativas. Algunas de estas aportaciones afectan exclusivamente a MIRC, como especialización de PPDRC en el campo de la planificación de los procesos de mecanizado e inspección.

- Se ha demostrado que la utilización de DOLCE (DUL) como ontología de base para el desarrollo de la ontología de dominio PPDRC facilita su interoperabilidad semántica con ontologías de otros ámbitos y permite considerar varios fundamentos ontológicos para conceptualizar con fidelidad la realidad a representar. En particular, el concepto *capability* definido en PPDRC se establece en base a las clases *characteristic* y *region* de DOLCE. Esto otorga a PPDRC un carácter general particularizable, por ejemplo en el caso de la ontología MIRC, y no limita los tipos de individuos de la clase *capability*.

- La reunión e integración, en PPDRC, de conceptos relevantes que provienen de ontologías y modelos para procesos y recursos, y que facilitan la interoperabilidad en sus respectivos dominios. De TOVE y PSL recoge los conceptos de proceso, constituido por actividades, y el de recurso, como aquel que tiene que estar presente durante la ejecución de actividades y procesos. MANDATE aporta, además del concepto de *capacity* presente en TOVE y PSL, la consideración de las *capabilities* del recurso, que le confieren la habilidad para ejecutar determinadas actividades obteniendo unas prestaciones. De la Teoría de Actividades se recoge la consideración de la dimensión social de las actividades que forman parte del proceso. De IDEF0 se recoge la clasificación de objetos según unos roles, en base a los cuales se definen como *Input*, *Output*, *Control* y *Mechanism*, estableciendo las formas de participación de un recurso en una actividad.

- El tratamiento homogéneo que se da en PPDRC a las actividades que conforman un plan, a los objetos que participan en la realización de estas actividades y a las características de dichos objetos, como son las *capabilities* de los recursos. Como parte de este tratamiento homogéneo se encuentra el concepto de interfaz, una entidad que relaciona el objeto que realiza la actividad y el objeto sobre el cual se realiza dicha actividad. Esta interfaz, que influye en las características resultantes del objeto de salida de la actividad, está regulada por un objeto que actúa como control.

- La posibilidad de modelar, con el nivel de detalle requerido acorde a cada situación, los recursos y procesos. El modelado de recursos permite definir su estructura (componentes) y su configuración (conexión y ajuste entre los

componentes), que establecen sus *capabilities* en la ejecución de actividades. Por su parte, el modelado de procesos permite establecer la estructura, dependencia, coordinación y lógica de ejecución (flujo) de dichos procesos.

- Una gran riqueza semántica en la concreción de las capacidades de un recurso, que se pone de manifiesto a través de una serie de predicados. De forma general, el predicado *parametrizes* cuantifica una *capability*. Su especialización *parametrizes_Object* asocia la *capability* de un recurso con la *region* que establece sus prestaciones, es decir, el valor que se puede obtener en la característica del objeto resultante. Este valor puede estar condicionado por las características del objeto de entrada. Para ello, el predicado *requires* establece los valores límite de las características del objeto de entrada para que puedan alcanzarse las prestaciones del recurso. Además, el predicado *parametrizes* posee otra especialización (*parametrizes_Occ*) que permite establecer las prestaciones del recurso que no se transmiten al objeto sobre el que se realiza la actividad, sino que caracterizan la propia ejecución de la actividad, por ejemplo: ratios de producción, potencia o tiempo, etc.

- El desarrollo de una metodología gráfica para la ontología MIRC, que facilita la comprensión del marco conceptual y la visualización de diversos elementos del plan. Dicha metodología se apoya en dos tipos de grafos bidimensionales (2D), que corresponden a sendas vistas o proyecciones de un grafo tridimensional (3D). Estos grafos 2D permiten la visualización del plan en su conjunto, con la secuencia de actividades de procesado y de configuración de los recursos (vista proyectada), y el detalle de cada una de las etapas que lo componen, con el tipo y cuantificación de las características de los objetos, recursos e interfaces asociados a la realización de la actividad (vista abatida).

- En la ontología MIRC se establece una taxonomía concreta para el ámbito del mecanizado y la inspección que afecta a las clases *Activity_Type*, *Resource*, *Capability* y *Region*:

    - Los tipos de actividades contemplados son *Preparation*, *Operation*, *Transport* y *Storage*. Dentro de la categoría *Preparation*, se incluyen las actividades de tipo *Loading* y *Setup*, vinculadas directamente a la preparación/configuración de los recursos; mientras que en las actividades de tipo *Operation* se incluyen las de tipo *Machining* e *Inspection*. Cada tipo de actividad de la ontología lleva asociado un tipo de *capability*.

    - En MIRC los recursos se clasifican atendiendo a tres criterios. Uno de ellos establece la diferenciación entre recursos según el tipo de prestación realizada (*Machine_Base*, *Tool*, *Fixture*, *Machine*, *Manipulator*, *Transport_System*, *Loading_Resource*, *Setup_Resource* y *Other_Resource_Group*). Otro de los criterios establece que los recursos pueden ser *Generic_Resources o Specific_Resources*, lo que permite agrupar sus *capabilities* para dar respuesta a la asignación de recursos genéricos para los propósitos de la planificación de procesos y de la planificación, programación y control de la producción, con diferentes grados de abstracción/concreción. Por último, la tercera clasificación

establece la diferencia entre *Resource_Elements* (*Machine_Base*, *Tool* y *Fixture*) y *Resource_Groups*, configurados a partir de otros recursos, y que tienen un conjunto de *capabilities* que deben ser entendidas como una única *capability* compleja a distinto nivel de detalle. A su vez, los *Resource_Group* pueden pertenecer a las clases siguientes: *Machine*, recurso compuesto de una *Machine_Base*, y de uno o más recursos de los tipos *Tool y Fixture;* y *Other_Resource_Group*, que incluye recursos de tipo *WorkStation* y *WorkCell*.

- Atendiendo a las distintas formas de cuantificar una característica, en MIRC se han establecido tres tipos de *regions* que consideran la forma de composición de sus valores: *Variability*, *Nominal_Value* y *Limit_Value*. Para las regions que pertenecen al tipo *Variability* se pueden aplicar leyes de composición estadística (cuadrática o similares); en las *regions* de tipo *Limit_Value* (como *Roughness*), se toma el valor máximo o el mínimo de todas las *regions* consideradas y en las *regions* de tipo *Nominal_Value* (como *Stroke*), la composición se realiza a partir de una expresión matemática simple, como puede ser la suma algebraica.

- La consideración en la ontología MIRC de tres especializaciones de las *regions* de tipo *Variability*, vinculadas con aspectos dimensionales y geométricos: *Dimensional_Variability* (DV), *Own_Geometric_Variability* (OGV) y *Reference_Geometric_Variability* (RGV). Las de tipo DV expresan variabilidades en longitudes y ángulos; las de tipo OGV representan variabilidades geométricas intrínsecas; y las de tipo RGV representan variabilidades de tipo geométrico (orientación y posición) entre elementos de entidades diferentes.

- El tratamiento integrado del mecanizado y de la inspección en la ontología MIRC. Tanto las herramientas de corte como las sondas de inspección pertenecen a la misma clase de recursos (*Tool*). Por otra parte, para caracterizar homogéneamente las actividades de tipo *Operation* (*Machining* e *Inspection*), se incluyen dos tipos de actividades que se ejecutan de forma concurrente, *Tool_Movement* y *Tool_Part_Interaction*, y que representan respectivamente el movimiento relativo entre la herramienta o sonda de medición y la pieza, y el tipo de interacción entre ellas.

- La ontología PPDRC y su especialización, la ontología MIRC, se han implementado mediante OWL y SWRL, lo que ha permitido solventar en parte las limitaciones expresivas de OWL y facilitar la inferencia de conocimiento a través de las propias ontologías. Esta implementación se ha complementado con el desarrollo de una herramienta que permite interrogar (mediante *queries*) a las ontologías acerca de aspectos relativos a la planificación, programación y control de procesos de ingeniería desde distintos puntos de vista, como los del planificador y el programador. Esta herramienta, que utiliza SPARQL, permite unir los predicados para generar una consulta compuesta que se muestra en un lenguaje natural, con la sintaxis y semántica de los conectores de SPARQL sobre los predicados de las ontologías. Su aplicación con la ontología MIRC permite extraer el conocimiento necesario para la toma de decisiones propias de las tareas de

asignación de recursos y validación de las soluciones adoptadas en la planificación del proceso de mecanizado e inspección.

3. Trabajos futuros

A pesar de las múltiples aportaciones realizadas con las ontologías PPDRC y MIRC en el marco del desarrollo colaborativo de producto/proceso, aún quedan otros aspectos que es necesario considerar y que constituirían proyectos de extensión de la tesis o nuevas líneas de investigación asociables a futuros trabajos. Entre éstas se pueden mencionar:

- La especialización de PPDRC a otras parcelas de la fabricación y verificación de productos. Concretamente, para la ontología MIRC, se propone extender su alcance a recursos complejos con un nivel de agregación superior a los de tipo *Machine*, como *Workstations*, *Workcells* y *Virtual_Workcells*. Algunos de estos recursos complejos están incluidos en la taxonomía de recursos de la ontología, pero en estos momentos MIRC no contempla los tipos de actividades específicos que corresponden a estos recursos, ni sus *capabilities*.

- Explotar la metodología gráfica desarrollada para la configuración de recursos y validación completa del plan de procesos en base a la ontología MIRC, para realizar un estudio comparativo entre esta metodología y otras que permiten la selección de recursos, como es el caso de las que utilizan grafos de tolerancias o cadenas de cotas. En este último caso, los estudios se realizan para una etapa concreta del plan, mientras que la metodología propuesta permite el tratamiento de cada etapa y del proceso en su conjunto.

- Aplicar la metodología gráfica propuesta para la configuración y validación de un plan de proceso no lineal en entornos de gran flexibilidad, propios de las empesas OKP, a entornos de fabricación menos flexibles, como es el caso de una línea de producción rígida, propia de un sistema dedicado.

- Profundizar en la naturaleza y la tipología de los objetos con rol de control, que ejercen su influencia en las interfaces de ejecución de la actividad. Influencia que se manifiesta especialmente en la cuantificación de las características resultantes de la ejecución de las actividades.

- Extender el rango de las características y *capabilities* más allá del dominio de la calidad dimensional y geométrica, teniendo en cuenta otras *capabilities* de los recursos vinculadas con diferentes medidas de la eficacia y eficiencia del plan, como son la tasa de arranque de material, la potencia y la gama de velocidades de corte, las velocidades de avance y el tiempo de ejecución, etc.

- Desarrollar una herramienta de co-planificación basada en la propuesta ontológica que posibilite el trabajo concurrente de varios planificadores sobre un plan de proceso único. Para ello sería necesario validar la metodología en un escenario en el que cualquiera de estos planificadores, que define una parte del plan de forma asíncrona, pueda agregarla al resto del plan creado de forma síncrona o asíncrona por otros planificadores. El desarrollo de esta herramienta estaría basado en la implementación de servicios Web para aprovechar los avances en las TIC's. Un sistema de co-planificación de este tipo proporcionaría un soporte más eficiente a

los procesos cooperativos entre socios, derribaría barreras tecnológicas para facilitar la colaboración, y mejoraría la interoperabilidad y la agilidad en la explotación del sistema. La arquitectura de estos servicios se organizaría por niveles, siendo el de más alto nivel, o nivel de aplicación, el de mayor interés. Éste podría servir, por ejemplo, para establecer mecanismos de respuesta a las *queries*, o disponer de una serie de *queries* tipo, particularizables para cada contexto/caso concreto, que sirvan de ayuda al planificador y que le orienten en el desarrollo de sus tareas.

- La integración de PPDRC con otras ontologías de producto, tanto del dominio del diseño como de la fabricación, sustentadas en fundamentos ontológicos que sean compatibles con los de PPDRC. Por ejemplo, ontologías de producto basadas en el concepto de *feature*. La integración de ontologías de alguno de estos tipos con la ontología PPDRC (o con la ontología MIRC) se realizaría utilizando los nodos de la red descrita en la propuesta del sistema Co-CAPP concebida para soportar el proceso de Desarrollo Integrado y Colaborativo de Producto, Procesos y Recursos. Nodos que incluyen y relacionan todos los elementos considerados en la planificación, como operaciones de fabricación e inspección, recursos y productos, así como las restricciones que limitan las secuencias de ejecución posibles.
- La especialización de PPDRC para otros campos como los de la construcción y la obra civil, que tienen similitud con el de la fabricación de piezas y productos industriales, y donde son claves las tareas de planificación y programación, que podrían estar soportadas por una ontología derivada de PPDRC.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Definición de una ontología integrada de procesos y recursos, para el desarrollo colaborativo de planes de proceso

Anexos

TESIS DOCTORAL

Presentada por:
Lorenzo Solano García
Dirigida por:
Pedro Rosado Castellano
Fernando Romero Subirón

Valencia, Noviembre 2015

Contenido

Anexo 1. Lógica

1. Introducción

La lógica es la disciplina que estudia los principios del razonamiento. Un razonamiento basado en abstracciones de la parte del mundo real que corresponden a un área de estudio o interés determinado, y que permite obtener conclusiones utilizando unos operadores lógicos que dependerán del tipo de lógica empleada.

Tabla A1.1. Tipos de lógica (Romero-Llop 2007).

Lógica	Tipo de abstracción que utiliza	Conclusiones alcanzadas
Proposicional	Hechos	Verdadero/Falso/Desconocido
Primer Orden	Hechos / Objetos / Relaciones	Verdadero/Falso/Desconocido
Temporal	Hechos / Objetos / Relaciones / Tiempo	Verdadero/Falso/Desconocido
Probabilística	Hechos	Grado de certeza
Difusa	Grado de verdad	Grado de certeza

Lógica proposicional, lógica de primer orden, lógica temporal, lógica probabilística y lógica difusa, son algunas de las lógicas existentes. Las tres primeras conducen a conclusiones en las que se determina si los hechos son verdaderos, falsos o desconocidos (los hechos siempre son verdaderos o falsos, pero es posible que esto se desconozca o no se pueda deducir), mientras que las lógicas probabilística y difusa permiten establecer grados de certeza. Además de la naturaleza de las conclusiones que se pueden alcanzar, otro criterio básico para clasificar las lógicas es el tipo de abstracción que utilizan. Como se aprecia en la Tabla A1.1, éstos pueden ser de varios tipos: (a) la lógica proposicional se basa en una abstracción de la realidad compuesta únicamente de hechos (p.e.: 'Este plato de garras es un utillaje'); (b) la lógica de primer orden (FOL) utiliza una abstracción del mundo en la que existen hechos, objetos y relaciones entre ellos (p.e.: 'Existe un plato de garras que permite embridar piezas cilíndricas de tamaño medio'); (c) en el caso de la

lógica temporal, la abstracción de la realidad incluye parámetros temporales, además de hechos, objetos y relaciones (p.e.: 'Este plato de garras formó parte de la máquina desde su primer *setup* en 2010'); (d) la lógica probabilística emplea una abstracción formada por hechos a los que se atribuye una cierta probabilidad (p.e.: 'Si un objeto de este taller es un utillaje, la probabilidad de que sea un plato de garras es del 15%'); y (e) la lógica difusa se basa en una abstracción de la realidad en la que existen grados de verdad en lugar de hechos (p.e.: 'Si la probabilidad de que este utillaje sea un plato de garras es del 50%, y la probabilidad de que forme parte de la máquina es del 20%, entonces la probabilidad de que el utillaje haya intervenido en la fabricación de la pieza es del 10%') (Romero-Llop 2007).

2. Lógica proposicional

La lógica proposicional o lógica de orden cero es un sistema formal en el que los elementos más simples son las variables proposicionales, que representan proposiciones y que tienen un valor de verdad definido (verdadero –V– o falso –F–). Además, esta lógica incluye constantes lógicas, o conectivas, que representan operaciones sobre proposiciones, capaces de formar otras proposiciones de mayor complejidad. Por tanto, la lógica proposicional permite realizar inferencias lógicas sobre proposiciones complejas, a partir de otras proposiciones simples, pero sin tener en cuenta la estructura interna de estas últimas. Por ejemplo, sea el siguiente razonamiento relativo al proceso de inspección de una pieza, compuesto por tres proposiciones:

(1) Es correcta o es incorrecta.
(2) No es correcta.
(3) Es incorrecta.

Si las premisas 1 y 2 son verdaderas, la conclusion será verdadera, y si las premisas son falsas, entonces la conclusión también podría serlo. La validez de este razonamiento no se debe al significado de las proposiciones 'es correcta' (p) y 'es incorrecta' (q), sino que depende del significado de las conectivas 'o' y 'no'. La lógica proposicional estudia el comportamiento de estas conectivas o constantes lógicas. En cuanto a las proposiciones, como 'es correcta', lo único que importa de ellas es que tengan un valor de verdad. Así, el razonamiento anterior podría reescribirse como:

```
p o q
No p
Por lo tanto, q
```

En la lógica proposicional, las conectivas lógicas se tratan como funciones de verdad. Es decir, como funciones que toman conjuntos de valores de verdad y devuelven valores de verdad.

En la Tabla A1.2 se muestran todas las conectivas lógicas de la lógica proposicional, incluyendo ejemplos de su uso en el lenguaje natural y los símbolos que se utilizan para representarlas en lenguaje formal (Wikipedia-3w).

Tabla A1.2. Conectivas lógicas de la lógica proposicional (Wikipedia-3w).

Conectiva	Expresión en el lenguaje natural	Ejemplo	Símbolo
Negación	No	No es correcta.	¬
Conjunción	Y	Está acabada y es correcta.	∧
Disyunción	O	Es correcta o es incorrecta.	∨
Condicional material	si… entonces	Si está inspeccionada, entonces está acabada.	→
Bicondicional	si y sólo si	Es incorrecta si y sólo si se excede la tolerancia.	↔
Negación conjuntiva	ni… ni	Ni está acabada ni está inspeccionada.	↓
Disyunción excluyente	o bien… o bien	O bien es correcta, o bien es incorrecta.	↮

3. Lógica de primer orden

La lógica de primer orden (FOL), también llamada lógica de predicados o cálculo de predicados, es un sistema formal diseñado para estudiar la inferencia en los lenguajes de primer orden, que son lenguajes formales con cuantificadores que alcanzan sólo a variables de individuo, y con predicados que expresan propiedades y relaciones cuyos argumentos son individuos o variables de individuo. Para la correcta comprensión del alcance de esta lógica se precisa una explicación de los conceptos anteriores (predicados, propiedades, relaciones, individuos, variables de individuo y cuantificadores), que se realiza a continuación.

Un predicado es una expresión que puede conectarse con otras expresiones para formar otro predicado de mayor extensión u oración. Por ejemplo, en la oración 'La torreta es un utillaje', el predicado 'La torreta' se conecta con el predicado 'es un utillaje'; mientras que la oración 'La herramienta se fija en la torreta', está compuesta de tres predicados conectados: 'La herramienta', 'se fija en' y 'la torreta'. En la lógica de primer orden, los predicados son tratados como funciones, con unos argumentos de entrada y un valor, o imagen, de salida. Por tanto, el predicado 'La torreta es un utillaje', que expresa una propiedad, puede formalizarse como:

`Utillaje(torreta)`, o más abreviademente como: `U(t)`

Del mismo modo, un predicado que expresa una relación, como 'La herramienta se fija en la torreta' puede formalizarse así:

`se_Fija_en (herramienta,torreta)`, o abreviando: `F(h,t)`

Este procedimiento puede extenderse a predicados que expresan relaciones entre más de dos entidades. Por ejemplo, el predicado 'La torreta conecta la herramienta con la máquina' puede formalizarse como:

`C(t,h,m)`

Una constante de individuo es una expresión que hace referencia a una entidad determinada. Por ejemplo 'torreta' y 'herramienta' son constantes de individuo, al igual que

las expresiones '1', '2', etc., que se refieren a números. Además de las constantes de individuo, la lógica de primer orden cuenta con variables (de individuo), que suelen representarse con las últimas letras minúsculas del alfabeto latino, y que no tienen una referencia determinada. Así, por ejemplo, haciendo uso de las variables, el predicado 'Esto es un utillaje' se representaría con la expresión:

`Utillaje(x)`, o abreviadamente: `U(x)`

En ella, hasta que no se determine a qué se refiere la variable 'x', no es posible asignar un valor de verdad al predicado 'Esto es un utillaje'. Las variables sirven también para formalizar relaciones. Por ejemplo, el predicado 'Esto es fija en aquello' se formaliza como: F(x,y). Y también pueden combinarse constantes de individuo con variables. Por ejemplo, el predicado 'Esto conecta la herramienta con la máquina' se formaliza como: C(x,h,m).

