# Herramienta de usuario final para la composición de servicios en el IoT

**Fabio Rubio Garrido**

Trabajo Final de Máster

Máster en Ingeniería del Software, Métodos formales y Sistemas de Información

Departamento de Sistemas Informáticos y Computación

Dirigido por:

Victoria Torres Bosch

Vicente Pelechano Ferragud

**A mi familia, siempre presente en su apoyo constante y cercano,**

**Y especialmente a Aixa y Alberto, mis queridos vecinos de abajo,**

**A la tita Ali, cariño de siempre y mecenazgo desinteresado,**

**Y a Inés, permanente soporte en línea y mi último refugio.**

## Resumen

Dos son las premisas de partida de este trabajo. En primer lugar, la constatación irrefutable del cambio de interacción máquina-usuario favorecido por la Web 2.0 y el Internet de las cosas. En segundo lugar, el hecho de que, año tras año, el uso de dispositivos móviles –en especial, los teléfonos inteligentes y las tabletas– haya visto incrementarse su extensión e intensidad.

La combinación de estos dos factores ha llevado, entre otras consecuencias, a favorecer un nuevo perfil de usuario final de dispositivos móviles que exige no ya solo aplicaciones que le proporcionen acceso a datos, sino que les proporcionen mecanismos ágiles y accesibles para generar nuevo contenido con aplicación tanto en su vida profesional como privada. Consideramos que aún no ha habido una respuesta satisfactoria a esta demanda.

Atendiendo a este diagnóstico, este trabajo propone una herramienta que permita al usuario final componer servicios, apoyándose principalmente en la interacción que proporciona el Internet de las cosas. El objetivo concreto perseguido consiste en posibilitar a usuarios finales, sin experiencia en programación o modelado de procesos, generar un modelo completo con interacción activa en servicios.



## Palabras clave

Composición de servicios, Internet de las cosas (IoT), Herramienta de desarrollo para usuario final.

## Índice

## Índice de Figuras

# 1. INTRODUCCIÓN

## 1.1 Planteamiento del problema

La idea de la Web 2.0 ha supuesto desde su aparición un cambio significativo en las estructuras de interacción entre el usuario y las herramientas digitales. Quedó atrás la figura pasiva del usuario en la Web o Internet, en tanto que se ha impuesto la necesidad de considerar al usuario como un protagonista activo, como un contribuidor más en el contenido de la red, siendo, por tanto, capaces de proponer y fomentar, con esa aportación diversa, una nueva concepción de la sociedad, que al tiempo que se informa y se comunica, genera conocimiento de forma desregulada y autónoma.

No es de extrañar, por tanto, el surgimiento masivo de herramientas que propicien este nuevo tipo de intercambio. Casos como las numerosas redes sociales, las wikis, las CMS's, etc. han puesto en evidencia este cambio de paradigma que no puede ya ser eludido. Se requieren nuevas formas de promover el uso de lo digital desde una nueva concepción del usuario con el fin de facilitar tanto el acceso como el desarrollo de nuevos contenidos y nuevas formas de interacción.

A esto se suman nuevos modelos de interconexión que van más allá de la interacción humano-máquina. En efecto, con el Internet de las cosas (IoT), se ha superado el diagrama convencional de la revolución digital, integrando así a lo real en la estructura misma. Con la interconexión digital de objetos cotidianos con Internet, fundamento este de la IoT, un nuevo cambio se suma a la idea de Web 2.0.

A este respecto, Gartner estima que existirán casi 26 billones de

dispositivos en IoT en 2020[1]. Si, por ejemplo, los aparatos de medición de temperatura, predicción de tormentas…, toda clase de productos de consumo, pudieran estar monitorizados permanentemente se evitaría el factor riesgo que toda interacción humana suma a lo real y, consecuentemente, se podrían realizar predicciones mucho más certeras. El caso de las predicciones meteorológicas es sintomático de ello. Habitualmente todos los parámetros involucrados en las mediciones meteorológicas se realizan gracias a múltiples aparatos: termómetros, detectores de saltos voltaicos, barómetros, audiómetros,... Toda la información que estos proporcionan queda compilada usualmente en servicios web que, no obstante, en raras ocasiones permite un uso global y simultáneo de las diferentes fuentes de información y sus respectivos servicios web. En cambio, es evidente que un uso compartido aumentaría considerablemente la capacidad predictiva y, en general, el valor añadido de esta información básica.

