UNIVERSIDAD POLITÉCNICA DE VALENCIA

Depto. Sistemas Informáticos y Computación

TESIS DE MÁSTER

Sistemas Multiagente Normativos

Autor: Natalia Criado Pacheco

Dirigida por: Dra. Estefanía Argente Villaplana

Grupo de Tecnología Informática - Inteligencia Artificial
Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
Camino de Vera, s/n
46020 Valencia, Spain
Junio, 2009

Índice general

Introducción

El paradigma de los sistemas multiagente (SMA) es una de las áreas que está recibiendo un mayor interés en los últimos años. Los agentes son entidades computacionales autónomas capaces de actuar dentro de entornos dinámicos y flexibles. Sin embargo, éstos no trabajan de forma aislada, sino que lo hacen formando parte de sociedades. En dichas sociedades no todos los agentes comparten unos mismos intereses. Para evitar la aparición de conflictos y asegurar la coordinación y cooperación, se introduce el concepto de *norma* dentro de los SMA.

Las normas han sido estudiadas desde distintas disciplinas como la filosofía, la psicología, el derecho, etc. Por ello resulta complicado ofrecer una definición del concepto de norma que cubra las diferentes aproximaciones.

Estos conceptos normativos también han sido estudiados dentro de la informática, especialmente dentro del ámbito de la Teoría de Agentes, como un mecanismo para asegurar la coordinación en los SMA Abiertos. En los últimos años los SMA tratan de cubrir grandes sistemas distribuidos, donde la información y las tareas están distribuidas entre varios agentes, para modelar sistemas complejos y dinámicos. Fruto de esta necesidad ha surgido un creciente interés sobre cómo incorporar a los agentes y los SMA conceptos extraídos de la teoría normativa.

Las normas definen lo que está permitido y prohibido en una sociedad. Especifican los beneficios y responsabilidades de sus miembros y, como consecuencia, los agentes son capaces de planificar sus acciones de acuerdo con el comportamiento esperado del resto de agentes. En general, todo tipo de actividad que requiere la coordinación y cooperación entre más de un agente es posible gracias a la introducción de normas que regulen dicha interacción.

Un *Sistema MultiAgente Normativo* es un SMA organizado por medio de mecanismos para representar, comunicar, distribuir, detectar, crear, modificar e imponer normas y procedimientos para deliberar acerca de las normas y detectar el quebrantamiento y satisfacción de las mismas [BvdTV08].

Motivación

El presente trabajo se enmarca dentro del proyecto **THOMAS** (Me-**TH**ods, Techniques and Tools for **O**pen **M**ulti-**A**gent **S**ystems), cuyo objetivo principal consiste en la investigación y desarrollo de la tecnología basada en agentes/sistemas multiagente necesaria para el desarrollo de organizaciones virtuales en entornos abiertos (sistemas abiertos de servicios basados en agentes). Dicha tecnología incluye la necesidad de la implementación de una plataforma multi-dispositivo de soporte adecuada, así como la inclusión de los servicios e infraestructura necesarios para una gestión inteligente de este tipo de organizaciones. En este sentido, las normas deben introducirse a medio plazo en la tecnología de agentes como una infraestructura para los Sistemas Abiertos [LMSW05].

El objetivo de este proyecto es avanzar y aportar soluciones en estas líneas. Para ello se propone:

- Desarrollar una arquitectura de sistema multiagente adecuada para la generación de organizaciones virtuales en entornos abiertos, así como una plataforma de soporte que permita la implantación de este tipo de sistemas.

- Desarrollar técnicas/métodos de coordinación inteligente de servicios en sistemas multiagente abiertos y descentralizados: localización inteligente de servicios (servicios de directorio, técnicas de equiparación sintáctica y semántica de servicios), así como generación y adaptación de servicios compuestos.

- Desarrollar mecanismos basados en estructuras organizativas y organizaciones virtuales para optimizar y regular la coordinación de servicios en sistemas multiagente abiertos.

Objetivos

El objetivo principal de este trabajo consiste en la implementación y diseño del elemento de la arquitectura encargado de llevar a cabo la gestión de las organizaciones virtuales. Más concretamente, dicho elemento se compone a su vez de: (i) un conjunto de servicios que permiten la gestión de los componentes estructurales de la organización; y (ii) un conjunto de técnicas y procedimientos que permiten el empleo de conceptos normativos como un mecanismo que permita coordinar a las distintas entidades que cooperan dentro de esta sociedad abierta.

Estructura del Trabajo

La memoria está organizada en dos partes. En la Primera Parte, se presenta el Marco Teórico del Trabajo que incluye el estado del arte de las principales áreas que dan soporte a la propuesta. En concreto, se ha realizado una revisión de las diferentes interpretaciones y aportaciones que se han llevado a cabo en el ámbito de los SMA Normativos. El capítulo 1 ofrece una visión general sobre las distintas concepciones del concepto de norma. Tras analizar los distintos conceptos normativos, en el capítulo 2 se introducen los formalismos lógicos para la representación de normas. En el capítulo 3 se realiza una revisión de las últimas propuestas dirigidas a la implementación de SMA normativos.

En la Segunda Parte de la memoria se presenta el Marco de la Propuesta donde se detallan las principales aportaciones del trabajo. En este trabajo proponemos un nuevo modelo normativo teórico adecuado para el diseño de sistemas abiertos (capítulo 4). En el capítulo 5 se describe el trabajo realizado dentro de la implementación de la arquitectura THOMAS. La implementación realizada ha sido validada mediante un caso de estudio en el capítulo 6. Finalmente, en el capítulo 7 se presentan las conclusiones, las aportaciones del trabajo de investigación acorde a los objetivos de la misma, las líneas de trabajo futuras y las publicaciones que se han elaborado durante la realización de la presente tesis de máster.

Parte I

Marco de Teórico. Estado del Arte

Capítulo 1

Concepto de Norma

Las normas han sido estudiadas desde distintas disciplinas. Además el concepto de norma es empleado comúnmente de un modo bastante ambiguo. Por ello resulta complicado ofrecer una definición del concepto de norma que cubra las diferentes aproximaciones. Dentro de la informática los conceptos normativos se han empleado para representar los derechos, deberes y compromisos de los agentes dentro de los SMA.

Existen numerosos trabajos dentro de la sociología, filosofía, psicología, derecho, etc. relacionados con la clasificación de normas [Mor56, The02]. En esta sección se analizarán las propuestas más relevantes con respecto al análisis de los diferentes tipos de normas y conceptos normativos dentro del área de los SMA.

1.1. Definición del Concepto de Norma

Los sistemas normativos tienen el propósito de coordinar la actividad de los miembros de una sociedad. Su existencia sólo tiene sentido si es aceptado por parte de los miembros de la sociedad. El concepto básico sobre el que se articulan los sistemas normativos es la *norma*. Las normas especifican los derechos y responsabilidades de los miembros de una sociedad, de modo que las normas tienen un significado prescriptivo.

En [BvdT08] se propone una clasificación de las normas en función de su propia naturaleza. Dicha clasificación ha sido tomada como referencia en muchos trabajos posteriores:

- *Normas Sustantivas:* definen las relaciones legales entre los miembros de la sociedad y el sistema normativo en sí mismo en términos de normas constitutivas y regulativas:

- *Normas constitutivas:* introducen nuevas clasificaciones abstractas de hechos y entidades, llamados *hechos institucionales*, o describen las consecuencias legales de las acciones en el sistema normativo. Este tipo de normas está basado en la noción de reglas *count-as* y proporcionan un mecanismo de abstracción, es decir, permiten definir la ontología empleada para la definición del comportamiento del sistema. Por ejemplo dentro de un contexto escolar es necesaria una norma constitutiva que defina que el hecho de que dos alumnos distintos presenten 2 trabajos o exámenes iguales se considera una copia.
- *Normas regulativas:* describen el comportamiento ideal de un sistema por medio de obligaciones, prohibiciones y permisos. Generalmente estas normas hacen referencia a conceptos más abstractos definidos mediante normas constitutivas. Continuando con el ejemplo escolar, dentro de este ámbito existe una norma que indica que los alumnos tienen prohibido copiar.

- *Normas Procedurales:* Este tipo de normas tienen enfoque instrumental, es decir, están dirigidas a conseguir el orden social especificado en términos de normas sustantivas. Una norma procedural asociada a la norma regulativa del ejemplo anterior especifica que en caso de copia los alumnos implicados serán suspendidos como sanción.

Las normas sustantivas y procedurales son complementarias. Las normas procedurales son el mecanismo por el cual los derechos y deberes, especificados mediante normas sustantivas, son defendidos e impuestos.

Básicamente esta clasificación organiza las normas en función de la finalidad de la norma, es decir, se define el comportamiento ideal, los hechos institucionales o los mecanismos de *enforcement* (imposición). Con independencia de la finalidad de las normas es posible hacer una clasificación de las mismas atendiendo a la entidad y las circunstancias en que éstas se crean. Más concretamente, en [Dig99] se propone una clasificación de las normas sociales que gobiernan el comportamiento de agentes autónomos en tres niveles:

1. *Nivel Privado:* es el nivel más bajo, en este nivel el agente considera sus diferentes obligaciones y objetivos y determina qué acciones llevará a cabo en consecuencia con ellos. En el nivel privado, el agente traduce tanto las normas sociales como sus propias normas de comportamiento en un concepto más básico denominado *preferencia* o *deseo*.

2. *Nivel Contractual:* en este nivel se describen *obligaciones* y *autorizaciones* entre agentes creados de forma explícita por un periodo de tiempo

limitado. De cada obligación o autorización se indica cómo surge, cómo se satisface y qué ocurre si es quebrantada. Tanto los contratos legales como los acuerdos informales establecidos entre los agentes pueden ser descritos de este modo.

3. *Nivel Institucional:* a este nivel pertenecen las normas de más alto nivel que regulan la coordinación de los agentes dentro de la sociedad. Esta clase describe las normas sociales generales. En este nivel se describen tanto las *obligaciones* como las *prohibiciones* y *permisos* de los agentes. Se definen generalmente de forma estática al diseñar el sistema. Puesto que describen el comportamiento general de los agentes es posible que en una situación en concreto un agente tenga un conflicto entre las diferentes convenciones establecidas siendo necesario un mecanismo para determinar la mejor alternativa de acuerdo con sus propias motivaciones.

Tras comentar brevemente las distintas clasificaciones del concepto de norma, a continuación comentaremos con más detalle los principales trabajos relacionados con la definición de normas pertenecientes al ámbito privado y al ámbito social. Las *normas sociales* comprenden tanto las normas institucionales como las derivadas de la interacción entre los agentes (*nivel contractual*). Por otro lado, las *normas privadas* comprenden las preferencias definidas a nivel interno de cada agente.

1.2. Normas Sociales

Este tipo de normas comprende todas las regulaciones que afectan a los agentes como consecuencia de su pertenencia a un entorno social. Esta categoría incluye tanto las normas pertenecientes al nivel contractual como las normas pertenecientes al institucional.

La dimensión social de las normas ha sido ampliamente estudiada dentro de la sociología. Uno de los trabajos más referenciados relacionados con el tratamiento de las normas dentro de la sociología es [Tuo95], quien estableció la siguiente separación entre las normas sociales:

- *R-normas*: este tipo de normas son aquellas que han sido creadas por una autoridad de la organización o institución.

- *S-normas*: son aquellas que surgen de las convenciones establecidas dentro de un grupo de agentes. Esta noción de *s-norma* está muy relacionada con el concepto de compromiso o contrato.

A continuación nos centraremos en el tratamiento que se le ha dado a las normas sociales dentro del área de la Inteligencia Artificial, más concretamente dentro del campo de los SMA.

1.2.1. Nivel Contractual: Compromiso Social

Un *compromiso* o *acuerdo* [Sin99] se define como una serie de condiciones que deben ser alcanzadas como consecuencia de una interacción entre dos agentes. Mediante estos acuerdos los agentes tratan de influir en el comportamiento de los agentes, aunque se mantiene la autonomía para aceptar los acuerdos.

Definición 1.2.1. *(Compromiso [Sin99]) Un compromiso c se define como una relación* $c = C(x, y, G, p)$*, donde :*

- *x e y son agentes;*
- *G es el grupo o contexto dentro del cual se define el acuerdo ;*
- *p es la condición para la satisfacción del acuerdo c.*

De modo que, c denota un acuerdo del agente x hacia el agente y.

En [Sin99] los compromisos son condiciones o acciones que deben ser alcanzadas por los agentes. Otro tipo de acuerdo definido en este mismo trabajo son las denominadas *políticas*. Una política es una expresión condicional que contiene acuerdos u operaciones sobre los acuerdos, de modo que las políticas o metacompromisos son expresiones condicionales que versan sobre algún compromiso.

1.2.2. Nivel Institucional: Normas Sociales

Las normas o leyes sociales son aquellas restricciones creadas por una autoridad de la institución para evitar los problemas existentes en dichas sociedades como consecuencia de la existencia de agentes autónomos y heterogéneos.

A. Ley Social

Uno de los primeros trabajos en esta línea es la propuesta de las *leyes sociales* realizada en [MT95]. En esta propuesta los autores sugieren diseñar los SMA de modo que las acciones de los agentes estén restringidas a aquellas que se corresponden con comportamientos socialmente aceptados.

Los sistemas donde el comportamiento de los agentes se determina mediante la definición de acciones restringidas reciben el nombre de *sistemas sociales artificiales*. Este componente, el *sistema social artificial*, es considerado como el componente principal en el diseño de SMA y debe ser definido explícitamente. Para ello, en primer lugar se formaliza un sistema multiagente, como sigue:

Definición 1.2.2. *(SMA [MT95]) Los autores definen un SMA como una tupla*
$S = \langle N, W, K_1, ..., K_n, A, Able_1, ..., Able_n, I, T\rangle$ *donde:*

- $N = \{1, ..., n\}$ *es un conjunto de agentes;*
- W *es un conjunto de mundos posibles;*
- $K_i \subseteq W \times W$ *son las relaciones de accesibilidad. Este tipo de funciones captura el conocimiento del agente* i*;*
- A *es un conjunto de acciones primitivas individuales;*
- $Able_i : W \rightarrow 2^A$ *es una función que determina las acciones posibles para un agente* i*.*
- I *es el conjunto de Inputs externos.*
- $T : W \times (A \times I)^n \rightarrow W \cup \{-\}$ *es una función de transición entre estados.* $T(w, (a, I)) = -$*, si existe una acción* a_i *en* (a, I) *tal que* $a_i \notin Able_i(w)$*.*

Los sistemas sociales se construyen en torno a una serie de *leyes sociales*. Más formalmente, una *ley social* se compone de una serie de funciones que restringen el conjunto de acciones "permitidas" para un agente en un momento dado. Las anteriores funciones $Able_i(w)$ pueden interpretarse como *leyes físicas*, ya que determinan las acciones que los agentes son físicamente capaces de realizar. Un primer paso para la definición formal de las *leyes sociales* consiste en ampliar la anterior definición de SMA para definir un sistema normativo que restrinja las acciones de los agentes:

Definición 1.2.3. *(Sistema Normativo [MT95]) Un sistema normativo* η *se define como* $\eta = \langle S, \{Legal_i\}_{i \leq n}\rangle$ *donde:*

- S *es un sistema multiagente;*
- $Legal_i : W \rightarrow 2^A$ *donde la función satisface las siguientes propiedades:*
  1. *(adecuación epistemologica)*$Legal_i(w) = Legal_i(w')$ *para todo* $(w, w') \in K_i$*;*

2. *(adecuación física)*$Legal_i(w) \subseteq Able_i(w)$ *para todo* i *y* w*;*
3. *(no trivialidad)*$Legal_i(w) \neq \emptyset$ *para todo* i *y* w*;*

Intuitivamente, $Legal_i(w)$ especifica qué acciones está autorizado el agente i a realizar en el estado w, de acuerdo con el sistema normativo.

En la anterior definición de sistema normativo no existe ningún elemento que garantice que los agentes del sistema respeten las normas establecidas. El *sistema social* es el encargado de evitar que aparezcan estas situaciones indeseables. De modo que para la definición formal de un *sistema social* se definen los conjuntos W_{soc} y G_{soc} que contienen los mundos y objetivos "socialmente aceptables".

Definición 1.2.4. *(Sistema Social [MT95]) Formalmente, un sistema social se define como* $\langle S, \{Legal_i\}_{i \leq n} \rangle, W_{soc}, G_{soc} \rangle$ *tal que:*

- S *es un sistema social;*
- W_{soc} *y* G_{soc} *que contienen los mundos y objetivos "socialmente aceptables";*
- *Un mundo* $w \in W$ *es legalmente alcanzable sólo si* $w \in W_{soc}$*;*
- *Para cada mundo accesible* w*, si el objetivo del agente* i *en* w *satisface* $g \in G_{soc}$*, entonces existe un plan legal para* i *desde el estado* w *e* i *realizará* g*.*

Desde esta perspectiva, el problema fundamental en el diseño de SMA es el de encontrar el conjunto de restricciones (ley social) sobre las acciones que los agentes pueden llevar a cabo en cada estado del sistema. Estas restricciones deben inducir un sistema en el cual los agentes puedan llevar a cabo los objetivos para los que fueron diseñados, alcanzando un balance adecuado entre los objetivos en conflicto de los distintos agentes.

Dentro de los SMA las leyes sociales cumplen un doble objetivo. Por un lado, reducen el conjunto de objetivos o estrategias que un agente es capaz de lograr. Por otro lado, esta restricción en el comportamiento de los *otros* agentes puede hacer posible que un agente consiga más objetivos y de una forma más eficiente. Esto es debido a que el sistema social reducen en cierta medida los conflictos existentes entre objetivos incompatibles dentro del SMA. Los diseñadores deben encontrar un balance adecuado entre los objetivos en conflicto de los distintos agentes. Es decir, el sistema social debe restringir los comportamientos de los agentes para evitar la aparición de conflictos o incompatibilidades, pero permitiendo que se satisfagan el mayor número de objetivos dentro de la sociedad. Sin embargo no siempre es posible evitar

todos los conflictos, por ello es necesario algún componente que describa el tratamiento de los conflictos (negociación, consenso, etc.).

Otro aspecto a tener en cuenta en el diseño de sistemas sociales es que este proceso se realiza *off-line*, es decir es llevado a cabo por los diseñadores antes del comienzo de la actividad. Los autores de la propuesta justifican esta elección en base a dos argumentos: el diseñador del sistema dispone de una mayor cantidad de recursos e información acerca del sistema y, por otro lado, el diseño *off-line* de un sistema social adecuado puede evitar la aparición de conflictos durante el transcurso de la actividad.

B. Normas para Agentes Autónomos

Recientemente han surgido varios trabajos centrados en ofrecen un modelo de *SMA Normativo* en el que sí se tenga en cuenta aspectos como la autonomía de los agentes y la dinamicidad del contexto normativo.

Uno de los trabajos más referenciados en esta línea es modelo normativo para agentes autónomos presentado en [LyLL03]. En este trabajo los autores proponen un modelo de normas que permita a los agentes conocer cuáles son las normas que les afectan y de este modo coordinar la acción de los diferentes agentes heterogéneos que cooperan dentro de un SMA Abierto.

Definición 1.2.5. *(Norma [LyLL03]) Una Norma se define como:*

addresses, beneficiaries: $\mathbb{P}$ *Agent*
normativegoals, reward, punishments: $\mathbb{P}$ *Goal*
context, exceptions: $\mathbb{P}$ *EnvState*
$normativegoals \neq \emptyset$; $addresses \neq \emptyset$; $context \neq \emptyset$
$context \cap exceptions = \emptyset$; $addresses \cap punishments = \emptyset$

Definición 1.2.6. *(Agente Autónomo Normativo [LyLL03]) Un Agente Autónomo Normativo se define como:*

capabilities: $\mathbb{P}$ *Action; beliefs:* $\mathbb{P}$ *Attribute*
goals: $\mathbb{P}$ *Goal; motivations:* $\mathbb{P}$ *Motivation*
norms: $\mathbb{P}$ *Norm*
$goals \neq \emptyset$; $motivations \neq \emptyset$
$norms \neq \emptyset$

Las normas no son un concepto estático sino que se encuentran en diferentes estados desde el momento de su creación hasta que es abolida. La Figura 1.1 muestra un diagrama de estados correspondiente a la dinámica de las normas [LyLL02]. El ciclo de vida de una norma comienza cuando un agente con capacidad para cambiar el contexto normativo publica (*Issue*) una nueva norma. A partir de este momento, la nueva norma es comunicada (*Spread*) a los miembros se la sociedad. Entonces tiene lugar el proceso de adopción de

la norma, por el cual un agente expresa su conformidad con la misma. Una vez que la norma ha sido adoptada, permanece inactiva o latente hasta que se satisfacen las condiciones para su aplicación (*Activation*). En los estados de excepción los agentes no están obligados a respetar las normas, por lo que las normas pueden ser desestimadas (*Dismissal*). En la mayoría de los casos esto no ocurrirá; con lo que las normas serán respetadas por los agentes, en cuyo caso se aplicará la recompensa asociada, o quebrantadas, en este caso se aplicará la sanción correspondiente.

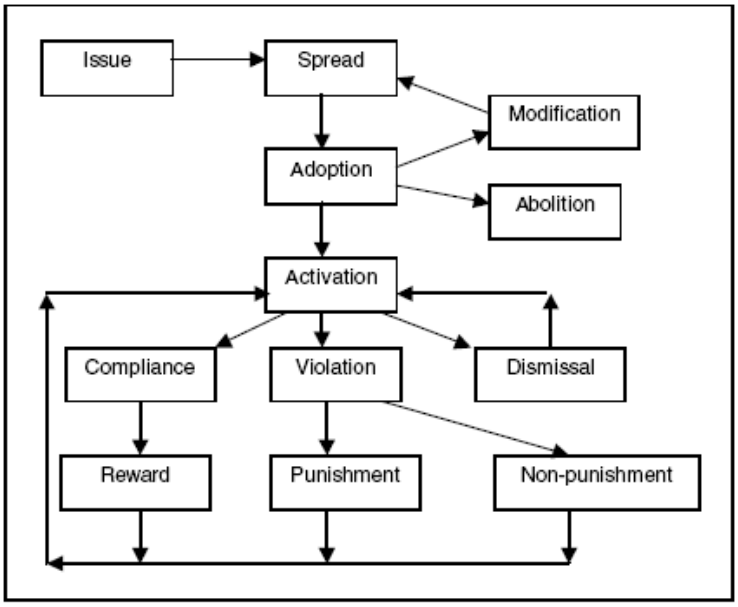

Figura 1.1: Dinámica de las normas

Tal y como se ha indicado en la imagen correspondiente a la dinámica de las normas, éstas pueden ser creadas por un agente con capacidad para cambiar el contexto normativo. Las capacidades para realizar cambios en el contexto normativo se definen mediante un conjunto de normas, llamadas *normas legislativas*, que permiten llevar a cabo las acciones de abolición o declaración de nuevas normas.

Definición 1.2.7. *(Norma Legislativa [LyLL03]) Formalmente la relación* $legislate : (Norm \times EnvState) \rightarrow 2$ *que permite definir a una norma* n *como una norma Legislativa en un determinado entorno* env *es:*

$$legislate(n, env) \Leftrightarrow (\exists normativeAction \in \{AbolishNorm, CreateNorm\}\ permitted(normativeAction, n, env))$$

Donde permitted es una función que determina para una acción, una norma y un estado si la acción contribuye a la satisfacción de los objetivos de la norma (normativegoals) en cuyo caso se considerará como permitida.

Los diferentes estados que conforman la dinámica de una norma permiten definir varios tipos de relaciones entre agentes. Estas relaciones se muestran en la figura 1.2. Básicamente se definen dos conjuntos de relaciones entre los agentes:

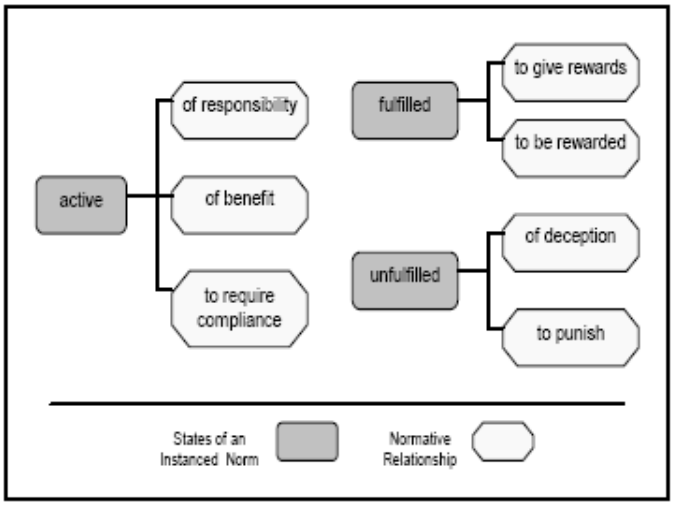

Figura 1.2: Relaciones Normativas

1. *Relaciones Legislativas.* Un agente es una *autoridad legislativa* para otro si el primero es un agente legislador en alguna de las organizaciones o sociedades a las que el segundo pertenece.

2. *Relaciones* vinculadas con las *normas activas*. Con respecto a una norma activa existen 3 tipos de relaciones:

   a) *Responsabilidad.* El agente al que va dirigida la norma (*addressed*) y que ha decidido adoptarla, tiene la responsabilidad de respetar la norma.

   b) *Beneficio.* Del mismo modo que ocurría con la responsabilidad se puede definir el beneficio como la relación contraria que se establece entre el agente beneficiario y él que está afectado por la norma.

c) *Exigencia.* Existe una relación de exigencia entre el agente encargado de llevar a cabo la recompensa o la sanción de una norma y el agente *addressed*.

Un *SMA Normativo* se define como un conjunto de *agentes normativos* que están regulados por una serie de *normas*.

Definición 1.2.8. *(SMA Normativo [LyLL03]) SMA Normativo se define como:*

$members$: $\mathbb{P}$ $NormativeAgent$
$generalNorms$, $legislationNorms$: $\mathbb{P}$ $Norms$
$environment$: $\mathbb{P}$ $EnvState$
$\forall ag \in members : ag.norms \cap generalNorms \neq \emptyset$
$\forall gn \in generalNorms : gn.addresses \subseteq members$
$\forall ln \in legislationNorms : legislate(ln, environment)$

Este *control* que el sistema ejerce sobre sus miembros puede observarse en tres aspectos diferentes.

- Primero, los agentes se conciben a sí mismos como parte de una sociedad. Ser miembro de una sociedad normativa implica que el agente está sujeto alguna de las normas del sistema.

- En segundo lugar, las normas deben incorporar mecanismos para hacer frente al comportamiento autónomo de los agentes (sanciones y recompensas). Las normas no se definen de forma aislada unas de otras, de modo que el cumplimiento o violación de una norma puede actuar como condición para la activación de otra norma. Un ejemplo interesante de este tipo de normas, son aquellas que se activan cuando se quebranta alguna norma para infringir un castigo al agente infractor. Del mismo modo se definen normas que llevan a cabo las recompensas asociadas al cumplimiento de una norma. Este tipo de normas tienen como objetivo motivar el respeto de las normas por parte de agentes autónomos.

- Por último, el sistema debe permitir que se produzcan modificaciones en el conjunto de normas definidas. En todas las sociedades debe existe la posibilidad de crear, modificar o eliminar nuevas normas. Sin embargo, estas capacidades suelen estar restringidas a ciertos agentes. Las normas destinadas a restringir quién y cuándo se pueden realizar estas acciones legislativas reciben el nombre de *normas legislativas* en el modelo propuesto.

En estos sistemas el comportamiento de los agentes no sólo está afectado por la existencia de de las normas sino también por las diferentes relaciones que se producen entre los agentes como consecuencia de las mismas.

1.3. Nivel Privado

A este nivel pertenecen las normas o reglas que guían el comportamiento de los agentes, es decir las intenciones del agente son vistas como acuerdos con él mismo acerca de la realización de una determinada acción o plan.

Los agentes deben traducir aquellas normas pertenecientes a los niveles superiores (compromisos y normas institucionales) que deseen satisfacer a intenciones de este nivel privado.

1.3.1. Formalización de las Normas Privadas en una arquitectura de Agente BDI

En [Dig99] se propone una formalización de las normas (esto incluye las normas de los 3 niveles) a las que se adhiere un agente BDI [RG91] mediante un concepto más básico: las preferencias o *deseos* condicionales. De modo que, el comportamiento de un agente viene completamente determinado por sus preferencias y el orden establecido entre ellas.