Un cuantificador es una expresión que se utiliza para indicar que una condición se cumple para un cierto número de individuos. El cuantificador universal (∀) permite expresar que una condición se cumple para todos los individuos a los que se hace referencia, mientras que con el cuantificador existencial (∃) se expresa que la condición se cumple para al menos uno de los individuos. Debe advertirse que, el valor de verdad de las expresiones en las que se utilizan cuantificadores depende del conjunto de individuos a los que se hace referencia, o dominio de discurso. A continuación se muestran sendos ejemplos de expresiones con cuantificadores:

Para todo x, x es utillaje. `∀x U(x)`

Existe al menos un x, tal que x es utillaje. `∃x U(x)`

La lógica de primer orden, que puede considerarse como una extensión de la lógica proposicional, incorpora las conectivas de la lógica proposicional. Combinando éstas con los predicados, constantes, variables y cuantificadores, es posible formalizar oraciones como las siguientes:

La torreta es un utillaje y es un recurso. `U(t) ∧ R(t)`

Si la torreta es un utillaje, entonces también es un recurso. `U(t) → R(t)`

Nada es utillaje y además máquina. `¬∃x (U(x) ∧ M(x))`

Todos los utillajes son recursos. `∀x (U(x) → R(x))`

Enlazando con la definición dada al principio de este apartado, puede afirmarse que la tarea de la lógica de primer orden consiste en determinar la validez de los razonamientos formados por expresiones donde las variables de individuo tiene cuantificadores. Por ejemplo:

Todos los utillajes son recursos. `∀x (U(x) → R(x))`

La torreta es un utillaje. `U(t)`

Por lo tanto, la torreta es un recurso. `R(t)`

4. Referencias

Romero Llop, R. 2007. “Especificación OWL de una Ontología para Teleeducación en la Web Semántica.” PhD diss., Universitat Politècnica de València.

Wikipedia-3w. Sitio web de “Wikipedia”; http://es.wikipedia.org/wiki/Portada.

Anexo 2. Web Semántica

1. Introducción

Mediante el razonamiento y la inferencia lógica se genera un conocimiento, cuya gestión y utilización se pretende mejorar con la Web semántica. Su antecedente, la denominada Web sintáctica, únicamente soportaba la representación y transmisión de los recursos (Romero-Llop 2007). Recursos, que posteriormente debían ser interpretados y relacionados entre sí de forma lógica por los usuarios con el consiguiente incremento de coste y pérdida de eficiencia. Para evitar esta situación, el World Wide Web Consortium (W3C) promovió el desarrollo de la Web semántica, con el objetivo inicial de conseguir que algunos tipos de información fuesen entendibles por las máquinas, para permitir a éstas realizar análisis más profundos y ayudar a las personas en sus tareas.

Para la construcción del marco de la Web semántica, en el que los datos son compartidos y reutilizados por distintas aplicaciones, empresas y comunidades, se han definido lenguajes con capacidad para capturar esos datos, representarlos uniformemente, especificar su semántica y publicarla en línea. Las ontologías son la infraestructura básica para la Web semántica. La idea misma de la Web semántica depende de la posibilidad de utilizar vocabularios compartidos para la descripción de contenido y capacidades, cuya semántica se describe en una forma razonablemente inequívoca y procesable por ordenador. Describir esta semántica, es decir, lo que se llama a veces el significado pretendido de los términos de un vocabulario, es exactamente el trabajo que hacen las ontologías para la Web semántica (Masolo et al. 2003).

Contrariamente a los trabajos de gestión del conocimiento existentes en el campo de la Inteligencia Artificial (AI), la Web semántica no asume ni garantiza la integridad o disponibilidad constante de la información. En este sentido, un aspecto importante de la Web semántica es la denominada 'suposición de mundo abierto', según la cual, una declaración no es necesariamente falsa si no puede probarse que es cierta. De hecho, esta declaración podría convertirse en realidad en el futuro, debido a que la Web semántica permite que cualquiera pueda añadir información en cualquier momento y en cualquier lugar. Incluso es posible ampliar de forma flexible una ontología definida en otra comunidad con conceptos y relaciones adicionales (Bräuer 2007). La base de conocimiento, que junto con el motor de inferencia sustenta los sistemas de la Web semántica, es un conjunto de sentencias escritas en un lenguaje formal, que pueden ir creciendo y variando en el tiempo. A partir de esta base de conocimiento, un agente puede sacar conclusiones y resolver el problema que se le ha planteado. Para ello, el motor de inferencia debe ser capaz de obtener nuevas sentencias a partir de las sentencias existentes en la base de conocimiento. Por ejemplo, debe poder determinar si una sentencia es: deducible;

imposible, es decir que lleva a contradicción; o no es imposible, pero no es deducible (Romero-Llop 2007).

2. Estructura de capas de la Web Semántica

Partiendo de una de las premisas de sus promotores (W3C), se propuso un funcionamiento de la Web semántica basado en la utilización de los sistemas ya existentes en la Web. De este modo, añadiendo a la Web convencional anotaciones semánticas se podría aprovechar todo el potencial de la misma. Para ello, la mejor opción es apoyarse en lenguajes conocidos y construir sobre ellos las distintas capas que soportan la Web semántica. Como se representa en la Figura A2.1, la Web semántica está basada en una arquitectura estratificada formada por: capa de símbolos e identificadores de recursos, capa de intercambio de datos, capa de aserciones, capa ontológica, capa lógica, capa de prueba, y capa de verdad y firma digital.

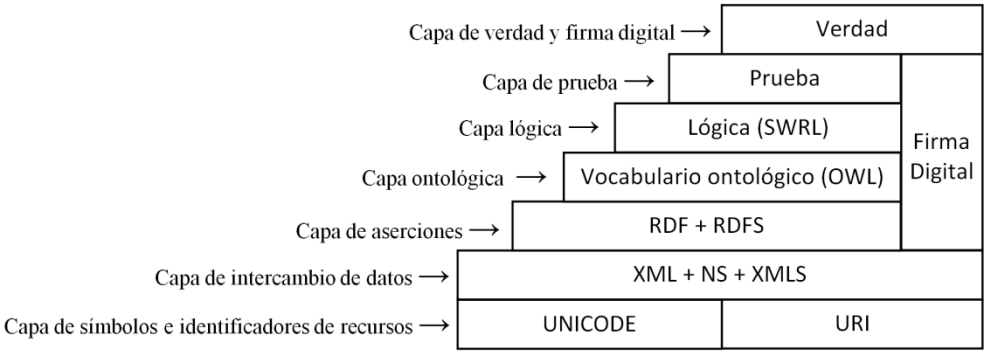


Figura A2.1. Estructura de capas de la Web semántica (ARC 2002).

La primera capa de la Web semántica está formada por Unicode y URIs (Uniform Resource Identifiers). Mediante Unicode se definen los símbolos con los que se construyen las restantes capas, mientras que los URIs sirven para identificar los recursos. Unicode es el estándar universal de codificación de caracteres usado para la representación de texto en computadores, y por tanto facilita la programación multi-idioma y multiplataforma (Romero-Llop 2007). Mediante los símbolos Unicode se define una capa, los URIs, que debe ser comprensible para los agentes inteligentes y que permite hacer referencia a un espacio de objetos o conceptos sobre los que se pretende ofrecer información. Un URI es una secuencia de caracteres en la que se utiliza la misma sintaxis (de ahí su carácter uniforme) para identificar un recurso abstracto o físico, y que puede abreviarse por medio de formas relativas, manteniendo siempre su identificación inequívoca.

En la capa de intercambio de datos se encuentran XML, XML Schema y NS (Name Spaces). Todos ellos utilizan los servicios/funcionalidades de la primera capa. XML es un sistema anidado de marcas que se ha convertido en el lenguaje para el intercambio de datos estructurados en la Web. Para definir la estructura que han de cumplir dichos datos, se puede recurrir a XML Schema (XMLS), que además permite comprobar si los datos se han generado y recibido de forma correcta. Esta capacidad para definir nuevas estructuras de datos de forma rápida y descentralizada dota a la Web de la flexibilidad necesaria, que

soslaya la ausencia de acuerdo previo entre los programadores respecto a la estructura de datos a utilizar. Los espacios de nombres (NS) se definen como un URI que representa a un conjunto de nombres dentro de la estructura de datos en la que están definidos. Un conjunto de términos comunes y definidos por la misma entidad pueden compartir el mismo espacio de nombres. A nivel práctico, los NS sirven para definir sufijos de URIs. Por ejemplo, en el documento XML de la expresión 2.1, para evitar la repetición del URI completo <http://ecommerce/schem:bill> se utiliza <edi:bill> en su lugar, facilitando su lectura y compactando la información.

```
<x xmlns:edi="http://ecommerce/schem">
   <edi:bill>1232</edi:bill>
</x>
```
[Expresión 2.1]

Los espacios de nombres son de gran importancia para los lenguajes RDF (Resource Description Framework), RDFS (RDF Schema) y OWL, ya que cada uno realiza una definición formal de palabras reservadas en su sintaxis, creando un espacio de nombres, que a su vez utiliza los espacios de nombres de los lenguajes de las capas inferiores.

RDF, que junto a RDFS constituye la denominada capa de aserciones, proporciona un modelo de datos básico para expresar aserciones acerca de recursos arbitrarios que pueden ser identificados a través de un URI. Una aserción RDF es una declaración acerca del valor de la propiedad de un recurso particular, cuyo significado es entendido por distintos procesos. Este modelo se basa en la idea de convertir las declaraciones de los recursos en expresiones con la forma sujeto-predicado-objeto, conocidas como tripletes. Los tripletes se representan con un arco que va desde el sujeto al objeto y que está etiquetado con la propiedad o predicado (Chang, Shin y Park 2003), y se pueden representar mediante XML (Figura A2.2 y Figura A2.3). El sujeto es el recurso, es decir aquello que se está describiendo. El predicado es la propiedad o relación que se desea establecer acerca del recurso. Por último, el objeto es el valor de la propiedad o el otro recurso con el que se establece la relación.

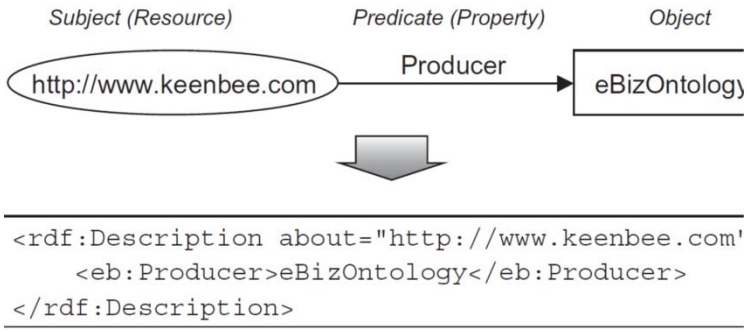


Figura A2.2. Modelo de datos RDF representado mediante XML (Chang, Shin y Park 2003).

Con las aserciones, también denominadas aseveraciones, sentencias, declaraciones o enunciados, realizadas mediante RDF se describen recursos, que se representan por medio de URIs o por medio de propiedades, las cuales, a su vez, también están representadas por URIs. El número de aserciones que se pueden realizar para describir un recurso es variable.

De este modo, para ir completando la descripción de cualquier recurso (sujetos, objetos e incluso propiedades) se proporciona la información incrementalmente: aserción a aserción. Considerando además que el recurso descrito en una aserción podría ser una propiedad de otra aserción distinta. La sintaxis de las aserciones está basada en XML, la capa inferior a RDF, e incluye marcas como la mostrada en la expresión 2.2, donde la etiqueta <Description> sirve para enmarcar la aserción. En dicha aserción, el atributo 'about' describe al sujeto, y cada elemento dentro de la descripción se considera una propiedad de ese sujeto.

```
<Description
about="http://www.dcom.upv.es/personal/Pedro">
    <trabajaCon
resource="http://www.dcom.upv.es/personal/Juan">
</Description>
```
[Expresión 2.2]

En la expresión 2.3 se muestran dos formas de describir a los seres humanos indicando que son una clase. En este caso, con RDF, se necesita una convención entre el emisor y el receptor de la información para especificar la semántica de la propiedad *Class*, lo que no sería necesario con RDFS, donde esta semántica ya está incluida en el lenguaje.

```
<Description
about="http://www.dcom.upv.es/general/SeresHumanos">
<rdf:type rdf:Class>
</Description>
<rdf:Class rdf:resource=
"http://www.dcom.upv.es/general/SeresHumanos">
```
[Expresión 2.3]

En la descripción de la expresión 2.4 se indica que el recurso Pedro es un objeto perteneciente a la clase ser humano.

```
<Description
about="http://www.dcom.upv.es/personal/Pedro">
   <rdf:type rdf:resource=
"http://www.dcom.upv.es/general/SeresHumanos">
</Description>
```
[Expresión 2.4]

En la expresión 2.5 se indica que 'trabajaCon' es una propiedad, que relaciona un ser humano (sujeto descrito por la marca *rdfs:domain*) con otro ser humano (objeto descrito por la marca *rdfs:range*).

```
<rdfs:Property rdf:ID="trabajaCon">
<rdfs:domain rdf:resource="#SeresHumanos"/>
<rdfs:range rdf:resource="#SeresHumanos"/></rdfs:Property>
```
[Expresión 2.5]

RDF es un lenguaje maduro con una gran aceptación dentro de las principales entidades relacionadas con la Web, y se han desarrollado herramientas que permiten realizar consultas (*queries*) mediante estándares de acceso a datos RDF como SPARQL, entre otros (Romero-Llop 2007). No obstante, RDF presenta algunas restricciones: no reserva palabras

clave para distinguir clases, entendidas como agrupaciones de objetos; y no distingue entre propiedades y elementos, ambos son considerados como recursos. Para solventar esta limitación se disponde de RDFS, que puede considerarse una extensión semántica de RDF para describir su vocabulario. De hecho, en la semántica de RDFS están incluidos términos como *Class*, *Property*, *type*, *subClassOf*, *range* o *domain*, que añaden soporte a las jerarquías de clase y propiedad, y a las restricciones del valor de las propiedades (Figura A2.3). De este modo, RDFS proporciona mecanismos para describir grupos de recursos relacionados y las relaciones entre dichos recursos, como las de herencia. No obstante, RDFS carece de restricciones de cardinalidad o existencia, lo que limita la semántica del lenguaje. Por esto, la posibilidad de crear ontologías mediante RDFS se limita a casos muy sencillos basados en relaciones taxonómicas (Bräuer 2007).

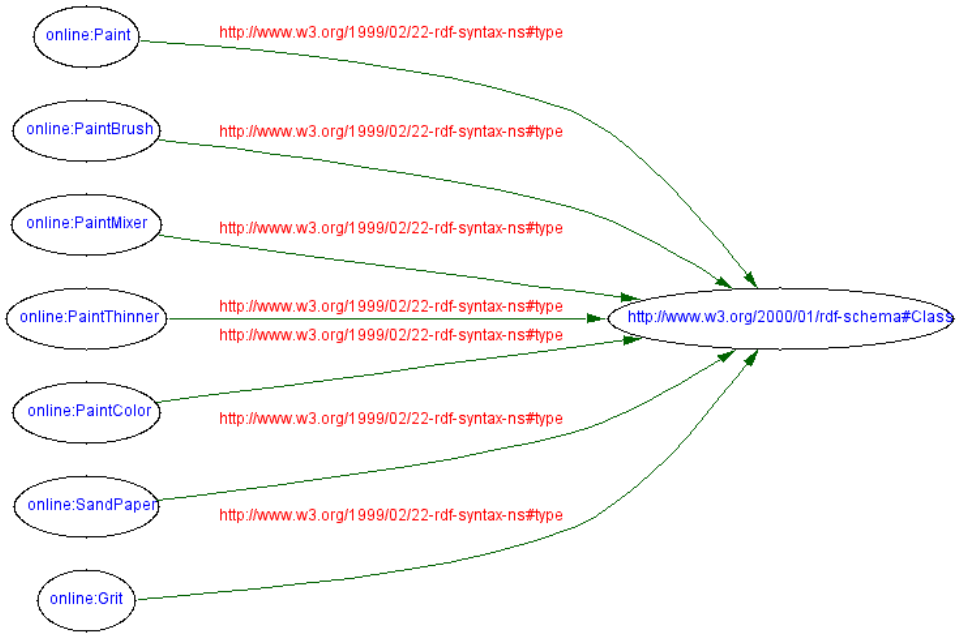


Figura A2.3. Grafo RDF para la definición de clases empleando la sintaxis abreviada (Lubell 2003).

Las capas ontológica y lógica, que tienen más relevancia para el propósito de esta tesis, se han descrito en el capítulo 3 (Ontologías). Sobre ellas se sitúa la capa de prueba, soportada por los razonadores, donde se devuelven ejemplos de objetos que cumplen unas restricciones definidas o se realizan las comprobaciones para determinar la veracidad de una suposición. El grado de fiabilidad que se le concede al razonamiento, y que se decide en la capa de verdad (Figura A2.1), dependerá de la confianza de las fuentes de información y la validez de la información original, y de la invariabilidad de dicha información desde que se generó. En este proceso de validación de la información, la firma digital (capa de firma digital) juega un papel clave, ya que garantiza la autenticidad de la autoría y de la fuente de la información (Romero-Llop 2007).

3. Referencias

ARC Advisory Group. 2002. "Collaborative Manufacturing Management Strategies." *ARC Advisory Group*, November.

Bräuer, M. 2007. "Design of a Semantic Connector Model for Composition of Metamodels in the Context of Software Variability." PhD diss., Technische Universität Dresden.

Chang, T., K. Shin y J. Park. 2003. "A Development Methodology for e-Work Ontology using RDF / RDFS and PSL." *Production Planning & Control* 14 (8): 766–777.

Lubell, J. 2003. "XML Representation of Process Descriptions." In Process Descriptions of Professional XML Meta Data. Wrox Press.

Masolo, C., S. Borgo, A. Gangemi, N. Guarino y A. Oltramari. 2003. WonderWeb Deliverable D18. Ontology Library (final). Laboratory for Applied Ontology ISTC CNR.

Romero Llop, R. 2007. "Especificación OWL de una Ontología para Teleeducación en la Web Semántica." PhD diss., Universitat Politècnica de València.

Anexo 3. KIF (Knowledge Interchange Format)

1. Introducción

Una ontología debe codificarse mediante un lenguaje que permita expresar los conceptos del dominio, de forma que éstos puedan ser manipulados y gestionados informáticamente atendiendo a su significado. Hay lenguajes de especificación de ontologías que han sido usados para representar ontologías instanciadas. Entre ellos se encuentra KIF (Knowledge Interchange Format), también conocido como CLIF (Common Logic Interchange Format), que puede considerarse la actualización de KIF tras su incorporación a Common Logic (CL), un lenguaje lógico de primer orden para el intercambio de conocimiento[1].

2. La sintaxis de KIF

Como se indica en ISO (2004), para la descripción de la sintaxis de KIF se utiliza BNF (Backus-Naur Formalism), una metasintaxis usada para expresar grámaticas libres de contexto: es decir, una manera formal de describir lenguajes formales. En KIF, la noción de *word* se considera primitiva. Una *expression* es una *word* o una secuencia finita de *expressions* (expresiones). Cuando se trata de una expresión compuesta, el uso de paréntesis sirve para delimitar los elementos que forman dicha expresión compuesta, y un conjunto de expresiones conforman una *knowledge base* (Figura A3.1).