Dicho esto, se presenta un problema de acuciante actualidad. Habida cuenta de la revolución en los medios de interacción digital ocasionados por la Web 2.0 y el Internet de las cosas, se requieren nuevos entornos de composición que satisfagan las necesidades de este nuevo perfil de usuario, que por lo general no tiene conocimientos expertos en programación. Entornos ampliamente extendidos como son Intalio, Activiti, Signavio o Bonita BPM, o lenguajes de composición de servicios y notaciones como Petri nets, EPC, YAWL, BPMN o UML Activity Diagrams distan mucho de cumplir con este nuevo orden. No están, en definitiva, al alcance del nuevo usuario.

1 http://www.gartner.com/newsroom/id/2636073

Adicionalmente, hasta la fecha, ordenadores y portátiles han acaparado mayormente los nuevos diseños de entornos para este tipo de composiciones, no cumpliendo con los nuevos requisitos de los usuarios tipo. Los datos constatables, a este respecto, no dejan lugar a duda alguna. Como defiende Gartner, en el segundo cuarto de 2013 la cantidad de teléfonos inteligentes vendidos a los usuarios finales alcanzó un total de 225 millones de unidades. Comparado con el segundo cuarto de 2012, significa un incremento del 46,5 por cien. Además, las ventas en el mundo de teléfonos móviles inteligentes han superado por primera vez las perspectivas de ventas de teléfonos convencionales, constituyendo el 51,8 por ciento de las ventas de teléfonos móviles.[2] Es cierto que hasta la fecha, los usuarios finales de estos terminales son fundamentalmente consumidores de información y servicios, pero a la vista de este masivo incremento en la adquisición de los nuevos dispositivos, y atendiendo a su disponibilidad, cabe suponer que en un futuro inmediato los teléfonos inteligentes y las tabletas puedan a su vez proporcionar medios para la creación de nuevas composiciones de servicios o información.

## 1.2 Solución propuesta

Por una parte, queda clara la transformación de las estructuras de interacción entre el usuario y la red, fundamentalmente advenida a raíz de la implantación de la Web 2.0 y del Internet de las cosas: ha de atenderse a una nueva figura de usuario final, que exige facilidad e

2 http://www.gartner.com/document/2573119

inmediatez en la composición de nuevos servicios o información. Por otra parte, el soporte de esta nueva demanda está cambiando a pasos agigantados. Así como pocos años atrás, el mercado estaba monopolizado por dispositivos como los ordenadores o los portátiles, recientemente ha quedado puesto de manifiesto el protagonismo que en breve (si no ya mismo) van a alcanzar los dispositivos móviles (fundamentalmente teléfonos y tabletas). En tal contexto, este trabajo pretende presentar una herramienta de usuario final para la construcción de composición de servicios para dispositivos móviles. Con este fin, el diseño de esta herramienta está inspirado por los cambios de patrones y técnicas para el desarrollo de aplicaciones de usuario final. Este trabajo ha sido generado en el contexto de la SITAC (Social Internet of Things, Apps by and for the Crowd), un proyecto ITEA 3 que tiene como objetivo hacer converger el Internet de las cosas y las redes sociales, dando lugar así a una mejor interacción entre humanos y dispositivos.

### 1.3 Estructura del trabajo

Este trabajo está estructurado como sigue:

- Sect.1 introduce el contexto en el que se enmarca esta propuesta.
- Sect.2 presenta un escenario que ilustra una aplicación práctica de la herramienta de usuario.
- Sect.3 introduce los fundamentos de la herramienta, cuáles son los patrones de cambio y el desarrollo usuario final.
- Sect.4 explica en detalle el diseño y la implementación de la

herramienta de usuario final.