Definición 1.3.1. *(Preferencia [Dig99]) Una preferencia se define como:*

$Pref_i(\phi|\varphi)$ *sii el agente i prefiere que sea cierto* ϕ *en cada situación en la que la condición* φ *sea cierta.*

Las preferencias son condicionales por el hecho de que son dependientes del entorno concreto en que se sitúa un agente.

Por ejemplo, si un agente desea cumplir con una obligación puede transformar la obligación en una preferencia del siguiente modo:

$$\forall i, j Pref_i(\phi|O_{ij}(\phi))$$

De modo que si el agente tiene una obligación($O_{ij}(\phi)$) acerca de una condición (ϕ) entonces prefiere que la condición (ϕ) sea cierta. Estas obligaciones pueden ser el resultado de las políticas internas del agente, las normas institucionales o derivarse de un compromiso adquirido por el agente.

Definición 1.3.2. *(Objetivos del agente [Dig99]) Los objetivos del agente de acuerdo con sus preferencias son:*

$$Goal_i(\phi|\varphi) \equiv Pref_i(\phi|\varphi) \wedge \neg\phi \wedge Achiev_i(\phi).$$

Los objetivos de un agente son todas aquellas preferencias no satisfechas que son alcanzables por el agente en cuestión. Esta definición permite que en un momento dado existan varios objetivos distintos para un agente dado. Para poder determinar qué objetivo perseguir en un momento dado es necesario establecer un orden entre las preferencias.

1.4. Conclusiones

En este capítulo se presenta un análisis de las diferentes definiciones otorgadas al concepto de norma. Estos trabajos abordan el problema de la definición teórica de lo que se entiende por norma dentro del área de la Teoría de Agentes. Estas definiciones van desde la definición de norma como una simple obligación o autorización impuesta a los agentes, hasta el concepto de norma como un compromiso adoptado por los agentes dentro de una sociedad.

La mayoría de las propuestas analizadas asumen que el diseño del Sistema Normativo, entendiendo por Sistema Normativo al conjunto de reglas que rigen una sociedad, se realiza de forma *off-line*. Es decir las normas se definen de forma estática antes de iniciarse el sistema. Esta asunción puede resultar adecuada para Sistemas Cerrados donde el diseñador tiene un control y conocimiento absoluto de lo que ocurrirá durante la interacción entre los agentes. Sin embargo, para modelar aquellos sistemas donde no es posible realizar ninguna asunción sobre el comportamiento de los participantes son necesarios mecanismos que permitan adaptar de forma *dinámica* el contexto normativo para adecuarlo a las necesidades actuales del sistema.

Por otro lado, los modelos comentados se centran principalmente en ofrecer un modelo de normas que dé soporte a uno de los tres niveles de normatividad: nivel privado, nivel contractual y nivel institucional.

Las principales características necesarias en un modelo de SMA normativo que permita el diseño de sistemas abiertos son:

- Debe cubrir los tres niveles de normatividad: institucional, contractual y privado.
- Debe ofrecer soporte para los distintos tipos de normas y permitir establecer la relación existente entre ellas.
- Debe permitir la definición dinámica del contexto normativo, es decir, la creación y derogación de nuevas normas.

Precisamente, en la segunda parte de esta tesis de máster, en el capítulo 4 sección 4.1, se propone un nuevo modelo de SMA Normativo diseñado con el objetivo de ofrecer soporte para la definición y modelado de SMA Normativos Abiertos, teniendo en cuenta estas características.

A continuación, en el siguiente capítulo se introducirán las diferentes propuestas lógicas relacionadas con la formalización de las normas, es decir, la definición de obligaciones, permisos y prohibiciones.

Capítulo 2

Lógicas para la Formalización de las Normas

En el capítulo anterior se han presentado varias propuestas que analizan los distintos tipos de normas. A pesar de que estos trabajos permiten identificar las normas que rigen una sociedad, es necesario algún formalismo para representar estos conceptos y que permita razonar acerca de las restricciones.

Uno de los primeros aspectos que se tomaron en consideración en el desarrollo de la lógica deóntica fue la distinción entre la lógica de las normas y la lógica de las posiciones normativas [ThT01]. Más concretamente, las normas tienen un significado prescriptivo, en este sentido preceptúan u ordenan el comportamiento deseado y por tanto no tienen un valor de verdad asociado. Por otro lado, las proposiciones normativas tienen un significado descriptivo, es decir, describen las normas que rigen una sociedad y las relaciones existentes entre los miembros de la misma como consecuencia de las normas vigentes. Esta distinción entre las normas y las proposiciones normativas puede verse como la diferencia entre aplicar y observar normas.

Tomando como punto de partida esta distinción entre la lógica de las normas y la lógica de las proposiciones normativas, en esta sección se presentan los trabajos más destacados relacionados con la formalización de los aspectos prescriptivos y descriptivos de las normas. Dentro de la formalización de las normas comentaremos con más detalle los operadores pertenecientes a la lógica deóntica y cómo estos operadores han sido incluidos dentro de otras lógicas, como las lógicas temporales, modales, dinámicas, etc. Por otro lado, comentaremos brevemente los trabajos relacionados con las proposiciones normativas como mecanismo para formalizar los aspectos descriptivos de un sistema normativo.

2.1. Lógica de Normas

Las normas determinan los derechos y deberes de los miembros de una sociedad en base a permisos, prohibiciones y obligaciones. Su significado es meramente prescriptivo y por este motivo no tienen un valor de verdad asociado.

A lo largo de esta sección comentaremos las propuestas más relevantes realizadas dentro de la lógica para permitir la formalización de este tipo de prescripciones. En primer lugar se describen las bases de la lógica deóntica, lógica que define operadores tales como la obligación, permiso, etc. Posteriormente se comentan otras variantes lógicas, tales como la lógica dinámica o la lógica temporal, que incorporan dentro de ellas a los operadores deónticos, permitiendo de este modo representar prescripciones normativas. Por último, se comenta un trabajo relacionado con el empleo de la lógica deóntica para representar las normas que determinan el comportamiento de los agentes.

2.1.1. Lógica Deóntica

El término 'deóntico' se deriva del griego antiguo *déon* que significa 'lo debido' [1] . La lógica deóntica es el campo de la lógica simbólica que se encarga de abordar conceptos normativos tales como la obligación, el permiso y la prohibición.

La *Standard Deontic Logic* (SDL) fue una de las primeras lógicas deónticas en ser axiomatizada y es, quizás, el sistema de lógica deóntica que ha sido más ampliamente estudiado.

A. Sintaxis de la SDL

La SDL se basa en la lógica proposicional y pertenece a una clase de las lógicas modales más conocidas: las "lógicas modales normales". Se trata de una lógica deóntica monádica, ya que su operador deóntico básico (**OB** obligación) es un operador *one-place*, es decir, que se aplica a una única frase o sentencia para obtener una sentencia compuesta.

Tomando como base el lenguaje proposicional clásico con un conjunto infinito de variables, los operadores de negación ($\neg$) e implicación ($\rightarrow$), el operador deóntico (**OB**), la axiomatización de la SDL se indica a continuación:

[1] http://www.science.uva.nl/ seop/entries/logic-deontic/

A1.Todas las tautologías de las wffs [2]	(TAUT)
A2. $\mathbf{OB}(p \rightarrow q) \rightarrow (\mathbf{OB}p \rightarrow \mathbf{OB}q)$	(**OB**-K)
A3. $\mathbf{OB}p \rightarrow \neg\mathbf{OB}\neg p$	(**OB**-D)
R1. Si $\vdash p$ y $\vdash p \rightarrow q$ entonces $\vdash q$	(MP)
R2. Si $\vdash p$ entonces $\vdash \mathbf{OB}p$	(**OB**-NEC)

TAUT es común a todos los sistemas de la lógica modal normal. **OB**-K, donde K es el axioma presente en todas las lógicas modales normales, indica que si una expresión condicional es obligatoria, cuando su antecedente sea obligatorio también lo será su consecuente. El axioma **OB**-D indica que si p es obligatorio su negación no lo es, es decir, no existen obligaciones en conflicto. MP es el axioma clásico de *Modus Ponens*. Por último, **OB**-NEC indica que si algo (p) es un teorema entonces la afirmación de que es obligatorio ($\mathbf{OB}p$) también lo es.

Cada uno de los principios deónticos (**OB**-K, **OB**-D y **OB**-NEC) son discutibles y, de hecho, existen multitud de trabajos reflejando las inconsistencias y paradojas presentes en la lógica deóntica [HPvdT07]. Algunas de estas paradojas se detallan a continuación [Ald07]:

1. Paradoja de Ross:	$\mathbf{OB}p \rightarrow \mathbf{OB}(p \vee q)$
2. No Free Choice Permission:	$(\mathbf{PE}p \vee \mathbf{PE}q) \leftrightarrow \mathbf{PE}(p \vee q)$
3. Paradoja del Buen Samaritano:	$p \rightarrow q \vdash \mathbf{OB}p \rightarrow \mathbf{OB}q$
4. Paradoja de Chisholdm:	$(\mathbf{OB}p \wedge \mathbf{OB}(p \rightarrow q) \wedge (\neg p \rightarrow \mathbf{OB}\neg q) \wedge \neg p) \rightarrow \bot$

B. Semántica de la SDL

La semántica de la lógica SDL se define en base a un modelo de Kripke, que se define como sigue.

Definición 2.1.1. *(Modelo de Kripke [Kri63]) Formalmente se define un modelo de Kripke (M) como una tupla $M = \langle W, A, V \rangle$ donde:*

- *W es el conjunto no vacío de mundos(estados) posibles*
- *A es una relación de accesibilidad entre los estados del mundo, es decir $A \subseteq W \times W$*
- *A cumple la siguiente propiedad $\forall i \exists j : Aij$*
- *V es una función de asignación de valores de verdad a las variables proposicionales para cada elemento de W*

[2] *well-formed formulas*

La expresión $M \models_i p$ denota que la proposición p en el mundo i es cierta de acuerdo con el modelo de Kripke M.

La semántica de los operadores deónticos es la siguiente:

PC: (Clausulas Estándar para los operadores de la lógica Proposicional.)
OB: $M \models_i \mathbf{OB}p : \forall j(Aij \rightarrow M \models_j p)$
PE: $M \models_i \mathbf{PE}p : \exists j(Aij \wedge M \models_j p)$
IM: $M \models_i \mathbf{IM}p : \forall j(Aij \longrightarrow M \models_j \neg p)$
GR: $M \models_i \mathbf{GR}p : \exists j(Aij \wedge M \models_j \neg p)$
OP: $M \models_i \mathbf{OP}p : \exists j(Aij \wedge M \models_j p) \wedge \exists j(Aij \wedge M \models_j \neg p)$

De modo que p es *obligatorio*(**OB**) en un estado i sii p es cierto en todos los estados accesibles desde i; p es *permisible* (**PE**) sii p es cierto en algún estado accesible desde i; p es *prohibido*(**IM**) sii no es cierto en ningún mundo accesible; p es *gratuito*(**GR**) sii la negación de p es cierta en algún estado i-accesible; y p es *opcional*(**OP**) si los estados donde se verifique tanto p como $\neg p$ son accesibles desde i.

C. Reducción de Anderson

En [And58] se propone la formalización de la lógica deóntica mediante una lógica modal. Esta reducción de la SDL a una *lógica modal* ha sido muy utilizada para modelar los sistemas normativos. Normalmente por *lógica modal* se entiende aquella rama de la lógica encargada del estudio de la formalización de expresiones del tipo "necesario" y "posible". Sin embargo, el término "lógica modal" suele utilizarse para categorizar a una familia de lógicas con reglas similares y diferentes símbolos. En este sentido, se dice que la lógica deóntica pertenece a la clase de la "lógica modal".

La *lógica modal* (K) consiste básicamente en una lógica proposicional extendida con los operadores modales $\Box$ y $\diamond$ que representan la *necesidad* y la *posibilidad* respectivamente. La propuesta de Anderson consiste básicamente en extender la lógica modal proposicional clásica con una constante proposicional (deóntica) "d", que representa la condición de que todas las prescripciones normativas se satisfacen.

De modo que se define el siguiente sistema de axiomas (Kd):

Kd.	A1: All Tautologies	(TAUT)
	A2. $\Box(p \rightarrow q) \rightarrow (\Box p \rightarrow \Box q)$	(K)
	A3. $\diamond d$	($\diamond d$)
	R1. If $\vdash p$ and $\vdash p \rightarrow q$ then $\vdash q$	(MP)
	R2. If $\vdash p$ then $\vdash \Box p$	(NEC)

Este sistema Kd se corresponde con un sistema de lógica modal al que se le ha añadido el axioma A3 que indica que el estado donde se cumplan todas las especificaciones normativas es posible. En otras palabras, la constante deóntica d divide el conjunto de estados en dos grupos: aquellos en los cuales se satisfacen las normas y aquellos estados en los que se ha producido la violación de alguna prescripción normativa. Anderson propuso la representación de todos los operadores de la SDL mediante el sistema Kd, tal y como se muestra a continuación:

$$\mathbf{OB}p =_{df} \Box(d \rightarrow p)$$
$$\mathbf{PE}p =_{df} \Diamond(d \wedge p)$$
$$\mathbf{IM}p =_{df} \Box(d \rightarrow \neg p)$$
$$\mathbf{GR}p =_{df} \Diamond(d \wedge \neg p)$$
$$\mathbf{OP}p =_{df} \Diamond(d \wedge p) \wedge \Diamond(d \wedge \neg p)$$

De modo que en Kd, p es *obligatorio* sii p es necesario para que todas las prescripciones normativas se cumplan, *permisible* sii p es compatible con d, *prohibido*(**IM**) sii p es incompatible con la satisfacción de las normas (d), *gratuito* sii la negación de p es compatible con d, y *opcional* sii tanto p como $\neg p$ son compatibles con d.

2.1.2. Extensiones de la Lógica Deóntica

Con frecuencia las obligaciones y permisos de los agentes son dependientes de las acciones y eventos pasados. Con el objetivo de representar estas relaciones la lógica deóntica se ha extendido con operadores pertenecientes a la lógica dinámica y la lógica temporal.

A. Lógica Dinámica

En [Mey87] se propone la reducción de la lógica deóntica como una variante de la lógica dinámica. La lógica dinámica puede considerarse como una lógica modal a la que se le han añadido axiomas para la gestión y tratamiento de las aserciones y las acciones. Ha sido empleada dentro del campo de la informática teórica para probar las propiedades y corrección de los programas.

La característica principal de la lógica dinámica es el operador $[\alpha]$ asociado a una acción α. La expresión $[\alpha]\phi$ representa que tras la acción α la condición ϕ es cierta. A continuación se muestra la equivalencia de las fórmulas deónticas mediante operadores de la lógica dinámica:

$$\mathbf{IM}\alpha =_{df} [\alpha]V$$
$$\mathbf{OB}\alpha =_{df} IM\neg\alpha$$
$$\mathbf{PE}\alpha =_{df} \neg IM\alpha$$

De modo que la *prohibición* ($\mathbf{IM}\alpha$) de α se redefine dentro de la lógica dinámica como $[\alpha]V$ que representa que tras la acción α se inserta una violación (V) en el estado. Los otros dos operadores deónticos básicos se definen en base a este operador de prohibición.

B. Lógica Temporal

En [DMWK96] se introducen operadores temporales a la lógica deóntica, así como meta-acciones para expresar decisiones y compromisos. Las nuevas fórmulas temporales, que permiten representar que una acción ha sido realizada o se tiene la intención de llevarse a cabo en un futuro, son:

- $Prev(\alpha)$ significa que el estado actual ha sido alcanzado al ejecutar la acción α.
- $Past(\alpha)$, cuyo significado es que la acción α ha sido realizada en el pasado
- $Int(\alpha)$ significa que se prefiere realizar la acción α a continuación.

Por otro lado este formalismo lógico es extendido con dos tipos de meta-acciones: *Decide* y *Commit*. La acción *Decide* establece qué acciones se pretenderán realizar en un cierto estado. La acción *Commit* no sólo implica la pretensión o deseo de realizar una cierta acción sino la obligación de llevarla a cabo.

2.1.3. Representación de las Normas mediante Lógica Deóntica

En [Dig99] se propone una aproximación basada en lógica deóntica para realizar el modelado de las normas. Estos conceptos normativos son necesarios para capturar el comportamiento de agentes autónomos. El uso de la lógica deóntica permite capturar tanto la autonomía de los agentes, como las dependencias sociales establecidas entre los agentes.

El concepto básico empleado para modelar las normas es la *obligación*. Ésta se define del siguiente modo:

$O_{ij}(\alpha)$ significa que el agente i está obligado por el agente j a realizar la acción α.

El concepto de *autorización* puede verse como el concepto homólogo de la obligación. Mediante él se describe la misma dependencia que la obligación pero desde el punto de vista del otro agente. Es decir, si un agente está

autorizado a demandar una acción a otro es porque este último está obligado a llevarla a cabo. Este concepto se modela mediante un predicado especial con dos argumentos, el agente y la actividad que está autorizado a realizar:

$auth(a, \alpha)$ significa que el agente a esta autorizado para realizar la acción α.

La *prohibición* en este modelo representa una limitación en el comportamiento de los agentes. Se modela como la negación de la autorización.

$$\neg auth(agent, action) \rightarrow Forbidden_{agent}(action)$$

El concepto de *permiso* solamente se emplea para indicar excepciones al comportamiento general de una regla o en caso de incertidumbre.

Además se permite definir las condiciones para la activación de una norma, o una fórmula si las expresamos mediante lógica deóntica, mediante el predicado:

$[\alpha]\phi$ significa que tras realizar la acción α la formula ϕ está activa.

Mediante el uso del operador $\rightarrow$ se pueden definir las violaciones de las normas así como las sanciones asociadas. Por ejemplo, supongamos una norma para un sistema de reserva de vuelos según la cual "la aerolínea (a) está obligada a transportar al pasajero (p)". Esta norma puede ser violada si el pasajero o la compañía cancelan el vuelo. La consecuencia de esta violación es que quien cancele el vuelo está obligado a pagar los costes. Esta norma quedaría definida como sigue:

$$[reservationFlight(a, p)]O_{ap}(transportPassanger)$$
$$[O_{ap}(transportPassanger) \rightarrow cancelFlight(a)]O_{ap}(payCost)$$
$$[O_{ap}(transportPassanger) \rightarrow cancelFlight(p)]O_{pa}(payCost)$$

2.2. Lógica de las Proposiciones Normativas

Tal y como se comentó en la introducción del presente capítulo, los formalismos lógicos para representar aspectos normativos son de dos tipos: los formalismos que permiten especificar las *normas* que rigen una sociedad y la formalización de las relaciones o *proposiciones normativas*. Tras realizar un análisis de los diferentes trabajos centrados en la definición formal de normas en la sección anterior, la presente sección tiene como objetivo resumir los aspectos fundamentales de la *teoría de las proposiciones normativas*.

Dentro del área de la filosofía, se enmarca principalmente la distinción entre las normas y las proposiciones o declaraciones normativas. Las proposiciones normativas tienen un valor de verdad asociado, es decir, pueden ser

ciertas o falsas. Las normas, por el contrario, pueden ser respetadas o no, pueden ser aplicadas, etc. pero no tiene sentido describirlas como verdaderas o falsas.

Los trabajos presentados a continuación están dirigidos a desarrollar una lógica de normas que tenga en consideración no sólo las normas sino el sistema del que forman parte. De modo que estas propuestas emplean los operadores de la lógica deóntica para representar las normas así como otras construcciones lógicas que permiten representar su estado. A continuación se describe la lógica de las posiciones normativas, los sistemas de transición entre estados, las lógicas temporales y la teoría de las *social expectations*.

2.2.1. Lógica de las Posiciones Normativas

La teoría de las *Posiciones Normativas* fue desarrollada inicialmente en [Kan72, Lin77]. En estos primeros trabajos quedó de manifiesto el hecho de que la lógica deóntica estándar (SDL), que consiste en aplicar un operador deóntico a las proposiciones, no resultaba adecuada para representar las relaciones jurisdiccionales. De este modo, el término *posición normativa* se define como el rango de posibles relaciones normativas que pueden darse dentro de una sociedad de agentes con respecto a la realización de un cierto tipo de acción o a una circunstancia dada. Ejemplos de estas relaciones son el derecho, deber, etc. Por lo tanto, tiene un carácter descriptivo y un valor de verdad asociado.

Definición 2.2.1. *(Lenguaje de las Posiciones Normativas) El lenguaje propuesto en [Kan72] consiste en una Lógica de Primer Orden (LPO) aumentada con las expresiones modales de obligación O y el operador modal Do para representar las acciones realizadas por los agentes.*

Mediante este lenguaje es posible combinar el operador deóntico de obligación con el operador de acciones. Un ejemplo de sentencia donde se combinan ambos operadores es: $O(Do(x, F))$, el agente x está obligado a realizar la acción F.

Definición 2.2.2. *(Postulados Lógicos de las Posiciones Normativas) Los postulados lógicos para los operadores O y Do se indican a continuación:*

1. *Si $F \rightarrow G$, entonces $O(F) \rightarrow O(G)$.*
2. *$(O(F) \wedge O(G)) \rightarrow O(F \wedge G)$.*
3. *$O(F) \rightarrow \neg O(\neg F)$.*
4. *Si $F \rightarrow G$ y $G \rightarrow F$, entonces $Do(x, F) \rightarrow Do(x, G)$.*
5. *$Do(x, F) \rightarrow F$.*

La idea fundamental de esta teoría es conseguir, dada una verdad en unas determinadas circunstancias, como por ejemplo $O(Do(a, F))$ (expresa que el agente a está obligado a llevar a cabo la acción F), determinar los permisos y obligaciones de otro agente b consecuentes con esta verdad. Estas posibilidades comprenden tanto los permisos y obligaciones de b (estas posiciones normativas reciben el calificativo de *básicas*), como las combinaciones de éstas. De este modo, en [Lin77] se estudian los diferentes tipos de posiciones normativas, definiéndose 26 tipos de relaciones normativas posibles. Cada una de estas posiciones es internamente consistente, mutuamente excluyente y en cada situación sólo una de ellas es cierta.

Tomando como base los trabajos anteriormente citados, en [Ser98] se realiza un refinamiento de esta teoría teniendo como referencia su aplicación dentro del ámbito de la informática. Así, se extiende la propuesta para determinar las relaciones o posiciones normativas existentes entre más de un par de agentes. Además se presenta una aplicación práctica mediante un algoritmo que realiza todo este proceso inferencial que permite determinar las posiciones normativas de un conjunto de agentes en una determinada situación.

2.2.2. Sistemas de Transición entre Estados. Lenguaje $nC+$

En [SC06] se presenta el lenguaje $nC+$ diseñado para representar aspectos normativos e institucionales de las sociedades. Es una extensión del lenguaje de acciones $C+$, formalismo que permite especificar y razonar acerca de los efectos de las acciones y la persistencia de los hechos a lo largo del tiempo [GLL$^+$04].

Definición 2.2.3. *(Sistema de Transición entre Estados) Un sistema de transición entre estados se define como una tupla $\langle S, A, R\rangle$, donde:*

- *S es un conjunto de estados*
- *A es un conjunto de etiquetas de transiciones (eventos)*
- *R es el conjunto de transiciones.*

La extensión deóntica de $nC+$ consiste en un mecanismo para etiquetar los estados y transiciones permitidas en un sistema de transición entre estados.

Definición 2.2.4. *(Sistema de Transición entre Estados Coloreado) Un sistema de transición entre estados coloreado se define como una tupla $\langle S, A, R, S_g, R_g\rangle$, donde:*

- $\langle S, A, R \rangle$ *es un sistema de transición entre estados.*
- $S_g \subseteq S$ *es el conjunto de estados permitidos.*
- $R_g \subseteq R$ *es el conjunto de transiciones permitidas.*

La clasificación de los estados como permitidos o prohibidos es demasiado simple para algunos ejemplos. Por este motivo se introduce el concepto de *graded transition system*[SC06], donde se define el conjunto de transiciones permitidas y una función de orden parcial que representa la bondad de un estado con respecto a otro en relación al cumplimento de las normas definidas en el sistema.

A partir de las definición de los sistemas de transición etiquetados se realiza un estudio de las propiedades de este tipo de sistemas, las acciones deónticas, así como su relación con las teorías causales [SC06].

2.2.3. Lógicas Temporales

La lógica temporal es un tipo lógica modal usada para describir un sistema de reglas y simbolismos para la representación y el razonamiento sobre proposiciones en las que tiene presencia el factor tiempo.

Dentro de las lógicas temporales, las más utilizadas en la representación de normas son las lógicas NTL y LTL, que se explican a continuación. Por otro lado, la lógica RTAL, aunque no se aplica directamente para la representación de normas, es la lógica empleada para la formalización de normas en nuestra propuesta (sección 4.2).

A. *Normative Temporal Logic* (NTL)

La NTL se basa en la lógica CTL, pero los operadores universales y existenciales han sido sustituidos por los operadores deónticos de obligación y permiso. A continuación se muestra una breve introducción a la lógica CTL y posteriormente se describe su extensión normativa (NTL).

***Computational Tree Logic* (CTL).** Es el sistema más representativo de la lógica temporal ramificada: la *Computational Tree Logic* (CTL)[Eme90]. La característica fundamental de esta lógica es que permite el empleo de los operadores temporales básicos combinados con los operadores de cuantificación sobre caminos.

Definición 2.2.5. *(Fórmulas Bien Formadas en CTL) El lenguaje de las fórmulas bien formadas en CTL se genera mediante la siguiente gramática:*

$$\phi ::= \bot \mid \top \mid p \mid (\neg\phi) \mid (\phi \wedge \phi) \mid (\phi \vee \phi) \mid (\phi \Rightarrow \phi) \mid (\phi \Leftrightarrow \phi) \mid AX\phi \mid EX\phi \mid AF\phi \mid EF\phi \mid AG\phi \mid EG\phi \mid A[\phi U \phi] \mid E[\phi U \phi]$$

donde:

- $\neg, \vee, \wedge, \Rightarrow$ *y* $\Leftrightarrow$ *son los operadores lógicos clásicos*
- $\bot$ *y* $\top$ *representan las constantes booleanas*
- *Los operadores temporales son los siguientes:*
  - *Cuantificadores sobre caminos*
    - *A* ϕ *- All:* ϕ *es cierto en todos los caminos desde el estado actual.*
    - *E* ϕ *- Exists: existe al menos un camino comenzando desde el estado actual en el que* ϕ *es cierto.*
  - *Operadores sobre los estados de un camino*
    - *X* ϕ *(Next):* ϕ *es cierto en el siguiente estado.*
    - *G* ϕ *(Globally):* ϕ *se tiene que cumplir en todo el camino.*
    - *F* ϕ *(Finally):* ϕ *eventualmente se cumple (en algún lugar del camino).*
    - ϕ *U* ψ *(Until):* ψ *se cumple en el estado actual o uno posterior, y* ϕ *se tiene que cumplir hasta esa posición. A partir de esa posición* ϕ *no es necesario que se siga cumpliendo.*

***Normative Temporal Logic* (NTL).** NTL extiende CTL ya que si consideramos $\eta_\oslash$ al sistema normativo vacío (aquel en el que no existe ninguna prohibición), el cuantificador universal A puede ser interpretado como $O_{\eta_\oslash}$ y muchos de los resultados y algoritmos desarrollados para trabajar con CTL pueden ser adaptados para la lógica NTL. En [gvdHRA+07] se presenta una axiomatización completa junto con la descripción semántica de la lógica NTL.

La definición formal de la lógica NTL se basa en el concepto de modelo o estructura de Kripke.

Definición 2.2.6. *(Estructura de Kripke [Kri63]) Una estructura de Kripke es una 4-tupla* $K = \langle S, S^0, R, V \rangle$*, donde:*

- S *es un conjunto finito no vacío de estados;*
- S^0 *es el conjunto de estados iniciales* ($S^0 \subseteq S$)*;*
- $R \subseteq S \times S$ *es una relación de transición entre estados;*

- *V es una función que asigna a cada estado un conjunto de variables proposicionales que son ciertas en dicho estado.*

Un *Sistema Normativo* se define como un conjunto de restricciones impuestas al comportamiento de los agentes dentro de un sistema. De una forma más precisa, un sistema normativo define, para cada transición de la estructura de Kripke, si la transición es considerada como legal o no.

Definición 2.2.7. *(Sistema Normativo) Formalmente, un sistema normativo η es un subconjunto de R (transiciones prohibidas), tal que $R \setminus \eta$ es una relación total. $R \setminus \eta$ denota las transiciones legales en η;de modo que esta condición de totalidad implica que para cada estado existe al menos una transición legal.*

La sintaxis de NTL viene definida por la siguiente gramática:

$$\varphi ::= \top \mid p \mid \neg\varphi \mid \varphi \vee \varphi \mid P_\eta \bigcirc \varphi \mid P_\eta(\varphi U \varphi) \mid O_\eta \bigcirc \varphi \mid O_\eta(\varphi U \varphi)$$

Por lo que respecta a la semántica de la lógica NTL, en [gvdHRA+07] se presenta una axiomatización completa junto con su descripción semántica.