A partir de las diez categorías primitivas de KIF (*words*) se construyen otras tres categorías más complejas (*variables*, *operators*, y *constants*). Una *variable* es una *word* cuyo primer carácter es '?' o '@'. Las variables que comienzan con '?' se denominan variables individuales, y se usan en la cuantificación de objetos individuales; mientras que las variables que comienzan con '@' se denominan variables de secuencia, y se usan en la cuantificación de secuencias de objetos. Los *operators* se emplean para formar expresiones complejas de varios tipos: los *term operators* se usan para formar términos complejos; los *sentence operators* se usan para formar sentencias complejas; los *rule operators* se usan

[1] CL proporciona un marco semántico para la lógica junto con las bases para un conjunto de formas sintácticas (dialectos) que comparten una semántica común (Delugach 2005). CLIF es uno de estos dialectos, siendo Conceptual Graph Interchange Format (CGIF), y una notación basada en XML para Common Logic (XCL) los dos restantes.

para formar reglas; y los *definition operators* se usan para formar definiciones. Una *constant* es cualquier palabra (*word*) distinta de *variable* y *operator*: las *object constants* designan objetos individuales; las *function constants* designan funciones sobre esos objetos; las *relation constants* designan relaciones, y las *logical constants* expresan condiciones acerca del mundo y son verdaderas o falsas. Hay cuatro tipos especiales de expresiones en KIF: *terms*, usados para designar objetos en el mundo que se describe; *sentences*, usadas para expresar hechos acerca del mundo; *rules*, usadas para expresar 'legal steps' o inferencia; y *definitions*, usadas para definir constantes. Estos tres últimos tipos de expresiones (*sentences*, *rules* y *definitions*) se agrupan en la categoría de *forms* (Figura A3.1).

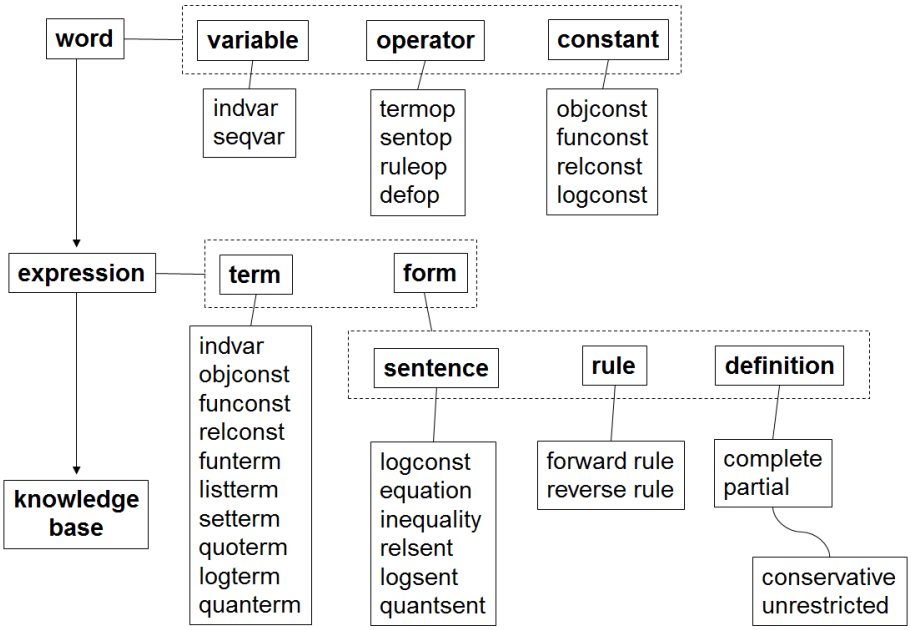
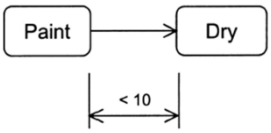


Figura A3.1. Estructura de los elementos sintácticos de KIF.

La Figura A3.2 incluye un ejemplo de expresión KIF en la que se utiliza la semántica de la ontología PSL (Process Specification Language) para especificar que el intervalo de tiempo entre dos operaciones debe ser inferior a 10. Para ello, el resultado de la evaluación lógica de la expresión en su conjunto, desde '*(forall...*' hasta '*…10)))))*', considerando la imbricación de los paréntesis y el operador *and* tiene que ser 'verdadero' (*true*). En esta expresión, se han subrayado las palabras reservadas de KIF; las funciones y relaciones de PSL se ha escrito en letra negrita y cursiva; y en letra normal las variables y constantes que intervienen como argumentos en dichas relaciones y funciones.




```
(forall (?occChangeColor)
        (implies
                (occurrence_of ?occChangeColor ChangeColor)
                (exists (?occPaint ?occDry)
                   (and (occurrence_of ?occPaint Paint)
                        (occurrence_of ?occDry Dry)
                        (subactivity_occurrence ?occPaint ?occChangeColor)
                        (subactivity_occurrence ?occDry ?occChangeColor)
                        (next_subocc ?occPaint ?occDry ChangeColor)
                        (lesser (timeduration (endof ?occPaint)
                                              (beginof ?occDry)) 10)))))
```

Figura A3.2. Restricción de tiempo entre dos operaciones expresada mediante KIF (Bock y Gruninger 2005).

3. Referencias

Bock, C. y M. Gruninger. 2005. “PSL: A Semantic Domain for Flow Models.” Journal of Software and Systems Modeling 4 (2): 209–231. doi: 10.1007/s10270-004-0066-x.

Delugach, H. 2005. “Common Logic in Support of Metadata and Ontologies”.

ISO (International Organization for Standardization). 2004. Industrial Automation System and Integration – Process Specification Language. Part 11, PSL-Core. ISO 18629–11, New York: American National Standards Institute.

Anexo 4. DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering)

1. Descripción de las categorías básicas de DOLCE

En este anexo se realiza la descripción con comentarios de las entidades más importantes que componen la ontología DOLCE (Figura A4.1) según los trabajos de Masolo et al. (2003) y Guarino (2006), así como los de otros autores cuyas referencias se indican en el texto.

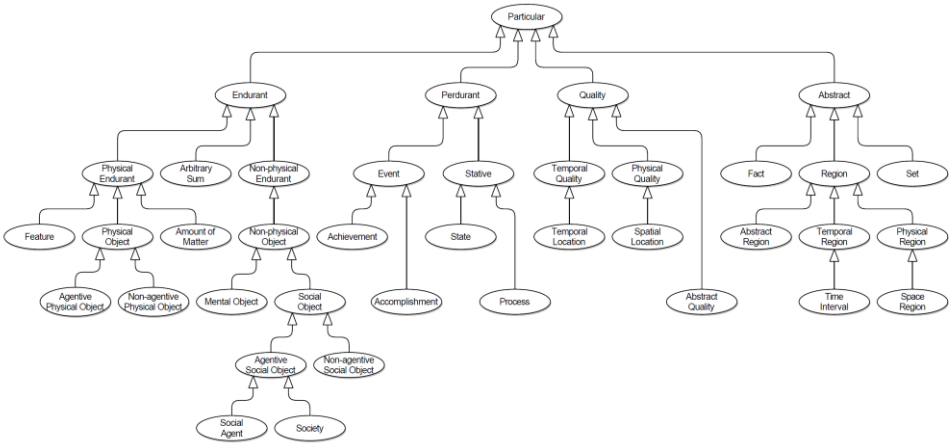


Figura A4.1. Taxonomía de las categorías básicas de DOLCE (Bräuer 2007).

Los *universals* son entidades que pueden tener instancias, que se denominan *particulars*. En la ontología DOLCE, las clases y predicados son *universals* y por tanto, pueden tener instancias. Cada una de ellas es un elemento *particular* que pertenece a la clase *Particular* para la que DOLCE establece la taxonomía mostrada en la Figura A4.1. En esta taxonomía es fundamental la distinción entre entidades de tipo *enduring* y *perduring* (*endurants y perdurants*) atendiendo a su comportamiento en el tiempo. A continuación se describen las clases del primer nivel de la ontología DOLCE (Figura A4.2):

- **Endurant**. Entidad en la que todas sus partes esenciales y sus propiedades significativas están presentes en todo momento en el que la entidad está presente. Un e*ndurant* es una entidad que está presente completamente en cualquier momento del tiempo que está presente (Borgo y Leitao 2004). Un *endurant* está en el tiempo y todas sus partes fluyen con él en el tiempo. Un *endurant* existe en el tiempo y puede genuinamente cambiar en el tiempo, en el sentido de que puede tener propiedades incompatibles en instantes de tiempo diferentes.
- **Perdurant**. Entidad que se extiende en el tiempo acumulando partes temporales diferentes, y por tanto, en cualquier instante de tiempo solo está presente parcialmente, en el sentido de que no todas sus partes están presentes (Borgo y Leitao 2004). En un *perdurant* solamente algunas de sus partes genuinas, que son partes temporales, están presentes cuando la entidad está presente. Un *perdurant* sucede en el tiempo, y no puede cambiar en el tiempo, ya que ninguna de sus partes mantiene su identidad en el tiempo.
- **Quality**. Entidad que es inherente a otra entidad. Las *qualities* pertenecen siempre a un *particular*, que puede ser incluso otra *quality*.
- **Abstract**. Entidad que no tiene ninguna *quality* espacial ni temporal y que no es una *quality*. Las únicas entidades abstractas consideradas en la versión de DOLCE descrita por Masolo et al. (2003) son las *Quality Regions*, o simplemente *Regions*. (Figura A4.3).

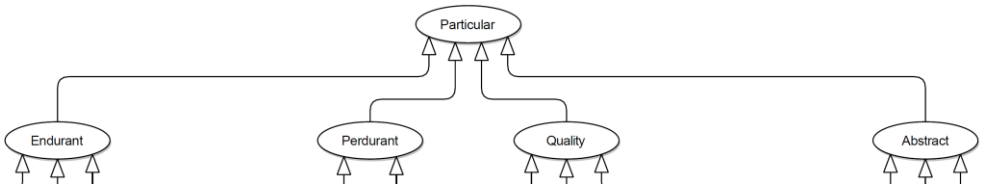


Figura A4.2. Primer nivel de la taxonomía de DOLCE (Bräuer 2007).

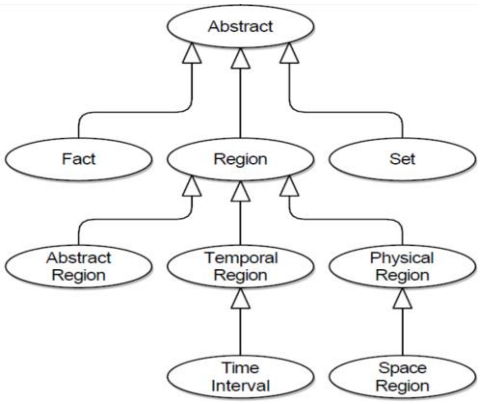


Figura A4.3. Especializaciones de la entidad *Abstract* en la ontología DOLCE (Bräuer 2007).

A continuación se presentan las especializaciones más relevantes de la clase *Endurant* (Figura A4.1):

- **Physical Endurant**. *Endurant* que tiene alguna *quality* espacial. Como se puede observar en la Figura A4.4, esta entidad tiene tres especializaciones.
- **Physical Object**. *Physical Endurant* que puede considerarse como una unidad. Es decir, sus instancias aunque estén compuestas o formadas por partes pueden considerarse como una entidad o un ente único.
- **Agentive y Non-agentive Physical Object**. Cuando a un *physical object* se le pueden atribuir creencias, deseos e intenciones, es de tipo *agentive*. En general, se asume que los *agentive physical objects* están constituidos por objetos *non-agentive physical objects*. Por ejemplo, una persona (*Agentive Physical Object)* está constituida por un organismo (*Non-agentive Physical Object)*. Otros ejemplos de n*on-agentive physical objects* son: casas, máquinas, piezas, etc.
- **Amount of Matter**. *Physical Endurant* que no puede considerarse como una entidad o ente único. Los *amounts of matter* son *endurants* incontables (como oro, acero, madera, arena, carne, etc.), a diferencia de los *physical objects* que son contables.
- **Feature**. Las *features* están albergadas en un *physical object* y solo tienen sentido en el conjunto donde se encuentran. Por ejemplo, el agujero de una arandela solo tiene sentido como la ausencia de material en la zona central de la misma. En general, una *feature* no es una parte del cuerpo de su anfitrión, en el sentido de algo que se pueda sustraer de ese cuerpo. Pero si esto fuera posible, la ausencia de la *feature* cambiaría la naturaleza del *physical object:* una arandela sin agujero no sería tal arandela, y en este sentido una *feature* puede confundirse con una *quality*. No ocurre lo mismo, por ejemplo con una máquina a la que se le quita un accesorio o un componente o se cambia su configuración: la máquina reconfigurada sigue siendo una máquina. Ejemplos típicos de *features* son las

denominadas entidades parásitas como agujeros, superficies y límites que son dependientes constantemente y genéricamente de *physical objects*, sus anfitriones.

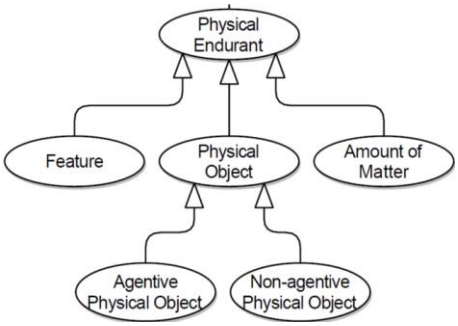


Figura A4.4. Especializaciones de la entidad *Physical Endurant* en la ontología DOLCE (Bräuer 2007).

- **Non-physical Endurant**. *Endurant* que no tiene *qualities* espaciales (Figura A4.5).
- **Non-physical Object**. Tiene dos especializaciones: *Social Object y Mental Object*, cuya diferencia se establece en función de que sus individuos sean o no dependientes genéricamente de una comunidad de agentes. Por ejemplo, una experiencia privada es un *mental object*. A su vez, dentro de los *social objects* se distinguen dos especializaciones:
    - **Agentive Social Object**. Los *agentive social objects* tienen creencias, deseos e intenciones –BDI– (Ferrario y Oltramari 2004). Por ejemplo, los *social agents*, como el presidente de los Estados Unidos o las *societies*, como CNR o Mercedes-Benz.
    - **Non-agentive Social Object**. Dependen genéricamente de las *societies* y no tienen creencias, deseos ni intenciones Por ejemplo, leyes, normas, tratados de paz, etc.

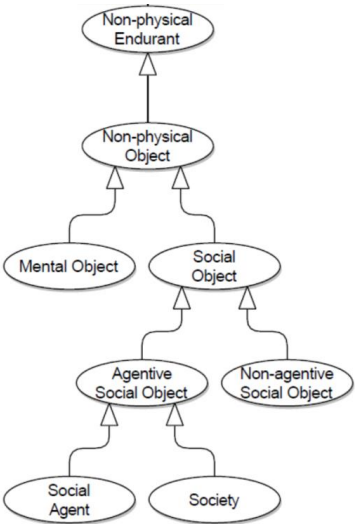


Figura A4.5. Especializaciones de la entidad *Non-Physical Endurant* en la ontología DOLCE (Bräuer 2007).

En la ontología DOLCE, los *perdurants* pueden ser de cuatro tipos distintos (*State*, *Process*, *Achievement* y *Accomplishment*) atendiendo a su carácter acumulativo, homeomérico y atómico (Figura A4.6 y Figura A4.7). Un *perdurant* es atómico si no tiene partes temporales. Un *perdurant* es acumulativo si al unirse o componerse con otro, se obtiene como resultado un *perdurant* del mismo tipo que ambos. Finalmente, un *perdurant* es homeomérico, si y si solo si, todas sus partes temporales están descritas por la misma expresión utilizada para describir el *perdurant* en su conjunto (Borgo y Leitao 2004).

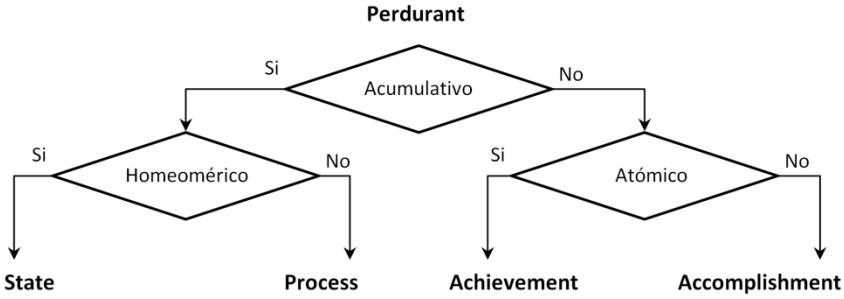


Figura A4.6. Tipos de *perdurants*, según información extraída de (Masolo et al. 2003).

- **Stative**. Especialización de la clase *Perdurant* cuyas instancias son acumulativas.
- **State**. *Perdurant* de tipo *Stative* y homeomérico.
- **Process**. *Perdurant* de tipo *Stative* y no homeomérico.

- **Event**. Especialización de la clase *Perdurant* cuyas instancias son no acumulativas o anti-acumulativas.
- **Achievement**. *Perdurant* no acumulativo (*Event*) y atómico.
- **Accomplisment**. *Perdurant* no acumulativo (*Event*) y no atómico (Figura A4.7).

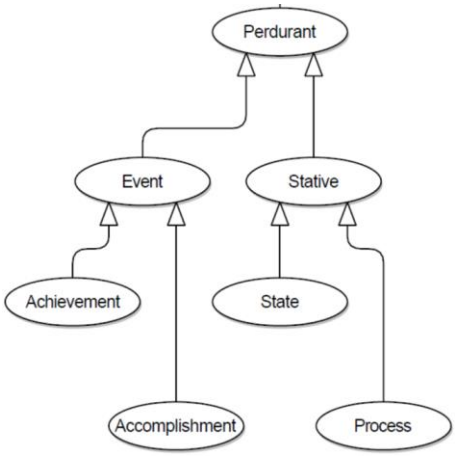


Figura A4.7. Especializaciones de la entidad *Perdurant* en la ontología DOLCE (Bräuer 2007).

En la Tabla A4.1 se muestran ejemplos para los tipos de *perdurants* expuestos anteriormente y ejemplos de aplicación en el contexto de la representación de un plan de proceso.

Tabla A4.1. Los *perdurants* y su *intention*.

	Achievement	**Accomplishment**	**State**	**Process**
Características	No acumulativo Atómico	No acumulativo No atómico	Acumulativo Homeomérico	Acumulativo No homeomérico
Ejemplo de perdurant	'Acabar de roscar'	'Operación de roscado' (considerando sus partes)	'Pieza roscada'	'Roscando' 'Roscado'
Intención	Sincronización	Describir partes de la operación y su secuencia	Precondición	Descripción genérica de la operación de roscado
Ejemplo de aplicación	La taladrina deja de proyectarse en el instante en que termina la operación de roscado (cuando se realiza el *achievement*)	La pasada de repaso se realiza después de diez pasadas de corte (se diferencia entre distintas partes del *accomplishment*)	La operación de tronzado requiere que la pieza esté roscada (no puede realizarse hasta que se haya alcanzado ese *state*)	Se ha seleccionado la herramienta número 2 para el roscado de la pieza (*process*)

Para finalizar la descripción de las entidades básicas de DOLCE, se presentan las especializaciones de *Quality* (Figura A4.8):

- **Temporal Quality**. *Quality* que es inherente a un *perdurant*, por ejemplo la ubicación temporal de un *perdurant*.
- **Physical Quality**. *Quality* que es inherente a un *physical endurant*, *por e*jemplo la ubicación espacial de un *endurant*.
- **Abstract Quality**. *Quality* que es inherente a un *non-physical endurant*.

Las *Qualities* pueden verse como las entidades básicas que pueden percibirse o medirse: formas, colores, tamaños, etc. En ocasiones este concepto se usa como sinónimo de propiedad, pero este no es el caso en DOLCE, ya que las *qualities* son *particulars* y las propiedades son *universals*. Este concepto de *Quality* es similar al *tropo* o instanciación de una propiedad, que aparece en la ontología de Hoffman y Rosenkrantz (Rosenkrantz 2006). En DOLCE no es posible que dos *particulars* puedan tener la misma *quality*, por tanto, cuando se dice que dos entidades tienen la misma *quality*, significa que tienen la misma posición en la *quality region*, llamada *quale*. Por ejemplo, si decimos que dos rosas tienen exactamente el mismo color, esto significa que sus *color qualities (quality)*, que son distintos, tienen la misma posición en el *color space (region)*, esto es, tienen el mismo *color quale*. Por tanto, el *quale* es una entidad abstracta que es una parte de una *quality region*. En la Figura A4.9 se representan los conceptos de *quality*, *quality region* y *quale* (indicado como *ql* en la misma).

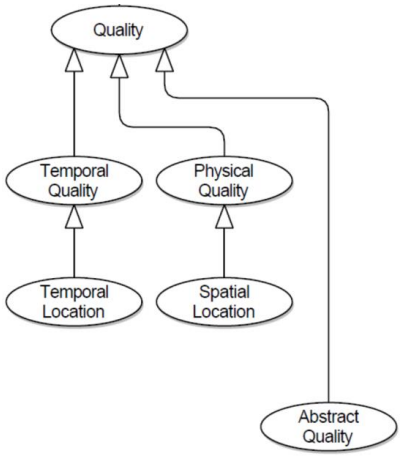


Figura A4.8. Especializaciones de la entidad *Quality* en la ontología DOLCE (Bräuer 2007).

Las *qualities* pueden ser directas e indirectas, y hay dos argumentos para esta distinción: el primero es el comportamiento simétrico de *perdurants* y *endurants* respecto a sus ubicaciones temporales y espaciales. Los *perdurants* tienen una ubicación temporal bien definida, mientras que su ubicación espacial proviene indirectamente de la ubicación espacial de sus participantes. Sin embargo, la mayoría de los *endurants*, los *physical endurants*, tienen una clara ubicación espacial, mientras que su ubicación temporal procede indirectamente de los *perdurants* en los que participan. El segundo argumento se debe a

que una *quality* puede tener múltiples dimensiones o *qualities*, por ejemplo, el color tiene tono, luminosidad, etc.

En DOLCE, las ubicaciones espacio-temporales son consideradas *qualities* individuales como el color y el peso. Por ejemplo, la ubicación temporal de un *physical object* pertenece a la *quality* tipo time, y su *quale* es un valor de la *region* espacio temporal.

Se asume que las *qualities* pertenecen a tipos de *quality* disjuntas, según el tipo de entidad a la que son inherentes directamente: las *temporal qualities* son inherentes directamente a *perdurants*, las *physical qualities* son inherentes directamente a *physical endurants* y las *abstract qualities* son inherentes directamente a *non-physical endurants*.

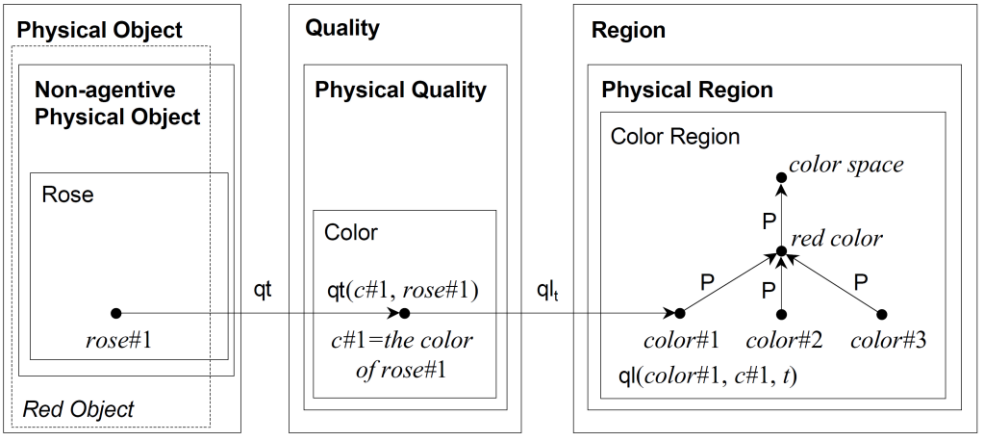


Figura A4.9. *Qualities* y *quality regions* (Masolo et al. 2003).

Un *endurant* puede cambiar a lo largo del tiempo, pudiendo presentar propiedades incompatibles en diferentes instantes de tiempo. Por tanto, todas las declaraciones realizadas en relación a las propiedades de un *endurant* deben hacerse en un marco temporal (Bräuer 2007).

A modo de resumen, en la Tabla A4.2 se presentan ejemplos de instancias de las entidades que ocupan la parte inferior del diagrama de la Figura A4.1, donde se representa la taxonomía de las entidades básicas de la ontología DOLCE.

Tabla A4.2. Ejemplos de categorías básicas de la ontología DOLCE (Masolo et al. 2003).

Categorías de DOLCE	Ejemplos
Abstract Quality	El valor de una calidad valorable.
Abstract Region	El valor convencional de un euro.
Accomplishment	Una conferencia, un ascenso o escalada, una representación. Una reconfiguración de una máquina.
Achievement	Alcanzar la meta del K2, una salida, una muerte. La finalización de una reconfiguración de una máquina.
Agentive Physical Object	Una persona humana (distinto de una persona jurídica).
Amount of Matter	Aire, oro, cemento.
Arbitrary Sum	Mi pie izquierdo y mi coche.
Feature	Un agujero, un golfo, una apertura, una frontera.
Mental Object	Una percepción.
Non-agentive Physical Object	Un martillo, una casa, un ordenador, un cuerpo humano.
Non-agentive Social Object	Una ley, un sistema económico, una moneda.
Physical Quality	El peso de una pluma, el color de una manzana.
Physical Region	El espacio físico, un área en el espectro de color, 80 Kg.
Process	Correr, escribir.
Social Agent	Una persona jurídica, un contrato.
Society	Fiat, Apple, el banco de Italia.
State	Estar sentado, estar abierto, ser feliz, ser rojo.
Temporal Quality	La duración de la Primera Guerra Mundial, el momento de inicio de los Juegos Olimpicos de 2000.
Temporal Region	El eje del tiempo, 22 de junio de 2002, un segundo.

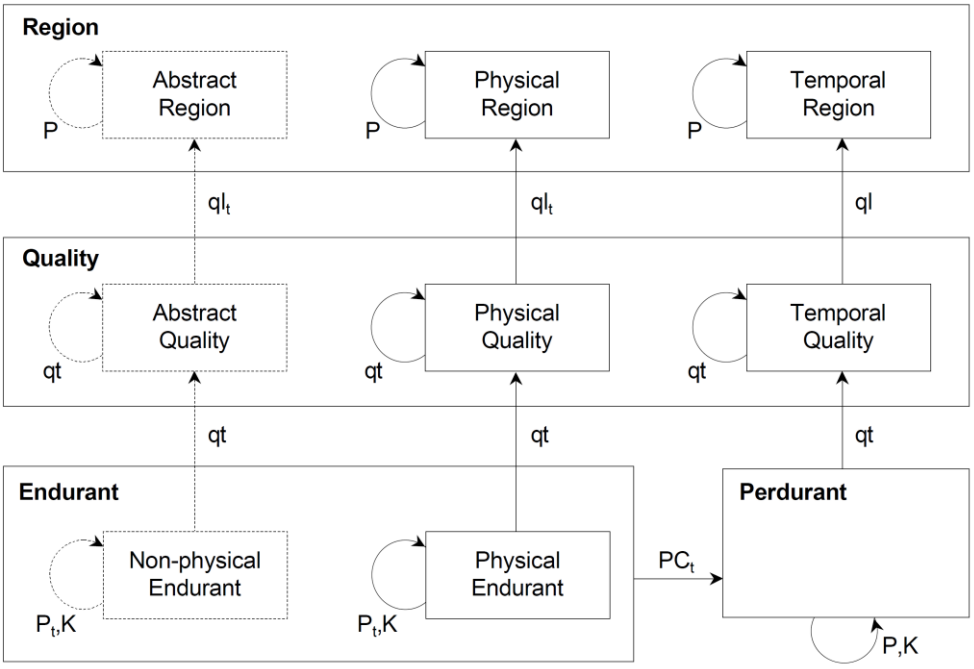
2. Relaciones básicas de DOLCE

Entre las relaciones básicas de DOLCE unas son temporales y otras atemporales (Masolo et al. 2003). A continuación se muestran las relaciones básicas más representativas de DOLCE:

- **Parthood**. La relación transitiva *Parthood* expresa que un individuo es parte de otro. Por ejemplo, la pieza *x* es parte del conjunto mecánico *y*.
- **Temporary Parthood**. La relación transitiva *Temporary Parthood* expresa que un individuo es parte de otro en un intervalo de tiempo determinado. Por ejemplo, la pieza *x* es parte del conjunto mecánico *y* durante el tiempo *t.*
- **Constitution**. La relación *Constitution* expresa que un individuo está constituido por otro individuo en un intervalo de tiempo determinado. Esta relación es diferente a la relación de identidad y a la relación *Parthood*. Por ejemplo, el ladrillo se hace con arcilla, pero ambos, arcilla y ladrillo no son lo mismo. Esta relación no es transitiva, de forma que siguiendo con el ejemplo anterior, la casa es de (está constituida por) ladrillos, el ladrillo es de arcilla, pero la casa no es de arcilla.
- **Participation**. Esta relación corresponde a la intuición de los *endurants* implicados en un *perdurant*. Los *endurants* participantes no son parte de los *perdurants*: solamente los *perdurants* pueden ser parte de otros *perdurants.* Por ejemplo, la máquina *x* participa en el proceso *y* durante el tiempo *t.*
- **Quality**. Mediante *Quality* se expresa la relación que existe entre una *quality* y el individuo al que es inherente. Por ejemplo, *x* es una quality de *y*.
- **Quale**. Mediante la relación *Quale* se expresa que una *quality* tiene un valor, durante un intervalo de tiempo, que es un *quale region*. Por ejemplo, *x* es el quale de *y* (durante *t*).

Las propiedades o relaciones consideradas pueden ser dependientes o independientes atendiendo al tiempo y al espacio. Las dependencias temporales pueden ser: específicas, cuando se establecen entre los elementos de la *extension* de la ontología, o genéricas, que solo pueden establecerse entre las propiedades de la ontología que forman parte de la *intension*. La dependencia espacial, añade a las anteriores condiciones de co-presencia (temporal) la condición de co-localización (espacial). De modo que, un *particular x* es dependiente constante y específicamente de otro *particular* y si y solo si, en cualqu*ier* instante de tiempo *t, x* no puede estar presente en *t* a menos que *y* también esté presente en *t*. Por ejemplo, una persona que depende de su cerebro. Del mismo modo, una propiedad ϕ es dependiente constante y genéricamente de otra propiedad ψ, si y solo si, *pa*ra cualquier instancia *x* de ϕ, en cualquier instante de tiempo *t, x* no puede estar presente en *t*, a menos que una cierta instancia *y* de ψ esté también presente en *t*. Por ejemplo, ser una persona depende de tener un cerebro.

En la Figura A4.10 se representan las entidades de alto nivel de la ontología DOLCE, junto con las relaciones básicas que se establecen entre ellas, y que han sido objeto de análisis en este anexo. Las líneas discontinuas de la parte izquierda del diagrama indican que los autores distinguen entre los *endurants* físicos y no físicos.



P (Parthood), K (Constitution), PC (Participation), qt(Quality), ql (Quale)

Figura A4.10. Relaciones primitivas de DOLCE entre categorías básicas (Masolo et al. 2003).

3. Referencias

Bock, C. 2006. "Interprocess communication in the process specification language". NIST Internal Report (NISTIR) 7348. National Institute of Standards and Technology.

Borgo, S. y P. Leitão. 2004. "The Role of Foundational Ontologies in Manufacturing Domain Applications". In: R. Meersman and Z. Tari (eds.), OTM Confederated International Conferences, ODBASE 2004, LNCS 3290, Springer, pp. 670–688.

Bräuer, M. 2007. "Design of a Semantic Connector Model for Composition of Metamodels in the Context of Software Variability." PhD diss., Technische Universität Dresden.

Ferrario, R. y A. Oltramari. 2004. "Towards a computational ontology of mind". In: Achille C. Varzi and Laure Vieu (eds.) Formal ontology in information systems, Proceedigs of the Intl. Conf. FOIS 2004. IOS Press, 287–297.

Guarino, N. 2006. *Formal Ontology for Information Systems Deliverable 3. General Ontology of Artefacts Including Subontology of Information Objects (final).*

Masolo, C., S. Borgo, A. Gangemi, N. Guarino y A. Oltramari. 2003. *WonderWeb Deliverable D18. Ontology Library (final)*. Laboratory for Applied Ontology ISTC CNR.

Rosenkrantz, G. 2006. “Ontological Categories.” Conference on The Metaphysics of E. J. Lowe, Department of Philosophy of the State University of New York at Buffalo, April 8–9.

Anexo 5. La ontología de recursos de TOVE

1. La ontología TOVE

Según Fox y Gruninger (1998) un modelo de empresa es una representación computacional de la estructura, actividades, procesos, información, recursos, personas, comportamiento, objetivos y restricciones de una empresa. El proyecto TOVE (TOronto Virtual Enterprise) tiene como objetivo el modelado de empresas, y como se aprecia en la Figura A5.1, el modelo propuesto se compone de diversas ontologías en el dominio de la fabricación que están relacionadas entre sí. Entre ellas, la ontología de recursos *Generic Enterprise Resource Ontology*, que tiene las características de ser genérica y reusable para una amplia variedad de aplicaciones. Esta ontología proporciona respuestas deductivas a preguntas de sentido común relativas a ciertas características de los recursos, como: divisibilidad, cantidad, ubicación, consumo, asignación, estructura y *capacity*. Particularmente, las preguntas de competencia de la ontología, son acerca de cómo cambian las propiedades de los recursos como resultado de las actividades, y acerca de la asignación de recursos en las tareas de programación (scheduling) a través del reconocimiento de las capacities de los recursos (Fadel, Fox y Gruninger 1994).

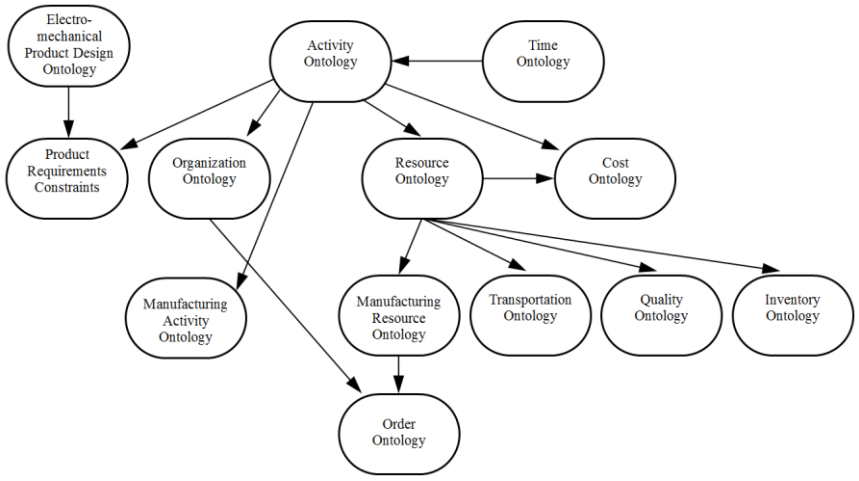


Figura A5.1. Toronto Virtual Enterprise Ontologies (Fox y Gruninger 1998).

Para TOVE, ser un recurso no es una propiedad innata de un objeto, sino una propiedad que deriva del rol que tiene el objeto respecto a la actividad. En otras palabras, el concepto de recurso está vinculado a su habilidad o capacidad para hacer algo.

Las llamadas propiedades primitivas de los recursos, sirven como punto de partida para la definición de la mayoría de las propiedades complejas de estos recursos. Estas propiedades primitivas se presentan a continuación, acompañadas en algunos casos de su representación en Prolog[2]. Así, por ejemplo, la propiedad *resource-know*, cuya sintaxis se muestra en la expresión 5.1 permite identificar un recurso `R` acerca del cual se desea razonar.

```
rknow(R)
```
[Expresión 5.1]

Cada recurso, identificado como `R`, tiene un rol (`Role`) respecto a una actividad (`A`), lo que se representa mediante la expresión 5.2.

```
role (R, A, Role)
```
[Expresión 5.2]

Los roles contemplados por TOVE son: materia prima, producto, instalación (*facility*), herramienta y operador. Además, en TOVE, existe la posibilidad de especificar la división de un recurso, ya sea físicamente (expresión 5.3) o funcionalmente (expresión 5.4). En ambas expresiones, `R2` representa la división o parte física/funcional del recurso `R`.

```
physical_division_of (R2, R)
```
[Expresión 5.3]

```
functional_division_of (R2, R)
```
[Expresión 5.4]

Por otra parte, se puede especificar la propiedad de un recurso de ser divisible respecto a una actividad sin afectar al rol del recurso respecto a la actividad. También se puede especificar, como se indica en la expresión 5.5, una unidad de medida cualitativa o cuantitativa (`Unit`) para un recurso (`R`) asociado a una actividad (`A`), cuyo identificador es Unit_ID.

```
Unit_of_measurement (R, Unit_ID, Unit, A)
```
[Expresión 5.5]

Las unidades de medida cualitativas, como {grande, medio, pequeño} y {bueno, malo}, se usan para especificar atributos de los recursos; mientras que las unidades de medida

[2] Prolog es un lenguaje lógico, y los programas en Prolog se componen de cláusulas de Horn, donde si es verdad el antecedente, entonces es verdad el consecuente. En Prolog no existen instrucciones de control. Su ejecución se basa en dos conceptos: la unificación y el *backtracking*. Gracias a la unificación, cada objetivo determina un subconjunto de cláusulas susceptibles de ser ejecutadas. Cada una de ellas se denomina punto de elección. Prolog selecciona el primer punto de elección y sigue ejecutando el programa hasta determinar si el objetivo es verdadero o falso. En caso de ser falso entra en juego el *backtracking*, que consiste en deshacer todo lo ejecutado situando el programa en el mismo estado en el que estaba justo antes de llegar al punto de elección. Entonces se toma el siguiente punto de elección que estaba pendiente y se repite de nuevo el proceso. Con ello, todos los objetivos terminan su ejecución en verdadero o en falso.

cuantitativas se usan para especificar los valores de ciertas magnitudes de los recursos, como peso, longitud y volumen. En esta misma línea, el predicado *measured by*, cuya sintaxis se muestra en la expresión 5.6, identifica el objeto (`Unit_id`) mediante el cual se mide un recurso `R` respecto a una actividad `A`.

```
measured_by (R, Unit_id , A)
```
[Expresión 5.6]

Por otra parte, TOVE establece que debe estar especificado el instrumento (objeto) con el que se efectúa la medición de cada unidad de medida.

TOVE permite especificar que un recurso es parte de (*component of*) otro recurso, lo que implica que este segundo recurso consta de uno o más sub-recursos. Un recurso puede ser un componente físico o funcional de otro recurso respecto a una actividad, y cualquiera de ellos tendrá un rol diferente al rol del recurso original. Para especificar la cantidad de un recurso se utiliza el predicado *rp* (*resource point*), que proporciona la información relativa a la cantidad del recurso en una unidad de medida, para un instante de tiempo y, en ocasiones, para una ubicación determinada. Existen dos variantes de *resource point*: 2D y 3D. En la primera, se especifica el recurso (`resource`), la cantidad (`Q`), el instante de tiempo (`time`) y la unidad de medida (`unit`), tal y como muestra la expresión 5.7. En la variante 3D de *resource point* (`rp`) se añade la especificación de la ubicación, indentificada mediante '`location`' en la expresión 5.8.

```
rp (resource, Q, time, unit)
```
[Expresión 5.7]

```
rp (resource, Q, time, location, unit)
```
[Expresión 5.8]

Existen además diversos predicados con los que se puede especificar la cantidad de un recurso que será consumida, usada o producida en un intervalo de tiempo y su unidad de medida. En relación a esto, se distingue entre los recursos continuos o incontables y los recursos discretos o contables. De forma análoga, es posible especificar si el recurso soporta una actividad en modo continuo o discreto.