En la práctica la representación explícita de los sistemas normativos mediante el empleo de estructuras de Kripke se hace impracticable debido al problema conocido como *state explosion problem*[Val98]. Por este motivo, en [AdHRA+07] se propone el lenguaje *Simple Reactive Modules Language* (SRML) para su presentación simbólica. Con este lenguaje las estructuras de Kripke (K) se implementan con una serie de agentes representados simbólicamente mediante el lenguaje SRML.

El problema del *Model Checking*[CGP99] constituye un problema computacional a tener en cuenta para cualquier lógica modal o temporal. En este caso concreto hay que considerar dos alternativas para el problema del *model checking* para la lógica NTL, en función de si ésta se representa de forma *explícita* (con una estructura de Kripke) o *simbólica* (con el lenguaje SRML).

En el caso de los sistemas normativos, la cuestión del *model checking* consiste en dos problemas: 1) el problema denominado *interpretado*, que consiste en dada una estructura de Kripke, un sistema normativo definido sobre ella y una formula NTL se debe comprobar si la fórmula es cierta; y 2) el problema *no-interpretado*, consistente en dada una estructura de Kripke y una formula NTL se debe determinar si existe un sistema normativo para el cual la fórmula sea cierta.

La tabla 2.1 muestra la complejidad [CC95] de resolver estos problemas en función del tipo de representación del modelo.

	Interpretado	No Interpretado
Explícita	P - completo	NP - completo
Simbólica	PSPACE - completo	EXPTIME - duro

Tabla 2.1: Complejidad del problema del *model checking* para la lógica NTL.

B. *Linear-Time Temporal Logic* (LTL)

La *Linear-Time Temporal Logic* (LTL) es una lógica que permite expresar propiedades respecto al orden en que ocurre una serie de eventos. La palabra "temporal", por tanto, no hace referencia al instante de tiempo en que ocurre un evento sino a la ordenación en el tiempo de los eventos.

En [ADM07] se propone la expresión de las normas mediante el empleo de la lógica temporal lineal (LTL).

Definición 2.2.8. *(Modelo LTL) Formalmente un modelo en LTL se define como* $M = \langle Agents, A, W\rangle$ *donde:*

- *Agents es un conjunto de agentes;*
- *A es el conjunto de acciones que los agentes pueden llevar a cabo en el sistema*
- $W = \langle \eta_1, \eta_2, \eta_3, ...\rangle$ *donde cada estado* $\eta_i = (v_i, N_i)$ *indica:*
  - v_i *es la función de evaluación para las proposiciones de LTL*
  - N_i : $Agents \rightarrow A$ *es una función que asigna a cada agente la acción que lleva a cabo en el estado i.*

La semántica de las fórmulas de la LTL se detalla a continuación.

Definición 2.2.9. *(Semántica LTL) Para cada modelo* $M = \langle Agents, A, W\rangle$*, fórmulas A y B,* $a \in Agents$*,* $\alpha \in A$ *y* $s, t, u, v \in N_0$*:*

$$
\begin{array}{rcl}
M, s \models A & \Leftrightarrow & v_s(A) = true \\
M, s \models \text{NOT } A & \Leftrightarrow & M, s \not\models A \\
M, s \models A \text{ AND } B & \Leftrightarrow & M, s \models A \wedge M, s \models B \\
M, s \models \text{DO}_a(\alpha) & \Leftrightarrow & N_s(a) = \alpha \\
M, s \models \text{DONE}_a(\alpha) & \Leftrightarrow & N_u(a) = \alpha \text{ para algún } u < s \\
M, s \models \text{NEXT } A & \Leftrightarrow & M, s+1 \models A \\
M, s \models \text{ALWAYS } A & \Leftrightarrow & \forall t \geq s : M, t \models A \\
M, s \models A \text{ UNTIL* } B & \Leftrightarrow & \text{si } \exists t \geq s : M, t \models B \\
 & & \text{entonces } \forall s \leq u < t : M, u \models A \\
 & & \text{sino } \forall v \geq s : M, v \models A
\end{array}
$$

Mediante esta lógica es posible capturar la semántica de los operadores deónticos de obligación y prohibición tal y como se muestra a continuación.

Definición 2.2.10. *(Obligaciones) Sea M un modelo, $a \in Agents$, $s,t,u,w \in N_0$ y P, D fórmulas de la LTL, se define:*
$M, s \models$OBLIGED$(a, P,$BEFORE $D) \Leftrightarrow_{def}$
$\exists t \geq s : M, t \models D$ AND
$(\forall s \leq u < t : M, u \models$NOT $viol(a, P, D))$ AND
$((\exists s \leq v < t : M, v \models P$ AND $M, v \models$ALWAYS NOT $viol(a, P, D))$ OR
$(\forall s \leq w < t : M, w \not\models P$ AND $M, t \models viol(a, P, D)))$

La obligación (OBLIGED$(a, P,$BEFORE $D)$) de un agente a de hacer cierto un predicado P antes de un determinado deadline D se define como: sea t el estado en el que el deadline se satisface $(M, t \models D)$, entonces para todo instante u anterior a t la norma aún no se puede definir como violada $(M, u \models$NOT $viol(a, P, D))$. Si el agente cumple la norma, entonces existirá un instante v anterior a t para el que la condición P sea cierta $(M, v \models P)$.

Definición 2.2.11. *(Prohibiciones) Sea M un modelo, $a \in Agents$, $s,t,u,w \in N_0$ y P, D fórmulas de la LTL, se define:*
$M, s \models$FORBIDDEN$(a, P,$BEFORE $D) \Leftrightarrow_{def}$
IF $\exists t \geq s : M, t \models D$ THEN
$(((\exists s \leq u < t : M, u \models P$ AND $M, u \models$NEXT $viol(a, P, D))$ AND
$(\forall s \leq w < u : M, w \not\models P$ AND $M, w \models$NOT $viol(a, P, D)))$ OR
$(\forall s \leq v < t : M, v \not\models P$ AND $M, v \models$NOT $viol(a, P, D)))$AND
$M, t \models$ALWAYS NOT $viol(a, P, D)))$
AND IF$\forall t \geq s : M, t \not\models D$ THEN $\forall u \geq s :$ IF $M, u \models P$ THEN
$M, u \models$NEXT $viol(a, P, D)$

Esta propuesta se completa con un mecanismo que permite a los agentes generar protocolos de actuación que respeten las reglas de la organización [ADM07]. El funcionamiento del mecanismo propuesto es el siguiente: a partir de las normas expresadas en LTL se construye un autómata; de dicho autómata se obtienen los *landmark patterns* u orden en que los estados deben ocurrir; estos patrones se someten a un proceso de restricción; finalmente estos estados ordenados se concretizan en una secuencia de acciones, mediante el empleo de un planificador, dando como resultado un protocolo consistente con las normas vigentes en la organización.

C. *Real-Time Agent Logic* (RTAL)

Para la formalización de las normas se ha optado por emplear la lógica *Real-Time Agent Logic* (RTAL). Dicha lógica combina la lógica temporal CTL, que se introdujo en la sección A , con la *Lógica Temporal de Acciones* (TLA). A continuación se introduce la teoría de la TLA. Posteriormente se describe la lógica RTAL.

Lógica Temporal de Acciones (TLA). La TLA [Lam94] es una lógica que permite razonar acerca de sistemas reactivos y concurrentes.

La TLA combina dos lógicas: una lógica de acciones, que es un lenguaje para la definición de predicados, funciones de estado y acciones; y una lógica temporal estándar.

Definición 2.2.12. *(Sintaxis de la TLA) La especificación BNF que permite definir expresiones en TLA es:*

$$\begin{aligned}
\langle generalFormula\rangle &::= \langle formula\rangle \mid \exists\langle variable\rangle : \langle generalFormula\rangle \\
&\quad \mid \langle generalFormula\rangle \wedge \langle generalFormula\rangle \\
&\quad \mid \neg\langle generalFormula\rangle \\
\langle formula\rangle &::= \langle predicate\rangle \mid \Box[\langle action\rangle]_{\langle stateFunction\rangle} \mid \neg\langle formula\rangle \\
&\quad \mid \langle formula\rangle \wedge \langle formula\rangle \mid \Box\langle formula\rangle
\end{aligned}$$

La TLA es una Lógica de Primer Orden Temporal. Sus fórmulas se construyen mediante:

- Los predicados usuales del cálculo de predicados:
  - Símbolos de variables.
  - Predicados de acción ($\langle action\rangle$) que son expresiones booleanas que contienen términos constantes, variables y variables que hacen referencia al siguiente estado.
  - Símbolos de predicados ($\langle predicate\rangle$) que son acciones sin variables referentes al siguiente estado.
  - Símbolos de función ($\langle stateFunction\rangle$), son expresiones no booleanas que contienen constantes como variables.
  - Los operadores clásicos $\neg$, $\wedge$, $=$.
- Los símbolos especiales $\Box$, $'$(representa el valor de una variable en el siguiente estado) y $\exists$.

En TLA los *algoritmos* pueden interpretarse como reglas que especifican secuencias válidas de estados. Un *comportamiento* es una secuencia infinita de estados. Por otro lado, una *acción* es una expresión booleana formada por variables, variables primas y símbolos constantes o invariantes. La expresión $x' = x+1$ es una acción donde x es una variable. La variable prima x' denota el valor de la variable x en el siguiente estado. A continuación se muestra la especificación de un sistema en el cual la variable x, que inicialmente tiene el valor 0, se incrementa (ϕ):

$Init_\phi$	$\equiv$	$(x = 0)$	Inicialmente, x es igual a 0.
M	$\equiv$	$x' = x + 1$	Siempre el valor de x en el siguiente estado (x') es igual a su valor en el estado actual más uno.
ϕ	$\equiv$	$Init_\phi \wedge \Box[M]_x$	

La fórmula $Init_\phi$ especifica el valor inicial de la variable. La acción M denota cómo cambia el sistema de un estado a otro. Por último la fórmula ϕ expresa el requisito de que el sistema siempre ejecuta una M-acción.

Las expresiones de la forma $[A]_f$ donde A es una acción y f es una función de estado se definen como:

$$[A]_f \equiv A \vee (f' = f)$$

donde f' denota el valor de f en el siguiente estado. De modo que un paso satisface la expresión $[A]_f$ sii se da la acción A o si el valor de f permanece inalterado. En el ejemplo anterior, $[M]_x \equiv [x' = x+1]_x$ permite aquellos pasos entre estados donde el valor de la x se incrementa en uno o la x permanece con el mismo valor.

Por último, las expresiones de la forma $\exists x : F$, donde F es una fórmula TLA, se satisfacen por un comportamiento sii existe alguna asignación de x para producir un comportamiento que satisfaga F.

***Real-Time Agent Logic* (RTAL)** Para la representación de restricciones se ha optado por el empleo de la lógica *Real-Time Agent Logic* (RTAL) propuesta en [Reb04].

La RTAL consiste básicamente en el empleo de la extensión temporal de la CTL con la lógica TLA. La CTL ha sido empleada tradicionalmente para la representación de sistemas concurrentes y para la especificación de los agentes inteligentes. Por otro lado, la TLA permite la representación de acciones. De entre las distintas variantes para la representación de las acciones la TLA ofrece una gran expresividad y además existe una gran cantidad de algoritmos y métodos para la comprobación de propiedades relacionadas con las acciones.

Definición 2.2.13. *(Sintaxis RTAL) Sean p un símbolo de predicado y a, b términos que se evalúan a un valor entero; la sintaxis de RTAL viene dada por la siguiente gramática:*

$$\begin{array}{rcl} (S1-S6)\ \phi & ::= & \mathsf{true} \mid p \mid \neg\phi \mid \phi \wedge \phi \mid \mathsf{A}\,\gamma \mid \mathsf{E}\gamma \\ (P1-P7)\ \gamma & ::= & \phi \mid \phi\,\mathsf{U}\,\phi \mid \mathsf{X}\,\phi \mid \phi\,\mathsf{U}^{[a,b]}\,\phi \mid \\ & & \mathsf{X}^{[a,b]}\,\phi \end{array}$$

El lenguaje de RTAL se genera a través de las fórmulas de estado (S1-S6) y de camino (P1-P5). El resto de expresiones con las conectivas y operadores temporales restantes pueden obtenerse como abreviaciones:

$$\begin{array}{rcl} \mathsf{A}\Diamond q & \equiv & \mathsf{A}(\mathsf{true}\,\mathsf{U}\,q) \\ \mathsf{E}\Diamond q & \equiv & \mathsf{E}(\mathsf{true}\,\mathsf{U}\,q) \\ \mathsf{A}\Box q & \equiv & \neg\mathsf{E}\Diamond\neg q \\ \mathsf{E}\Box q & \equiv & \neg\mathsf{A}\Diamond\neg q \\ \mathsf{AX}q & \equiv & \neg\mathsf{EX}\neg q \end{array}$$

Una fórmula de RTAL se interpreta sobre una estructura temporal $M = (S, R, V)$ donde S es un conjunto de estados, R es una relación binaria total sobre S y V es una función de evaluación de proposiciones sobre estados. Se define el conjunto de caminos completos de M a partir del estado s_0 como $\Pi(M, s_0) = \{(s_0, s_1, s_2, ...) |\ \forall i\,(s_i, s_{i+1}) \in R\}$.

Definición 2.2.14. *(Semántica RTAL) Sea M una estructura temporal como la anterior, s un estado y $x = (x_0, x_1, x_2, ...)$ un camino completo. La*

semántica de RTAL se define a continuación:

$$
\begin{array}{lll}
(S1)\ (M,s) & \models & \mathsf{true} \\
(S2)\ (M,s) & \models & p \textit{ iff } p \in V(s) \\
(S3)\ (M,s) & \models & \neg\phi \textit{ iff } (M,s) \not\models \phi \\
(S4)\ (M,s) & \models & \phi \wedge \phi' \textit{ iff } (M,s) \models \phi \wedge (M,s) \models \phi' \\
(S5)\ (M,s) & \models & \mathsf{A}\gamma \textit{ iff } \forall\, x \in \Pi(M,s) \cdot (M,x) \models \gamma \\
(S6)\ (M,s) & \models & \mathsf{E}\gamma \textit{ iff } \exists\, x \in \Pi(M,s) \cdot (M,x) \models \gamma \\
(P1)\ (M,x) & \models & \phi \textit{ iff } (M,x_0) \models \phi \\
(P2)\ (M,x) & \models & \phi\, \mathsf{U}\, \phi' \textit{ iff } \exists i \mid (M,x_i) \models \phi \wedge \forall j < i (M,x_j) \models \phi' \\
(P3)\ (M,x) & \models & \mathsf{X}\phi \textit{ iff } (M,x_1) \models \phi \\
(P4)\ (M,x) & \models & \phi\, \mathsf{U}^{[a,b]}\, \phi' \textit{ iff } \exists i, t \leq i \leq s \cdot (M,x_i) \models \phi \wedge \forall j, 0 \leq j < i, \\
 & & (M,x_j) \models \phi' \\
(P5)\ (M,x) & \models & \mathsf{X}^{[a,b]}\, \phi \textit{ iff } \exists i, t \leq i \leq s \cdot (M,x_i) \models \phi
\end{array}
$$

Tras introducir brevemente la sintaxis y la semántica de la lógica RTAL a continuación se describe cómo representar una norma en RTAL.

Definición 2.2.15. *(Regla en RTAL) Según la formulación desarrollada en [Reb04] una regla relativa a una acción sobre el entorno se define del siguiente modo:*

$$A\Box(\pi \rightarrow E\ \Box^{[a,b]}[Try(\alpha)]\phi)$$

donde A y E son los cuantificadores universal y existencial sobre caminos, respectivamente.

La fórmula anterior se lee como: siempre que se cumpla π, se verifica que habrá al menos un camino en el que siempre, tras la ejecución de la acción α entre los instantes de tiempo a y b se cumple que ϕ es cierta.

2.2.4. Social Expectations

En [Cra05, Cra07, CW07] se emplea el concepto de *social expectation* para representar las normas. El término *social expectation* abarca cualquier restricción presente o futura resultado de una serie de reglas que expresan las convenciones sociales a las que están sujetos los agentes. Estas convenciones sociales pueden ser consecuencia de normas establecidas, compromisos adoptados o la propia experiencia de los agentes. Más concretamente, una *expectativa* se define como una restricción acerca de la secuencia esperada de futuros estados.

En [Cra07, CW07] se presenta una variante de la lógica temporal ($hyMITL^{\pm}$) que permite la representación de expectativas mediante reglas condicionales acerca de observaciones pasadas y presentes, cuyas consecuencias imponen restricciones en los futuros estados.

A. Sintaxis del lenguaje $hyMITL^{\pm}$

Las fórmulas del lenguaje $hyMITL^{\pm}$ se definen mediante la siguiente gramática:

$$\begin{array}{rcl} \phi & ::= & p \mid \neg\phi \mid \phi \wedge \phi \mid \forall x.\phi_x \mid \mathbf{X}^{+}\phi \mid \mathbf{X}^{-}\phi \\ & & \phi \mathbf{U}_I^{+}\phi \mid \phi \mathbf{U}_I^{-}\phi \mid \mathbf{A}\phi \mid \mathbf{E}\phi \mid \downarrow^{u} x.\phi_x \mid I \\ I & ::= & (-\infty, +\infty) \mid [b, b] \mid [b, b) \mid (b, b] \mid (b, b) \\ b & ::= & a \mid +d \mid -d \end{array}$$

donde:

- p es una fórmula atómica de un lenguaje de primer orden L.
- ϕ_x denota una fórmula ϕ en la que la variable x es libre (no acotada por $\forall$ o $\downarrow$).
- $\downarrow$ es el operador de "enlace", empleado para enlazar una variable a un término o valor en el instante de tiempo actual.
- u es una unidad de selección en el operador $\downarrow$, haciendo referencia a la granularidad temporal deseada (por ejemplo años o minutos) para asignar valor a la variable x.
- a y d son términos que denotan valores absolutos y duraciones temporales, respectivamente.
- Los operadores temporales $\mathbf{U}(until)$ y $\mathbf{X}(next\ state)$ pueden aplicarse tanto en la dirección futura $(+)$ como en la dirección pasada $(-)$.
- $\mathbf{A}$ (all) y $\mathbf{E}(some)$ son los cuantificadores de caminos definidos en la lógica temporal.

B. Semántica del lenguaje $hyMITL^{\pm}$

En [Cra05] se especifica la semántica completa del lenguaje propuesto.

Definición 2.2.16. *(Semántica $hyMITL^{\pm}$)La semántica de los operadores y extensiones fundamentales presentes en el lenguaje es la siguiente:*

$\langle M, p, V\rangle \models \forall x.\phi_x$ *sii* $\forall d \in D, \langle M, p, V[d/x]\rangle \models \phi_x$
$\langle M, p, V\rangle \models \boldsymbol{X}^{+}\phi$ *sii* $\langle M, p^1, V\rangle \models \phi$
$\langle M, p, V\rangle \models \boldsymbol{X}^{-}\phi$ *sii para algún camino* $q, q^1 = p$ $\langle M, q, V\rangle \models \phi_x$
$\langle M, p, V\rangle \models{\downarrow^u} x.\phi_x$ *sii* $\langle M, p, V[floor(\tau(p), u^M)/x]\rangle \models \phi_x$

donde:

- D es el dominio de aplicación de las variables.
- M es un modelo de la forma $\langle S, <, \tau, \prec, floor\rangle$ siendo:
  - S un conjunto de estados;
  - $<$ una relación de orden total en el tiempo;
  - τ es una función de S en el dominio del tiempo;
  - $\prec$ es una relación que asocia cada estado con su predecesor; y
  - $floor$ es la función que redondea el valor de tiempo de acuerdo con la granularidad deseada.
- V es una asignación de valores a las variables del modelo, la notación $V[d/x]$ representa una asignación de valores idéntica a V con la excepción de la asignación del valor d a la variable x.
- p y q son secuencias de estados y p^i es la subsecuencia del camino p a partir del estado p_i.

C. Representación de Normas mediante el lenguaje $hyMITL^{\pm}$

El lenguaje de reglas que permite la definición de normas mediante el concepto de expectativas sociales se define del siguiente modo:

$$\mathbf{AG}^{+}\forall_{1\leq i\leq n} x_i(\phi \rightarrow \psi)$$

Las reglas definidas mediante este lenguaje se interpretan del siguiente modo: dado el estado actual y la historia de todos los estados anteriores, para cada regla se realiza un proceso de *matching* de la parte izquierda (ϕ) contra el estado actual, con lo que resulta un conjunto de instanciaciones de las partes derechas (ψ). Estas expectativas se añaden al conjunto de expectativas actuales y se comprueba cuáles de ellas se han satisfecho o violado. Todas aquellas expectativas que no pueden ser evaluadas en un momento dado se mantienen activas en el siguiente estado.

Además se propone un algoritmo para ser empleado en tiempo de ejecución dentro de un SMA o individualmente por los agentes para monitorizar cuándo se generan, satisfacen o violan las expectativas. Este algoritmo también puede ser empleado individualmente por los agentes para monitorizar las expectativas del resto de agentes.

2.3. Conclusiones

A lo largo de este capítulo se han introducido los formalismos más importantes empleados para la representación formal de las normas y las proposiciones normativas.

Los primeros trabajos en la representación de normas establecieron los fundamentos sintácticos y semánticos de la lógica deóntica. En este sentido los trabajos descritos en la sección 2.1 permiten definir los derechos, deberes y responsabilidades de los agentes en base a permisos, obligaciones y prohibiciones. Sin embargo, los SMA necesitan de un formalismo lógico que permita representar las normas de forma explícita teniendo en cuenta el sistema normativo al que pertenecen.

En la sección 2.2 se han descrito las propuestas lógicas relacionadas con la formalización de las denominadas proposiciones normativas. Estos trabajos amplían las propuestas realizadas para la formalización de normas permitiendo la representación del sistema normativo en sí. Para cada una de las alternativas propuestas se ha ofrecido una visión general de su sintaxis e interpretación. Lógicamente el problema de la formalización de las proposiciones normativas es un problema mucho más complejo que la mera representación de las normas. La evaluación de la utilidad de cada una de las propuestas debe considerar dos aspectos fundamentales: su capacidad expresiva y la posibilidad de razonar sobre las propiedades del sistema. Como ya se ha comentado todas las propuestas analizadas disponen de algoritmos y métodos para analizar y razonar acerca del sistema normativo en sí. Sin embargo podemos destacar dos formalismos por encima de los demás: la lógica NTL y la lógica RTAL. Para la primera propuesta se han presentado numerosos e interesantes trabajos relacionados con el estudio de las propiedades de los sistemas normativos expresados mediante este formalismo [ÅvdHW07, ÅvdHW08]. Por otro lado la lógica RTAL, que está basada en la lógica de acciones TLA, ha sido la alternativa empleada para realizar la formalización de las normas en nuestra propuesta de modelo (sección 4.2). Los motivos para la selección de la RTAL es que la lógica TLA ofrece una gran expresividad y además existe una gran cantidad de algoritmos y métodos para la comprobación de propiedades relacionadas con las acciones.

Tras ofrecer una alternativa para la definición de los SMA Normativos se plantea el problema de cómo razonar sobre ellos. Razonar sobre el sistema normativo implica tanto considerar las propiedades de adecuación y consistencia del sistema en sí, como el razonamiento individual de los agentes cuyas interacciones están reguladas por las normas del sistema. El siguiente capítulo aborda con detalle la implementación de los SMA normativos.

Capítulo 3

Implementación de Sistemas Multiagente Normativos

Como se ha visto en los primeros capítulos del presente documento, existen distintos trabajos que describen diferentes aspectos de las normas desde un punto de vista más teórico o formal. Sin embargo estos modelos no ofrecen la posibilidad de representar computacionalmente los distintos tipos de normas ni, por lo tanto, de proporcionar mecanismos que permitan a los agentes razonar y cooperar dentro de SMA Normativos.

Recientemente, están surgiendo estudios centrados en una visión más operacional de las normas, definiendo cómo se puede dar una interpretación computacional a las normas y cómo la teoría normativa puede ser incluida dentro del diseño y ejecución de los SMA.

Como ya se ha comentado anteriormente las sociedades están reguladas por normas y, consecuentemente, los agentes que deseen formar parte de dicha sociedad necesitarán ser capaces de razonar sobre dichas normas. Sin embargo, ningún proceso de razonamiento podrá llevarse a cabo por los agentes si previamente no se define un modelo de normas. En esta sección se introducen las propuestas más relevantes realizadas hasta el momento con el objetivo de dotar a los agentes de un modelo normativo que les permita conocer cuáles son los efectos de las normas y, por lo tanto, conocer la influencia de las mismas sobre la consecución de sus objetivos.

3.1. Lenguaje Normativo

Existen numerosos trabajos que presentan diferentes lenguajes normativos para la representación de restricciones en el comportamiento de los agentes. En [VSAD04] se propone un lenguaje normativo de propósito general. A

continuación se muestra un resumen de la gramática de este lenguaje:

<norm>:= <deontic_clause>(a, <situation> ⟨IF <proposition>⟩)
<deontic_clause>:= OBLIGED | PERMITTED | FORBIDDEN
<situation>:= <proposition> | DO <action> |
<proposition> <temporal_clause> deadline |
DO <action> <temporal_clause> deadline
<temporal_clause>:= BEFORE | AFTER

De modo que una norma define un control deóntico (<deontic_clause>) sobre un agente (a) y controla una situación (<situation>). Estas situaciones pueden definirse sobre una condición(<proposition>) o una acción (DO <action>). Una proposición representa una condición expresada en alguna lógica proposicional. Además las normas pueden definir condiciones para su activación (IF <proposition>) así como restricciones temporales (<temporal_clause>).

Este lenguaje ha sido empleado como referencia para otras propuestas de lenguajes normativos. Por ejemplo, en [GCNRA05] se ha realizado una extensión de esta propuesta con el objetivo de definir normas dentro de las *Instituciones Electrónicas* (IE). Las IE son un método de modelado del entorno en el que interactúan los agentes; es decir, restringen el comportamiento de los agentes para asegurar que éstos interactúen en condiciones válidas. Los dos componentes fundamentales de las IE son las *ilocuciones* y las *escenas*. Las *escenas* son especificaciones de protocolos multiagente basados en roles. Una escena define secuencias válidas de interacciones entre agentes que adoptan diferentes roles. Por otro lado, las acciones que los agentes pueden realizar dentro de una IE son actos de habla o *ilocuciones* (acciones que se realizan diciendo algo)[Est02]. Por lo tanto, la propuesta de lenguaje normativo realizada en este trabajo incluye la posibilidad de expresar normas con restricciones temporales así como la posibilidad de expresar condiciones o sanciones que afecten al estado institucional (los atributos observables de los agentes y su entorno). Por otro lado, como las acciones en las IE son actos de habla, las acciones expresadas mediante el nuevo lenguaje propuesto están limitadas a la pronunciación de ilocuciones.

En [dS07] se realiza una extensión de la propuesta anterior con la noción de acciones *no dialógicas*. Las acciones no dialógicas son aquellas que no están relacionadas con interacciones entre los agentes. En este trabajo se propone una extensión del lenguaje normativo para tener en consideración acciones no dialógicas en la especificación de las normas.

En esta sección se han descrito diferentes propuestas de lenguajes normativos. La existencia de estos lenguajes normativos permite la definición formal

de restricciones sobre el comportamiento de los agentes. A continuación se describe un trabajo que aborda aspectos operacionales de las normas.

3.2. Aspectos Operacionales de las Normas

En [VSAD04] los autores hacen un estudio sobre los aspectos *operacionales* de las normas que deben ser incorporados en las plataformas de agentes para facilitar la implementación de las normas. Básicamente este trabajo se centra en los mecanismos de imposición (*enforcement*) y aspectos de verificabilidad que están directamente relacionados con la imposición.

A continuación se comentan brevemente las guías de implementación relacionadas con los distintos tipos de normas y los componentes de cada una de ellas.

3.2.1. *Enforcement* de las Normas

Los mecanismos de *enforcement* están muy relacionados con los *addresses*(destinatarios) de la norma. En función del *addresse* es posible distinguir cuatro tipos de normas:

- **Normas que afectan a entidades fuera del alcance del sistema**. En este caso no es posible emplear ningún mecanismo de *enforcement*. Un ejemplo de este tipo de norma sería la obligación de un usuario de proporcionar información verdadera sobre su persona.