TOVE prevé distintas formas de utilización de los recursos por parte de las actividades. Por ejemplo, impidiendo que un mismo recurso pueda ser utilizados por dos actividades simultáneamente, o estableciendo que un recurso sea parcial o totalmente inutilizable por una actividad si está asignado a otras actividades. Para esto último, además se precisa que la cantidad del recurso disponible se actualice dinámicamente conforme van finalizando las actividades a las que el recurso está asignado. Según lo anterior, un recurso estará disponible para una actividad si la cantidad del recurso puede soportar la actividad y no existe la restricción de uso simultáneo entre la actividad que requiere el recurso y las que ya están haciendo uso del mismo.

El concepto de *capacity* se define como el máximo conjunto de actividades que puede usar/consumir un recurso simultáneamente en un instante de tiempo dado. Si el recurso es físicamente indivisible, su *capacity* indica que una actividad puede usar/consumir el recurso. Pero si el recurso es físicamente divisible, su *capacity* representa el número de actividades que un recurso puede soportar simultáneamente. La determinación de la *capacity* de un recurso está ligada a la programación (scheduling) y puede resultar de gran complejidad cuando el recurso es divisible funcionalmente o cuando, debido a la

heterogeneidad en el uso del recurso, la cantidad del recurso precisada por la actividad y el tiempo de procesado de la actividad no son proporcionales.

Otros aspectos de interés contemplados por TOVE en relación a los recursos, tienen que ver con su selección y preparación. Por ejemplo: (a) especificar la historia de uso o consumo del recurso antes de un instante de tiempo dado, mostrando una lista de las actividades que han sido soportadas por el recurso durante un periodo de tiempo cuyo instante final es igual o inferior al instante de tiempo dado; (b) especificar la configuración de un recurso en relación a una actividad, configuración que se mantendrá al menos hasta el instante de finalización de la actividad, estableciendo una restricción de uso simultáneo entre dos actividades que precisen el mismo recurso pero con distinta configuración; (c) especificar la duración requerida en la preparación de un recurso para ser usado en una actividad, incluyendo el consumo de tiempo para el cambio de configuración y para el cambio de ubicación; y (d) especificar el o los recursos alternativos que pueden ser usados o consumidos por una actividad, lo que resulta de utilidad para el caso de rotura de una máquina, o la no disponibilidad de un recurso en general.

Los estados están vinculados al desarrollo de las actividades. En TOVE, un estado representa lo que es cierto para el desarrollo de una actividad o lo que es cierto tras la realización de la actividad. La situación o valor de un estado depende de la situación de los recursos que la actividad usa o consume. Por ejemplo, el estado de una actividad consumiendo un recurso puede ser *posible* o *terminado*. El estado será *posible* si el recurso está disponible para la actividad, no ha sido asignado a otra actividad, y la actividad no está ejecutándose; mientras que el estado será *terminado* si la actividad ya ha sido soportada por el recurso y realizada.

2. Referencias

Fadel, F. G., M. S. Fox y M. Gruninger. 1994. “A generic enterprise resource ontology.” Paper presented at the 3rd IEEE workshop on enabling technologies. Infraestructure for collaborative enterprises, Morgantown, April 17–19.

Fox, M. S. y M. Gruninger. 1998. “Enterprise modeling.” *AI Magazine* 19 (3): 109–121.

Anexo 6. La ontología Product and Processes Development Resource Capabilities (PPDRC)

1. Introducción

En este anexo se enumeran y describen las clases, relaciones, reglas e individuos que constituyen la *intension* de la ontología PPDRC, junto con su descripción. La información contenida en este anexo se ha extraído fundamentalmente de la última versión de la ontología PPDRC editada en Protégé (PPDRC 2013) y que puede ser consultada a través del enlace http://www.coapp.es/ontologies/2013/0/PPDRC_v1.owl

2. Clases de la ontología PPDRC

La Figura A6.1 muestra la taxonomía de clases de la ontología PPDRC tal y como aparece en el editor de ontologías Protégé. A continuación se describen cada unas de estas clases, incluyendo, cuando procede, algún ejemplo relativo a su interpretación o utilización en la ontología.

- **ActivityOccurrence.** La entidad *ActivityOccurrence* constituye, junto con *Object* y *Region*, el grupo de tres entidades mutuamente excluyentes del primer nivel de la ontología PPDRC. *ActivityOccurrence* corresponde al concepto de *perdurant* de la ontología DOLCE. Un *perdurant* es una entidad en la que solamente algunas de sus partes genuinas están presentes cuando la entidad está presente. La entidad *ActivityOccurrence* representa a cualquier realización (occurrence) de una actividad. Las *ActivityOccurrences* u occurrences utilizan y producen o transforman objetos. Los individuos pertenecientes a la clase *ActivityOccurrence*

tienen un único inicio y final, definidos respectivamente por sendos instantes de tiempo, y son la realización de una única actividad.

- **Arboreal**. Es una *ActivityOccurrence* primitiva, y por tanto, una especialización de la clase *ActivityOccurrence*, que forma parte del *Occurrence_tree*. Este concepto de *Occurrence_tree* sirve para representar el conjunto de todas las posibles actividades atómicas que pueden realizarse en el marco de una actividad. El *Occurrence_tree* de una actividad representa todo lo que puede ocurrir durante la ejecución de una actividad, incluyendo *ActivityOccurrences* (realizaciones) que no forman parte de esa actividad, y que pueden ocurrir entre las realizaciones de sus subactividades. El *Occurrence_tree* se puede utilizar para expresar restricciones respecto a las secuencias de realizaciones permitidas para un proceso en particular, por ejemplo, para especificar lo que debe pasar, lo que puede pasar, y lo que no debe pasar.

- **Initial**. *Initial* es una *ActivityOccurrence* de tipo *Arboreal* que establece el inicio de una secuencia de *ActivityOccurrences* de tipo *Arboreal*. En otras palabras, *Initial* es el *root* (ver *Object properties*) de un *Occurrence_tree*.

- **Sucessor**. *Sucessor* es una especialiación de la clase *ActivityOccurrence* cuyo axioma de definición establece que sus individuos tienen alguna relación de tipo *successorOf* con individuos de la clase *ActivityOccurrence*. En otras palabras, sus individuos suceden (va a continuación) de otros individuos de la clase *ActivityOcurrence*.

- **Workflow**. La entidad *Workflow* es un tipo de *ComplexOccurrence* en la que necesariamente tiene que mostrarse su configuración. En otras palabras, en la occurrence de una actividad de tipo *Workflow* debe especificarse la estructura y la secuencia de sus partes temporales. Los *workflows* son un tipo de occurrence compleja con un objetivo común compartido por uno o varios agentives, que se coordinan para alcanzar este objetivo.

- **Object**. Una de las tres entidades de primer nivel de la ontología, que junto con *ActivityOccurrence* y *Region* forma un grupo de tres clases mutuamente excluyentes. Corresponde al concepto de *endurant* de la ontología DOLCE. El *endurant* es la entidad en la que todas sus partes esenciales están presentes en todo momento en el que la entidad está presente. Un *endurant* es una entidad que está presente completamente en cualquier momento que está presente. Un *endurant* está en el tiempo, existe en el tiempo y puede genuinamente cambiar en el tiempo.

- **Agentive**. Las entidades de tipo *Agentive* pueden tener objetivos. Sus individuos son *Objects* que tienen alguna relación de tipo *hasGoal* con individuos de la clase *Goal*. El cumplimiento de sus objetivos es lo que guía las iniciativas de los *agentives* durante la realización de las actividades. La entidad *Agentive* se corresponde con aquellos objetos, como las personas, que tienen creencias, deseos e intenciones –BDI– y cuyo comportamiento está marcado por la motivación.

- **Social_Agent**. La entidad *Social_Agent* es una clase definida, cuyo axioma de definición establece que sus individuos están caracterizados por la *Capability ManageWorflow*. La entidad *Social_Agent* pertenece simultáneamente a las clases

Agentive y *SocialObject*. Por ejemplo, una persona es *Social_Agent*, puesto que además de tener carácter agentive, es un concepto social. *Social_Agent* es una entidad caracterizada por la *Capability* de lanzar y gestionar *Workflows*.

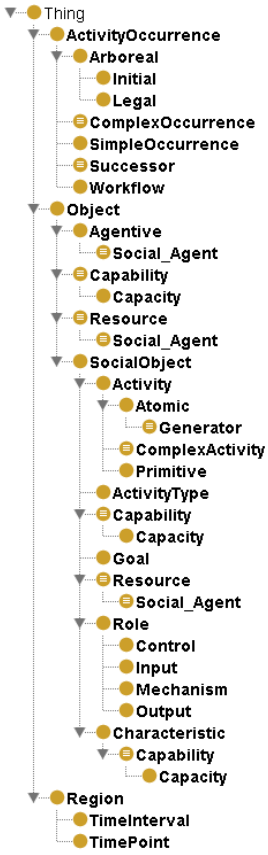


Figura A6.1. Taxonomía de clases de la ontología PPDRC (PPDRC 2013).

- **Capability**. La entidad *Capability* es una clase definida cuyo axioma de definición establece que sus individuos pertenecen a la clase *Characteristic* y tienen alguna relación de tipo *executing* con algún individuo de la clase *ActivityType*. En otras palabras, la entidad *Capability* es el objeto social (*Social_Object*) o definición compartida que caracteriza el uso de un recurso en un tipo de actividad, expresando unas competencias para ejecutar actividades de ese tipo y en su caso, el nivel de prestación alcanzado en las ejecuciones de ese tipo de actividades. Por ejemplo, desarrollar el conjunto mecánico de una bomba de engranajes es una actividad, mientras que la competencia para desarrollar conjuntos mecánicos simples es una capability. Del mismo modo, rectificar los dientes de un engranaje de la bomba es una actividad, mientras que la obtención de superficies complejas

con un nivel de prestación caracterizado por una rugosidad media de 0,2 micrometros es una capability.

- **Capacity**. La entidad *Capacity* es una especialización de la entidad *Capability* que se expresa en términos de '*amount of production*'. Por ejemplo, la *capacity* asociada a un recurso que tiene competencias de diseño permite determinar el ratio de desempeño del recurso para diseñar, o la cantidad de trabajo a desarrollar por el recurso diseñando, en un horizonte temporal definido. Del mismo modo, la *capacity* de un recurso puede expresar su tasa de producción (en mm^3 / min) para rectificar superficies planas con un determinado grado de rugosidad.

- **Resource**. La entidad *Resource* es una especialización de *SocialObject* definida en la ontología mediante un axioma en el que se establece que sus individuos son objetos caracterizados por alguna *capability*. En otras palabras, cualquier objeto que tiene una relación de tipo *characterizedBy* con un individuo de la entidad *Capability* pertenece a la clase *Resource*. Por tanto, todos los individuos de la clase *Resource* tienen la competencia o la habilidad de ejecutar al menos un tipo de actividad. Las entidades *Resource* y *Characteristic* son disjuntas.

- **SocialObject**. La entidad *SocialObject* es una especialización de la clase *Object* que responde al concepto de descripciones compartidas que permiten el entendimiento mutuo entre los miembros de una comunidad.

- **Activity**. La entidad *Activity* es una especialización de *SocialObject* que corresponde al concepto de actividad. Las actividades representan todo aquello que puede hacerse o realizarse y constituyen un patrón que puede repetirse. Este concepto abstracto de actividad como un patrón repetible, se concreta con cada una de sus ejecuciones que suceden en el tiempo y que denominamos occurrences. Las entidades *Activity* y *Characteristic* son disjuntas.

- **Atomic**. La entidad *Atomic* es una especialización de la entidad *Activity*. Una actividad es atómica si y solo si es primitiva o si es la superposición concurrente de un conjunto de actividades primitivas. En cualquier caso, una actividad atómica es diferente de una actividad compleja. En PSL, las actividades que participan en agregaciones concurrentes se pueden considerar como subactividades de actividades atómicas. Por simplicidad, PSL representa las actividades concurrentes como subactividades de una actividad atómica.

- **Generator**. La entidad *Generator* se ha definido en la ontología como una especialización de la entidad *Atomic* cuyos individuos tienen alguna relación de tipo *occurrences* con individuos de la clase *Arboreal*. En otras palabras, *Generator* es una *Activity* de tipo *Atomic* cuyas occurrences son elementos del Occurrence_tree.

- **ComplexActivity**. La entidad *ComplexActivity* está definida mediante un axioma que establece que sus individuos pertenecen a la clase *Activity* y tienen alguna relación de tipo *subactivities* con individuos de la clase *Activity*. Es decir, *ComplexActivity* es una actividad que tiene subactividades.

- **Primitive**. La entidad *Primitive* es una especialización de la clase *Activity* y es distinta de la entidad *ComplexActivity*. Es decir, es una actividad que no tiene subactividades.
- **ActivityType**. La entidad *ActivityType* es una especialización de *SocialObject* que representa la abstracción de un grupo o tipo de actividades, y que es diferente de la entidad *Characteristic*.
- **Goal**. La entidad *Goal* es una especialización de la entidad *SocialObject* diferente de la entidad *Characteristic*. Los *Goals* son los objetivos. Las entidades de tipo *Agentive* son las únicas que pueden tener *Goals*.
- **Role**. La entidad *Role* es una especialización de *SocialObject* distinta de la entidad *Characteristic*. *Role* es un objeto social que se identifica con la manifestación de unas *capabilities* en la realización/ejecución de una actividad (*ActivityOccurrence*). Los recursos son los únicos objetos que pueden tener roles. Mientras que una *capability* está ligada a un tipo de actividad, un individuo de la clase *Role* establece el tipo de participación de un recurso en una *ActivityOccurrence* concreta. Un recurso puede participar en una misma actividad con roles diferentes según las *capabilities* que exhiba. Por ejemplo, un recurso que tiene la *capability* para dirigir, puede no participar como director en la occurrence de una actividad. Los individuos de la clase *Role* solamente pueden relacionarse, mediante *isPresent*, con *ActivityOccurrences* de tipo *Arboreal* o *ActivityOccurrences* primitivas.
- **Control**. Rol de control. Es el rol de un recurso que interviene en la realización de una actividad controlándola o regulándola.
- **Input**. Rol de entrada. Es el rol de un recurso que interviene como entrada en la realización de una actividad.
- **Mechanism**. Rol de mecanismo. Es el rol de un recurso que ejecuta la actividad.
- **Output**. Rol de salida. Es el rol de un recurso resultante de la realización de una actividad.
- **Characteristic**. La entidad *Characteristic* es una especialización de la entidad *SocialObject* cuyos individuos expresan las cualidades de otros individuos de la ontología. La entidad *Characteristic* es distinta de las entidades *Resource*, *Goal*, *ActivityType*, *Role* y *Activity*.
- **Region**. Una de las tres entidades de primer nivel de la ontología, que junto con *ActivityOccurrence* y *Object*, forman un grupo de entidades mutuamente excluyentes. El concepto de *region* sirve para expresar cualidades de otros conceptos. *Region* representa el espacio en el que se encuentran los valores (*quales*) correspondientes a una cualidad (*quality*). Un *quale* expresa el valor que toma una cualidad o característica entre los valores establecidos para la misma.
- **TimeInterval**. La entidad *TimeInterval* es una especialización de la clase *Region* que representa una característica temporal incremental inherente a las *ActivityOccurrences* y a la existencia de los objetos.

- **TimePoint**. La entidad *TimePoint* es una especialización de la clase *Region* que representa una característica absoluta de tipo tiempo inherente a las *ActivityOccurrences* y a la existencia de los objetos.

3. Relaciones de la ontología PPDRC

3.1. Object Properties

La Figura A6.2 y la Figura A6.3 muestran, en el modo en que aparecen en el editor de ontologías Protégé, las relaciones (*Object Properties*) que pueden establecerse entre individuos de la ontología PPDRC. Las relaciones entre clases y las propiedades de estas clases se reúnen en los predicados de la ontología. A continuación, se describe brevemente cada una de estas relaciones y se incluye, en algunos casos, un breve comentario y/o un ejemplo relativo a su interpretación o utilización en la ontología.

- **after**. El predicado *after* (*TimePoint*, *TimePoint*) expresa la relación de ordenación temporal 'posterior a'. Ejemplo: ?t1 *after* ?t2. El *TimePoint* ?t1 es posterior al *TimePoint* ?t2. La relación *after* es transitiva y su inversa es la relación *before*.

- **before**. El predicado *before* (*TimePoint*, *TimePoint*) expresa la relación de ordenación temporal 'anterior a'. Ejemplo: ?t1 *before* ?t2. El *TimePoint* ?t1 es anterior al *TimePoint* ?t2. La relación *before* es transitiva y su inversa es la relación *after*.

- **beginOf**. El predicado *beginOf* (*TimePoint*, *ActivityOccurrence* / *Object*) expresa el instante de tiempo en el que se inicia la realización de una actividad o el instante de tiempo a partir del cual un objeto empieza a estar disponible. Ejemplo: ?t *beginOf* ?x. El *TimePoint* ?t es el inicio de ?x. La inversa de la relación *beginOf* es la relación *beginsAt*.

- **beginsAt**. El predicado *beginsAt* (*ActivityOccurrence* / *Object*, *TimePoint*) expresa la relación entre la realización de una actividad y el instante de tiempo en el que se inicia, o la relación entre un objeto y el instante de tiempo a partir del cual dicho objeto está disponible. Ejemplo: ?x *beginsAt* ?t. ?x se inicia o está disponible a partir del *TimePoint* ?t. La inversa de la relación *beginsAt* es la relación *beginOf*.

- **behaves**. El predicado *behaves* (*Role*, *Capability*) establece el tipo de relación de un recurso en una *ActivityOccurrence* concreta. Por ejemplo, un recurso que participa con rol de tipo *Mechanism* en la realización de una actividad de inspección, usando su capability para obtener mediciones con una incertidumbre inferior a 0,1 mm. La inversa de la relación *behaves* es la relación *isUsed*.

- **capabilityRequiredBy**. El predicado *capabilityRequiredBy* (*Capability*, *ActivityOccurrence*) expresa la necesidad de una determinada *Capability* para realizar una *ActivityOccurrence* concreta. La inversa de esta relación es *requiresCapability*.

- **characterizedBy**. El predicado *characterizedBy* (*Object*, *Characteristic*) establece la relación de un *Object* con sus *Characterisics*. La relación inversa de *characterizedBy* es *characterizes*.

- **characterizes**. El predicado *characterizes (Characteristic, Objetc)* establece la relación entre una *Characteristic* y el *Object* al que caracteriza. La inversa de la relación *characterizes* es *characterizedBy*.

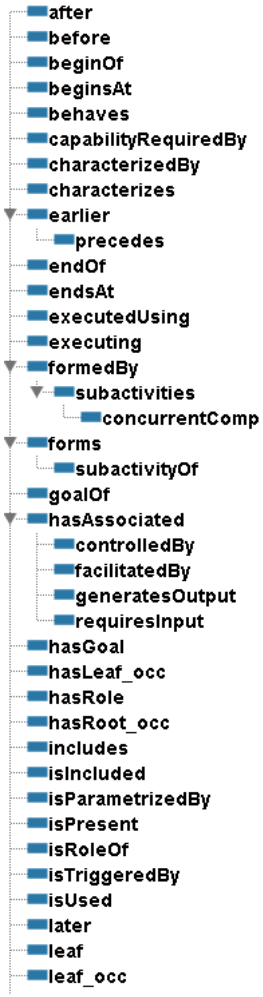


Figura A6.2. *Object Properties* de la ontología PPDRC (PPDRC 2013).