- **Normas que afectan a agentes externos**. Este grupo debe ser altamente controlado y es para el cual tienen más sentido los mecanismos de *enforcement*. De los agentes externos no podemos conocer sus estados mentales, tan sólo se puede observar su comportamiento en términos de mensajes (públicos) y acciones (visibles). En este caso el *enforcement* depende de la verificabilidad de los predicados y las acciones que están presentes en estas normas.

- **Normas sobre agentes internos.** Los mecanismos en este caso son similares a los anteriores con la salvedad de que los agentes son internos y se tiene un acceso y control relativo sobre sus estados mentales.

- **Normas sobre agentes *enforcers***. Estas normas afectan a un grupo de agentes internos especiales encargados de velar por el cumplimiento de las normas en última instancia. En este caso las normas deben estar definidas de forma explicita en el código de los agentes *enforcer*.

No sólo el *addresse* de las normas sino también los elementos presentes en la expresión de la norma afectan al *enforcement*. La implementación del *enforcement* está compuesta por tres procesos relacionados: i) detección de cuándo una norma está activa, ii) detección de las violaciones, y iii) manejo de las violaciones. A continuación se ofrece una caracterización de las normas en función de si i) se refieren a un estado o acción, ii) son condicionales, iii) incluyen un *deadline*, o iv) conciernen a otras normas. Estas características están muy relacionadas con los procesos de detección de la activación y violación, como se verá a continuación.

- **Normas que se refieren a un estado o acción.** En este caso se consideran normas sin restricciones temporales; de modo que este tipo de normas estarán activas en todo momento.
  - Norma concerniente a una *condición* o predicado P. Para determinar cuándo se produce una infracción de la norma es necesario comprobar si P se satisface. En general esta comprobación es indecidible.
  - Norma concerniente a una *acción* A.
    - No pueden existir obligaciones incondicionales sobre acciones ya que esto implicaría que el agente debe ejecutar dicha acción todo el tiempo.
    - Para el caso de normas incondicionales de permiso es necesario comprobar que el agente que ejecuta una acción juega el rol adecuado.
    - Para las normas de prohibición la acción abstracta A se traduce en las acciones concretas α y se detecta la ocurrencia de α. La implementación de este mecanismo de detección se basa en la creación de una lista negra de acciones y un trigger o alarma que se dispara cada vez que se ejecuta una acción de la lista.
- **Normas condicionales.** En el caso de normas condicionales (C) es necesario detectar la activación (cuándo la condición C se satisface) de la norma y la desactivación (cuando el predicado P o la acción A se satisfacen o la condición C pasa a ser falsa). Otro aspecto importante es la consideración del tiempo necesario entre la activación y desactivación para que los agentes puedan reaccionar y cumplir la norma.
  - En el caso de las normas de obligación el mecanismo de *enforcement* dependerá de la verificabilidad de C (detección de la activación) y la verificabilidad de P o A (detección de la desactivación).

  - Para las normas de permiso o prohibición el orden de comprobaciones cambia: primero se detecta la ocurrencia de P o A y entonces se comprueba la condición C.

- **Normas temporales.** Un tipo de norma condicional es aquella en la cual la condición hace uso de operadores temporales (BEFORE, AFTER).

  - En el caso de permisos y prohibiciones el procedimiento es similar a las normas condicionales: primero se detecta la ocurrencia de la acción o el predicado y entonces se comprueban las restricciones temporales.
  - En el caso de las normas de obligación el *deadline* debe ser comprobado antes de la ocurrencia de la acción o predicado. Los *deadline* no son sencillos de implementar, la solución propuesta por los autores consiste en incluir en la plataforma un mecanismo *clock trigger* que envía una señal cuando termine un *deadline*.

- **Normas sobre otras normas.** Cuando este tipo de normas están dirigidas a un agente interno de la plataforma el mecanismo de *enforcement* se implementa como un objetivo del agente. Si el destinatario de la norma es un agente externo a es necesario confiar en el agente externo para que éste ejecute las sanciones y reparaciones necesarias; o disponer de un agente superior encargado de controlar al agente a.

3.2.2. Aspectos de Verificabilidad

Como ya se ha indicado con anterioridad, la detección de las violaciones de las normas depende de la posibilidad de realizar las comprobaciones necesarias. Las propiedades más importantes de estas comprobaciones son: a) si las comprobaciones son *verificables* y b) si las restricciones son *computacionales* (si pueden ser comprobadas en cualquier momento con un consumo de recursos razonable). Atendiendo a estos dos criterios podemos distinguir los distintos tipos de normas [VSAD04]:

- Normas computacionalmente verificables: la verificación de las condiciones, predicados y acciones puede realizarse fácilmente en cualquier momento. En este caso la verificación puede realizarse cada vez que se necesite.

- Normas que no son computacionalmente verificables pero lo pueden ser mediante el empleo de recursos extras: este tipo de normas requiere la

inclusión de mecanismos y estructuras de datos para su comprobación. Las alarmas y *clock triggers* son ejemplos de estos mecanismos.

- Normas no computacionalmente verificables: para su comprobación son necesarios demasiados recursos o tiempo y por ello no se puede realizar en cualquier momento. La verificación de la norma no se realiza en todo momento sino que se restringe a unos determinados instantes de tiempo como pueden ser: cuando el sistema no está demasiado ocupado, de forma periódica o mediante comprobaciones aleatorias.

- Normas no verificables. A este tipo de normas pertenecen aquellas que no son decidibles, o no son observables. Lógicamente estas comprobaciones no se pueden delegar en ningún elemento del sistema.

3.3. Implementación de Normas

En [GCRASV06] se presenta un formalismo basado en reglas que permite la definición de restricciones en el comportamiento de los agentes así como la programación *norm-oriented* de las Instituciones Electrónicas. Este formalismo es concebido por los autores como un "lenguaje máquina" sobre el cual se pueden construir lenguajes normativos de mayor nivel.

3.3.1. Características Deseables del Lenguaje

El objetivo fundamental de este trabajo es producir un lenguaje que soporte la especificación de un mecanismo de coordinación en SMA a través de la definición de normas. Las características deseables de este lenguaje son:

1. El lenguaje debe permitir la *gestión y definición explícita de las proposiciones normativas*, es decir, de las obligaciones, permisos y prohibiciones de los agentes.

2. Debe ser de *propósito general*, de forma que se permita la especificación del mayor número de SMA normativos.

3. Este lenguaje se propone como un "lenguaje máquina" sobre el cual se desarrollen lenguajes normativos de más alto nivel. Por este motivo es necesario que el lenguaje soporte la definición de los diferentes conceptos normativos definidos. Además el lenguaje no sólo debe proporcionar suficiente expresividad, sino ofrecer *patrones de diseño* que guíen y faciliten el diseño de SMA.

4. El lenguaje debe ser *declarativo* y disponer de un mecanismo de ejecución. Otra de las ventajas de la naturaleza declarativa del lenguaje es el permitir la verificación de las propiedades de las especificaciones.

3.3.2. Aproximación Basada en Reglas

Los autores proponen un lenguaje de reglas que representa cómo cambian las proposiciones normativas a medida que los agentes interactúan. Estas reglas son de la forma $LHS \rightarrow RHS$, donde LHS contiene una representación del estado que produce la activación de la norma. RHS contiene las actualizaciones o cambios que se producen en el estado actual. Las actualizaciones posibles son la adición($\oplus$) o eliminación($\ominus$) de hechos o fórmulas.

Esta propuesta adopta la semántica usual de las reglas de producción. Cada regla se aplica de forma exhaustiva mediante un proceso de *matching* de la LHS contra el estado actual; empleando los valores de las variables obtenidos para instanciar la RHS. La imagen 3.1 muestra cómo este proceso de aplicación de reglas se produce a medida que los agentes intercambian mensajes. El diagrama muestra cómo desde un estado inicial Δ_0 se genera otro estado Δ_1. En el estado Δ_0 existe un conjunto de agentes $ag_1, ..., ag_n$ que intercambian un conjunto de mensajes $\alpha_1^0, ..., \alpha_n^0$ (esto se representa por $\updownarrow$). Tras el intercambio de mensajes las reglas son aplicadas exhaustivamente (representado por $\overset{*}{\rightsquigarrow}$) teniendo como resultado el estado Δ_1.

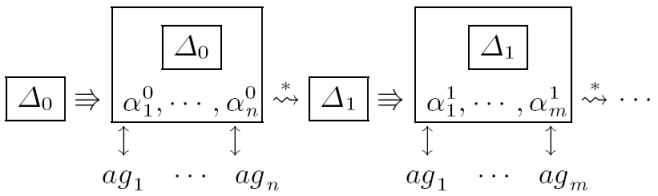

Figura 3.1: Semántica de las reglas

3.3.3. Aplicación a las Instituciones Electrónicas

Este trabajo extiende el trabajo realizado en [Est02] sobre las *instituciones electrónicas* (IEs) proporcionando un nivel normativo explícito. Básicamente su propuesta consiste en traducir las normas expresadas en el lenguaje normativo a reglas Jess [GCNRA05]:

- *Normas condicionales:* este tipo de normas se traducen a reglas Jess poniendo las condiciones dentro de la parte izquierda de la regla (LHS).

- *Normas dependientes de otras acciones o del tiempo:* este tipo de normas son aquellas que tienen una expresión del tipo BEFORE, AFTER o BETWEEN seguida de una acción o una expresión temporal.
  - BEFORE: Las acciones que deben realizarse antes de una determinada condición se implementan mediante una regla que inserta un hecho de tipo violación si se cumple la condición y la acción no se ha llevado a cabo. Por otro lado los permisos o prohibiciones que están activos antes de la condición son añadidos inicialmente a la base de hechos y se define una nueva regla que los elimina cuando se cumple la condición.
  - AFTER: Para las normas activas después de una condición se crea una nueva regla que añade la norma cuando la condición se satisface.
  - BETWEEN: Por último, las normas que están activas durante un intervalo de tiempo se descomponen en dos reglas Jess como si estuvieran formadas por una condición de tipo BEFORE y otra AFTER.

A. Ejemplos de Normas

Estos predicados representan los operadores deónticos pero no sus relaciones. Las relaciones entre ellos se capturan mediante las reglas definidas, como se indica en el apartado anterior. A continuación se muestra un ejemplo de regla:

$$\begin{pmatrix} att(S,W,I)\wedge \\ per(S,W,I)\wedge\neg prh(S,W,I) \end{pmatrix} \rightsquigarrow \begin{pmatrix} \oplus att(S,W,I)\bullet \\ \ominus utt(S,W,I) \end{pmatrix}$$

Esta regla representa que si cualquier agente trata de pronunciar una ilocución válida, es decir, que no está prohibida y está permitida, entonces este intento de ilocución se convierte en una ilocución pronunciada.

La siguiente regla muestra cómo especificar la sanción de una regla:

$$\begin{pmatrix} att(S,W,inform(A_{g1},A_{g2},info(C)))\wedge \\ prh(S,W,inform(A_{g1},A_{g2},info(C)))\wedge \\ oav(A_{g1},rep,V_{Rep})\wedge(V'_{Rep}=V_{Rep}-10) \end{pmatrix} \rightsquigarrow \begin{pmatrix} \oplus oav(A_{g1},rep,V'_{Rep})\bullet \\ \ominus oav(A_{g1},rep,V_{Rep}) \end{pmatrix}$$

La anterior regla representa que cuando un agente A_{g1} intenta revelar una información secreta C a otro agente A_{g2}, A_{g1} es sancionado mediante un cambio en el valor de sus atributos. En este caso la sanción supone reducir la reputación del agente mediante un cambio en el valor del atributo *rep*.

3.3.4. Evolución del Lenguaje: Reglas *Event-Condition-Action*

Para finalizar con las propuestas de lenguajes normativos en IE comentaremos el trabajo presentado en [GC07]. En este artículo se presenta una evolución del lenguaje de reglas anteriormente comentado con la noción de *evento*.

La idea básica consiste en ofrecer un formalismo que permita la gestión y regulación de los eventos generados concurrentemente por los agentes en un SMA. El lenguaje de reglas permite expresar los efectos de los eventos así como representar las situaciones en las cuales estos deben ser ignorados, forzados, esperados o sancionados. A continuación se muestran las reglas principales del lenguaje propuesto:

$$\begin{aligned} ECA-Rule &:= \texttt{on}\ events\ \texttt{if}\ conditions\ \texttt{do}\ actions \\ if-Rule &:= \texttt{if}\ conditions\ \texttt{do}\ actions \\ ignore-Rule &:= \texttt{ignore}\ events\ \texttt{if}\ conditions \\ prevent-Rule &:= \texttt{prevent}\ conditions\ \texttt{if}\ conditions \\ force-Rule &:= \texttt{force}\ events\ \texttt{on}\ events\ \texttt{if}\ conditions\ \texttt{do}\ actions \end{aligned}$$

Este lenguaje permite la representación de normas no sólo en términos de acciones permitidas, obligadas o prohibidas, sino cuál es la semántica computacional de esas restricciones. Por ejemplo, en el caso de una prohibición permite determinar si la prohibición implica que la acción debe ser evitada por el sistema o por el contrario si los agentes que la lleven a cabo serán sancionados por ello.

3.4. Proporcionando Información Normativa a los Agentes

En [FCBL06, FCB+08] los autores proponen una alternativa, denominada DynaCROM, para permitir que los agentes que se encuentran dentro de SMA Abiertos puedan mantener de forma automática información normativa actualizada. Esta propuesta no es un mecanismo para forzar el cumplimiento de las normas. Más bien es una herramienta para mantener informados a los agentes de las normas que les afectan, pero éstos mantienen su autonomía para decidir adoptar las normas.

La propuesta DynaCROM consiste en: (i) seguir una estrategia *top-down* para el modelado contextual de las normas, (ii) emplear una ontología para representar la semántica de las normas y (iii) utilizar un mecanismo de inferencia para determinar las normas vigentes en un contexto dado.

A. Modelado Conceptual de las Normas

En esta propuesta, las normas definen qué acciones están permitidas, obligadas y prohibidas. Estas normas son modeladas de acuerdo con los cuatro niveles de abstracción siguientes: *Entorno*, *Organización*, *Rol* e *Interacción*. Estos niveles reciben el nombre de *contextos normativos* y se diferencian en las áreas de influencia o fronteras de sus normas. Las *Normas de Entorno* afectan a todos los miembros de la sociedad normativa; las *Normas de la Organización* se aplican a todos los agentes que son miembros de la misma; finalmente las *Normas del Rol* y *Normas de Interacción* afectan a los agentes que asumen dicho rol o que participan en la interacción, respectivamente.

B. Representación de las Normas mediante una Ontología

Para permitir que los agentes heterogéneos sean capaces de procesar los contextos normativos en los que se encuentran, DynaCROM emplea ontologías para representar los contextos normativos y sus datos (normas). Esta ontología se muestra en la figura 3.2.

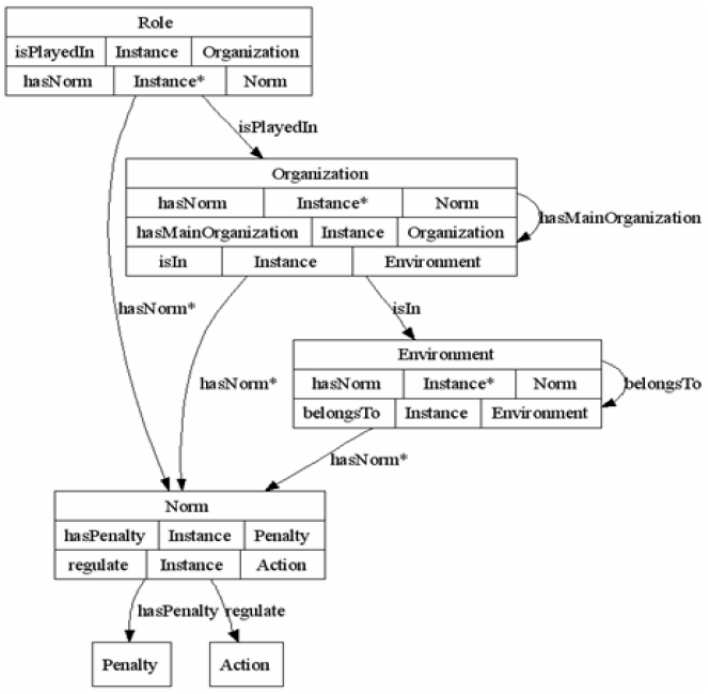

Figura 3.2: Ontología normativa propuesta en DynaCROM

C. Inferencia de Normas Contextuales

Tras la definición y organización de las normas, siguiendo una estrategia top-down, y su representación explicita a través de instancias de la ontología, DynaCROM emplea un conjunto de reglas para determinar los diferentes contextos normativos. Este proceso se resume en tres fases: en primer lugar se analiza la ontología con las normas definidas, tras esto se analizan las reglas que permiten componer los contextos normativos y por último se infiere una nueva ontología resultante de aplicar las reglas de inferencia a la ontología original.

D. Detalles de Implementación

El proceso de inferencia de normas contextuales de DynaCROM ha sido implementado en JAVA y encapsulado como un comportamiento JADE, con el objetivo de que los agentes puedan añadir este comportamiento sin necesidad de realizar ninguna implementación extra. Lo único que es necesario es extender e instanciar la ontología con los conceptos del dominio y, por otro lado, definir las reglas de composición de contextos normativos.

3.5. Conclusiones

A lo largo de este capítulo se han descrito varios trabajos relacionados con la implementación de normas que permiten el desarrollo de aplicaciones de SMA Normativos.

Estas propuestas se basan en el marco de las IE, las cuáles proporcionan un marco para la cooperación entre agentes heterogéneos. Sin embargo, no son un entorno abierto en su sentido más amplio, ya que los agentes interactúan sólamente a través de la infraestructura proporcionada por la IE. Por lo tanto, el comportamiento de los agentes externos está completamente controlado por la institución, que permite o no a los agentes pronunciar ciertas ilocuciones, controladas a través de los agentes *governors*. Por ello, las normas son impuestas a los agentes de forma automática. La mediación de la institución impide que los agentes se desvíen del comportamiento deseado.

Además, el hecho de que todas las comunicaciones se realizan a través de la IE permite una fácil aplicación y cumplimiento de las normas. En este sentido, la existencia de un *middleware*, que actúa como mediador en la comunicación entre los agentes, evita la necesidad de tomar en consideración las limitaciones que existen en entornos abiertos. El término *limitación* se refiere al hecho de que una entidad necesita información adicional y capacidades extra para actuar como supervisor o controlador de la norma. Más concretamente, esas

limitaciones están relacionadas con la detección de hechos y las capacidades adicionales necesarias para imponer normas a otros agentes.

La definición de SMA Normativos es más similar a los llamados *Partially Controlled MAS*(PAC) [RIB96]. El PAC es un SMA en el que hay dos tipos de agentes: los agentes *controlables*, que son controlados directamente por el diseñador del sistema; y los agentes *incontrolables*, que no están bajo el control directo del diseñador. Así, los agentes *incontrolables* podrán desviarse del comportamiento ideal. Como consecuencia de ello, tiene que haber un mecanismo de control para motivar a los agentes a obedecer las normas. Este control es delegado en algunos agentes controlables que actúan como supervisores normativos.

Por lo tanto, mayores esfuerzos deben hacerse a fin de proporcionar mecanismos que permitan la implementación de las normas dentro de los SMA Normativos, donde el control normativo se delega a ciertos representantes de dicho sistema. Como ya se argumentó, estos agentes tienen una capacidad limitada para hacer cumplir las normas. Por lo tanto, debe realizarse un análisis acerca de la verificación de normas en los SMA Normativos, antes de abordar el problema de la implementación de normas.

Parte II

Marco de la Propuesta

Capítulo 4

Propuesta de Modelo SMA Normativo

En la siguiente sección se propone un nuevo modelo de SMA Normativo. Dicho modelo toma como punto de partida la clasificación de las normas realizada en el capítulo 1 de este trabajo y ofrece soporte para los distintos tipos de normas.

A continuación se propone un formalismo para la definición de las restricciones o normas del modelo. Para la formalización de las normas se ha optado por emplear la lógica RTLA descrita en la sección 2.2.3-C.

4.1. Propuesta de Modelo Normativo Global [CVA09]

Tras analizar los modelos normativos existentes y detectar sus carencias (capítulo 1) realizaremos a continuación nuestra propuesta de modelo normativo. Este modelo se basa en las propuestas previas de modelos normativos y está fuertemente inspirado en los sistemas legales de las sociedades humanas [Mit06, Alc96]. A continuación se describe cada uno de sus componentes.

Definición 4.1.1. *(SMA Normativo) Un SMA Normativo (NMAS) se define como una tupla NMAS=*$\langle M, A, R, E, G, N, F, W, T\rangle$ *donde:*

- M *es un conjunto de agentes miembros del NMAS;*
- A *es un conjunto de acciones primitivas individuales. Se define* λ *como la acción nula, es decir, mediante esta acción se representa la no realización de ninguna acción;*
- R *es el conjunto de roles definidos dentro del NMAS;*

- *E es el conjunto de elementos observables del entorno;*
- *$G = \{g_0, g_1, ...\}$ es el conjunto de grupos o unidades organizativas definidas en el NMAS, donde cada $g_i \subseteq M$. Se define g_0 como una unidad básica que contiene a todos los miembros del NMAS ($g_0 = M$);*
- *N es el contexto normativo del SMA, es decir, el conjunto de normas que regulan el funcionamiento del SMA;*
- *F es el conjunto de hechos básicos. Comprende los hechos relacionados con la realización de acciones ($Do(a)$ y $Done(a)$) y los relacionados con la observación de hechos del entorno ($Obs(e)$).*
  $(\forall a \in A : Do(a) \in F \wedge Done(a) \in F) \wedge (\forall e \in E : Obs(e) \in F)$;
- *W es una secuencia infinita de estados de la forma $W = \langle w_1, w_2, ...\rangle$ donde cada estado es de la forma $w_i = (v_i, \sigma_i, R_i, G_i)$ donde:*
  - *v_i es una función de evaluación que determina los predicados ciertos en un estado para un conjunto de agentes;*
  - *$\sigma_i : M \rightarrow A$, es una función que asigna a cada agente una acción en un estado i;*
  - *$R_i : M \rightarrow 2^R$, función que asigna a cada agente los roles asumidos por ese agente en un estado i;*
  - *$G_i : M \rightarrow 2^G$, función que asigna a cada agente las unidades a las que pertenece en un estado i.*
- *$T : W \times W$, función de transición entre estados;*

Esta definición del NMAS se basa en el contexto normativo N. Dicho contexto normativo se define como:

Definición 4.1.2. *(Contexto Normativo) El contexto normativo (N) de un NMAS comprende tanto las normas impuestas por la institución como las normas definidas mediante algún contrato. Más formalmente $N = I \cup C$ donde:*

- *$I = I_{Regulative} \cup I_{Constitutive} \cup I_{Procedural}$ normas institucionales. Comprenden tanto las normas sustantivas como las normas procedurales*
- *$C = C_{Regulative} \cup C_{Constitutive} \cup C_{Procedural}$ es el conjunto de normas contractuales*

Como se puede observar el modelo de NMAS propuesto es muy similar a los modelos de SMA tradicionales con la salvedad de que dentro de él se han definido una serie de normas, tanto institucionales (I) como derivadas de la interacción entre los agentes (C), que permiten regular el comportamiento del sistema. Estas normas se clasifican en dos grandes grupos: las *normas sustantivas*, que permiten definir el comportamiento ideal del sistema; y las *normas procedurales*, que determinan el modo en el que se aplican las normas en el sistema. A continuación se detalla cómo el modelo permite definir cada uno de los tipos de normas.

4.1.1. Normas Sustantivas

Las normas sustantivas tradicionalmente se han clasificado en: *normas constitutivas*, que permiten definir el significado institucional de los hechos y acciones; y las *normas regulativas* que permiten definir el comportamiento deseado del sistema a través de permisos, obligaciones y prohibiciones.

A continuación se detalla cómo el modelo propuesto permite definir cada tipo de normas.

A. Normas Constitutivas

Los códigos legales no sólo contienen las prescripciones normativas, sino que en gran parte contienen definiciones para la determinación de categorías y hechos que son empleados en la definición de las normas. Este tipo de normas, que permiten dotar de un significado institucional a los diferentes hechos, elementos del entorno, etc., reciben el nombre de *normas constitutivas*.

Tomando como punto de partida los trabajos sobre la formalización de normas constitutivas condicionales presentados en [BvdT04, BvdT05] en este modelo proponemos el empleo del concepto de norma constitutiva para la declaración de hechos y conceptos abstractos sobre los cuales se definen las normas regulativas.

Definición 4.1.3. *(Norma Constitutiva) En general una norma constitutiva* $n_{constitutive}$ *se define como* $n_c = x$ *COUNT-AS y* $IN\ c$ *donde:*

- x *representa un concepto o conjunto de conceptos básicos. Básicamente es una expresión de la Lógica de Primer Orden (LPO) definida sobre términos pertenecientes al conjunto de hechos básicos* F.
- y *representa un concepto más abstracto.*
- c *representa el contexto en el cual la abstracción del término básico* x *en el término de más alto nivel* y *es válida. Se define como una condición*

booleana definida sobre términos pertenecientes al conjunto de hechos básicos F

- *El significado de una norma constitutiva (expresado mediante lógica dinámica) es:*
  x COUNT-AS y IN c $\equiv [c]x \rightarrow y$. *La regla representa que si la condición c es cierta en el contexto actual, el concepto abstracto y se define como x.*

Este tipo de normas se usa dentro del NMAS en los niveles de interacción e institucional para definir categorías, conceptos y hechos que permiten describir el comportamiento ideal del sistema. De modo que gracias a las normas constitutivas es posible definir dos conjuntos de hechos: los hechos institucionales ($F_{Institutional}$) y los hechos definidos dentro de un contrato entre dos agentes ($F_{Contractual}$).

Definición 4.1.4. *(Hechos Institucionales) Formalmente el conjunto de hechos institucionales* ($F_{Institutional}$) *se define como:*

$$\forall f_i \in F_{Institutional} \exists n_c \in I_{Constitutive} : n_c = x\ COUNT - AS\ f_i\ in\ c$$

Un subconjunto muy importante de las normas constitutivas institucionales está formado por aquellas que permiten definir las acciones y procesos *legislativos*, a través de los cuales el contexto legal es modificado. Como un ejemplo, la siguiente norma define el proceso por el cual una norma institucional es abolida:

$$\begin{gathered} do_a(abolish(n)) \text{ COUNT-AS } do_I(abolish(n)) \\ \text{IN } \exists i :\ \sigma_i(a) = abolish(n)\ \wedge\ LEGISLATOR \in R_i(a) \\ \wedge\ n \in I \end{gathered}$$

Esta norma establece que si un agente (a) lleva a cabo la acción correspondiente a la abolición de una norma ($abolish(n)$) , ésto se considerará como si la institución en sí misma (I) aboliera la norma sólo si el agente en cuestión es un legislador de la institución ($LEGISLATOR \in R_i(a)$) y la norma es una norma institucional ($n \in I$).

Definición 4.1.5. *(Hechos Contractuales) Formalmente el conjunto de hechos contractuales* ($F_{Contractual}$) *se define como:*

$$\forall f_i \in F_{Contractual} \exists n_c \in C_{Constitutive} : n_c = x\ COUNT - AS\ f_i\ in\ c$$

La siguiente norma define un concepto abstracto como resultado de una acuerdo entre dos agentes:

$$do_s(deliver(item, d)) \text{ COUNT-AS } done_s(fulfillContract)$$
$$\text{IN } \exists i, j : \ j < i \ \wedge \ \sigma_i(s) = deliver(item, d) \ \wedge$$
$$\sigma_j(s) = contract(item, cd) \ \wedge \ d <= cd$$

Esta norma establece que si un agente vendedor (s) lleva a cabo la acción de entrega ($deliver(item, d)$) de un item ($item$) en una fecha concreta (d) enterior o igual a la fecha de entrega acordada ($d <= cd$) entonces se considerará como un contrato satisfecho ($done_s(fulfillContract)$).

En resumen, el conjunto de normas constitutivas $N_{Constitutive}$ comprende tanto las normas constitutivas formuladas por la institución como las definidas a nivel contractual.

Definición 4.1.6. *(Normas Constitutivas) El conjunto de normas constitutivas $N_{Constitutive}$ se define del siguiente modo:*

$$N_{Constitutive} = I_{Constitutive} \cup C_{Constitutive}$$

es decir, está formado por las normas institucionales constitutivas ($I_{Constitutive}$) y las normas contractuales constitutivas ($C_{Constitutive}$).

B. Normas Regulativas

Las normas regulativas son aquellas que tomando como base los operadores deónticos de prohibición, obligación y permiso permiten definir el comportamiento ideal del sistema. El comportamiento deseado del sistema se puede especificar a varios niveles: nivel institucional, nivel de interacción y nivel privado. En este apartado comentaremos cómo formalizar las normas regulativas pertenecientes a cada uno de los tres niveles de normatividad.

- **Nivel Institucional:** las normas institucionales comprenden a aquellas sentencias normativas mediante las cuales la institución en sí misma determina los derechos y deberes de los miembros de la sociedad. En nuestra propuesta se ha distinguido entre aquellas normas generales que regulan el comportamiento de los miembros de una sociedad y las reglas específicas para los distintos grupos de agentes.

  - Las *Normas Generales* (NG) son aquellas que definen los derechos y deberes de los miembros de la sociedad.

Definición 4.1.7. *(Norma General) Una Norma General ng se define mediante la expresión γ formalizada mediante lógica deóntica que afecta al conjunto de los miembros de la sociedad. Dicha expresión γ se define en base a conceptos definidos mediante normas institucionales constitutivas ($F_{Institucional}$).*

Un ejemplo de norma general se muestra a continuación:

$$FORBIDDEN\ kill \Leftrightarrow_{def}$$
$$[\forall a \in M\ \wedge\ \forall i \in \mathbb{N}\ \wedge\ \sigma_i(a) = kill]$$
$$V(a, kill, i)$$

En concreto, esta norma define que está prohibido matar dentro de la organización. Su definición mediante la lógica deóntica determina que si cualquier agente (a) lleva a cabo la acción de matar ($\sigma_i(a) = kill$) en un determinado estado (w_i) se considerará como una violación ($V(a, kill, i)$).

Las normas generales se estructuran en niveles, de modo que primero se crean unas normas generales y después se definen normas más concretas que regulan situaciones más específicas. Este hecho puede dar lugar a conflictos normativos, lo que hace necesario la definición de algún criterio para la precedencia entre normas. En el caso de las normas generales, el criterio conocido como *lex specialis* [BvdT03a], que da precedencia a las normas más específicas sobre las normas generales, es el empleado para resolver los conflictos normativos.

- Por otro lado, se define como una *Regla* a la norma que regula el comportamiento de los agentes que asumen un determinado rol dentro de un grupo.

  Definición 4.1.8. *(Regla) Una Regla se define como un par $reg = \langle \gamma, r, g \rangle$, donde:*

  - *γ es la expresión deóntica que define la norma, se define en base a conceptos definidos mediante normas institucionales constitutivas.*
  - *$r \in R$ es el rol que se ve directamente afectado por lo descrito en la norma*
  - *$g \in G$ es el grupo dentro del cual es aplicable la norma*

  A continuación se muestra un ejemplo de regla para controlar el

comportamiento de los agentes que asumen el rol doctor:

$$\langle FORBIDDENrevealMedicSecret,$$
$$doctor, hospital\rangle \Leftrightarrow_{def}$$
$$[\forall a \in M \ \wedge \ \forall i \in \mathbb{N} \ \wedge \ \sigma_i(a) = revealMedicSecret$$
$$doctor \in R_i(a) \ \wedge \ hospital \in G_i(a)]$$
$$V(a, revealMedicSecret, i)$$

Esta norma establece que está prohibido para cualquier agente (a) que sea un doctor ($doctor \in R_i(a)$) en algún hospital ($hospital \in G_i(a)$) revelar secretos médicos ($\sigma_i(a) = revealMedicSecret$).

Como un concepto homólogo al de la regla, se define el Reglamento asociado a un rol como el conjunto de reglas que prescriben el comportamiento esperado de los agentes que asumen dicho rol.

Definición 4.1.9. *(Reglamento) El reglamento de un rol r se define del siguiente modo:*

$$Reglamento(r) = \{reg \in Reg : reg = \langle \gamma, r, g\rangle\}$$

donde Reg es el conjunto de reglas establecidas por la institución.

Del mismo modo, se define el Reglamento de un grupo como el conjunto de reglas que definen el comportamiento ideal de los miembros de un grupo.

Definición 4.1.10. *(Reglamento) El reglamento de un grupo g se define del siguiente modo:*

$$Reglamento(g) = \{reg \in Reg : reg = \langle \gamma, r, g\rangle\}$$

donde Reg es el conjunto de reglas establecidas por la institución.

Tal y como quedaba reflejado en el modelo general de NMAS, un agente m en un estado w_i del sistema asume un determinado conjunto de roles definido por $R_i(m)$. Puesto que las reglas pueden ser distintas e incluso entrar en conflicto unas con otras es necesario establecer algún criterio que permita determinar el conjunto de reglas que afectan a un agente en un momento dado. Por ello dentro de la institución se define una relación de orden total entre los elementos de R. De este modo se determina que en caso de que exista un conflicto entre las reglas que afectan a un agente, aquellas normas relativas al rol más prioritario prevalecerán sobre el resto de reglas en conflicto.

En resumen el conjunto de normas regulativas institucionales está formado por las normas generales y las reglas de la institución.

Definición 4.1.11. *(Normas Regulativas Institucionales) El conjunto de normas regulativas institucionales $I_{Regulative}$ se define como:*

$$I_{Regulative} = NG \cup Reg$$

donde:

- $\forall ng \in NG : ng = \langle \gamma \rangle$
- $\forall reg \in Reg : reg = \langle \gamma, r, g \rangle$

A continuación la Figura 4.1 contiene un esquema que recoge los principales estados y transiciones de la dinámica de las normas institucionales. Básicamente una norma institucional se crea cuando un agente con capacidades legislativas la *promulga*. A partir de ese momento la institución emplea alguno de sus mecanismos para hacerla *pública* y darla a conocer a sus miembros. En ese momento la norma pasa a estar *activa*. Ocasionalmente puede darse un estado de *excepción* y no estar vigente durante un periodo de tiempo. Finalmente la norma se cancela cuando un agente *legislador* la *deroga*.

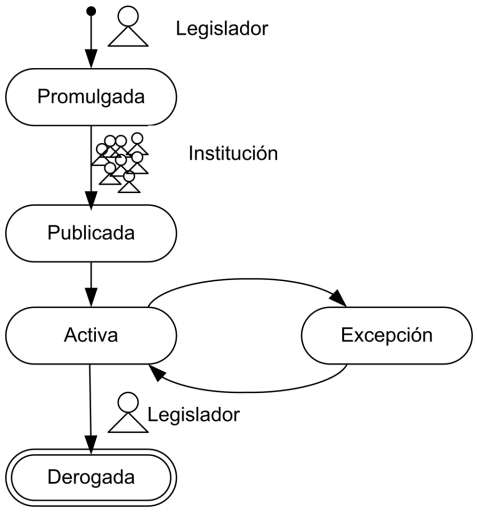

Figura 4.1: Dinámica de las normas institucionales

- **Nivel Interacción:** a este nivel pertenecen todas aquellas prescripciones de comportamiento que son resultado de la interacción entre dos agentes. Es decir, a este nivel pertenecen aquellos compromisos que los agentes adquieren a lo largo de su ejecución. En nuestra propuesta hemos distinguido dos tipos de compromisos: aquellos realizados de un modo informal y aquellos formalizados a nivel institucional (contratos).

  - Por el término *Compromiso* tradicionalmente se entiende una obligación contraída entre dos agentes. Nuestra propuesta permite definir los compromisos de un modo más general mediante los operadores de la lógica deóntica. De modo que un compromiso puede ser relativo a una obligación, un permiso o una autorización. Por ejemplo, como consecuencia de una negociación un agente podría estar autorizado a usar los recursos de otro agente.

    Definición 4.1.12. *(Compromiso) Se define un compromiso (commit)* $c_{Regulative}$ *como* $c_{Regulative} = \langle i, j, \gamma \rangle$ *donde:*

    - $i, j \in M$ *representan a los dos agentes involucrados por el compromiso.*
    - γ *expresión deóntica definida sobre* $N_{constitutive}$*. Es decir, un permiso, obligación o prohibición definida sobre hechos institucionales o definidos mediante algún compromiso.*

    Como ejemplo, un agente (a) puede ser autorizado ($PERMITTED\ do_a(employResource(R_b))$) a emplear los recursos de otro (R_b) como resultado de un proceso de negociación entre ellos:

    $$\langle PERMITTED\ employResource(R_b), a, b \rangle$$

  - Los *Contratos* son un subconjunto de los compromisos que han sido oficializados dentro de la institución. Es decir, aquellos compromisos formalizados a nivel institucional por algún representante legal de la institución (*notario*) reciben el nombre de contratos.

    Definición 4.1.13. *(Contrato) Se define un contrato (contract) como un compromiso con reconocimiento a nivel institucional. Las diferencias entre compromiso y contrato a nivel prescriptivo (regulativo) no existen, sólo se determinan a nivel procedural.*

    De modo que el conjunto de contratos ($Contract$) es un subconjunto de los compromisos ($Contract \subseteq C_{Regulative}$).

    La Figura 4.2 contiene un esquema que recoge los principales estados y transiciones de la dinámica de los compromisos y contratos.

Tal y como ya se ha indicado, un compromiso se crea como consecuencia de una interacción entre dos agentes o grupos de agentes actuando como una entidad. Si los agentes participantes en un compromiso lo desean, pueden darle un significado institucional a través de un proceso de formalización llevado a cabo por un representante del sistema institucional (*notario*) que tiene como resultado la creación de un nuevo *contrato*. No sólo se institucionalizarán los aspectos regulativos del contrato sino también todos aquellos términos y conceptos abstractos empleados en la definición del contrato. A partir de este momento el contrato pasa a estar *activo*; *excepcionalmente* podrá estar no vigente durante un periodo de tiempo. Finalmente, el contrato se eliminará del sistema cuando un agente que actúe en nombre de la institución (*notario*) lo *derogue*. Dicha cancelación podrá realizarse de forma automática al transcurrir el periodo de validez del contrato o bien el *notario* podrá derogarlo atendiendo a una solicitud de los agentes vinculados.

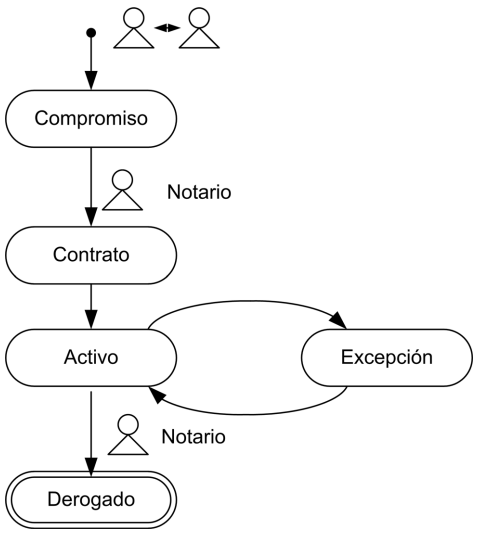

Figura 4.2: Dinámica de los contratos

- **Nivel Privado**: A este nivel pertenecen aquellas normas básicas que rigen el comportamiento de los agentes. Estas normas básicas reciben el nombre de primitivas en nuestro modelo y se definen tal y como se indica a continuación.

  Definición 4.1.14. *(Primitiva) Se define una primitiva de comporta-*

miento como una expresión primitiva $\rho(c|a)$, donde $a \in A$ y c es una expresión de primer orden definida sobre elementos de F.

En cada situación es posible que un agente tenga más de una primitiva a aplicar, por ello es necesario establecer una ordenación entre las primitivas que permita determinar la acción que se llevará a cabo en cada momento.

A continuación la tabla 4.1 contiene un resumen de los diferentes tipos de normas definidos en esta sección.

Nivel	**Tipo**	**Descripción**	
Institucional	Norma General	Norma impuesta por una institución a sus miembros	
	Reglamento	Reglamento especifico de un subconjunto de la sociedad	
Interacción	Contrato	Compromiso entre dos entidades reconocido por alguna autoridad legal	Significado Institucional
	Compromiso	Compromiso informal entre dos entidades	Significado Privado
Individual	Primitiva	Pauta de comportamiento individual	

Tabla 4.1: Tabla Resumen de las Normas Regulativas

A modo de resumen definiremos el conjunto de normas regulativas como:

Definición 4.1.15. *(Normas Regulativas)El conjunto de normas regulativas $N_{Regulative}$ se define como la unión de las normas regulativas institucionales y las normas regulativas contractuales:*

$$N_{Regulative} = I_{Regulative} \cup C_{Regulative}$$

donde:

- $\forall i_{Regulative} \in I_{Regulative} : i_{Regulative} \in (NG \cup Reg)$
- $\forall c_{Regulative} \in C_{Regulative} : c_{Regulative} = \langle i, j, \gamma \rangle$

4.1.2. Normas Procedurales

Este tipo de normas son las que determinan el modo en que se aplican las normas dentro del sistema. Es decir, las normas procedurales son aquellas que permiten a los agentes encargados de controlar una norma detectar su violación y llevar a cabo las acciones apropiadas.

Definición 4.1.16. *(Norma Procedural) Una Norma Procedural $n_{procedural}$ es aquella que relaciona una norma regulativa con los mecanismos que permiten su aplicación $n_{procedural} = \langle n_{Regulative}, violation, sanction, reward \rangle$ donde:*

- $n_{Regulative} \in N_{Regulative} = I_{Regulative} \cup C_{Regulative}$ *representa la norma regulativa o prescripción a la que afecta la norma procedural.*
- *violation es una condición definida sobre términos de* F*. Dicha condición establece las condiciones en las cuales la norma se puede considerar como violada.*
- $sanction, reward$ *son expresiones que describen las acciones que se llevarán a cabo si la norma es quebrantada o respetada, respectivamente. Este tipo de expresiones se define como una secuencia de descripciones de acciones que se llevarán a cabo en cada caso.*

A continuación se muestra un ejemplo de norma procedural:

$$\langle FORBIDDENkill, do_a(kill),$$
$$do_I(imprison(a,t)) \wedge t \in [5,10], - \rangle \Leftrightarrow_{def}$$
$$[\forall a \in M \ \wedge \ \forall i \in \mathbb{N} \ \wedge \ \sigma_i(a) = kill]$$
$$V(a, kill, i) \ \wedge \ do_I(imprison(a,t)) \ \wedge \ t \in [5,10]$$

Dicha norma determina que si cualquier agente (a) en un cierto estado (i) lleva a cabo la acción de matar ($\sigma_i(a) = kill$), entonces esto se considerará como una violación ($V(a, kill, i)$) y será encarcelado por la organización a modo de sanción ($do_I(imprison(a,t))$).

Se define el conjunto de normas procedurales asociadas a una norma regulativa institucional como:

Definición 4.1.17. *(Normas Procedurales Institucionales) El conjunto de normas procedurales institucionales* $I_{Procedural}$ *se define como:*

$$\forall i_{Procedural} \in I_{Procedural} :$$
$$i_{Procedural} = \langle i_{Regulative}, violation, sanction, reward \rangle \wedge i_{Regulative} \in I_{Regulative}$$

Se define el conjunto de normas procedurales asociadas a una norma regulativa contractual, derivada de una interacción, como:

Definición 4.1.18. *(Normas Procedurales relativas a un Compromiso) El conjunto de normas procedurales relativas a compromisos* $C_{Procedural}$ *se define como:*

$$\forall c_{Procedural} \in C_{Procedural} :$$
$$c_{Procedural} = \langle c_{Regulative}, violation, sanction, reward \rangle \wedge c_{Regulative} \in C_{Regulative}$$

Por último el conjunto de normas procedurales se define como la unión de las normas procedurales institucionales y las normas procedurales relativas a un compromiso:

Definición 4.1.19. *(Normas Procedurales) El conjunto de normas procedurales institucionales* $N_{Procedural}$*:*

$$N_{Procedural} = I_{Procedural} \cup C_{Procedural}$$

4.2. Expresión de Normas mediante la *Real-Time Agent Logic* (RTAL)

En el sección anterior se ha propuesto un nuevo modelo para representar SMA Abiertos, donde el comportamiento de los agentes se controla mediante normas.

En este modelo tanto las normas institucionales como los contratos se formalizan en base a una formula (γ) que es una expresión deóntica definida en base a hechos y acciones básicas o a hechos abstractos definidos mediante una norma constitutiva. Más concretamente, para la representación de este tipo de restricciones se ha optado por el empleo de la lógica *Real-Time Agent Logic* (RTAL) propuesta en [Reb04].

Tal y como se indicó en la sección 2.2.3-C, la RTAL consiste básicamente en el empleo de la extensión temporal de la CTL con la lógica TLA.

4.2.1. Normas Regulativas

A continuación se propone la expresión de normas en nuestro modelo como fórmulas de RTAL. Tomando como punto de partida la expresión de reglas temporales mediante la lógica RTAL, tal y como se ha detallado con anterioridad, y la interpretación de los operadores deónticos basada en la lógica dinámica (sección 2.1.2); se define la "traducción" de una prohibición del siguiente modo:

Definición 4.2.1. *(Prohibición en RTAL) Formalmente se define una prohibición en RTAL como una fórmula de la siguiente forma:*

$$A\Box(\pi \rightarrow A\ \Box^{[e_{start},e_{end}]}[\alpha]V)$$

donde:

- π *es la condición booleana para la activación de la norma definida en base a hechos básicos y/o institucionales.*
- e_{start}, e_{end} *eventos de activación y desactivación. Determinan el periodo de vigencia de la norma. Se definen en base a la detección de hechos básicos y/o institucionales.*

- *α acción controlada por la norma. Dicha acción se especifica mediante la lógica TLA.*
- *V símbolo de predicado auxiliar empleado para representar un estado de violación normativo.*

Esta fórmula representa que siempre que la condición π sea cierta, se verifica que para todos los caminos, tras ejecutar la acción α dentro del intervalo de activación de la norma ($[e_{start}, e_{end}]$) se produce una violación (V).

Por otro lado, las obligaciones quedarían expresadas tal y como se indica a continuación.

Definición 4.2.2. *(Obligación en RTAL) Formalmente se define una obligación en RTAL como:*

$$A\Box(\pi \rightarrow A\ \Box^{[e_{start},e_{end}]}[\neg\alpha]V)$$

donde π, e_{start}, e_{end}, a y V se definen como en la definición anterior (4.2.1).

Esta fórmula representa que siempre que la condición π sea cierta, se verifica que para todos los caminos, la no ejecución de la acción α dentro del intervalo de activación de la norma ($[e_{start}, e_{end}]$) produce una violación (V).

Por último, los permisos quedarían expresadas tal y como se indica a continuación.

Definición 4.2.3. *(Permiso en RTAL) Formalmente se define una obligación en RTAL como:*

$$A\Box(\pi \rightarrow A\ \Box^{[e_{start},e_{end}]}[\alpha]\neg V)$$

donde π, e_{start}, e_{end}, a y V se definen como en la definición anterior (4.2.1).

Esta fórmula representa que siempre que la condición π sea cierta, se verifica que para todos los caminos, la ejecución de la acción α dentro del intervalo de activación de la norma ($[e_{start}, e_{end}]$) no produce una violación (V).

4.3. Conclusiones

Tal y como quedó de manifiesto en el capítulo 1, se han propuesto diferentes modelos normativos con el objetivo de permitir el diseño de sociedades de agentes complejas y dinámicas. Este tipo de escenarios, caracterizados por

la heterogeneidad y dinamicidad de los agentes participantes y los entornos, necesitan un concepto de norma más amplio.

En este capítulo se describe un nuevo modelo de norma que extiende los trabajos anteriores tratando de ofrecer una solución a la problemática de los sistemas abiertos. Las principales características de este modelo son:

- Ofrece soporte para los diferentes niveles de normatividad.
- Tiene en consideración los diferentes tipos de normas, es decir, normas regulativas, constitutivas y procedurales.
- Permite una definición estructurada del sistema normativo.
- Ofrece mecanismos para la creación y derogación de normas, a través de normas constitutivas que determinan la dinámica del sistema legal.

Para finalizar este capítulo, se ha tratado el problema de la formalización de las restricciones deónticas. La formalización propuesta consiste en el empleo de la lógica RTAL y la interpretación de los operadores deónticos mediante la lógica dinámica. De modo que conceptos tales como la obligación y prohibición son representados como fórmulas pertenecientes a la RTAL.

Capítulo 5

Propuesta de Arquitectura: THOMAS

La evolución tecnológica de los últimos años (Internet, la WWW, Comercio Electrónico, conectividad inalámbrica, etc.) ha dado lugar a un nuevo paradigma de computación: 'la computación como interacción' [LMSW05]. En este nuevo paradigma, la computación es algo que ocurre mediante y a través de la comunicación entre entidades computacionales. Desde esta aproximación la computación es una actividad inherentemente social, en lugar de solitaria, llevando a nuevas formas de concebir, diseñar, desarrollar y manejar sistemas computacionales. Un ejemplo de la influencia de este punto de vista es el modelo emergente del software como un servicio, por ejemplo en las *arquitecturas orientadas a servicios* (SOA) [Tuo08]. La tecnología de agentes/sistemas multiagente resulta especialmente prometedora como soporte a este nuevo paradigma de computación como interacción. Las organizaciones dinámicas de agentes que se auto-ajustan para obtener ventajas a partir de su entorno actual son cada vez más importantes [DD06]. Estas organizaciones podrían aparecer en sociedades de agentes dinámicas o emergentes, tales como las sugeridas por los dominios Grid, las redes peer-to-peer u otros ambientes en los cuales los agentes se agrupan de forma dinámica para ofrecer servicios compuestos. Los factores sociales en las organizaciones de sistemas multiagentes también son cada vez más importantes para estructurar las interacciones en mundos abiertos y dinámicos. En esta interesante área, enmarcada en la más general de Sistemas Inteligentes, podemos identificar dos grandes líneas de trabajo:

- Dinamicidad/Regulación: flexibilidad para permitir la entrada y salida de los agentes, evolución de la estructura organizativa, mecanismos de regulación, etc.

- Heterogeneidad: diferentes tipos de agentes con capacidades diversas, coordinación en tiempo de ejecución (requiere descripción semántica de las capacidades/servicios), diferentes dispositivos (recursos físicos de los dispositivos), diferentes canales de comunicación (wireless, wireline, etc).

En el marco del proyecto THOMAS (comentado en la introducción de este trabajo), se está trabajando en el desarrollo de un arquitectura de agentes abstracta que permita el modelado de sistemas organizativos abiertos. El objetivo del presente capítulo es avanzar y aportar soluciones en estas líneas, para ello se propone:

- Desarrollar una arquitectura de sistema multiagente adecuada para la generación de organizaciones virtuales en entornos abiertos, así como una plataforma de soporte que permita la implantación de este tipo de sistemas.

- Desarrollar mecanismos basados en estructuras organizativas y organizaciones virtuales para optimizar y regular la coordinación de servicios en sistemas multiagente abiertos.

- Incluir mecanismos de control del comportamiento de los agentes empleando un lenguaje normativo.

En las siguientes secciones se presenta el trabajo realizado en los ámbitos anteriores.

5.1. Modelo de Organización Virtual [CAJB09]

Las *Organizaciones Virtuales* (VOs) son un conjunto de individuos e instituciones que necesitan coordinar recursos y servicios mediante una serie de fronteras institucionales [Fos01, BHvdT05]. Se trata, por lo tanto, de Sistemas Abiertos formados por la agrupación y la colaboración entre entidades heterogéneas. Han sido empleadas como un paradigma para el desarrollo de SMA [FGM03]. Las VO permiten el modelado de sistemas con un alto nivel de abstracción, incluyendo tanto la perspectiva individual y la perspectiva organizacional, así como la adaptación dinámica de los modelos organizacionales y los cambios producidos en el entorno [DD06], formando grupos con límites de visibilidad [FGM03].

Tomando como punto de partida la Teoría de las Organizaciones Humanas [Daf03], en [CAJB09] definimos un *Modelo de Organización Virtual* (VOM) que permite la descripción de los principales aspectos de una organización: su

estructura, comportamiento, dinámica, normas y su entorno. Básicamente, el VOM permite definir cuáles son las entidades del sistema, cómo se relacionan entre ellas y con su entorno, cuál es su funcionalidad, qué servicios se necesitan y/o ofrecen, cómo estas entidades asumen cierta funcionalidad predefinida en el sistema y cuáles son sus restricciones de comportamiento. La principal contribución del VOM es que está enfocado hacia el modelado de sociedades abiertas en las cuales agentes heterogéneos y autónomos trabajan juntas. Por otro lado, el modelo ofrece soporte para la integración de las tecnologías de los Servicios Web y los SMA. Por este motivo la funcionalidad proporcionada por las entidades se describe, publicita y accede a través de servicios.

Nuestra propuesta de VOM considera los cuatro aspectos fundamentales de toda VO: (i) la *Dimensión Estructural* describe los componentes del sistema y sus relaciones; (ii) la *Dimensión Funcional* detalla la funcionalidad del sistema; la (iii) *Dimensión Normativa* define los mecanismos empleados por la sociedad para influenciar en el comportamiento de sus miembros; por último, la (iv) *Dimensión del Entorno* define el entorno o ambiente de una organización en base a sus recursos y cómo los agentes pueden percibirlos y actuar sobre ellos.

A continuación se describe con detalle cada una de las cuatro dimensiones definidas en el VOM.

5.1.1. Dimensión Estructural

La *Dimensión Estructural* (Figura 5.1a) describe cuáles son los elementos del sistema (agentes y unidades organizacionales) y cómo se relacionan entre sí.

El modelo define una *Unidad Organizativa* (UO) como una entidad social básica que representa al conjunto mínimo de agentes que llevan a cabo una actividad determinada, siguiendo un patrón de comunicación y cooperación predefinido [AJB08, APA+07]. Esta asociación puede verse desde una perspectiva global como una única entidad; ya que persigue unos objetivos, ofrece y necesita unos servicios y juega un rol específico dentro de otras unidades. Las UO representan un punto de encuentro virtual; en este sentido, los agentes pueden entrar o salir de ellas de forma dinámica a través de la adopción de roles dentro de la UO.

Una UO está formada por diferentes entidades(*Entity*) (relación *has_member*) a lo largo de su ciclo de vida, que pueden ser tanto agentes individuales u otras UO vistas como una única entidad. Las UOs presentan diferentes topologías y relaciones comunicativas en función de su entorno, el tipo de actividades que se llevan a cabo y su propósito. Las topologías básicas son:

(i) *Jerarquía Simple*, en la cual existe un agente supervisor que controla al resto de miembros; (ii) el *Equipo*, que es un grupo de agentes que comparten un objetivo común, colaborando y cooperando entre ellos; y (iii) la estructura *Plana*, que representa una agrupación en la cual no hay ningún agente con capacidad para controlar al resto de miembros. Empleando estas tres topologías básicas es posible definir estructuras organizativas más complejas, como federaciones, burocracias, congregaciones o coaliciones, tal y como se explica en [HL04].

En una UO se define una serie de roles que pueden ser adoptados por sus miembros (relación *has_role*); del mismo modo se definen las relaciones existentes entre los roles (*has_ relationship*). El concepto de Rol (*Role*) se define mediante tres atributos: *Visibilidad*, *Accesibilidad* y *Posición*. La *Visibilidad* indica si los agentes pueden obtener información acerca de este rol desde fuera de la unidad en la que se define (*publico*) o sólo es posible hacerlo desde dentro (*privado*). La propiedad de *Accesibilidad*, basada en [DVSD04], considera dos tipos de roles: (a) roles *internos*, en los cuales se delegan ciertas funcionalidades relacionadas con el control del resto de miembros de la unidad, por ello son asignados a agentes internos de la plataforma; y (b) los roles *externos* puede ser asumidos por cualquier agente, cuya funcionalidad podría no haber sido implementada por el diseñador del sistema. Finalmente, la *Posición* de un rol determina su posición estructural dentro de la unidad, como lo son por ejemplo las posiciones de supervisor y subordinado.

El concepto de Relación (*Relationship*), que se define en [LyLL02], representa las conexiones sociales entre *Roles*. Una relación de *información* establece una conexión entre los agentes con capacidad para conocerse entre sí e intercambiar información. La relación de *monitorización* implica el proceso de control y monitorización sobre alguna actividad llevada a cabo por un agente. Finalmente, la relación de *supervisión* implica que un agente tiene poder sobre otro, es decir, que el primero delega uno o más objetivos o tareas al agente subordinado.

5.1.2. Dimensión Funcional

La *Dimensión Funcional* (Figura 5.1b) detalla la funcionalidad del sistema en base a servicios, tareas y objetivos.