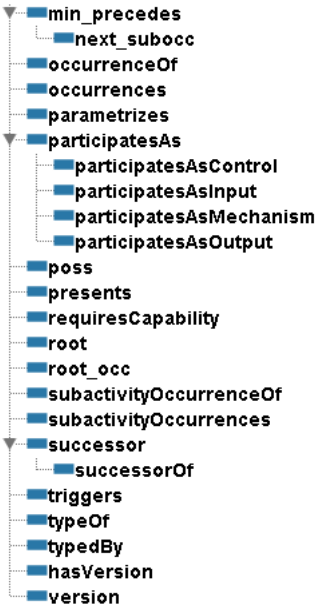


Figura A6.3. *Object Properties* de la ontología PPDRC (PPDRC 2013).

- **earlier.** El predicado *earlier* (*Arboreal*, *Arboreal*) expresa la relación de ordenación temporal 'anterior a'. Ejemplo: occ1 *earlier* occ2. La *ActivityOccurrence* occ1 es anterior a la *ActivityOccurrence* occ2. La relación *earlier* es transitiva y su inversa es la relación *later*.

- **precedes**. El predicado *precedes* (*Arboreal*, *Legal*) expresa la relación ordenación temporal 'precede a'. Ejemplo: occ1 *precedes* occ2. La *ActivityOccurrence Arboreal* occ1 es anterior a la occurrence *Legal* occ2, y todas las ocurrences entre occ1 y occ2 son legales.

- **endOf.** El predicado *endOf* (*TimePoint*, *ActivityOccurrence* / *Object*) expresa el instante de tiempo en el que finaliza la realización de una actividad o el instante de tiempo a partir del cual un objeto deja de estar disponible. Ejemplo: ?t *endOf* ?x. El *TimePoint* ?t es el final de ?x. La inversa de la relación *endOf* es la relación *endsAt*.

- **endsAt.** El predicado *endsAt* (*ActivityOccurrence* / *Object*, *TimePoint*) expresa la relación entre la realización de una actividad y el instante de tiempo en el que finaliza, o la relación entre un objeto y el instante de tiempo a partir del cual dicho objeto deja de estar disponible. Ejemplo: ?x *endsAt* ?t. ?x finaliza o deja de estar disponible a partir del *TimePoint* ?t. La inversa de la relación *endsAt* es la relación *endOf*.

- **executedUsing**. El predicado *executedUsing* (*ActivityType*, *Capability*) expresa la relación que existe entre un *ActivityType* y la *Capability* necesaria para la ejecución de actividades de ese tipo. Su inversa es *executing*.

- **executing.** El predicado *executing* (*Capability*, *ActivityType*) expresa la relación entre la *Capability* y el *ActivityType* al que pertenece la actividad que se está ejecutando haciendo uso de dicha *Capability*. Su inversa es *executedUsing*.

- **formedBy.** El predicado *formedBy* (*entity*, *entitiy*) se puede aplicar sobre cualquiera de las entidades de primer nivel de la ontología (*Object*, *ActivityOccurrence* y *Region*). *FormedBy* expresa relaciones de composición entre entidades del mismo tipo: composición entre objetos, composición temporal entre occurrences y composición entre regions. El predicado *formedBy* aporta el mecanismo para estructurar las entidades de la ontología a distintos niveles de detalle o agregación. Por ejemplo, el desarrollo de un proceso es una occurrence compleja, que para un nivel de agregación alto puede considerarse una occurrence simple. Del mismo modo, según el punto de vista considerado, un equipo humano encargado del desarrollo de un producto puede verse como una entidad elemental o como una entidad compleja formada por cada uno de los subequipos. Este predicado *formedBy* expresa relaciones transitivas y reflexivas que se limitan a identificar las entidades que constituyen otra entidad, sin especificar otros tipos de relaciones entre dichas entidades. La transitividad de este predicado aplicada al ejemplo anterior implica que si el equipo de desarrollo para un producto está formado por varios grupos de trabajo y cada grupo de trabajo está formado por unas personas, entonces, el equipo de desarrollo también estará formado por esas personas. La inversa de *formedBy* es *forms*.

- **subactivities**. La relación transitiva *subactivities* (*Activity*, *Activity*) es una especialización de *formedBy*. Ejemplo: ?a *subactivities* ?act expresa que ?act es una subactividad de ?a. La inversa de *subactivities* es *subactivityOf*.

- **concurrentComp**. La relación transitiva *concurrentComp* es una especialización de *subactivities* que expresa la composición concurrente de actividades. Ejemplo: ?c *concurrentComp* ?a expresa que ?c está compuesta concurrentemente por ?a.

- **forms**. La relación transitiva y reflexiva *forms* (*entity*, *entity*) expresa la relación entre una entidad y la entidad de la que forma parte. Su inversa es la relación *formedBy*.

- **subactivityOf**. La relación transitiva *subactivityOf* (*Activity*, *Activity*) es una especilización de la relación *forms*, que expresa la relación entre una actividad y la actividad de la que forma parte. Su inversa es la relación *subactivities*.

- **goalOf**. El predicado *goalOf* (*Goal, Agentive*) expresa la relación entre el objetivo (*Goal*) y la entidad *Agentive* que tiene dicho objetivo. Su inversa es la relación *hasGoal*.

- **hasAssociated**. El predicado *hasAssociated* establece la relación entre la realizacion de una activiad primitiva (*Primitive*) y el recurso que participa en la ejecución de dicha *ActivityOccurrence*. Su inversa es la relación *participatesAs*.

- **controlledBy**. El predicado *controlledBy* es una especialización de *hasAssociated* que establece la relación entre la realizacion de una actividad primitiva (*Primitive*) y el recurso que participa en la ejecución de dicha *ActivityOccurrence* con rol de tipo *Control*. Su inversa es la relación *participatesAsControl*.

- **facilitatedBy**. El predicado *facilitatedBy* es una especialización de *hasAssociated* que establece la relación entre la realizacion de una actividad primitiva (*Primitive*) y el recurso que participa en la ejecución de dicha *ActivityOccurrence* con rol de tipo *Mechanism*. Su inversa es la relación *participatesAsMechanism*.

- **generatesOutput**. El predicado *generatesOutput* es una especialización de *hasAssociated* que establece la relación entre la realizacion de una actividad primitiva (*Primitive*) y el recurso que participa en la ejecución de dicha *ActivityOccurrence* con rol de tipo *Output*. Su inversa es la relación *participatesAsOutput*.

- **requiresInput**. El predicado *requiresInput* es una especialización de *hasAssociated* que establece la relación entre la realizacion de una actividad primitiva (*Primitive*) y el recurso que participa en la ejecución de dicha *ActivityOccurrence* con rol de tipo *Input*. Su inversa es la relación *participatesAsInput*.

- **hasGoal**. El predicado *hasGoal* (*Agentive*, *Goal*) expresa la relación entre un *Agentive* y su objetivo (*Goal*). Las entidades de tipo *Agentive* son las únicas que pueden tener objetivos, siendo el cumplimiento de estos objetivos lo que guía las iniciativas de los *Agentives* durante la realización de las actividades. La inversa de *hasGoal* es la relación *goalOf*.

- **hasLeaf_occ.** El predicado *hasLeaf_occ*, que relaciona dos *ActivityOccurrences*, por ejemplo Occ1 *hasLeaf_occ* Occ2, expresa que la primera *ActivityOccurrence* (Occ1), que es compleja, tiene por última *ActivityOccurrence* a Occ2. La relación inversa de *hasLeaf_occ* es *leaf_occ*.

- **hasRole**. El predicado *hasRole* (*Object*, *Role*) expresa la relación que existe entre un individuo de la clase *Object* y su *Role*. La inversa de *hasRole* es isRoleOf.

- **hasRoot_occ**. El predicado *hasRoot_occ*, que relaciona dos *ActivityOccurrences*, por ejemplo Occ1 *hasRoot_occ* Occ2, expresa que la primera *ActivityOccurrence* (Occ1), que es compleja, tiene por primera *ActivityOccurrence* a Occ2. La relación inversa de *hasRoot_occ* es *root_occ*.

- **includes.** El predicado *includes*, que se puede establecer entre cualquier pareja de entidades del mismo tipo, expresa la relación entre una entidad y otra entidad incluida en la primera. Esta relación, que es transitiva y reflexiva, tiene por inversa a la relación *isIncluded*.

- **isIncluded**. El predicado *isIncluded*, que se puede establecer entre cualquier pareja de entidades del mismo tipo, expresa la relación entre una entidad y otra entidad que incluye a la primera. Esta relación, que es transitiva y reflexiva, tiene por inversa a la relación *includes*.

- **isParametrizedBy.** El predicado *isParametrizedBy* (*Region*, *Capability*) expresa la relación entre una *region* y la *capability* que cuantifica. Su inversa es *parametrizes*.

- **isPresent.** El predicado *isPresent* (*Role*, *Arboreal*) expresa que un rol de un recurso está presente en una *ActivityOccurrence* de tipo *Arboreal*. Es la inversa de la relación *presents*.
- **isRoleOf.** El predicado *isRoleOf* (*Role*, *Object*) expresa la relación entre un rol y el objeto al que pertenece. Su inversa es la relación *hasRole*.
- **isTriggeredBy.** El predicado *isTriggeredBy* (*Workflow*, *Social_Agent*) expresa la relación entre un *Workflow* y el *Social_Agent* que lo lanza. Su inversa es la relación *triggers*.
- **isUsed.** El predicado *isUsed* (*Capability*, *Role*) establece la relación de una *Capability* con el rol de un recurso que está siendo utilizado en la realización de una actividad. La inversa de esta relación es *behaves*.
- **later**. El predicado *later* (*Arboreal*, *Arboreal*), que es transitivo, expresa la relación 'es posterior a' entre dos *ActivityOccurrences Arboreal*. Su inversa es *earlier*.
- **leaf**. La relación *leaf*, que se establece entre la *ActivityOccurrence* de una actividad atómica y una *Activity*, expresa que dicha *ActivityOccurrence* es la última de las que componen la *Activity*.
- **leaf_occ**. El predicado *leaf_occ*, que se establece entre *ActivityOccurrences*, por ejemplo Occ1 *leaf_occ* Occ2, expresa que la *ActivityOccurrence* Occ1 es la última de las que forman parte de la *ActivityOccurrence* Occ2. La relación inversa de *leaf_occ* es *hasLeaf_occ*.
- **min_precedes**. El predicado *min_precedes*, que se establece entre *ActivityOccurrences* de actividades atómicas, por ejemplo, occ1 *min_precedes* occ2, expresa que la *ActivityOccurrence* occ1 precede a la *ActivityOccurrence* occ2, donde ambas (occ1 y occ2) pertenecen a la misma actividad.
- **next_subocc**. El predicado *next_subocc*, que es una especialización de *min_precedes*, se establece entre dos *ActivityOccurrences*. Por ejemplo, occ1 *next_subocc* occ2, que expresa que la *ActivityOccurrence* occ2 es inmediatamente posterior a la *ActivityOccurrence* occ1. Ambas, occ1 y occ2, pertenecen a la misma actividad, y no existe ninguna otra *ActivityOccurrence* de la actividad entre ellas.
- **occurrenceOf**. El predicado *occurrenceOf* (*ActivtyOccurrence*, *Activity*) expresa la relación que existe entre una *ActivityOccurrence* y la actividad que realiza. Su inversa es la relación *occurrences*.
- **occurrences.** Es la relación entre una *Activity* y una *ActivtyOccurrence*, que expresa que la primera se realiza mediante la segunda. Su inversa es la relación *occurrenceOf*.
- **parametrizes.** El predicado *parametrizes* (*Capability*, *Region*) expresa la relación entre una *capability* y la *region* que la parametriza. Su inversa es la relación *isParametrizedBy*.

- **participatesAs**. El predicado *participatesAs* establece la relación entre un recurso que participa en la realización de una actividad primitiva (*Primitive*) y dicha *ActivityOccurrence*. Su inversa es la relación *hasAssociated*.

- **participatesAsControl**. Este predicado, que es una especialización de *participatesAs*, establece la relación entre un recurso que participa con un rol de tipo *Control* en la realización de una actividad primitiva (*Primitive*) y dicha *ActivityOccurrence*. Su inversa es la relación *controlledBy*.

- **participatesAsInput**. Este predicado, que es una especialización de *participatesAs*, establece la relación entre un recurso que participa con un rol de tipo *Input* en la realización de una actividad primitiva (*Primitive*) y dicha *ActivityOccurrence*. Su inversa es la relación *requiresInput*.

- **participatesAs Mechanism**. Este predicado, que es una especialización de *participatesAs*, establece la relación entre un recurso que participa con un rol de tipo *Mechanism* en la realización de una actividad primitiva (*Primitive*) y dicha *ActivityOccurrence*. Su inversa es la relación *facilitatedBy*.

- **participatesAsOutput.** Este predicado, que es una especialización de *participatesAs*, establece la relación entre un recurso que participa con un rol de tipo *Output* en la realización de una actividad primitiva (*Primitive*) y dicha *ActivityOccurrence*. Su inversa es la relación *generatesOutput*.

- **poss.** El predicado *poss*, que se establece entre una *Activity* y una *ActivityOccurrence*, expresa que dicha *Activity* puede realizarse después de la *ActivityOccurrence*.

- **presents.** El predicado *presents* (*Primitive*, *Role*) expresa la relación que existe entre una *ActivityOccurrence* primitiva y el rol que está presente en la misma. Su inversa es la relación *isPresent*.

- **requiresCapability**. El predicado *requiresCapability* (*ActivityOccurrence*, *Capability*) expresa que para realizar una *ActivityOccurrence* concreta se precisa una determinada *Capability*. La inversa de esta relación es *capabilityRequiredBy*.

- **root**. El predicado *root*, establece la relación que existe entre la primera de las *ActivityOccurrences* (atómicas) de una actividad y dicha actividad.

- **root_occ**. El predicado *root_occ*, que se establece entre *ActivityOccurrences*, por ejemplo, occ1 *root_occ* occ2, expresa que la *ActivityOccurrence* occ1 es la primera entre las *ActivityOccurrences* de occ2.

- **subactivityOccurrenceOf**. El predicado *subactivityOccurrenceOf*, que se establece entre dos *ActivityOccurrences*, expresa que la primera de ellas es una subocurrencia de la segunda. Esta relación, que es transitiva, tiene como inversa la relación *subactivityOccurrences*. Ejemplo: occ1 subactivityOccurrenceOf occ2. La ocurrencia occ1 es una subocurrencia de la ocurrencia occ2. Inversa de subactivityOccurrences.

- **subactivityOccurrences**. El predicado *subactivityOccurrence*s, que se establece entre dos *ActivityOccurrences*, expresa que la primera de ellas tiene como

subocurrencia a la segunda. Esta relación, que es transitiva, tiene como inversa la relación *subactivityOccurrenceOf*.

- **successor**. El predicado *successor* (*ActivityOccurrence*, *Successor*) que se establece entre dos *ActivityOccurrences*, expresa que la segunda es una ActivityOccurrence de tipo *Successor* de la primera.
- **triggers**. El predicado *triggers* (*Social_Agent*, *Workflow*) expresa la relación entre un *Social_Agent* y el *Workflow* que es lanzado por el mismo. Su inversa es la relación *isTriggeredBy*.
- **typeOf**. Relación entre *ActivityType* y *Activity* que expresa el tipo de actividad al que pertenece una actividad determinada. Inversa de la relación *typedBy*.
- **typedBy**. Relación entre *Activity* y *ActivityType* que expresa que una actividad es de un tipo determinado. Inversa de la relación *typeOf*.
- **hasVersion.** El predicado transitivo *hasVersion* (*Object*, *Object*) expresa las relaciones que existen entre las versiones de un objeto, por ejemplo un recurso. Estas versiones (*versions*) pueden corresponder, por ejemplo, a su situación actual, frente a su estado futuro planificado; o a la capability ideal proporcionada por el fabricante, frente a la capability real conocida tras la verificación de los resultados obtenidos con el mismo. El predicado *hasVersion*, cuya inversa es *version*, permite establecer relaciones entre recursos diferenciados únicamente por los valores (*regions*) de sus características.
- **version**. El predicado transitivo *version* (*Object*, *Object*) expresa la relación que existe entre un objeto, por ejemplo un recurso, que es la versión (*version*) de un segundo objeto, y dicho objeto. Este predicado es la inversa de *hasVersion*.

3.2. Data Properties

La Figura A6.4 presenta, tal y como se muestran en el editor de ontologías Protégé, las relaciones (*Data Properties*) que pueden establecerse entre individuos de la ontología PPDRC y sus datos. A continuación, para cada una de estas propiedades se incluye un breve comentario y/o un ejemplo relativo a su interpretación o utilización en la ontología.

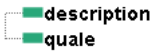


Figura A6.4. *Data Properties* de la ontología PPDRC (PPDRC 2013).

- **description**. Expresa la descripción de un individuo.
- **quale**.Valor cualidad. Expresa el valor que corresponde a una *Region*. Por ejemplo, la *region* ?tab de tipo *TimeInterval* tendrá una valor (*quale*) de 2 segundos, si las *regions* de tipo *TimePoint* ?ta y ?tb tienen los valores (*quales*) de 12 y 14 segundos respectivamente.

4. Reglas de la ontología PPDRC

La Figura A6.5 muestra las reglas establecidas en la ontología PPDRC y escritas en SWRL en el modo en que aparecen en el editor de ontologías Protégé. A continuación, para cada una de estas reglas se incluye un breve comentario y/o un ejemplo relativo a su interpretación o utilización en la ontología.

beginsAt(?sub, ?t) , root_occ(?sub, ?oc) -> beginsAt(?oc, ?t)
participatesAsMechanism(?r, ?oc2) , participatesAsOutput(?r, ?oc1) -> earlier(?oc1, ?oc2)
after(?x, ?y) , before(?x, ?y) -> sameAs(?x, ?y)
beginsAt(?x, ?in) , endsAt(?x, ?fin) -> differentFrom(?in, ?fin)
beginsAt(?y, ?ini) , earlier(?x, ?y) , endsAt(?x, ?fin) , differentFrom(?fin, ?ini) -> before(?fin, ?ini)
subactivities(?x, ?y) , subactivities(?y, ?x) -> sameAs(?x, ?y)
Control(?rol) , hasRole(?r, ?rol) , isPresent(?rol, ?occ) -> participatesAsControl(?r, ?occ)
TimePoint(?x) , TimePoint(?y) , quale(?x, ?u) , quale(?y, ?v) , lessThan(?u, ?v) , differentFrom(?x, ?y) -> before(?x, ?y)
behaves(?r, ?cap) , isPresent(?r, ?occ) , occurrenceOf(?occ, ?a) , typedBy(?a, ?at) -> executing(?cap, ?at)
Resource(?x) , behaves(?y, ?z) , hasRole(?x, ?y) -> characterizedBy(?x, ?z)
participatesAsInput(?r, ?oc2) , participatesAsOutput(?r, ?oc1) -> earlier(?oc1, ?oc2)
executing(?cap, ?at) , occurrenceOf(?occ, ?a) , typedBy(?a, ?at) -> requiresCapability(?occ, ?cap)
endsAt(?sub, ?t) , leaf_occ(?sub, ?oc) -> endsAt(?oc, ?t)
beginsAt(?x, ?in) , endsAt(?x, ?fin) -> before(?in, ?fin)
Input(?rol) , hasRole(?r, ?rol) , isPresent(?rol, ?occ) -> participatesAsInput(?r, ?occ)
Output(?rol) , hasRole(?r, ?rol) , isPresent(?rol, ?occ) -> participatesAsOutput(?r, ?occ)
earlier(?x, ?y) -> differentFrom(?x, ?y)
Mechanism(?rol) , hasRole(?r, ?rol) , isPresent(?rol, ?occ) -> participatesAsMechanism(?r, ?occ)

Figura A6.5. Reglas de la ontología PPDRC (PPDRC 2013).

- **after(?x, ?y) , before(?x, ?y) -> sameAs(?x, ?y)**

  Si un TimePoint ?x es posterior a otro TimePoint ?y, y el TimePoint ?x es anterior al TimePoint ?y entonces, los TimePoints ?x e ?y son iguales.

- **beginsAt(?sub, ?t) , root_occ(?sub, ?oc) -> beginsAt(?oc, ?t)**

  Si una ActivityOccurrence ?sub se inicia en un TimePoint ?t, y ?sub es la primera subactivityOccurrence de ?oc entonces, ?oc se inicia en el TimePoint ?t.

- **beginsAt(?x, ?in) , endsAt(?x, ?fin) -> before(?in, ?fin)**

  Si una ActivityOccurrence o un Object ?x se inicia en un TimePoint ?in, y si esa ActivityOccurrence u Object ?x finaliza en un TimePoint ?fin entonces, el TimePoint ?in es anterior al TimePoint ?fin.

- **beginsAt(?x, ?in) , endsAt(?x, ?fin) -> differentFrom(?in, ?fin)**

  Si una ActivityOccurrence o un Object ?x se inicia en un TimePoint ?in, y si esa ActivityOccurrence u Object ?x finaliza en un TimePoint ?fin entonces, el TimePoint ?in es distinto del TimePoint ?fin.

- **beginsAt(?y, ?ini) , earlier(?x, ?y) , endsAt(?x, ?fin) , differentFrom(?fin, ?ini) -> before(?fin, ?ini)**

  Si una ActivityOccurrence Arboreal ?y se inicia en un TimePoint ?ini, una ActivityOccurrence Arboreal ?x finaliza en un TimePoint ?fin, ?x es anterior a ?y, y los TimePoints ?fin e ?ini son distintos entonces, el TimePoint ?fin es anterior al TimePoint ?ini.

- **behaves(?r, ?cap) , isPresent(?r, ?occ) , occurrenceOf(?occ, ?a) , typedBy(?a, ?at) -> executing(?cap, ?at)**

  Si un rol ?r exhibe una capability ?cap y está presente en una ActivityOccurrence ?occ que es una realización (ocurrencia) de una actividad ?a de tipo ?at entonces, la capability ?cap corresponde a la ejecución de una actividad de tipo?at.

- **Control(?rol) , hasRole(?r, ?rol) , isPresent(?rol, ?occ) -> participatesAsControl(?r, ?occ)**

  Si un recurso ?r tiene un rol ?rol que es de tipo Control y está presente en una ActivityOccurrence ?occ entonces, el recurso ?r participa como control en la ActivityOccurrence ?occ.

- **earlier(?x, ?y) -> differentFrom(?x, ?y)**

  Si una ActivityOccurrence Arboreal ?x es anterior a una ActivityOccurrence Arboreal ?y entonces, las ActivityOccurrences ?x e ?y son distintas.

- **endsAt(?sub, ?t) , leaf_occ(?sub, ?oc) -> endsAt(?oc, ?t)**

  Si una ActivityOccurrence ?sub finaliza en un TimePoint ?t, y ?sub es la última subactivityOccurrence de ?oc entonces, ?oc finaliza en el TimePoint ?t.

- **executing(?cap, ?at) , occurrenceOf(?occ, ?a) , typedBy(?a, ?at) -> requiresCapability(?occ, ?cap)**

  Si ?occ es la realización (ocurrencia) de una actividad ?a de tipo ?at, y la capability ?cap corresponde a la ejecución de una actividad de tipo ?at entonces, para la realización de la ActivityOccurrence ?occ se requiere la capability ?cap.

- **Input(?rol) , hasRole(?r, ?rol) , isPresent(?rol, ?occ) -> participatesAsInput(?r, ?occ)**

  Si un recurso ?r tiene un rol ?rol que es de tipo Input y está presente en una ActivityOccurrence ?occ entonces, el recurso ?r participa como entrada en la ActivityOccurrence ?occ.

- **Mechanism(?rol) , hasRole(?r, ?rol) , isPresent(?rol, ?occ) -> participatesAsMechanism(?r, ?occ)**

  Si un recurso ?r tiene un rol ?rol que es de tipo Mechanism y está presente en una ActivityOccurrence ?occ entonces, el recurso ?r participa como mecanismo en la ActivityOccurrence ?occ.

- **Output(?rol) , hasRole(?r, ?rol) , isPresent(?rol, ?occ) -> participatesAsOutput(?r, ?occ)**

  Si un recurso ?r tiene un rol ?rol que es de tipo Output y está presente en una ActivityOccurrence ?occ entonces, el recurso ?r participa como salida en la ActivityOccurrence ?occ.

- **participatesAsInput(?r, ?oc2) , participatesAsOutput(?r, ?oc1) -> earlier(?oc1, ?oc2)**

  Si un recurso ?r participa con rol de tipo Input en la ActivityOccurrence ?oc2 y participa con rol de tipo Output en la ActivityOccurrence ?oc1, siendo que ?oc1 y ?oc2 son realizaciones de actividades primitivas entonces, la ActivityOccurrence ?oc1 es anterior a la ActivityOccurrence ?oc2.

- **participatesAsMechanism(?r, ?oc2) , participatesAsOutput(?r, ?oc1) -> earlier(?oc1, ?oc2)**

  Si un recurso ?r participa con rol de tipo Mechanism en la ActivityOccurrence ?oc2 y participa con rol de tipo Output en la ActivityOccurrence ?oc1, siendo que ?oc1 y ?oc2 son realizaciones de actividades primitivas entonces, la ActivityOccurrence ?oc1 es anterior a la ActivityOccurrence ?oc2.

- **Resource(?r) , behaves(?y, ?z) , hasRole(?x, ?y) -> characterizedBy(?x, ?z)**

  Si un recurso ?r tiene un rol ?y que exhibe una capability ?z entonces, el recurso ?r está caracterizado por la capability ?z.

- **subactivities(?x, ?y) , subactivities(?y, ?x) -> sameAs(?x, ?y)**

  Si ?y es una subActivityOccurrence de ?x y ?x es una subActivityOccurrence de ?y entonces, las ActivityOccurrences ?x e ?y son iguales.

- **TimePoint(?x) , TimePoint(?y) , quale(?x, ?u) , quale(?y, ?v) , lessThan(?u, ?v) , differentFrom(?x, ?y) -> before(?x, ?y)**

  Si ?u, el valor del TimePoint ?x, es menor que ?v, el valor del TimePoint ?y, y los TimePoints ?x e ?y son distintos entonces, ?x es anterior a ?y.

5. Individuos de la ontología PPDRC

El individuo *ManageWorkflow* forma parte de la *intension* de la ontología, ya que interviene en la definición de *Social_Agent*. Una entidad caracterizada por la *Capability ManageWorkflow*, que le permite lanzar y gestionar *ConversationalWorkflows.*

Esto se expresa en la ontología mediante el axioma *characterizedBy value ManageWorflow*, que sirve para definir la clase *Social_Agent*.

6. Referencias

PPDRC ontology. 2013. Accessed June http://www.coapp.es/ontologies/2013/0/PPDRC_v1.owl.

Anexo 7. La ontología Manufacturing and Inspection Resource Capabilities (MIRC)

1. Introducción

En este anexo se enumeran y describen las clases, relaciones y reglas que forman parte de la *intension* de la ontología MIRC. La información contenida en este anexo se ha extraído fundamentalmente de la última versión de la ontología MIRC editada en Protégé (MIRC 2014) y que puede ser consultada a través del enlace http://coapp.webs.upv.es/ontologies/2014/0/MIRC_v1.owl.txt.

2. Clases de la ontología MIRC

La ontología MIRC especializa las clases definidas por la ontología PPDRC, y por ello las clases definidas por MIRC incluyen todas las clases de PPDRC. En las figuras A7.1 a A7.7 se muestran todas las clases de la ontología MIRC, apareciendo en letra normal las clases importadas directamente sin cambios de PPDRC y en letra negrita las clases propias de MIRC y aquéllas sobre las que se han añadido nuevos axiomas. A continuación, y exclusivamene para estas últimas, se incluye un breve comentario y/o un ejemplo relativo a su interpretación o utilización en la ontología. La explicación y comentarios referidos a las clases de la ontología PPDRC pueden consultarse en el anexo 6.

- **Operation**. Una *Operation* es una agrupación concurrente de actividades de las clases *Tool-Part_Interaction* y *Tool_Movement*, donde *Tool-Part_Interaction* representa la forma de interacción entre herramienta y pieza, para arrancar material o para inspeccionar la pieza, y *Tool_Movement* representa el movimiento relativo entre la herramienta o sonda de medición y la pieza.
- **Part_Process**. Los individuos de la clase *Part_Process* están formados por individuos de la clase *Phase* e individuos de la clase *Transport-Storage*.
- **Phase**. Los individuos de la clase *Phase* están formados por individuos de la clase *Loading* e individuos de la clase *Operation*.

- **Preparation**. Las *Preparations* son tipos de actividades que determinan las *Capabilities* de los recursos de mecanizado e inspección como una única entidad, a través de acciones que implican la formación de un *Resource_Group* y/o la modificación y caracterización de éste.

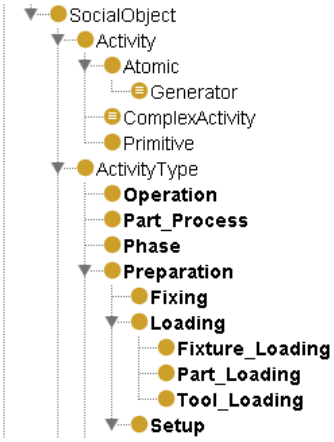


Figura A7.1. Taxonomía de clases de la ontología MIRC (MIRC 2014).

Las clases *Operation*, *Tool-Part_Interaction*, *Part_Process*, *Preparation*, *Tool_Movements*, *Phase* y *Transport_Storage* son disjuntas entre sí.

- **Fixing**. Es un tipo de *Preparation* distinto de *Loading* y *Setup* que representa una operación de fijación o embridaje. *Fixing*, *Loading* y *Setup* son clases disjuntas entre sí.
- **Loading**. Es una especialización de la entidad *Preparation*. El resultado de la realización de una actividad de tipo Loading es un objeto diferente de los objetos que participan con rol de entrada en dicha realización. No obstante, el objeto de salida mantendrá algunas características de los objetos de entrada y adquirirá otras características nuevas, propias del conjunto. Las tres especializaciones de *Loading* que se describen a continuación corresponden a clases disjuntas.
- **Fixture_Loading**. Especialización de *Loading*. En la realización de una actividad perteneciente al tipo *Fixture_Loading* se obtiene un recurso de tipo *Fixture* montado en la máquina.
- **Part_Loading**. Especialización de *Loading*. En la realización de una actividad perteneciente al tipo *Part_Loading* se obtiene un recurso que incluye una pieza montada en el utillaje o en la máquina.
- **Tool_Loading**. Especialización de *Loading*. En la realización de una actividad perteneciente al tipo *Tool_Loading* se obtiene un recurso que incluye una herramienta montada en la máquina.

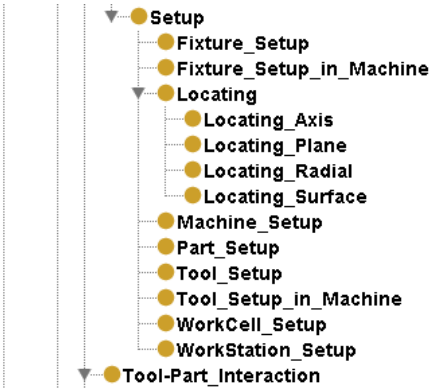


Figura A7.2. Taxonomía de clases de la ontología MIRC (MIRC 2014).

- **Setup**. Las actividades de tipo *Setup*, que incluyen una medida y/o corrección, cuantifican las capabilities de un *Resource_Group* directamente, como si de una entidad única se tratara. Las especializaciones de *Setup* que se describen a continucación corresponden a clases disjuntas.
- **Fixture_Setup**. Especialización de *Setup*. En la realización de una actividad perteneciente al tipo *Fixture_Setup* se realiza el *setup* de un utillaje para la sujeción de la pieza.
- **Fixture_Setup_In_Machine**. Especialización de *Setup*. En la realización de una actividad perteneciente al tipo *Fixture_Setup_In_Machine* se realiza el *setup* de un utillaje que está montado en la máquina.
- **Locating**. Especialización de *Setup*. En la realización de una actividad perteneciente al tipo *Locating* se realiza la localización de la pieza restringiendo alguno de sus grados de libertad. Las especializaciones de *Locating* son disjuntas.
- **Locating_Axis**. Especialización de *Locating* en la que se localiza un eje de la pieza, normalmente materializado a partir de dos puntos.
- **Locating_Plane**. Especialización de *Locating* en la que se facilita la localización de un plano, por ejemplo a partir de tres puntos de la pieza.
- **Locating_Radial**. Especialización de *Locating* en la que se localiza un punto de la pieza.
- **Locating_Surface**. Especialización de *Locating* en la que se posiciona una superficie de la pieza completamente irregular en la que no es posible identificar ejes, planos u otros elementos con los que definir éstos.
- **Machine_Setup**. Especialización de *Setup*. En la realización de una actividad perteneciente al tipo *Machine_Setup* se realiza el *setup* de la máquina.
- **Part_Setup**. Especialización de *Setup*. En la realización de una actividad perteneciente al tipo *Part_Setup* se realiza el *setup* de la pieza.

- **Tool_Setup**. Especialización de *Setup*. En la realización de una actividad perteneciente al tipo *Tool_Setup* se realiza el *setup* de la herramienta.
- **Tool_Setup_In_Machine**. Especialización de *Setup*. En la realización de una actividad perteneciente al tipo *Tool_Setup_In_Machine* se realiza el *setup* de una herramienta previamente montada en la máquina.
- **WorkCell_Setup**. Especialización de *Setup*. En la realización de una actividad perteneciente al tipo *WorkCell_Setup* se realiza el *setup* de una *WorkCell*.
- **WorkStation_Setup**. Especialización de *Setup*. En la realización de una actividad perteneciente al tipo *WorkStation_Setup* se realiza el *setup* de una *WorkStation*.

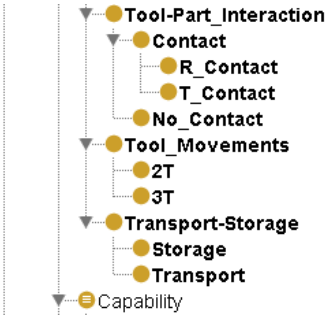


Figura A7.3. Taxonomía de clases de la ontología MIRC (MIRC 2014).

- **Tool-Part_Interaction**. *Tool-Part_Interaction* es un tipo de actividad que representa la forma de interacción entre herramienta y pieza, para arrancar material o para inspeccionar la pieza.
- **Contact**. Especialización de *Tool-Part_Interaction* en la que la interacción entre herramienta y pieza se realiza por contacto entre ambas. *Contact* es distinto de *No_Contact*.
- **R_Contact**. Especialización de *Contact* en la que la herramienta tiene un movimiento de rotación.
- **T_Contact**. Especialización de *Contact* en la que la herramienta tiene un movimiento de traslación.
- **No_Contact**. Especialización d**e** *Tool-Part_Interaction* en la que la interacción entre herramienta y pieza se realiza sin contacto entre ambas. *No_Contact* es distinto de *Contact***.**
- **Tool_Movements**. *Tool_Movement* es la combinación de movimientos de traslación y de rotación que definen las estrategias de las operaciones de mecanizado e inspección.
- **2T**. Especialización de *Tool_Movements* que corresponde a un movimiento de traslación contenido en un plano.

- **3T**. Especialización de *Tool_Movements* que corresponde a un movimiento de traslación en tres dimensiones.
- **Transport-Storage**. *ActivityType* que agrupa las actividades de tipo *Storage* y *Transport*.
- **Storage**. Especialización de *Transport-Storage* que representa actividades de almacenamiento. *Storage* es distinto de *Transport*.
- **Transport**. Especialización de *Transport-Storage* que representa actividades de transporte. *Transport* es distinto de *Storage*.

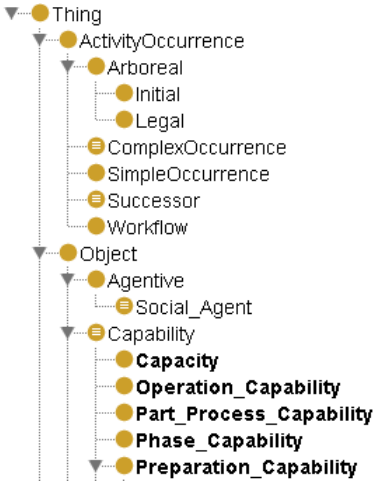


Figura A7.4. Taxonomía de clases de la ontología MIRC (MIRC 2014).

Las entidades que se describen a continuación: *Operation_Capability*, *Transport-Storage_Capability*, *Preparation_Capability*, *Capacity*, *Phase_Capability*, *Tool-Part_Interaction_Capability*, *Tool_Movements_Capability* y *Part_Process_Capability* son disjuntas entre sí.

- **Operation_Capability**. *Capability* para realizar actividades de tipo *Operation.* Cada individuo de la clase *Operation_Capability* está formado por un individuo de la clase *Tool-Part_Interaction_Capability* y por un individuo de la clase *Tool_Movements_Capability*.
- **Part_Process_Capability**. *Capability* para realizar actividades de tipo *Part_Process*. Cada individuo de la clase *Part_Process_Capability* está formado por un individuo de la clase *Operation_Capability* y por un individuo de la clase *Transport-Storage_Capability*.
- **Phase_Capability**. *Capability* para realizar actividades de tipo *Phase*.
- **Preparation_Capability**. *Capability* para realizar actividades de tipo *Preparation.*

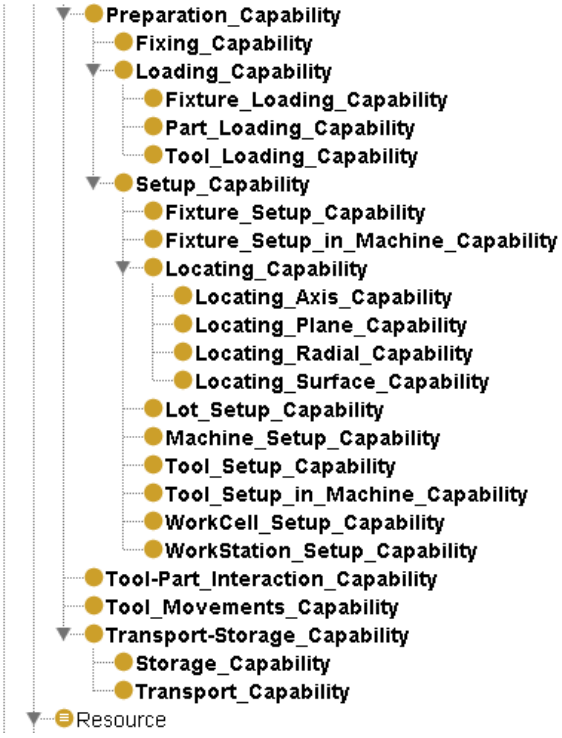


Figura A7.5. Taxonomía de clases de la ontología MIRC (MIRC 2014).

Las clases correspondientes a *Fixing_Capability*, *Loading_Capability* y *Setup_Capability* son disjuntas.