Como ya se comentó en la introducción del capítulo, nuestra propuesta está centrada en la integración de las tecnologías SOA y SMA. Los servicios representan la funcionalidad que los agentes ofrecen a otras entidades, independientemente del agente concreto que haga uso de ella. Los servicios pueden ser atómicos (tareas simples) o estar formados por diferentes tareas. Estas tareas pueden ser realizadas por el agente que ofrece el servicio o bien

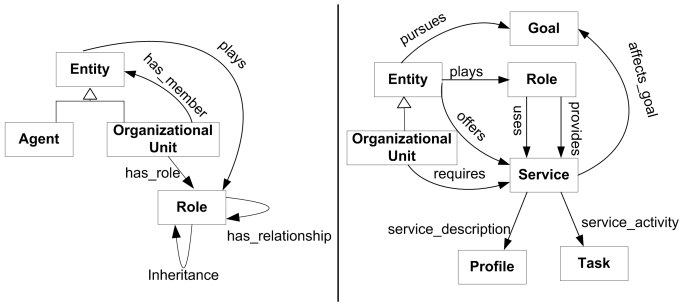

Figura 5.1: a) Dimensión Estructural; b) Dimensión Funcional

delegarse a otros agente, a través de la invocación, orquestación o composición de servicios. El VOM permite describir la funcionalidad de los agentes en términos de servicios, a través de la ontología estándar para la definición de servicios OWL-S[1]. De este modo, en VOM se describe la funcionalidad (representada mediante el concepto de *Service*) y el modo en el que se lleva a cabo (*service_ activity*) mediante el empleo de OWL-S; lo cual permite estandarizar la funcionalidad de los agentes y el modelado de los SMA siguiendo una visión orientada a servicios.

Una entidad *Entity* se describe mediante un identificador y una relación de pertenencia a una unidad en la que juega (*plays*) un rol determinado. Además puede ofrecer (*offers*) alguna funcionalidad al resto de entidades. Su comportamiento está motivado por los objetivos que persigue (*pursues*). Por otro lado, una UO puede hacer pública su necesidad de algún servicio (relación *requires*), de modo que los agentes externos puedan decidir participar dentro de ella como proveedores de dicha funcionalidad. Cada servicio tiene uno o más roles que se encargan de su provisión (*provides*) y otros roles que los emplean (*uses*). Además, cada servicio tiene una cierta influencia sobre los objetivos del sistema (relación *affects_ goal*).

5.1.3. Dimensión del Entorno

Tomando como punto de partida la Teoría de las Organizaciones Humana [Daf03], la *Dimensión del Entorno* puede ser considerada desde dos perspectivas: (i) desde una perspectiva *estructural*, el entorno se describe en términos de los elementos que lo componen, es decir, sus recursos y aplicaciones; y (ii) desde una perspectiva *funcional*, el entorno se describe en base a cómo los agentes lo perciben y actúan sobre él.

[1]http://www.w3.org/Submission/OWL-S/

El VOM incluyen tanto las perspectivas funcional y estructural del entorno. Esta propuesta se centra en describir los componentes del entorno, la percepción y actuación sobre esos elementos y la definición de los permisos para acceder a él. La *Dimensión del Entorno* (Figura 5.2a) define cada elemento del ambiente como un *Recurso*, que representa a un componente del entorno. Dicho componente pertenece a una entidad (*has_resource*), que puede ser tanto un agente individual como una UO. En este último caso, es necesaria una entidad que controle los permisos de acceso a dicho elemento (*has_portControl*). Los recursos son percibidos y accedidos a través un *EnvironmentPort*. Por otro lado, el concepto *ServicePort* representa el registro de un servicio en el directorio de servicios (*registers*) o su empleo (*serves* o *requests*). Cada puerto es empleado por uno o más roles (*uses_port*) definidos en la organización.

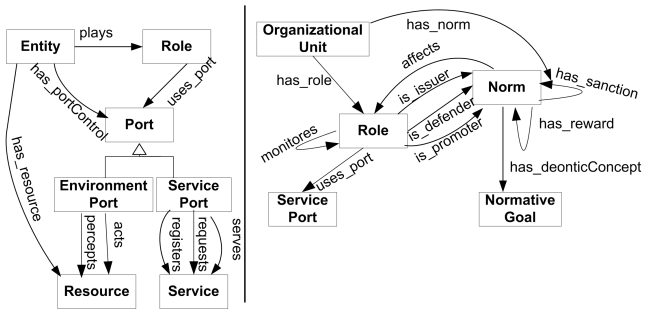

Figura 5.2: a) Dimensión del Entorno; b) Dimensión Normativa

5.1.4. Dimensión Normativa

La *Dimensión Normativa* (Figura 5.2b) describe las normas organizacionales y los objetivos normativos que los agentes deben seguir incluyendo las sanciones y recompensas.

Los sistemas Abiertos, en los cuales agentes heterogéneos y auto-interesados trabajan juntos, necesitan de algún mecanismo para restringir el comportamiento de los agentes pero manteniendo un cierto grado de autonomía. La *Dimensión Normativa* del VOM define un mecanismo de control que trata de: (i) promover aquellos comportamientos que resultan satisfactorios para la organización, es decir, acciones que contribuyen a la consecución de los objetivos globales; y (ii) evitar acciones perjudiciales, es decir, acciones que conduzcan a estados indeseables del sistema. De modo que, la dimensión normativa permite especificar el comportamiento deseado de los miembros

en términos de acciones que los agentes deben o no llevar a cabo. Más concretamente, las normas determinan la funcionalidad de los roles a través de la restricción de las acciones permitidas y definen las consecuencias de las acciones en términos de sanciones y recompensas.

Cada UO tiene un conjunto de normas que restringen el comportamiento de sus miembros (*has_norm*). Una norma *afecta* a un rol directamente, quien estará *obligado*, *autorizado* o tendrá *prohibido* realizar una determinada acción (*has_deonticConcept*). Las *acciones* en nuestra propuesta son la solicitud, registro o provisión de servicios. Las sanciones y recompensas se expresan mediante normas. El rol *issuer* es el encargado de controlar el cumplimento de la norma, mientras que los roles *defender* y *promoter* se encargan de la aplicación de las sanciones y recompensas, respectivamente. Todos estos roles pueden ser asumidos por un mismo agente o por distintos agentes.

5.2. Arquitectura de Organización Virtual [CGJ+ss]

Teniendo en cuenta el modelo anterior de Organización Virtual se ha propuesto una arquitectura para dar soporte a SMA Abiertos de gran escala. La arquitectura propuesta, conocida como THOMAS [CGJ+ss], tiene como principal objetivo conseguir integrar la tecnología de SMA y SOA como base para el desarrollo de Organizaciones Virtuales. Los agentes pueden ofrecer e invocar servicios a otros agentes o entidades de un modo transparente. Del mismo modo, los agentes tienen acceso a la infraestructura de la arquitectura a través de un conjunto de servicios. La Figura 5.3 contiene un esquema de la arquitectura. Los servicios ofrecidos por la arquitectura THOMAS están organizados en diferentes módulos o componentes:

- **Service Facilitator** (SF). Este componente proporciona el mecanismo a través del cual las organizaciones y agentes pueden ofrecer y descubrir servicios. En este sentido, el SF redefine el *Directory Facilitator* de FIPA, permitiendo la gestión de servicios de una forma más elaborada, siguiendo los estándares propuestos en SOA. De modo que ofrece servicios simples y complejos tanto a los agentes individuales como a las organizaciones de agentes. Su funcionalidad es similar a la de un servicio de páginas amarillas y un servicio de páginas verdes.

- **Organization Management System** (OMS). Este componente se encarga de la gestión de las organizaciones y sus entidades. De modo

que lleva a cabo la gestión del ciclo de vida de las organizaciones así como la gestión de sus componentes estructurales (unidades, roles y normas).

- **Platform Kernel** (PK). Comprende los servicios básicos de gestión de la plataforma de agentes. Integra tanto al componente AMS como a la red de comunicaciones definidas por el estándar FIPA.

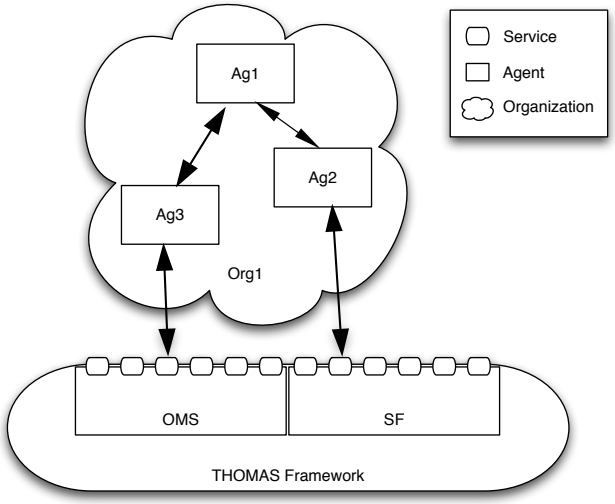

Figura 5.3: Arquitectura THOMAS

Los servicios proporcionados por la arquitectura pueden ser registrados y publicitados con el objetivo de permitir a los agentes conocer su existencia y cómo y cuándo hacer uso de ellos.

El OMS requiere de un lenguaje normativo para el control del comportamiento de los agentes del sistema. Por ello, nos centraremos a continuación en la descripción de este componente.

5.2.1. Organization Management System [CAJB07]

El OMS [CAJB07] se encarga se realizar la gestión del ciclo de vida de las organizaciones, es decir, cómo se crean las UO, qué entidades participan dentro de ellas, cómo estas entidades se relacionan unas con otras y qué roles asumen dichas entidades en cada momento. Para llevar a cabo este proceso

de gestión el OMS ofrece a los agentes un conjunto de servicios clasificados en:

- **Servicios Estructurales**, que permiten a los agentes solicitar al OMS realizar cambios en la especificación estructural y normativa. Este conjunto comprende servicios para añadir/eliminar normas (*RegisterNorm*, *DeregisterNorm*), añadir/eliminar roles (*RegisterRole*, *DeregisterRole*) y crear nuevas UO o eliminarlas (*RegisterUnit*, *DeregisterUnit*). Gracias a la publicación de estos servicios se permite a los agentes modificar la estructura de la organización a lo largo del tiempo.

- **Servicios Informativos**, proporcionan información acerca del estado actual de la organización, detallando cuáles son los roles definidos en una UO (*InformUnitRoles*), los roles asumidos por un agente (*InformAgentRoles*), los miembros de una UO (*InformMembers*), la cantidad de miembros (*InformQuantity*), su estructura interna (*InformUnit*), y los servicios y normas relacionados con un rol específico (*InformRoleProfiles*, *InformRoleNorms*).

- **Servicios Dinámicos**, que permiten definir cómo los agentes pueden adoptar roles dentro de las UOs (*AcquireRole*, *LeaveRole*) o cómo los agentes pueden ser obligados a dejar un rol específico (*Expulse*), normalmente como resultado de la aplicación de una sanción. Mediante estos servicios se permite a los agentes externos participar en el sistema.

Este conjunto de servicios para la gestión del ciclo de vida de las organizaciones permite definir las características de especificación y administración de los componentes estructurales de la organización (roles, unidades y normas) y su dinámica (entrada/salida de entidades). Sin embargo, es necesario definir algún mecanismo de control que permita establecer quién puede hacer uso de estos servicios y bajo qué condiciones. Este tipo de control se ha definido a través de normas. La siguiente sección describe el lenguaje normativo desarrollado para controlar el acceso a los servicios.

5.3. Lenguaje Normativo [ACJB08]

Los Sistemas Normativos han sido definidos como un mecanismo para permitir la cooperación dentro de SMA Abiertos. En este sentido, las normas son un método de persuasión para obtener el comportamiento deseado de los agentes. Además, las normas pueden verse como un herramienta de coordinación para organizar los SMA, ya que con ellas se especifica el comportamiento

deseado de los miembros de la sociedad [BvdTV08]. En relación a esta segunda concepción de las normas, nuestra propuesta consiste en emplear las normas para regular las organizaciones de agentes. Más concretamente, este trabajo tiene el propósito de aplicar la teoría normativa para definir el modo en el que los agentes pueden modificar la estructura de la organización (normas, unidades y roles) y sus componentes de ejecución, con el objetivo de adaptar la organización a los cambios producidos en el entorno. Con este propósito se ha desarrollado un lenguaje normativo para controlar la dinámica de la organización así como una implementación de estas normas.

5.3.1. Descripción del Lenguaje Normativo

Esta propuesta de lenguaje se basa en los trabajos realizados para la definición de normas dentro de las *Instituciones Electrónicas* (IEs) [GCNRA05]. Estos trabajos definen un lenguaje para controlar los actos de habla (ilocuciones) que los agentes pueden pronunciar dentro de una IE. Además, se propone una extensión de este lenguaje para permitir la definición de normas relativas a acciones no-dialógicas [dS07]. Nuestro lenguaje toma estas aproximaciones como punto de partida y las extiende ofreciendo soporte para la gestión de organizaciones. Las principales contribuciones del lenguaje propuesto son: (i) permite la definición de las consecuencias de las normas a través de las sanciones y recompensas; (ii) ofrece soporte para conceptos organizacionales como roles o UOs; (iii) la definición de las funcionalidades de los agentes a través del estándar OWL-S aumenta la expresividad de las normas.

Las normas definen los derechos y responsabilidades de los agentes en términos de las acciones que los agentes están o no autorizados a llevar a cabo. Las acciones se han dividido en dos categorías: aquellas relacionadas con los aspectos organizacionales; y aquellas acciones relativas al acceso a servicios. Por lo tanto, se han definido dos tipos fundamentales de normas:

- **Normas Organizacionales**: están relacionadas con los servicios ofrecidos por el OMS a los miembros de la organización. Establecen la dinámica de la organización, por ejemplo la gestión de los roles (cardinalidad e incompatibilidad entre los roles) y el protocolo por el cual los agentes pueden adquirir los roles.

- **Normas Funcionales**: definen la funcionalidad de los roles en términos de los servicios que pueden solicitar/proveer, el orden de solicitud de los servicios, las condiciones en las que se pueden solicitar, el protocolo que debe seguirse, etc. Establecen la gestión de acceso a los servicios en base al resultado previo de otras solicitudes, estado del entorno, etc.

<ext_norm>	::=	<norm> [SANCTION(<norm>)] [REWARD(<norm>)]
<norm>	::=	<deontic_concept> <entity> <action> [<temporal_situation>] [IF <if_condition>] | *norm_id*
<deontic_concept>	::=	OBLIGED | FORBIDDEN | PERMITTED
<entity>	::=	agent: role − unit | role − unit | entity_id
<agent>	::=	?*variable* | *agent_id*
<role>	::=	?*variable* | *role_id*
<unit>	::=	?*variable* | *unit_id*
<entity_id>	::=	*agent_id* | *role_id* | *unit_id*
<action>	::=	<functional_action> | <organizational_action>
<temporal_situation>	::=	BEFORE <situation> | AFTER <situation > | BETWEEN(<situation> , <situation>)

Tabla 5.1: Sintaxis BNF de las Normas

A. Sintaxis del Lenguaje Normativo

La Tabla 5.1 contiene la sintaxis BNF del lenguaje propuesto. En nuestra propuesta, se define una norma por medio de un predicado deóntico (<*deontic_concept*>), un rol a quien le afecta la norma (<*entity*>) y una acción (<*service_action*>). El término <*temporal_situation*> establece una condición temporal para la activación de la norma, relacionada con la provisión de otros servicios. Además, puede contener una condición sobre el estado que determina las condiciones de activación de la norma (<*if_condition*>). La sintaxis de estas condiciones se define en la Tabla 5.4.

Por un lado, las *Normas Organizacionales* están relacionadas con acciones que permiten a los agentes solicitar servicios al OMS (<*org_service*>) para adoptar roles, registrar nuevas normas, etc. La sintaxis de las acciones organizacionales se detalla en la Tabla 5.2.

Por otro lado, las *Normas Funcionales* se definen en términos de acciones relacionadas con la publicación (REGISTER), provisión (SERVE) o empleo de servicios (REQUEST). La sintaxis BNF de este tipo de acciones se detalla en la Tabla 5.3.

La condición sobre el estado para la activación de una norma se muestra en la Tabla 5.4. Dicha condición booleana (<*if_condition*>) puede expresarse sobre variables, identificadores, resultados normativos (<*normative_result*>) o resultados de servicio (<*service_result*>). Los resultados de un servicio son los estados correspondientes a un fallo en la provisión o a una ejecución satisfactoria. El término <*situation*> se emplea en la definición de la condición

<organizational_action>	::=	REQUEST <org_service> MESSAGE(<msg_cont>)
<org_service>	::=	<structural_service> | <dynamic_service> | <informative_service>
<structural_service>	::=	RegisterNorm | RegisterRole | DeregisterNorm | DeregisterRole | DeregisterUnit | RegisterUnit
<informative_service>	::=	InformUnitRoles | InformAgentRoles | InformMembers | InformRoleProfiles | InformRoleNorms | InformUnit | InformQuantity
<dynamic_service>	::=	AcquireRole | LeaveRole | Expulse

Tabla 5.2: Sintaxis BNF de las Acciones Organizacionales

<functional_action>	::=	<serv_publication> | <serv_provision> | <serv_usage>
<serv_publication>	::=	REGISTER *service_name* PROFILE <profile_desc> [PROCESS<process_desc>]
<service_provision>	::=	SERVE *service_name* PROCESS <process_desc> [MESSAGE(<msg_cont>)]
<service_usage>	::=	REQUEST *service_name* MESSAGE(<msg_cont>)

Tabla 5.3: Sintaxis BNF de las Acciones Funcionales

<if_condition>	::=	NOT(<cond_exp>) | < cond_exp >
<cond_exp>	::=	<condition> | NOT <condition> |
		<condition> AND <cond_exp> |
		<condition> OR <cond_exp>
<condition>	::=	<variant> <operator> <variant> |
		<normative_result>(<norm>)
<normative_result>	::=	FAILED | SATISFIED
<operator>	::=	> | < | >=| <= | = | ⊆
<variant>	::=	<atomic> | <formula> | *value*
<atomic>	::=	*identifier variable* | *variable* | *identifier*
<formula>	::=	*identifier*(<args>) | <service_result> |
		QUANTITY(<args>)
<args>	::=	<variant> [, <variant>]
<situation>	::=	*deadline* | <service_result> |
		[entity] < service_action>
<service_result>	::=	RESULT(*service_name*, <entity>, *ContextTime*)

Tabla 5.4: Sintaxis BNF de las Condiciones

temporal. Puede expresarse como un *deadline*, una acción o el resultado de un servicio.

Tal y como se ha comentado con anterioridad, la especificación de la funcionalidad a través de los estándares de los servicios Web (OWL-S) permite definir las funcionalidades de una forma más expresiva: (i) representado las precondiciones y efectos de los servicios; (ii) describiendo las funcionalidades globales como servicios complejos que están compuestos de servicios atómicos, de modo que la especificación de un servicio complejo permite definir cómo los comportamientos de los agentes son orquestados; y (iii) describiendo la funcionalidad de dos modos: servicios que las entidades ofrecen y servicios necesitados. De modo que los conceptos de la Computación Orientada a Servicios, tales como ontologías, modelos de proceso, coreografía, facilitadores, medidas de calidad de servicio (QoS), etc. pueden ser aplicadas a los SMA. Nuestra propuesta de lenguaje normativo ofrece soporte para especificar el conocimiento relativo a un servicio siguiendo la ontología OWL-S. El *profile* (<*profile_desc*>) de un servicio contiene sus parámetros, precondiciones y postcondiones. La descripción del proceso (<*process_desc*>) ofrece una descripción detallada de las funciones que realiza el servicio. Estas acciones se enlazan unas con otras por medio de diferentes estructuras de control: CONNECTS indica un orden secuencial entre dos acciones; JOIN indica una concurrencia entre acciones y un paso de sincronización final; IF-THEN-ELSE y WHILE-DO define la clásica estructura de control. Finalmente, el *grounding* proporciona los detalles acerca de cómo interoperar con un servi-

<profile_desc>	::=	[INPUT(<param_list>)] [OUTPUT (<param_list>)] [PRE(<cond_exp>)][POST(<cond_exp>)] | *profile_id*
<process_desc>	::=	*process_id* | *?variable* | <action> CONNECTS <process_desc> | <action> JOIN <process_desc> | IF <cond_exp> THEN(<process_desc>) [ELSE (<process_desc>)] | WHILE <cond_exp> DO(<process_desc>)
<msg_cont>	::=	[SENDER(<entity>)] [RECEIVER (<entity>)] [PERFORMATIVE (*performative_id*)] CONTENT (<args>)
<action>	::=	*task_id*(<param_list>) | <service_usage>
<param_list>	::=	*variable* : *type* [,param_list]

Tabla 5.5: Sintaxis BNF del *Service Process* y *Service Profile*

cio a través de mensajes (<*msg_cont*>). La Tabla 5.5 contiene la sintaxis correspondiente al perfil, proceso y mensajes de solicitud de un servicio.

B. Semántica del Lenguaje Normativo

Para la expresión del significado de las normas se ha empleado la lógica dinámica descrita en la sección 2.1.2. A continuación se muestra la semántica del lenguaje normativo:

$$\phi : FORBIDDEN\ \alpha \equiv [\alpha]V$$
$$\phi : OBLIGGED\ \alpha \equiv [\neg\alpha]V$$
$$\phi : PEMITTED\ \alpha \equiv [\alpha]\neg V$$

$$\phi : \phi' SANCTION\ \alpha \equiv \phi' \wedge [V]DO(\alpha)$$
$$\phi : \phi' REWARD\ \alpha \equiv \phi' \wedge [\neg V]DO(\alpha)$$

$$\phi : \phi' BEFORE\ \alpha \equiv \phi' \vee DONE(\alpha)$$
$$\phi : \phi' AFTER\ \alpha \equiv [\alpha]\phi'$$
$$\phi : \phi' BETWEEN(\alpha_1, \alpha_2) \equiv [\alpha_1]\phi' \vee DONE(\alpha_2)$$

$$\phi : \phi' IF \beta \equiv \beta \rightarrow \phi'$$

Donde:

- α representa la descripción de una acción.
- β es una descripción de un estado.

- V, $DO(\alpha)$ y $DONE(\alpha)$ son los predicados clásicos para representar los estados de violación, las acciones que se llevarán a cabo a continuación y una acción que ya ha sido realizada, respectivamente.

- ϕ representa una norma.

De este modo una norma (ϕ) del tipo $FORBIDDEN\,\alpha$ se interpreta como que tras la ocurrencia de la acción α se inserta una violación en el estado (V). Si dicha norma ϕ tiene una recompensa asociada $\phi' : \phi REWARD\,\alpha'$ entonces si no se produce la violación ($[\neg V]$) se realizará la acción de recompensa ($DO(\alpha')$). Por último supongamos que la norma ϕ' está activa tras la ocurrencia de un evento α'' ($\phi'' : \phi' AFTER\,\alpha'$); esto implica que la norma ϕ'' está activa tras la ocurrencia de α' ($[\alpha']\phi'$).

C. Ejemplos del Lenguaje Normativo

Tras especificar tanto la sintaxis como la semántica del lenguaje normativo comentaremos algunos ejemplos de normas.

Las normas permisivas, como la detallada en la Norma 1, autorizan a un agente o rol a realizar una acción. En este ejemplo concreto la norma permite a un agente solicitar la adquisición del rol *Subordinated*.

$$\begin{aligned} &agent : ?role\, PERMITTED\, REQUEST\, AcquireRole \\ &MESSAGE(CONTENT(RoleID\ \text{“}Subordinated''\text{)}) \end{aligned} \tag{1}$$

Normalmente las obligaciones y prohibiciones emplean sanciones y recompensas como un método de persuasión. Por ejemplo, la Norma 2 obliga a un agente *Supervisor* a solicitar el servicio de *AddUnit* tras 10 segundos si él es el único *Supervisor* dentro de la organización. Si el agente *Supervisor* no respeta la norma, entonces se le expulsará de la organización como sanción.

$$\begin{aligned} ?agent : Supervisor\, OBLIGED\, REQUEST\, AddUnit& \\ IF\, InformQuantity(\text{“}Supervisor'') = 1& \\ BEFORE(10'')& \\ SANCTION\, oms\, OBLIGED\, PROVIDE\, Expulse& \\ MESSAGE(CONTENT(?agent, \text{“}Supervisor''))& \end{aligned} \tag{2}$$

Por último, la Norma 3 contiene un ejemplo de una norma funcional. Esta norma obliga a un agente *Provider* a registrar su propio servicio de *Search-*

Service.

$$\begin{aligned} &?agent : Provider\, OBLIGED \\ &REGISTER\, SearchService \\ &PROFILE \\ &INPUT(ServiceDescription : String) \\ &OUTPUT(ServiceID : Identifier) \end{aligned} \tag{3}$$

5.4. Implementación del OMS [CJBA09]

Tras describir las principales características del lenguaje normativo propuesto, pasaremos a comentar aspectos relacionados con la implementación del agente OMS (*Organizational Management System* [CJBA09]). Dicho componente de THOMAS permite la gestión de los componentes estructurales de la organización (roles, unidades y normas), así como el control de la dinámica de la organización.

La implementación de los servicios que componen el OMS se ha realizado a través de un sistema de reglas implementado en Jess. Dicho sistema mantiene una base de hechos que representa el estado de la organización y controla aquellas normas que el OMS es capaz de verificar. A continuación se explica el control de las normas verificables realizado por el OMS y se describe la implementación de los servicios.

5.4.1. Implementación del Control Normativo

Tal y como se ha comentado anteriormente, el acceso a los servicios proporcionados por el OMS es controlado por medio de la definición de normas. Sin embargo, el OMS no podrá controlar todas las normas definidas en el sistema, ya que las capacidades del OMS como controlador de normas están limitadas. Por este motivo, antes de comentar los detalles relacionados con la implementación de las normas, se describen a continuación los aspectos relacionados con la verificabilidad de las normas.

A. Verificabilidad de las Normas

El OMS no actúa como mediador en las comunicaciones entre los agentes, situación que sí ocurre con los agentes governor de las Instituciones Electrónicas (IEs). En este sentido el OMS no es una entidad centralizada que controle

<if_condition>	::=	NOT(<cond_exp>) | < cond_exp >
<cond_exp>	::=	<condition> | NOT <condition> | (<condition> AND <cond_exp>) | (<condition> OR <cond_exp>)
<condition>	::=	<variant> <operator> <variant>
<operator>	::=	> | < | >=| <= | =
<variant>	::=	<informative_service>(<args>) | *value*

Tabla 5.6: Sintaxis de las condiciones verificables por el OMS

las interacciones que los agentes llevan a cabo, sino que se encarga de controlar aquellas normas que regulan el acceso a los servicios proporcionados por el OMS.

El conjunto de normas verificables por el OMS es un subconjunto de las normas organizacionales, es decir, de las normas que regulan el acceso a los servicios proporcionados por el OMS. Más concretamente, las normas verificables son permisos y prohibiciones relativas al acceso a los servicios del OMS. Este tipo de normas es controlable ya que el OMS comprueba antes de proporcionar un servicio si el agente solicitante está autorizado para ello. Por otro lado, las obligaciones no pueden ser impuestas por el OMS ya que éste no tiene capacidad para forzar a los agentes a llevar a cabo una acción. Sin embargo, el OMS puede detectar la violación de una obligación y llevar a cabo la correspondiente sanción. Por el contrario, si la norma es respetada entonces el OMS llevará a cabo las acciones especificadas en la recompensa.

La Tabla 5.6 contiene la sintaxis BNF correspondiente a las condiciones verificables por el OMS. Estas condiciones están relacionadas con los servicios informativos del OMS (<*variant*>) que informan acerca de aspectos como la cardinalidad de los roles, etc. Por lo que respecta a los eventos detectables, la Tabla 5.7 muestra su sintaxis. Básicamente estos eventos son la solicitud (<*ServiceRequest*>) y provisión (<*ServiceProvision*>) de los servicios ofrecidos por el OMS. Finalmente, las sanciones y recompensas se definen a través de normas que obligan al OMS a solicitar o proporcionar un determinado servicio.