- **Fixing_Capability**. Especialización de *Preparation_Capability*. *Capability* para realizar actividades de tipo *Fixing*.
- **Loading_Capability**. Especialización de *Preparation_Capability*. *Capability* para realizar actividades de tipo *Loading*. Esta clases se especializa en las siguientes clases disjuntas:
- **Fixture_Loading_Capability**. Especialización de *Loading_Capability*. *Capability* para realizar actividades de tipo *Fixture_Loading*.
- **Part_Loading_Capability**. Especialización de *Loading_Capability*. *Capability* para realizar actividades de tipo *Part_Loading*.
- **Tool_Loading_Capability**. Especialización de *Loading_Capability*. *Capability* para realizar actividades de tipo *Tool_Loading*.
- **Setup_Capability**. Especialización de *Preparation_Capability*. *Capability* para realizar actividades de tipo *Setup*, que tiene varias especializaciones, todas ellas disjuntas, que se presentan a continuación:

- **Fixture_Setup_Capability**. Especialización de *Setup_Capability*. *Capability* para realizar actividades de tipo *Fixture_Setup*.
- **Fixture_Setup_In_Machine_Capability**. Especialización de *Setup_Capability*. *Capability* para realizar actividades de tipo *Fixture_Setup_In_Machine*.
- **Locating_Capability**. Especialización de *Setup_Capability*. *Capability* para realizar actividades de tipo *Locating*. Sus cuatro especializaciones, *Locating_Plane_Capability*, *Locating_Axis_Capability*, *Locating_Radial_Capability* y *Locating_Surface_Capability* son disjuntas.
- **Locating_Axis_Capability**. Especialización de *Locating_Capability*. *Capability* para realizar actividades de tipo *Locating_Axis*.
- **Locating_Plane_Capability**. Especialización de *Locating_Capability*. *Capability* para realizar actividades de tipo *Locating_Plane*.
- **Locating_Radial_Capability**. Especialización de *Locating_Capability*. *Capability* para realizar actividades de tipo *Locating_Radial*.
- **Locating_Surface_Capability**. Especialización de *Locating_Capability*. *Capability* para realizar actividades de tipo *Locating_Surface*.
- **Lot_Setup_Capability**. Especialización de *Setup_Capability*. *Capability* para realizar actividades de tipo *Lot_Setup*.
- **Machine_Setup_Capability**. Especialización de *Setup_Capability*. *Capability* para realizar actividades de tipo *Machine_Setup*.
- **Tool_Setup_Capability**. Especialización de *Setup_Capability*. *Capability* para realizar actividades de tipo *Tool_Setup*.
- **Tool_Setup_In_Machine_Capability**. Especialización de *Setup_Capability*. *Capability* para realizar actividades de tipo *Tool_Setup_In_Machine*.
- **WorkCell_Setup_Capability**. Especialización de *Setup_Capability*. *Capability* para realizar actividades de tipo *WorkCell_Setup*.
- **WorkStation_Setup_Capability**. Especialización de *Setup_Capability*. *Capability* para realizar actividades de tipo *WorkStation_Setup*.
- **Tool-Part_Interaction_Capability**. *Capability* para realizar actividades de tipo *Tool-Part_Interaction*.
- **Tool_Movements_Capability**. *Capability* para realizar actividades de tipo *Tool_Movements*.
- **Transport-Storage_Capability**. *Capability* para realizar actividades de tipo *Transport-Storage*.
- **Storage_Capability**. Especialización de *Transport-Storage_Capability*. *Capability* para realizar actividades de tipo *Storage*. *Storage_Capability* es distinto de *Transport_Capability*.

- **Transport_Capability**. Especialización de *Transport-Storage_Capability*. *Capability* para realizar actividades de tipo *Transport*. *Transport_Capability* es distinto de *Storage_Capability*.

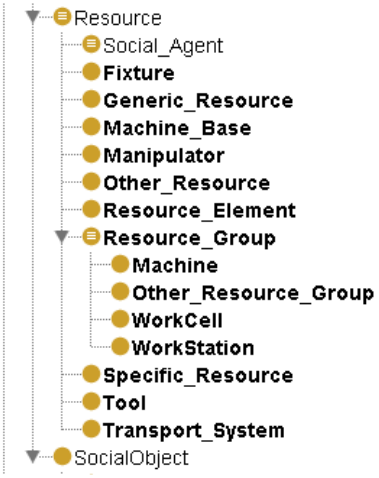


Figura A7.6. Taxonomía de clases de la ontología MIRC (MIRC 2014).

Los Resources descritos a continuación: *Fixture*, *Machine_Base*, *Manipulator*, *Other_Resource*, *Tool* y *Transport-System* corresponden a clases disjuntas.

- **Fixture**. Utillaje para la sujeción de piezas. Los individuos de la clase *Fixture* están caracterizados por capabilities para realizar actividades de tipo localización/fijación de piezas.
- **Machine_Base**. Los individuos de la clase *Machine_Base* están caracterizados por capabilities para realizar actividades de tipo *Tool_Movement.*
- **Manipulator**. Manipulador. Los individuos de la clase *Manipulator* están caracterizada por capabilities para realizar actividades de manipulación de piezas, herramientas y utillajes.
- **Other_Resource**. *Resource* distinto de las entidades *Fixture*, *Machine_Base*, *Transport_System*, *Manipulator* y *Tool*.
- **Tool.** Herramienta/sonda de medición. Los individuos de la clase *Tool* están caracterizados por capabilities para realizar actividades de tipo *Tool-Part_Interaction*.
- **Transport_System**. Sistema de transporte. Los individuos de la clase *Transport_System* están caracterizados por *Transport_Capabilities*.
- **Resource_Element**. Recurso elemental, que no está integrado por otros recursos. *Resource_Element* es distinto de *Resource_Group*.
- **Resource_Group**. Recurso complejo, que está integrado por otros recursos elementales (*Resource_Elements*) o complejos (*Resource_Groups*).

Resource_Group es distinto de *Resource_Element*. Los *Resource_Group* descritos a continuación: *Machine*, *Other_Resource_Group*, *WorkCell* y *WorkStation* son disjuntos.

- **Machine**. *Resource_Group* que participa con rol de mecanismo en la realización de actividades de tipo *Operation.* Un individuo de la clase *Machine* está integrado por un individuo del tipo *Machine_Base* y uno o varios individuos de las clases *Tool* y *Fixture.*
- **WorkCell**. Recurso complejo con un nivel de agregación superior a los de *tipo Machine.* Los individuos de la clase *WorkCell* están integrados por individuos de tipo *WorkStation y* están *caract*erizados por *Capabilities* par*a realiz*ar actividades de tipo *Part_Process*.
- **WorkStation**. Recurso complejo con un nivel de agregación superior a los de tipo *Machine.* Los individuos de la clase *WorkStation* están integrados por individuos de tipo *Machine* y están caracterizados por *Capabilities* para realizar actividades de tipo *Phase*.
- **Other_Resource_Group**. Recurso complejo distinto de los recurso de tipo *Machine*, *WorkCell* y *WorkStation*.
- **Generic_Resource**. Un *Generic_Resource* debe entenderse como un recurso abstracto cuya participación en la realización de una actividad concreta se traduce en la participación en dicha actividad de uno de los *Specific_Resources* incluidos en este *Generic_Resource*. Un *Generic_Resource* es distinto de un *Specific_Resource.*
- **Specific_Resource**. Un *Specific_Resource* debe entenderse como un recurso concreto que puede estar incluido en *Generic_Resource*, pero que es distinto de éste.

Las regions *Limit_Value*, *TimeInterval*, *TimePoint*, *Nominal_Value* y *Variability* corresponden a clases disjuntas.

- **Limit_Value**. Las *Regions* pertenecientes al tipo *Limit_Value* se caracterizan por la forma de componer sus valores, que se realiza tomando el valor máximo o el mínimo de todas las *regions* consideradas.
- **Clamp_Dimension**. *Region* de tipo *Limit_Value* que especifica las dimensiones máximas o mínimas de la pieza que puede fijarse en un *fixture*.
- **Interaction_Dimension**. *Region* de tipo *Limit_Value* que representa la dimensión máxima en la que puede interaccionar una herramienta.
- **MRR**. *Region* de tipo *Limit_Value* que especifica la tasa se arranque de material o *Material_Removal_Rate*.
- **Pass_Dimension**. *Region* de tipo *Limit_Value* que especifica el tamaño de la sección de paso en utillajes como platos de garras o alimentadores de barra.
- **Roughness**. *Region* de tipo *Limit_Value* que especifica la rugosidad superficial.

- **Tool_Pass_Dimension**. *Region* de tipo *Limit_Value* que especifica el diámetro mínimo requerido por herramientas de mecanizado de interiores o por instrumentos de medición para acceder a la zona de mecanizado o inspección respectivamente.

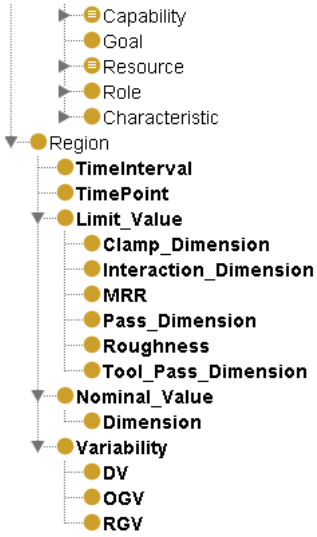


Figura A7.7. Taxonomía de clases de la ontología MIRC (MIRC 2014).

- **Nominal_Value**. Las *Regions* pertenecientes al tipo *Nominal_Value* se caraterizan por la forma de componer sus valores, que se realiza a partir de una expresión matemática simple, como puede ser la suma algebraica.
- **Dimension**. *Region* de tipo *Nominal_Value* que representa la dimensión medida en la dirección del movimiento de avance asociado a la capability.
- **Variability**. Las *regions* pertenecientes al tipo *Variability* se caraterizan por la forma de componer sus valores, que se realiza aplicando reglas de composición cuadrática o similares.
- **DV**. *Dimensional_Variability* (*DV*) es una especialización de *Variability*. Las *regions* de tipo *DV* expresan variabilidades en longitudes y ángulos.
- **OGV**. *Own_Geometric_Variability* (*OGV*) es una especialización de *Variability*. Las *regions* de tipo *OGV* representan variabilidades geométricas intrínsecas, por ejemplo las asociadas a características de planitud o de redondez.
- **RGV**. *Reference_Geometric_Variability* (*RGV*) es una especialización de *Variability*. Las *regions* de tipo *RGV* representan variabilidades de tipo geométrico (orientación y posición) entre elementos de entidades diferentes, por ejemplo, como son las asociadas a características de paralelismo o de perpendicularidad.

3. Relaciones de la ontología MIRC

3.1. Object Properties

La ontología MIRC especializa la relaciones (*Object Properties*) definidas por la ontología PPDRC, y por ello las relaciones definidas por MIRC incluyen todas las de PPDRC. En las figuras A7.8 a A7.10 se muestran todas las relaciones de la ontología MIRC, apareciendo en letra normal las relaciones importadas directamente sin cambios de PPDRC y en letra negrita las relaciones propias de MIRC y aquéllas sobre las que se han añadido nuevos axiomas. A continuación, y exclusivamene para estas últimas, se incluye un breve comentario y/o un ejemplo relativo a su interpretación o utilización en la ontología. La explicación y comentarios referidos a las relaciones de la ontología PPDRC pueden consultarse en el anexo 6.

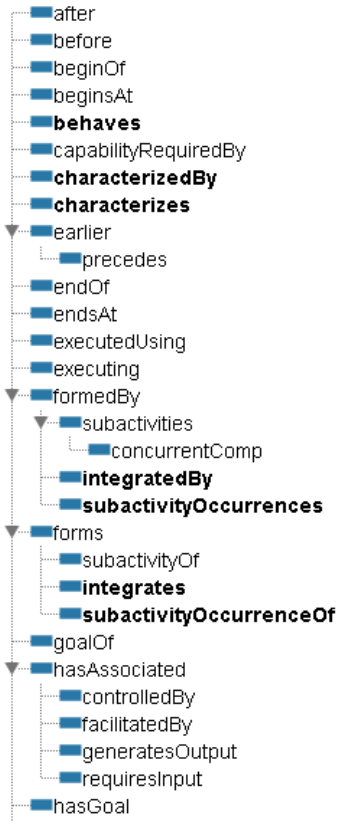


Figura A7.8. *Object Properties* de la ontología MIRC (MIRC 2014).

- **integratedBy**. El predicado *integratedBy* (*Resource*, *Resource*), que es una especialización de *formedBy*, expresa la agrupación de los recursos que configuran un *Resource_Group*. *integratedBy* es una relación transitiva cuya inversa es *integrates*.

- **integrates**. El predicado *integrates* (*Resource*, *Resource*), que es una especialización de *forms* expresa la relación que existe entre un recurso elemental (*Resource_Element*) o complejo (*Resource_Group*) y el recurso del que forma parte. *integrates* es una relación transitiva cuya inversa es *integratedBy*.

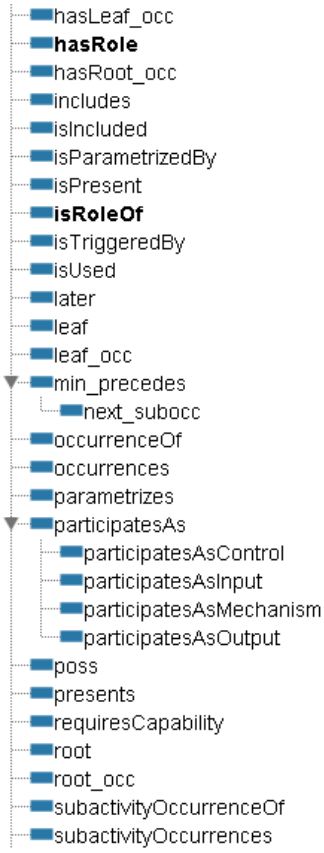


Figura A7.9. *Object Properties* de la ontología MIRC (MIRC 2014).

- **isRequiredBy**. El predicado *isRequiredBy* (*Region*, *Capability*), cuya inversa es *requires*, expresa la relación existente entre los requisitos exigidos a las características del objeto sobre el que se realiza la actividad y el nivel de prestación de la *Capability* del recurso ejecutando actividades de ese tipo.

- **relatedTo**. El predicado *relatedTo* (*Object*, *Object*) establece la relación que existe entre recursos físicos que están conectados a través de interfaces. Por tanto, este predicado facilita más información acerca de los recursos que forman un recurso complejo que la suministrada por el predicado integratedBy, que solamente muestra la agrupación de varios recursos.

- **relatedToPart**. El predicado *relatedToPart* es una especialización de *relatedTo* que expresa el sentido de conexión (hacia la interfaz de la pieza) entre los recursos que forman un *Resource_Group*. Su inversa es el predicado *relatedToTool*.
- **relatedToTool**. El predicado *relatedToTool* es una especialización de *relatedTo* que expresa el sentido de conexión (hacia la interfaz de la herramienta) entre los recursos que forman un *Resource_Group*. Su inversa es el predicado *relatedToPart*.
- **requires**. El predicado *requires* (*Capability*, *Region*) expresa los requisitos exigidos a las características del objeto sobre el que se realiza la actividad, en relación al nivel de prestación de la *capability* del recurso ejecutando actividades de ese tipo. En este sentido, el predicado *requires* implica la existencia de restricciones sobre el objeto que recibe la actividad para que la *Capability* del recurso pueda considerarse válida. Por ejemplo, si la *capability* de un recurso mecanizando una superficie tiene una relación *parametrizes* con la region Ra = 2 µm y una relacion *requires* con la region Ra = 10 µm, la prestación alcanzada por el recurso en el mecanizado de superficies será de 2 µm de rugosidad media, siempre que la superficie del objeto sobre el que se realiza la actividad tenga una rugosidad media menor o igual a 10 µm. La inversa de *requires* es el predicado *isRequiredBy*.

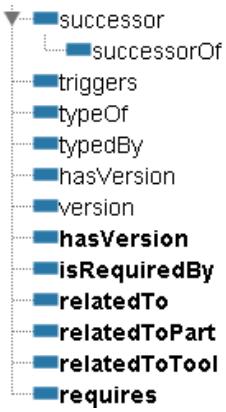


Figura A7.10. *Object Properties* de la ontología MIRC (MIRC 2014).

3.2. Data Properties

En la Figura A7.11 se muestran las relaciones (*Data Properties*) que pueden establecerse entre individuos de la ontología MIRC y sus datos. La descripción de estas propiedades, que se importan de la ontología PPDRC, se encuentra en el anexo 6.

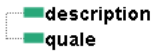


Figura A7.11. *Data Properties* de la ontología MIRC (MIRC 2014).

4. Reglas de la ontología PPDRC

La ontología MIRC especializa las reglas definidas por la ontología PPDRC, y por ello las reglas de la ontología MIRC incluyen todas las de PPDRC. En la Figura A7.12 se muestran todas las relaciones de la ontología MIRC, apareciendo en letra normal las relaciones importadas directamente de PPDRC y en letra negrita las relaciones propias de MIRC. A continuación, y exclusivamene para estas últimas, se incluye un breve comentario y/o un ejemplo relativo a su interpretación o utilización en la ontología. La explicación y comentarios referidos a las reglas de la ontología PPDRC pueden consultarse en el anexo 6.

Resource(?R) , Resource(?r) , characterizes(?Cap, ?r) , integrates(?r, ?R) -> characterizes(?Cap, ?R)
Activity(?Act) , ActivityOccurrence(?Occ) , Resource(?I) , Resource(?O) , Loading(?ActType) , occurrenceOf(?Occ, ?Act) , participatesAsInput(?I, ?Occ) , participatesAsOutput(?O, ?Occ) , typedBy(?Act, ?ActType) -> integrates(?I, ?O)
beginsAt(?sub, ?t) , root_occ(?sub, ?oc) -> beginsAt(?oc, ?t)
participatesAsMechanism(?r, ?oc2) , participatesAsOutput(?r, ?oc1) -> earlier(?oc1, ?oc2)
after(?x, ?y) , before(?x, ?y) -> sameAs(?x, ?y)
beginsAt(?x, ?in) , endsAt(?x, ?fin) -> differentFrom(?in, ?fin)
beginsAt(?y, ?ini) , earlier(?x, ?y) , endsAt(?x, ?fin) , differentFrom(?fin, ?ini) -> before(?fin, ?ini)
subactivities(?x, ?y) , subactivities(?y, ?x) -> sameAs(?x, ?y)
Control(?rol) , hasRole(?r, ?rol) , isPresent(?rol, ?occ) -> participatesAsControl(?r, ?occ)
behaves(?r, ?cap) , isPresent(?r, ?occ) , occurrenceOf(?occ, ?a) , typedBy(?a, ?at) -> executing(?cap, ?at)
Resource(?x) , behaves(?y, ?z) , hasRole(?x, ?y) -> characterizedBy(?x, ?z)
participatesAsInput(?r, ?oc2) , participatesAsOutput(?r, ?oc1) -> earlier(?oc1, ?oc2)
executing(?cap, ?at) , occurrenceOf(?occ, ?a) , typedBy(?a, ?at) -> requiresCapability(?occ, ?cap)
endsAt(?sub, ?t) , leaf_occ(?sub, ?oc) -> endsAt(?oc, ?t)
beginsAt(?x, ?in) , endsAt(?x, ?fin) -> before(?in, ?fin)
Input(?rol) , hasRole(?r, ?rol) , isPresent(?rol, ?occ) -> participatesAsInput(?r, ?occ)
TimePoint(?x) , TimePoint(?y) , quale(?x, ?u) , quale(?y, ?v) , lessThan(?u, ?v) , differentFrom(?x, ?y) -> before(?x, ?y)
Output(?rol) , hasRole(?r, ?rol) , isPresent(?rol, ?occ) -> participatesAsOutput(?r, ?occ)
earlier(?x, ?y) -> differentFrom(?x, ?y)
Mechanism(?rol) , hasRole(?r, ?rol) , isPresent(?rol, ?occ) -> participatesAsMechanism(?r, ?occ)

Figura A7.12. Reglas de la ontología MIRC (MIRC 2014).

- **Resource(?R), Resource(?r), characterizes(?Cap, ?r), integrates(?r, ?R) -> characterizes(?Cap, ?R)**

  Si un recurso ?r que está caracterizado por una capability ?Cap forma parte de otro recurso ?R, entonces el recurso ?R está caracterizado por la capability ?Cap.

- **Activity(?Act), ActivityOccurrence(?Occ), Resource(?I), Resource(?O), Loading(?ActType), occurrenceOf(?Occ, ?Act), participatesAsInput(?I, ?Occ), participatesAsOutput(?O, ?Occ), typedBy(?Act, ?ActType) -> integrates(?I, ?O)**

  Si los recursos ?I y ?O participan respectivamente como entrada y salida en la *ActivityOccurrence* ?Occ de una actividad de tipo *Loading*, entonces el recurso ?I forma parte del recurso ?O.

5. Referencias

MIRC ontology. 2014. Accessed February. http://coapp.webs.upv.es/ontologies/2014/0/MIRC_v1.owl.txt