B. Implementación del Control Normativo [CJA09]

Tras caracterizar el conjunto de normas controlables por el OMS, en este apartado se describe la implementación del mecanismo de razonamiento

<sit>	::=	<event> | (<event> AND <sit>) | (<event> OR <sit>)
<event>	::=	<ServiceRequest> | <ServiceProvision> | *deadline*
<ServiceRequest>	::=	[<entity>] REQUEST <organizational_action>
<ServiceProvision>	::=	(<ServiceRequest>) = <ServiceResult>
<ServiceResult>	::=	Not-Allowed | Error | Provided

Tabla 5.7: Sintaxis de las eventos detectables por el OMS

que permite al OMS considerar las normas verificables en su proceso de toma de decisiones. Este mecanismo de razonamiento se ha implementado de un modo general, permitiendo que cualquier agente que sea informado por la organización acerca de sus normas pueda emplear este mecanismo para establecer:

- El conjunto de servicios que el agente en cuestión puede registrar, es decir, el conjunto de implementaciones de servicios que está autorizado a proporcionar.
- El conjunto de servicios que puede solicitar a otros miembros de la organización, es decir, los servicios que el agente en cuestión está autorizado a utilizar.
- Las condiciones para la provisión y solicitud de servicios, es decir, cuándo puede un servicio ser solicitado o proporcionado, qué agentes pueden hacer uso de él, etc.

Este proceso de razonamiento normativo ha sido implementado a través de un sistema de reglas en Jess[2]. La Figura 5.4 muestra una visión esquemática del proceso. Básicamente, una norma expresada mediante el lenguaje deóntico se transforma en una instancia del *template* norma. Esta instancia se añade como un hecho en el proceso de razonamiento Jess. El sistema de reglas contiene reglas para detectar la activación y satisfacción de las normas. De modo que, cuando las condiciones de activación de una norma se satisfacen la norma pasa a ser una norma activa. El conjunto de normas activas es el que controla el acceso a los servicios del OMS en un momento dado.

El carácter dinámico de las normas hace necesaria una función que permita registrar una nueva norma como una creencia del agente. Del mismo modo, también existe una función para eliminar o modificar una norma existente. El proceso llevado a cabo por esta función es:

[2]http://herzberg.ca.sandia.gov/

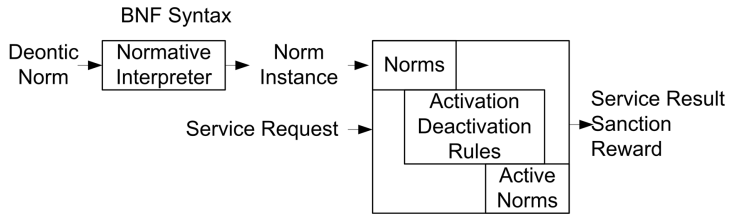

Figura 5.4: Proceso de Razonamiento Normativo

- (i) *Análisis Normativo.* La especificación formal de las normas se traduce en una norma operacional computable por el sistema de reglas.
- (ii)*Detección de la activación de las normas.* Cada vez que se cumplen las condiciones de activación de la norma ésta pasa a formar parte del conjunto de normas activas

A continuación se describen estos dos procesos junto con la explicación de la función que permite al OMS tener en cuenta las normas en su proceso de toma de decisiones.

Análisis Normativo. La fase de análisis normativo tiene dos propósitos diferentes: por un lado, analiza si la nueva norma pertenece al subconjunto de normas que pueden ser controladas por el OMS; por otro lado, realiza la traducción de la expresión formal de la norma en un hecho normativo que puede ser controlado por el sistema de reglas. Este proceso se realiza a través de un intérprete que ha sido automáticamente construido empleando la herramienta JavaCC[3]. A modo de ejemplo, la Norma (1a) expresada mediante el lenguaje normativo descrito en la Sección 5.3, se traduce en el hecho normativo mostrado en (1b). En concreto, esta norma permite a aquellos agentes que asuman el rol supervisor solicitar el servicio de expulsión al OMS.

$$\begin{aligned} &PERMITTED?agent : supervisorREQUESTExpulse \\ &AFTER([?agent]REQUESTAquireRole \end{aligned} \tag{4}$$

$$\updownarrow$$

$$\begin{aligned} &(norm(deonticConceptpermitted)(entityagent?agentrolesupervisor) \\ &(serviceIDExpulse) \end{aligned} \tag{5}$$

[3]https://javacc.dev.java.net/

Detección de la Activación de las Normas. La fase anterior permite al agente OMS traducir la especificación formal de las normas en hechos normativos computables por el sistema Jess. Tal y como se ha comentado anteriormente, no todas las normas están activas en un momento dado. Por este motivo, son necesarias un conjunto de reglas que permitan detectar la activación y desactivación de normas. De modo que el OMS sólo tenga en consideración aquellas normas que están activas en cada momento.

La implementación de las normas implica la detección de las situaciones temporales y las condiciones. Esta implementación está basada en los trabajos propuestos en la implementación de normas dentro de las IE. El modo en el que el OMS interpreta las normas depende del tipo deóntico de la norma (obligación, permiso o prohibición). De modo que las normas se clasifican en: *normas de acceso a los servicios*, que definen permisos y prohibiciones relativas al uso de los servicios proporcionados por el OMS; y *normas de obligación*, que especifican una obligación de realizar una solicitud de servicio. A continuación, se describen detalles los relativos a los dos tipos de normas controladas por el OMS:

- *Normas de acceso a los servicios.* Tal y como se ha mencionado con anterioridad, estas normas permiten la definición de prohibiciones y permisos relativos al uso de los servicios proporcionados por el OMS. De un modo más formal, la semántica operacional de las normas de acceso a los servicios, expresada mediante reglas *Event-Condition-Action* (ECA-reglas), es la siguiente:

$$\begin{aligned} \textbf{on } event_{start} \textbf{ if } if_{condition} \textbf{ do } \oplus permitted(a) \\ \textbf{on } a \textbf{ if } permitted(a) \textbf{ do } \oplus provided(a) \\ \textbf{on } event_{end} \textbf{ do } \ominus permitted(a) \\ \textbf{if } not(if_{condition}) \textbf{ do } \ominus permitted(a) \end{aligned} \tag{6}$$

$$\begin{aligned} \textbf{on } event_{start} \textbf{ if } if_{condition} \textbf{ do } \oplus forbidden(a) \\ \textbf{on } a \textbf{ if } forbidden(a) \textbf{ do } \oplus notAllowed(a) \\ \textbf{on } event_{end} \textbf{ do } \ominus forbidden(a) \\ \textbf{if } not(if_{condition}) \textbf{ do } \ominus forbidden(a) \end{aligned} \tag{7}$$

Las cuatro primeras ECA-reglas (6) describen la semántica de los permisos, mientras que las cuatro últimas (7) representan la semántica de las normas de prohibición. De acuerdo con estas reglas, una norma de acceso a un servicio se controla mediante la definición de cuatro nuevas

reglas en el sistema. La primera regla detecta la activación de la norma e inserta el permiso ($\oplus permitted(a)$) o prohibición ($\oplus forbidden(a)$). Si la acción (a) ocurre y está permitida, entonces el servicio será proporcionado ($\oplus provided(a)$). Por el contrario, si a está prohibida, entonces un hecho del tipo *notAllowed* se inserta en el sistema de reglas. Las ultimas dos reglas eliminan del sistema experto la norma cuando ésta se desactiva, es decir, cuando la condición ($if_{condition}$) no es cierta o cuando se detecta la ocurrencia del hecho de finalización ($event_{end}$). La Tabla 5.8 contiene la sección de código que permite añadir una norma al sistema. La regla (i) de dicho código detecta la ocurrencia del evento de activación y activa la norma. La regla (ii) se encarga de desactivar la norma si la condición de activación no es cierta. Finalmente, la regla (iii) desactiva la norma en el caso de que el evento de finalización se detecte.

- *Normas de obligación.* Este tipo de normas no pueden ser implementadas directamente por el OMS, ya que este agente no puede forzar a otros agentes a llevar a cabo una determinada acción. Sin embargo, el OMS puede persuadir a otros agentes a través de las sanciones y recompensas asociadas a las normas. A continuación se muestra la descripción formal de la semántica de las reglas de obligación:

$$\begin{aligned} \textbf{on } event_{start} \textbf{ if } if_{condition} \textbf{ do } &\oplus expected(a) \\ \textbf{on } a \textbf{ if } expected(a) \textbf{ do } &\ominus expected(a) \bullet reward \\ \textbf{on } event_{end} \textbf{ if } expected(a) \textbf{ do } &\ominus expected(a) \bullet sanction \end{aligned} \tag{8}$$

Tal y como muestra la semántica de una norma de obligación, su implementación consiste en controlar la activación de la obligación. En ese momento el OMS espera la realización de la acción a, es decir, el OMS inserta una nueva expectativa ($\oplus expected(a)$). Si la acción se realiza antes del *deadline* ($event_{end}$) entonces la recompensa se lleva a cabo. En caso contrario el OMS llevará a cabo la sanción.

Razonamiento Normativo. En este apartado se muestra el modo en que el OMS interpreta las normas, es decir, cómo éste delibera acerca del conjunto de normas activas durante su proceso de toma de decisiones. En concreto, cada vez que el OMS recibe una solicitud de servicio por parte de algún agente *cliente*, el OMS debe revisar el conjunto de normas activas para determinar si debe o no proporcionar dicho servicio.

```
(defrule newNorm
?f<-( norm (normid ?normid) (deonticConcept ?deon) (if ?condition)
  (before ?before) (after ?after)...)
(test(or (eq ?deon forbidden)(eq ?deon permitted)))
=>
;i) Activation Rule
(build (str-cat
  '(defrule activateNorm '?normid'
   '?condition'
   '?after'
    =>
   (assert(activenorm(normid '?normid') (deonticConcept '?deon')))))'
  )
)
;ii) Conditional Deactivation Rule
(bind ?ruledelIF (str-cat
  '(defrule deactivateNormIF'?normid'
   ?f<-(activenorm(normid '?normid') (deonticConcept '?deon') )
   (not '?condition')
   =>
   (retract ?f))'
  )
)
;iii) Temporal Deactivation Rule
(build (str-cat   '(defrule deactivateNormBEFORE'?normid
   ?f<-(activenorm(normid '?normid') (deonticConcept '?deon'))
   '?before'
   =>
   (retract ?f))'
  )
))
```

Tabla 5.8: Función que controla la creación de una nueva norma de acceso a servicio

La determinación de las acciones permitidas para una agente que actúe como *cliente* de los servicios del OMS se realiza empleando un criterio conocido como *lex specialis* [BvdT03b]. Este criterio para la precedencia entre normas establece que en caso de conflicto entre normas, las normas más específicas prevalecen sobre el resto de normas. De modo que el proceso de comprobación de las normas comienza comprobando los permisos del agente *cliente*, tal y como se muestra en la Tabla 5.9 (i). Si no hay ninguna norma

se consideran las prohibiciones (ii), ya que los permisos se han definido como una excepción a una norma más general de prohibición. En el último caso, los permisos y prohibiciones asociadas a los roles asumidos por el agente *cliente* se comprueban (iii y iv). Si no existe ninguna norma, entonces el servicio se considera como permitido por defecto (v).

```
(deffunction isAllowed (?agent ?service $?info)
;i)Checks agent addressed permissive norms
(if(<0 (countAgentActiveNorms ?agent permitted ?service))
then(return TRUE))
;ii)Checks agent addressed prohibition norms
(if(<0 (countAgentActiveNorms ?agent forbidden ?service))
then(return FALSE))
;iii)Checks role addressed permissive norms
(if(<0 (countRoleActiveNorms ?agent permitted ?service))
then(return TRUE))
;iv)Checks role addressed prohibition norms
(if(<0 (countRoleActiveNorms ?agent forbidden ?service))
then(return FALSE))
;v)None norm is found
(return TRUE))
```

Tabla 5.9: Código de la función que permite comprobar si una acción está permitida

5.4.2. Implementación de los Servicios del OMS

En la implementación de los servicios se debe tener en consideración el registro del estado organizacional dentro del sistema de reglas así como la existencia de normas que regulan el control a dichos servicios. Cuando se envía una petición de servicio al OMS, éste registra el hecho en el sistema y lleva a cabo la función correspondiente a la provisión del servicio.

La Figura 5.5 contiene la sección de código correspondiente a la función que lleva a cabo el proceso correspondiente a la gestión de una solicitud de servicio:

- (i) En primer lugar debe comprobarse el contexto normativo para determinar si el agente *cliente* está autorizado o no a realizar dicha solicitud de servicio de acuerdo con el estado institucional. Esta funcionalidad se ha descrito en el apartado anterior (apartado de Razonamiento Normativo).

```
(deffunction serviceRequest (?agentid ?serviceid $?info)
i) check normative context
(if (not(isAllowed ?agentid ?serviceid ?info))
 then
  (assert(servicerequest(agentid ?agentid)(serviceid ?serviceid)
   (info ?info)(state not-allowed)))
  (return 'not-allowed'))
ii) service execution
(if (not(apply ?serviceid ?agentid ?info))
 then
  (assert(servicerequest(agentid ?agentid)(serviceid ?serviceid)
   (info ?info)(state error)))
  (return 'error'))
(assert(servicerequest(agentid ?agentid)(serviceid ?serviceid)
   (info ?info)(state provided)))
(return 'provided'))
```

Figura 5.5: Función que gestiona la solicitud de un servicio

- (ii) Si el agente está autorizado entonces el servicio se proporciona.

Básicamente, el proceso de provisión de un servicio proporcionado por el OMS consiste en una actualización de las reglas o hechos pertenecientes al sistema de reglas. A continuación, a modo de ejemplo, el código del servicio que permite registrar una nueva norma se muestra en la Tabla 5.10. Dicho proceso consiste en los siguientes pasos:

- (i) Verificación de los datos de entrada, es decir, comprobar la adecuación de la sintaxis de la norma así como la unicidad del identificador de la norma.
- (ii) Determinación de la verificabilidad de la norma, es decir, determinar si el control de la norma debe realizarse por el OMS.
- (iii) Comprobación de inconsistencias. Tomando como referencia los trabajos presentados en [KVGCN07, VKN07], la insistencia entre normas se define como una situación en la cual la misma acción se define como permitida u obligatoria y como prohibida al mismo tiempo. La detección *off-line* de todas las inconsistencias no es posible, ya que las condiciones de activación de las normas se basan en la detección de eventos y hechos que pueden ocurrir durante la ejecución. Por el momento, la detección de inconsistencias realizada por el OMS está

restringida a la detección de *inconsistencias estáticas*, que son situaciones en las cuales la misma acción se define incondicionalmente como permitida o prohibida al mismo tiempo.

- (iv) El hecho correspondiente a la nueva norma se añade en el sistema de reglas.

```
(deffunction addnorm (?agentid ?normid ?norminfo)
(if (> 0 (count-query-results getnorm ?normid))
 then(return FALSE))
i) check norm syntax
(bind ?interpreter
 (new NormativeLanguageParser.NormativeInterpreter))
(bind ?content (call ?interpreter checkNorm ?norminfo))
(assert(norm (normid ?normid)(norminfo ?norminfo)))
(if (eq ?content '') then(return FALSE))
ii) checks norm verificable
(bind ?interpreter
 (new omsNormativeLanguageParser.NormativeInterpreter))
(bind ?content (call ?interpreter checkNorm ?norminfo))
(if (eq ?content '') then(return TRUE))
iii)check inconsistencies
(if (eq (checkConsistency ?content) FALSE) then (return FALSE))
iv)assert new organizational norm
(assert-string
 (str-cat (organizationalnorm (normid ' ?normid ') '?content')'))
(return TRUE))
```

Tabla 5.10: Servicio del OMS que permite registrar una nueva norma

El resto de servicios necesarios para la gestión de unidades, roles, normas y la dinámica de la organización se han implementado en el sistema de reglas de un modo similar.

5.5. Conclusiones

A lo largo de esta sección se ha descrito el trabajo realizado dentro del proyecto THOMAS, cuyo objetivo principal consiste en la investigación y desarrollo de la tecnología basada en agentes/sistemas multiagente necesaria para el desarrollo de organizaciones virtuales en entornos abiertos (sistemas

abiertos de servicios basados en agentes). Dicha tecnología incluye la necesidad de una metodología para el análisis y diseño de este tipo de sistemas, la implementación de una plataforma multi-dispositivo de soporte adecuada, así como la inclusión de los servicios e infraestructura necesarios para una gestión inteligente de este tipo de organizaciones. En concreto, el trabajo realizado en esta tesis de máster está directamente relacionado con el desarrollo de una plataforma y los servicios necesarios para gestionar dichas organizaciones virtuales. A modo de resumen el trabajo realizado consiste en:

1. Definición de un *modelo de organización virtual* basado en cuatro conceptos clave: la unidad organizativa, el servicio, el entorno y la norma.

2. Definición de una *Arquitectura de Sistemas MultiAgent* que ofrece soporte para dichas organizaciones virtuales.

3. Desarrollo del componente de dicha arquitectura encargado de gestionar las organizaciones (OMS). Esta tarea se descompone a su vez en:

   a) Definición de los servicios proporcionados por el OMS.

   b) Definición del lenguaje normativo.

   c) Implementación del mecanismo de provisión de servicios así como de las herramientas que permiten el control de normas.

El siguiente capítulo contiene un caso real de sistema multiagente abierto empleado para validar la propuesta realizada en esta tesis de máster. En concreto, se trata de un sistema de gestión de viajes, donde distintas entidades requieren uno o varios servicios de otras entidades, llamadas oferentes. Estos oferentes representan a cadenas hoteleras, compañías áreas, etc., que ofrecen información sobre hoteles, vuelos, así como la posibilidad de realizar reservas e incluso el pago adelantado de las mismas.

Capítulo 6

Caso de Estudio

En el presente capítulo se desarrolla un caso de estudio de un sistema de agencia de viajes, empleando el modelo de organización virtual propuesto en este trabajo, a fin de evaluar la validez de la propuesta. Para ello se realiza el modelado de los componentes estructurales, funcionales, normativos y del entorno del sistema. Por otro lado, a fin de validar la utilidad de la arquitectura THOMAS para la implementación de sistemas abiertos se ha realizado un análisis de diferentes escenarios de ejecución para ese caso de estudio.

6.1. Introducción

En esta sección, se ha diseñado un ejemplo de una agencia de viajes empleando el modelo VOM propuesto en la sección 5.1 del capítulo anterior. Este sistema actúa como un punto de encuentro regulado, en el que proveedores y clientes se ponen en contacto entre ellos siendo posible el empleo y provisión de servicios turísticos. Un caso de estudio similar ha sido modelado utilizando las instituciones electrónicas en otros trabajos anteriores [STPW07, DDT+07]. Este caso de estudio ha sido implementado en un programa de demostración que se encuentra disponible junto con un prototipo de la arquitectura THOMAS [1].

En este sistema, los agentes pueden buscar y hacer reservas de hoteles y vuelos. El ejemplo *TravelAgency* es una aplicación que facilita la interconexión entre los clientes (particulares, empresas, agencias de viajes) y los proveedores (cadenas hoteleras, líneas aéreas). Se trata, por tanto, de una arquitectura orientada a servicios. Por lo tanto, el sistema delimita los servicios que puede ofrecer y/o proporcionar cada agente, a través de normas. Sin

[1] http://www.dsic.upv.es/users/ia/sma/tools/Thomas/index.html

embargo, la funcionalidad interna de estos servicios es responsabilidad de los agentes que los ofrezcan.

6.2. Modelado del Sistema [CAJB09]

A lo largo de esta sección se detalla cómo el sistema *TravelAgency* ha sido modelado empleando para ello el modelo VOM (sección 5.1). A continuación se describen las componentes estructurales, funcionales, del entorno y normativas del caso de estudio.

6.2.1. Descripción Estructural

En el sistema *TravelAgency*, se identifican tres roles diferenciados que pueden ser asumidos por los agentes:

- **Customer:** este rol representa a los agentes que solicitan servicios al sistema. Se trata de un rol accesible, por lo tanto puede ser asumido por agentes externos. Teniendo en cuenta los distintos tipos de productos del sistema, este rol se especializa en los roles *HotelCustomer* y *FlightCustomer*.

- **Provider:** este rol ofrece y provee servicios. De modo que un agente *Provider* ofrece un servicio de búsqueda de hoteles y vuelos; y, en algunos casos, un servicio de reserva de habitaciones y plazas de avión. Del mismo modo que ocurría con el rol cliente, los proveedores se especializan en los roles *HotelProvider* y *FlightProvider*.

- **Payee:** este rol proporciona el servicio de pago de las reservas. Representa a una institución bancaria a través de la cual se lleva a cabo el pago por adelantado de las reservas. Se trata de un rol interno, se modo que los agentes externos no podrán adoptarlo.

El caso de estudio se ha modelado como una unidad principal (*TravelUnit*) dentro de la cual hay dos unidades organizativas (*HotelUnit* y *FlightUnit*), que se están dedicadas a servicios relacionados con hoteles y vuelos, respectivamente. Estas unidades asumen el rol de *Provider* dentro de la unidad principal. La Figura 6.1 muestra una descripción parcial de este sistema.

6.2.2. Descripción Funcional

Además de la descripción de la estructura del sistema, dentro de la unidad principal (*TravelAgency*) se ofrecen tres servicios: *TravelSearch*, *Reserve*

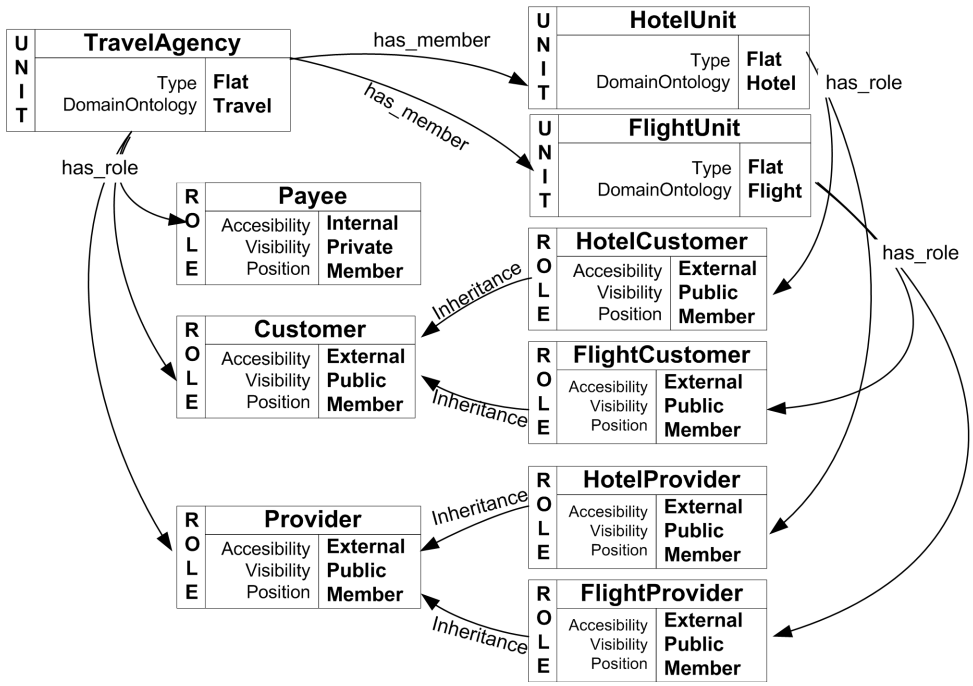

Figura 6.1: Descripción Estructural de la *TravelAgency*

y *Payment*. Los servicios de la agencia de viajes se especializan, dentro de cada unidad, de acuerdo con cada tipo de producto (hoteles y vuelos). En consecuencia, los servicios registrados dentro de la unidad son *HotelUnit* son *SearchHotel* y *ReserveHotel*. Del mismo modo, dentro de la unidad *FlightUnit* se han definido servicios especializados sobre el dominio de los vuelos. La descripción de un servicio consiste en la especificación OWL-S del mismo. La especificación de los perfiles de los servicios se han ampliado para poder representar los roles cliente y proveedor, así como el identificador de la unidad dentro de la cual se proporciona el servicio. Los perfiles de los servicios ofrecidos dentro de la unidad *TravelAgency* están contenidos en la Tabla 6.1. Las Tablas 6.2 y 6.3 contienen una breve descripción de los servicios ofrecidos dentro de las unidades *HotelUnit* y *FlightUnit*, respectivamente.

6.2.3. Descripción del Entorno

Las organizaciones de agentes no están aisladas, por el contrario, se encuentran situadas en un entorno concreto. En concreto, dentro de la *TravelAgency* hay una base de datos que contiene la información geográfica. Por lo tanto; este elemento, perteneciente a la dimensión del entorno, puede mode-

Service		**Description**		**UnitID**
TravelSearch		Search for travel information		TravelAgency
ClientRole	**ProviderRole**	**Inputs**	**Outputs**	
Customer	Provider	city:string country:string	[city ok] company:string location:string price:float	[not in city] error
Service		**Description**		**UnitID**
Reserve		Make a reservation		TravelAgency
ClientRole	**ProviderRole**	**Inputs**	**Outputs**	
Customer	Provider	company:string location:string date:time numRsv:integer	[available] RsvTicket:Reserve price:float [advanced payment] IBAN:integer	[not available] "completed"
Service		**Description**		**UnitID**
Payment		Advanced payment of a reservation		TravelAgency
ClientRole	**ProviderRole**	**Inputs**	**Outputs**	
Customer	Payee	RsvTicket:Reserve amount:float IBAN:string bankcard:string	[ok] Invoice:string	[error] bankError

Tabla 6.1: Perfiles de los Servicios ofrecidos en *TravelAgency*

Service		**Description**		**UnitID**
SearchHotel		Search for information about hotels in a city		HotelUnit
ClientRole	**ProviderRole**	**Inputs**	**Outputs**	
HotelCustomer	HotelProvider	city:string country:string category:integer	[city ok] HotelChain:string HotelName:string RoomRate:float Address:string	[not in city] error
Service		**Description**		**UnitID**
ReserveHotel		Make an hotel reservation		HotelUnit
ClientRole	**ProviderRole**	**Inputs**	**Outputs**	
HotelCustomer	HotelProvider	company:string location:string date:time numRsv:integer nights:integer	[available] RsvTicket:Reserve price:float [advanced payment] IBAN:string	[not available] "completed" OptDate:date

Tabla 6.2: Perfiles de los Servicios ofrecidos en *HotelUnit*

larse como un *recurso* perteneciente a la unidad *TravelAgency* que puede ser empleado por todos los miembros de dicha unidad.

6.2.4. Descripción Normativa [ACJB08]

Este ejemplo está regulado a través de diferentes tipos de normas como: normas de incompatibilidad entre roles, normas de cardinalidad, normas que determinan la funcionalidad de los roles, etc. A continuación se muestran ejemplos de normas para el caso de estudio *TravelAgency*. Dichas normas han sido especificadas mediante el lenguaje normativo descrito en la sección

Service		**Description**		**UnitID**
SearchFlight		Search for information about flights		HotelFlight
ClientRole	**ProviderRole**	**Inputs**	**Outputs**	
FlightCustomer	FlightProvider	cityFrom:string countryFrom:string cityTo:string countryTo:string date:time	[cityFrom, cityTo ok] FlightCompany:string FlightType:string price:float HourDept:time HourArr:time NumConnect:integer	[not in city] error
Service		**Description**		**UnitID**
ReserveFlight		Make a flight reservation		HotelFlight
ClientRole	**ProviderRole**	**Inputs**	**Outputs**	
FlightCustomer	FlightProvider	company:string location:string numRsv:integer date:time cityTo:string HourDept:time HourArr:time	[available] RsvTicket:Reserve price:float [advanced payment] IBAN:string	[not available] "completed"

Tabla 6.3: Perfiles de los Servicios ofrecidos en *FlightUnit*

5.3 del capítulo anterior:

- **Normas de Incompatibilidad entre Roles**.
  - El rol *Payee* es incompatible con los roles *Provider* y *Customer*.

    FORBIDDEN Payee REQUEST AcquireRole
    MESSAGE (CONTENT (role "Provider", unit "TravelAgency")
    OR CONTENT (role "Customer", unit "TravelAgency"))

  - El rol *Customer* es incompatible con el rol *Provider*.

    FORBIDDEN Customer REQUEST AcquireRole
    MESSAGE (CONTENT (role "Provider", unit "TravelAgency"))

- **Normas de Cardinalidad.**
  - Cardinalidad del rol *Customer*=(0,20). La cardinalidad máxima del rol se especifica mediante la siguiente norma:

    FORBIDDEN Member REQUEST AcquireRole
    MESSAGE (CONTENT (role "Customer", unit "TravelAgency")
    IF QUANTITY(Customer)$\geqslant$20

  - Cardinalidad del rol *Provider*=(3,∞). La restricción de cardinalidad mínima impuesta sobre el rol *Provider* ha sido modelada a través de una única norma. Esta norma prohíbe a los agentes solicitar el servicio *TravelSearch* si el número de proveedores es menor que 3:

FORBIDDEN Member REQUEST TravelSearch
IF QUANTITY(Provider)⩽3

- **Normas Funcionales.**

  1. **Funcionalidad Incondicional.**
     - El rol *Customer* no puede registrar ningún servicio; sólo está autorizado a solicitar servicios.

       FORBIDDEN Customer REGISTER TravelSearch
       PROFILE TravelSearchProfile
       FORBIDDEN Customer REGISTER Reserve
       PROFILE ReserveProfile

  2. **Funcionalidad Condicional.**
     - El rol *Customer* no puede solicitar el servicio *Reserve* para una determinada información (hotel o vuelo) si previamente no ha realizado una búsqueda acerca de dicha información.

       OBLIGED Customer REQUEST TravelSearch
       MESSAGE (CONTENT (city ?c, country ?co))
       BEFORE Customer REQUEST Reserve
       MESSAGE(CONTENT(company ?cm city ?c))

6.3. Escenario de Ejecución [EdV09]

Esta sección ilustra un posible escenario de ejecución del caso de estudio. Este ejemplo ilustra cómo el empleo de los estándares de servicios web para proporcionar y publicitar los servicios permite a los agentes externos descubrir y hacer uso las funcionalidades ofrecidas por la propia arquitectura THOMAS y servicios ofrecidos por agentes. En concreto, dicho escenario muestra cómo un agente externo se une a la plataforma THOMAS, se registra como cliente y lleva a cabo una solicitud de servicio (Figura 6.2).

C1 es un agente que representa a un cliente interesado en información sobre hoteles. Los agentes externos entran a formar parte de la plataforma THOMAS mediante el envío al OMS de una solicitud del servicio de adquisición del rol *Member* dentro de la unidad *Virtual* (Figura 6.2 mensaje 1). A continuación, el OMS comprueba todas las restricciones (la existencia de la unidad y el rol, las restricciones de incompatibilidad entre roles, etc.) y registra al agente C1 como un nuevo miembro de THOMAS (mensaje 2).

El nuevo agente de cliente (C1) pide al SF el servicio *SearchService* a fin de encontrar los servicios de su interés (mensaje 3). El resultado de este servicio figura en el mensaje 4. Entonces, C1 emplea el servicio *InformService*

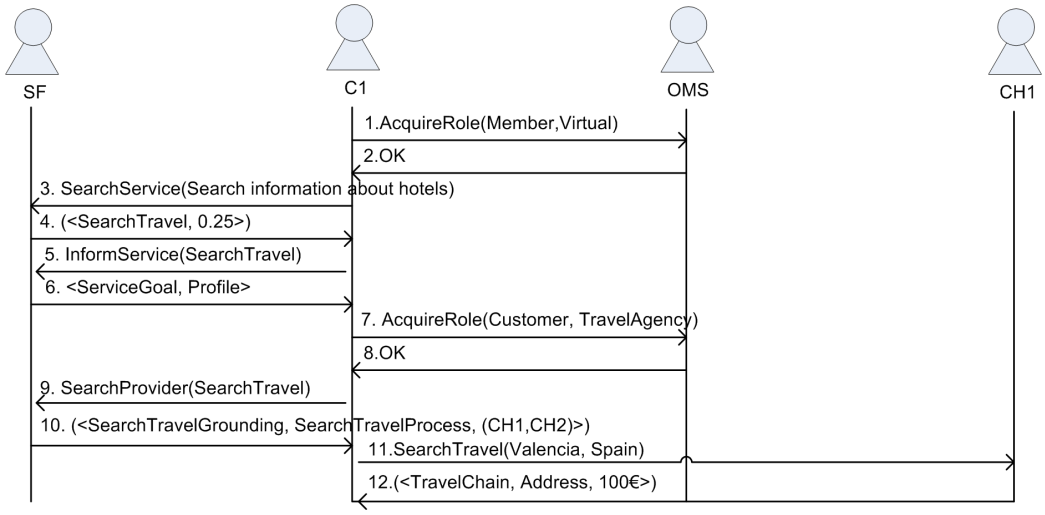

Figura 6.2: Registro de un cliente y petición de servicio

para saber dentro de qué unidad de la organización se presta el servicio de *SearchTravel* (mensajes 5 y 6). Tal y como se indica en el perfil de servicio que figura en la Tabla 6.1, C1 debe adquirir el rol *Customer* para poder solicitar dicho servicio (mensajes 7 y 8).

Una vez que el agente C1 desempeña papel de *Customer*, emplea el servicio *SearchProvider* con el fin de saber quiénes son los proveedores de este servicio (mensajes 9 y 10). Como muestra el mensaje 10 de la Figura 6.2, se ha registrado previamente una implementación del servicio *SearchTravel*. Esta implementación cuenta con dos proveedores distintos (CH1 y CH2). C1 decide realizar una petición del servicio *SearchTravel* al agente CH1 (mensajes 11 y 12), quien le contesta con la información solicitada.

6.4. Conclusiones

En este capítulo se ha mostrado la aplicabilidad del modelo propuesto en el capítulo anterior para dar soporte al diseño de un sistema multiagente abierto. En concreto, se ha realizado el diseño de la estructura organizativa, la funcionalidad y el entorno del sistema, así como la especificación de las normas que regulan un caso de estudio correspondiente a una agencia de viajes. Por otro lado, varios escenarios de ejecución del ejemplo de la agencia de viajes han sido empleados para validar la arquitectura THOMAS propuesta en el capítulo anterior. Un ejemplo de estos escenarios se ha descrito en la sección anterior. Además dicho caso de estudio se ha tomado como ba-

se para implementar un programa de demostración que permite validar la implementación de la arquitectura THOMAS [2].

[2] http://www.dsic.upv.es/users/ia/sma/tools/Thomas/index.html

Capítulo 7

Trabajo Futuro y Conclusiones

En este último capítulo se recogen las principales conclusiones de este trabajo, las líneas futuras de investigación, así como las publicaciones relacionadas, producidas como resultado del trabajo de investigación realizado.

7.1. Aportaciones

El presente trabajo se enmarca en el contexto del proyecto THOMAS (comentado en la introducción de la tesis de máster). Dicho proyecto tiene entre sus objetivos el desarrollo de un arquitectura de agentes abstracta que permita el modelado de sistemas organizativos abiertos. De acuerdo a los objetivos que nos planteamos en un principio, las principales aportaciones de esta tesis, contenidas en la segunda parte de este documento, son:

- Un *Modelo Normativo* que extiende los trabajos anteriores tratando de ofrecer una solución a la problemática de los sistemas abiertos. Además este modelo permite la formalización de las restricciones deónticas mediante el empleo de la lógica RTAL.

- Diseño e implementación del *Organization Management System* (OMS), encargado de la gestión del ciclo de vida de las organizaciones. Gracias a este componente, se pueden definir unidades organizativas con una estructura interna determinada. Para estas unidades se establecen las normas que controlan los comportamientos de sus entidades, así como los servicios que se requieren y se ofrecen en ellas.

Las ventajas que nos ofrece el OMS como elemento gestor de organizaciones frente a otras alternativas son:

1. El OMS ofrece soporte para las organizaciones virtuales como una abstracción social que permite coordinar a los agentes autónomos. Mediante el empleo de normas es posible establecer un control en el comportamiento de los agentes y asegurar el orden social.

2. La funcionalidad del OMS se ha diseñado siguiendo una aproximación orientada a servicios. De modo que los servicios del OMS se han implementado como servicios Web independientes de la plataforma de agentes, con lo que es posible la coordinación entre agentes que pertenecen a plataformas distintas.

3. Los servicios del OMS se registran y publicitan a través de la arquitectura THOMAS permitiendo que agentes externos los descubran y sean capaces de hacer uso de ellos.

4. Los servicios del OMS permiten a los agentes cambiar en tiempo de ejecución la estructura de la organización; permitiendo de este modo adaptar de forma dinámica la organización a los cambios que se produzcan en el entrono.

7.2. Líneas Abiertas

Como conclusión principal podemos señalar que todavía queda mucho camino y trabajo por realizar dentro del área de los SMA Normativos. Si bien en los últimos años el área de los Sistemas Normativos es uno de los campos que está recibiendo un mayor interés, se está muy lejos todavía del objetivo de proporcionar marco normativo que permita a los agentes el razonamiento en entornos abiertos teniendo en cuenta aspectos como la confianza y la reputación.

Los SMA Normativos han sido descritos y analizados tradicionalmente empleando lógica deóntica, describiendo las relaciones lógicas entre las obligaciones y los permisos. Sin embargo, existe una separación entre las teorías de la lógica deóntica y los SMA Normativos. A continuación se comentan y clasifican los desafíos y retos abiertos con el objetivo de lograr superar esta separación. Estas cuestiones pendientes las podemos clasificar en tres categorías básicas y estrechamente relacionadas:

1. Desarrollo de un lenguaje computacional para la especificación de los Sistemas Normativos. Uno de los aspectos fundamentales para conseguir alcanzar la implantación de las normas dentro de la teoría de los

SMA es la existencia de un lenguaje que permita especificar formalmente los sistemas normativos. Esta área puede a su vez dividirse en dos secciones:

a) Desarrollo de un lenguaje computacional para la especificación de los Sistemas Normativos. Este lenguaje no sólo debe tener la expresividad suficiente como para permitir la definición de sistemas normativos complejos; sino que debe poseer propiedades computacionales que permitan razonar sobre el sistema normativo.

b) Razonamiento sobre el Sistema Normativo. Dentro de este apartado se encuentran los trabajos relacionados con el análisis de los Sistemas Normativos tales como: el *model-checking*, la verificación y análisis de las especificaciones, consistencia, simplificación, etc.

2. Razonamiento Normativo Individual. Los agentes normativos son aquellos que son conscientes de las normas a las que están sujetos y que, por lo tanto, son capaces de tenerlas en consideración para tomar sus decisiones. Dentro de este ámbito las principales cuestiones abiertas son:

   a) Estudio de aspectos sobre la adopción de las normas por parte de los agentes, así como de la resolución de conflictos entre los intereses de los agentes y las normas del sistema.

   b) Técnicas de persuasión acerca de la adopción de normas: sanciones y recompensas, autoridad y delegación.

   c) Estudios sobre la emergencia de las normas dentro de sociedades.

3. Implementación de SMA Normativos. Desarrollo de las herramientas software necesarias para permitir la implementación de aplicaciones reales de SMA Normativos. Estas herramientas deben ofrecer facilidades a las tareas de:

   a) Diseño de aplicaciones. Sirviendo de guía a los diseñadores de los sistemas, estas herramientas deben facilitar el proceso de especificación de las sociedades de agentes, así como la definición de las normas que permitan coordinar la acción de los agentes.

   b) Implementación de las aplicaciones multiagente normativas. Las herramientas de implementación no sólo deben ofrecer soporte para los agentes autónomos normativos, sino que deben permitir realizar cambios dinámicos en la estructura normativa con el fin de facilitar al sistema adaptarse a los cambios que se produzcan en el entorno.

El trabajo realizado en esta tesis de máster supone un avance en esta última línea de trabajo. En concreto, el objetivo fundamental de este trabajo es permitir la generación de organizaciones virtuales de agentes en entornos abiertos. La plataforma THOMAS ofrece soporte para la implementación de aplicaciones multiagente. Estas aplicaciones emplean conceptos normativos como un elemento que permite la coordinación y cooperación entre agentes autónomos y heterogéneos. El posterior estudio de cómo estos agentes toman en consideración dichas normas en su proceso de toma de decisiones es una cuestión que se ha plantado como trabajo futuro.

7.3. Publicaciones Asociadas

[CVA09] N. Criado, V.Botti, E. Argente
A Normative Model For Open Agent Organizations
International Conference on Artificial Intelligence (ICAI). In Press (2009)

[EdV09] E. del Val, N. Criado, M. Rebollo, E. Argente, V. Julian
Service-Oriented Framework for Virtual Organizations
International Conference on Artificial Intelligence (ICAI). In Press (2009)

[HCAJ09] S. Heras, N. Criado, E. Argente and V. Julian
A Dialogue-Game Approach for Norm-based MAS Coordination
International Conference on Hybrid Artificial Intelligence Systems (HAIS). Vol. 1 N. 5572 pp. 468-475. (2009)

[CJBA09] N. Criado, V. Julian, V. Botti and E. Argente
A Norm-based Organization Management System
International Workshop Coordination, Organizations, Institutions and Norms (COIN). Pp. 1-16.(2009)

[CAJB09] N. Criado, E. Argente, V. Julian and V. Botti
Designing Virtual Organizations
International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS) Vol. 55 pp. 319-328. (2009)

[CJA09] N. Criado, V. Julian and E. Argente
Towards the Implementation of a Normative Reasoning Process

International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS) Vol. 55 pp. 440-449. (2009)

[CAJB08] N. Criado, E. Argente, V. Julian and V. Botti
Servicios Organizacionales Para La Plataforma De Agentes Spade
IEEE Latin America Transactions Vol. 6 N. 6 pp. 550-555. (2008)

[ACBJ08] E. Argente, N. Criado, V. Botti and V. Julian
Norms for Agent Service Controlling
European Workshop on Multi-Agent Systems (EUMAS) pp. 1-15. (2008)

[ACJB08] E. Argente, N. Criado, V. Julian and V. Botti
Designing Norms in Virtual Organizations
Congrés Internacional de l'Associació Catalana d'Intel·ligència Artificial (CCIA) Vol. 184 pp. 16-23. (2008)

[CAJB07] N. Criado, E. Argente, V. Julian and V. Botti
Organizational Services for SPADE agent platform
Workshop Internacional sobre Aplicaciones Prácticas de Agentes y Sistemas Multiagente (IWPAAMS) Vol. 1 pp. 31-40. (2007)

Bibliografía

[ACBJ08] E. Argente, N. Criado, V. Botti, and V. Julian. Norms for agent service controlling. In *EUMAS*, 2008.

[ACJB08] E. Argente, N. Criado, V. Julian, and V. Botti. Designing norms in virtual organizations. In *CCIA*, volume 184, pages 16–23. IOS Press, 2008.

[AdHRA+07] Thomas Agotnes, Wiebe Van der Hoek, Juan A. RodrÃguez-Aguilar, Carles Sierra, and Michael Wooldridge. The simple normative system language (snl). pages 1–12, 2007.

[ADM07] H.M. Aldewereld, F.P.M. Dignum, and J-J.Ch. Meyer. From norms to interaction patterns: Deriving protocols for agent institutions. In *Workshop on Programming Multi-Agent Systems (ProMAS2007)*, pages 22–37, 2007.

[AJB08] E. Argente, V. Julian, and V. Botti. Mas modelling based on organizations. In *9th Int. Workshop on Agent Oriented Software Engineering (AOSE08)*, pages 1–12, 2008.

[Alc96] C.E. Alchourrón. On law and logic. *Ratio Juris*, 9(4):331–348, 1996.

[Ald07] H.M. Aldewereld. *Autonomy vs. Conformity : an Institutional Perspective on Norms and Protocols*. PhD thesis, Universiteit Utrecht, 2007.

[And58] A. R. Anderson. A reduction of deontic logic to alethic modal logic. *Mind*, 67:100–103, 1958.

[APA+07] E. Argente, J. Palanca, G. Aranda, V. Julian, V. Botti, A. García-Fornes, and A. Espinosa. Supporting agent organizations. In *CEEMAS'07*, volume 4696 of *LNAI*, pages 236–245. Springer, 2007.

[ÅvdHW07] Thomas Ågotnes, Wiebe van der Hoek, and Michael Wooldridge. Normative system games. In *AAMAS*, page 129, 2007.

[ÅvdHW08] Thomas Ågotnes, Wiebe van der Hoek, and Michael Wooldridge. Robust normative systems. In *AAMAS (2)*, pages 747–754, 2008.

[BHvdT05] Guido Boella, Joris Hulstijn, and Leendert W. N. van der Torre. Virtual organizations as normative multiagent systems. In *HICSS*. IEEE Computer Society, 2005.

[BvdT03a] G. Boella and L. van der Torre. Permissions and obligations in hierarchical normative systems. In *Proc. of ICAIL 03, Edimburgh*, 2003.

[BvdT03b] G. Boella and L. van der Torre. Permissions and obligations in hierarchical normative systems. In *Proc. of ICAIL 03, Edimburgh*, 2003.

[BvdT04] Guido Boella and Leendert van der Torre. Regulative and constitutive norms in normative multiagent systems. In *KR2004: Principles of Knowledge Representation and Reasoning*, pages 255–265. AAAI Press, 2004.

[BvdT05] Guido Boella and Leendert W. N. van der Torre. Constitutive norms in the design of normative multiagent systems. In *CLIMA VI*, volume 3900 of *LNCS*, pages 303–319, 2005.

[BvdT08] Guido Boella and Leendert van der Torre. Substantive and procedural norms in normative multiagent systems. *J. Applied Logic*, 6(2):152–171, 2008.

[BvdTV08] Guido Boella, Leendert van der Torre, and Harko Verhagen. Introduction to the special issue on normative multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 17:1–10, 2008.

[CAJB07] N. Criado, E. Argente, V. Julian, and V. Botti. Organizational services for spade agent platform. In *IWPAAMS*, volume 1, pages 31–40. Universidad de Salamanca, 2007.

[CAJB08] N. Criado, E. Argente, V. Julian, and V. Botti. Servicios organizacionales para la plataforma de agentes spade. *IEEE Latin America Transactions*, 6:550–555, 2008.

[CAJB09] N. Criado, E. Argente, V. Julian, and V. Botti. Designing virtual organizations. In *PAAMS*, volume 55 of *Advances in Soft Computing*, pages 440–449, 2009.

[CC95] Allan Cheng and Allan Cheng. Complexity results for model checking, March 10 1995.

[CGJ+ss] C. Carrascosa, A. Giret, V. Julian, M. Rebollo, E. Argente, and V. Botti. Service oriented mas: An open architecture. In *AAMAS 09*, In Press.

[CGP99] Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. *Model Checking*. The MIT Press, Cambridge, Massachusetts, 1999.

[CJA09] N. Criado, V. Julian, and E. Argente. Towards the implementation of a normative reasoning process. In *PAAMS*, volume 55, pages 319–328, 2009.

[CJBA09] N. Criado, V. Julian, V. Botti, and E. Argente. A norm-based organization management system. In *AAMAS Workshop on Coordination, Organization, Institutions and Norms in Agent Systems (COIN)*, 2009.

[Cra05] Stephen Cranefield. A rule language for modelling and monitoring social expectations in multi-agent systems. In *AAMAS Workshops*, volume 3913 of *Lecture Notes in Computer Science*, pages 246–258. Springer, 2005.

[Cra07] Stephen Cranefield. Modelling and monitoring social expectations in multi-agent systems. In *Coordination, Organizations, Institutions, and Norms in Agent Systems II*, number 4386 in LNAI, pages 308–321. Springer, Berlin Heidelberg, 2007.

[CVA09] N. Criado, V.Botti, and E. Argente. A normative model for open agent organizations. In *International Conference on Artificial Intelligence (ICAI)*, 2009.

[CW07] Stephen Cranefield and Michael Winikoff. Verifying social expectations by model checking truncated paths. 2007.

[Daf03] R. Daft. *Organization Theory and Design*. South-Western College, 2003.

[DD06] V. Dignum and F. Dignum. A landscape of agent systems for the real world. Technical report, Inst. Information and Computing Sciences, Utrecht University, 2006.

[DDT+07] Frank Dignum, Virginia Dignum, John Thangarajah, Lin Padgham, and Michael Winikoff. Open agent systems??? In Michael Luck and Lin Padgham, editors, *AOSE*, volume 4951 of *Lecture Notes in Computer Science*, pages 73–87. Springer, 2007.

[Dig99] Frank Dignum. Autonomous agents with norms. *Artif. Intell. Law*, 7(1):69–79, 1999.

[DMWK96] Frank Dignum, John-Jules Ch. Meyer, Roel Wieringa, and Ruurd Kuiper. A modal approach to intentions, commitments and obligations: Intention plus commitment yields obligation. In *Deontic Logic, Agency and Normative Systems, DEON '96: Third International Workshop on Deontic Logic in Computer Science, Sesimbra, Portugal, 11-13 January 1996*, Workshops in Computing, pages 80–97. Springer, 1996.

[dS07] Viviane Torres da Silva. Implementing norms that govern non-dialogical actions. In *Normative Multi-agent Systems*, number 07122 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2007. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany.

[DVSD04] Virginia Dignum, Javier Vázquez-Salceda, and Frank Dignum. OMNI: Introducing social structure, norms and ontologies into agent organizations. In *PROMAS*, volume 3346 of *Lecture Notes in Computer Science*, pages 181–198. Springer, 2004.

[EdV09] M. Rebollo E. Argente V. Julian E. del Val, N. Criado. Service-oriented framework for virtual organizations. In *International Conference on Artificial Intelligence (ICAI)*, 2009.

[Eme90] E. Allen Emerson. Temporal and modal logic. In *Handbook of Theoretical Computer Science. Volume B*, pages 995–1072. North-Holland, Amsterdam, 1990.

[Est02] Marc Esteva. *Electronic Institutions: From Especification To Development*. PhD thesis, Universidad Politécnica de Cataluña, 2002.

[FCB+08] Carolina Felicissimo, Ricardo Choren, Jean-Pierre Briot, Carlos Lucena, Caroline Chopinaud, and Amal El Fallah Seghrouchni. *Providing contextual norm information in open multi-agent systems*, volume 4898 of *Lecture Notes in Artificial Intelligence*, pages 19–36. Springer-Verlag, 2008.

[FCBL06] Carolina Felicissimo, Ricardo Choren, Jean-Pierre Briot, and Carlos Lucena. Supporting regulatory dynamics in open mas. In *International Workshop on Coordination, Organization, Institutions and Norms in Agent Systems (COIN'2006)*, Hakodate, Japon, 5 2006. 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'2006), ACM Press.

[FGM03] Jacques Ferber, Olivier Gutknecht, and Fabien Michel. From agents to organizations: An organizational view of multi-agent systems. In *AOSE*, volume 2935 of *Lecture Notes in Computer Science*, pages 214–230. Springer, 2003.

[Fos01] Ian T. Foster. The anatomy of the grid: Enabling scalable virtual organizations. In *CCGRID*, pages 6–7. IEEE Computer Society, 2001.

[GC07] Andrés García-Camino. Ignoring, forcing and expecting concurrent events in electronic institutions. In *COIN III: Coordination, Organization, Institutions and Norms in Agent Systems. Revised Selected Papers from the 2007 Workshop Series*, volume 4870 of *Lecture Notes in Computer Science*, pages 15–26. Springer, 2007.

[GCNRA05] Andrés García-Camino, Pablo Noriega, and Juan A. Rodríguez-Aguilar. Implementing norms in electronic institutions. In *EUMAS*, pages 482–483, 2005.

[GCRASV06] Andrés García-Camino, Juan A. Rodríguez-Aguilar, Carles Sierra, and Wamberto Weber Vasconcelos. Norm-oriented programming of electronic institutions. In *AAMAS*, pages 670–672. ACM, 2006.

[GLL+04] Enrico Giunchiglia, Joohyung Lee, Vladimir Lifschitz, Norman McCain, and Hudson Turner. Nonmonotonic causal theories. *Artificial Intelligence*, 153(5–6):49–104, 2004.

[gvdHRA+07] Thomas Ågotnes, Wiebe van der Hoek, Juan A. Rodriguez-Aguilar, Carles Sierra, and Michael Wooldridge. On the logic of normative systems. In *Normative Multi-agent Systems*, number 07122 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2007.

[HCAJ09] S. Heras, N. Criado, E. Argente, and V. Julian. A dialogue-game approach for norm-based mas coordination. In *4th International Conference on Hybrid Artificial Intelligence Systems (HAIS-09)*, volume 1 of *LNAI*, pages 468–475. Springer, 2009.

[HL04] Bryan Horling and Victor Lesser. A survey of multi-agent organizational paradigms. *Knowl. Eng. Rev.*, 19(4):281–316, 2004.

[HPvdT07] Jörg Hansen, Gabriella Pigozzi, and Leendert van der Torre. Ten philosophical problems in deontic logic. In *Normative Multi-agent Systems*, number 07122 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2007.

[Kan72] Stig Kanger. Law and logic. *Theoria*, 38:105–132, 1972.

[Kri63] S. Kripke. Semantical considerations on modal logic. *Acta Philosophica Fennica*, 16:83–94, 1963.

[KVGCN07] Martin J. Kollingbaum, Wamberto Weber Vasconcelos, Andrés García-Camino, and Timothy J. Norman. Conflict resolution in norm-regulated environments via unification and constraints. In *DALT*, volume 4897 of *Lecture Notes in Computer Science*, pages 158–174. Springer, 2007.

[Lam94] Leslie Lamport. The temporal logic of actions. *Transactions on Programming Languages and Systems (TOPLAS)*, 16(3):872–923, May 1994.

[Lin77] Lars Lindahl. *Position and Change: A Study in Law and Logic.* D. Reidel Publishing Co., Dordrecht, 1977.

[LMSW05] Michael Luck, Peter McBurney, Onn Shehory, and Steve Willmott. Agent technology: Computing as interaction: A roadmap for agent-basedcomputing. Technical report, Agentlink, 2005.

[LyLL02] F. López y López and M. Luck. A model of normative multi-agent systems and dynamic relationships. In *RASTA*, volume 2934 of *Lecture Notes in Computer Science*, pages 259–280. Springer, 2002.

[LyLL03] F. López y López and M. Luck. Modelling norms for autonomous agents. In *ENC*, pages 238–245. IEEE Computer Society, 2003.

[Mey87] J.-J. Ch. Meyer. A different approach to deontic logic: deontic logic viewed as a variant of dynamic logic., December 1987.

[Mit06] Matthias Mittag. A legal theoretical approach to criminal procedure law: The structure of rules in the german code of criminal procedure. *German Law Journal*, 8, 2006.

[Mor56] R. T. Morris. A typology of norms. *American Sociological Review*, 31:610–613, 1956.

[MT95] Yoram Moses and Moshe Tennenholtz. Artificial social systems. *Computers and Artificial Intelligence*, 14(6), 1995.

[Reb04] Miguel Rebollo. *Imaginación en Tiempo Real. Representación y Gestión de Creencias Temporales para Agentes en Entornos Dinámicos*. PhD thesis, DSIC, Universidad Politécnica de Valencia, 2004.

[RG91] A. S. Rao and M. P. Georgeff. Modeling rational agents within a BDI-architecture. Technical Report 14, Australian AI Institute, Carlton, Australia, 1991.

[RIB96] M. Tennenholtz R. I. Brafman. On partially controlled multi-agent systems. *Journal of Artificial Intelligence Research*, 4:477–507, 1996.

[SC06] Marek J. Sergot and Robert Craven. The deontic component of action language nC+. In *DEON*, volume 4048 of *Lecture Notes in Computer Science*, pages 222–237. Springer, 2006.

[Ser98] Marek J. Sergot. Normative positions. In *Norms, Logics and Information Systems. New Studies in Deontic Logic and Computer Science*, pages 289–310. IOS Press, Amsterdam, 1998.

[Sin99] Munindar P. Singh. An ontology for commitments in multi-agent systems. *Artif. Intell. Law*, 7(1):97–113, 1999.

[STPW07] Carles Sierra, John Thangarajah, Lin Padgham, and Michael Winikoff. Designing institutional multi-agent systems. *Agent-Oriented Software Engineering VII*, LNCS 4405:84–103, 2007.

[The02] G. Therborn. Back to norms! on the scope and dynamics of norms and normative action. *Current Sociology*, 50:863–880, 2002.

[ThT01] Leendert Van Der Torre and Yao hua Tan. Dynamic normative reasoning under uncertainty: How to distinguish between obligations under uncertainty and prima facie obligations, February 08 2001.

[Tuo95] Raimo Tuomela. *The Importance of Us*. Stanford University Press, 1995.

[Tuo08] Raimo Tuomela. *Service-Oriented Modeling: Service Analysis, Design, and Architecture*. John Wiley & Sons, Inc., 2008.

[Val98] A. Valmari. The state explosion problem. *Lecture Notes in Computer Science*, 1491:429–528, 1998.

[VKN07] Wamberto Weber Vasconcelos, Martin J. Kollingbaum, and Timothy J. Norman. Resolving conflict and inconsistency in norm-regulated virtual organizations. In *AAMAS*, 2007.

[VSAD04] Javier Vázquez-Salceda, Huib Aldewereld, and Frank Dignum. Implementing norms in multiagent systems. In *Multiagent System Technologies, Second German Conference, MATES 2004, Erfurt, Germany, September 29-30, 2004, Proceedings*, volume 3187 of *Lecture Notes in Computer Science*, pages 313–327. Springer, 2004.