- Sect.5 presenta algunos trabajos relacionados.
- Sect.6 cierra el texto a modo de conclusión y provee algunas recomendaciones para trabajos futuros.

# 2 FUNDAMENTOS

En la presente sección, se describen someramente los dos ámbitos de investigación en los que nos hemos basado para el desarrollo de este trabajo. En primer lugar, se presentan las técnicas de las que se puede hacer uso para el desarrollo de propuestas para usuarios finales. En segundo lugar, se evalúan los pros y los contras de las mismas en la creación de modelado de procesos. Es de destacar el hecho de que los cambios de patrones y las ventajas que estas introducen permiten describir servicios de composición con un alto grado de abstracción.

## 2.1 Desarrollo orientado a usuario final

Dada esta gran proliferación de dispositivos móviles y tabletas se accede a una gran cantidad de nuevos usuarios que demandan unas nuevas características para sus dispositivos que atiendan a las nuevas necesidades. De esta forma, se ven obligados a introducirse en el mundo del desarrollo que hasta hace no mucho estaba restringido a una reducida cantidad de especialistas con los suficientes conocimientos para la programación. Para suplir este problema, desde hace unos años se ha estado activamente investigando sobre este tema, desarrollando las técnicas necesarias para crear el mejor entorno posible. Lieberman lo define así:

> El desarrollo orientado a usuarios finales puede ser definido como un conjunto de métodos, técnicas y herramientas que permite a los usuarios de software, que sin actuar como profesionales de desarrollo software, puedan de la misma manera crear, modificar y extender artefactos de software.[1]

Se entiende por “artefactos” cualquier comportamiento automático o secuencia de control (como petición a base de datos o reglas gramáticas) que puedan ser descritos como paradigmas de programación. Adicionalmente, puede entenderse por “artefactos” parámetros o cambios realizados en el comportamiento de una aplicación, o incluso puede referirse a la creación de nuevo contenido, no necesariamente computacional.

Así pues, el desarrollo orientado a usuarios finales no sólo debe permitir a profesionales realizar sus sofisticadas y completas creaciones sino que debe ser lo suficiente amigable, familiar y rápidamente comprensible para que un usuario sin conocimiento alguno de programación pueda generar y manipular sus propios contenidos con la suficiente confianza para su uso.

De entre las técnicas empleadas para el desarrollo orientado a usuarios finales, cabe desatacar:

- Programación por demostración

Es una técnica que permite al usuario enseñar al robot u ordenador una serie de tareas a realizar, en lugar de escribir una serie de comandos necesarios para su interpretación por la máquina. Este, a su vez, lo repite tantas veces como sea necesario mientras el usuario corrige los posibles errores y lo refina, dando una solución completa.

- Programación por ejemplos

Es una técnica de programación gracias a la cual el usuario crea una aplicación realizando distintas tareas. El ordenador registra y crea el código necesario para poder realizarlo repetidamente. Se diferencia principalmente de PbD en que en este tipo de programación el usuario entrega una solución prototipada, mientras que el PbD es un conjunto de tareas que no tienen por qué estar definidas con anterioridad.

- Programación visual

Es una técnica de programación que se basa en la manipulación por parte del usuario de elementos gráficos o iconos que, con unas ciertas reglas (que incluye sintaxis, expresiones visuales, símbolos matemáticos...), se ordenan y dan lugar a una secuencia de tareas que internamente crearía el programa deseado (por ejemplo, Blockly, Kawa, Cimple o UML Generator).

- Generación de código

Mediante la utilización de macros o comandos, el usuario es capaz de generar una aplicación sin la necesidad de poseer grandes conocimientos de programación, la máquina utiliza las macroinstrucciones ordenadas previamente para la construcción del código completo (por ejemplo, Visual Basic for Aplications o Microsoft Macro Assembler).

Estas técnicas pueden ser utilizadas individualmente o en combinación, haciendo que ciertas debilidades que poseen algunas puedan ser

suplidas por otra(s) y hacer así más fácil la interacción con el usuario. Recuérdese que este, probablemente, no sea capaz de testear, revisar o depurar la solución. En este sentido, es habitual utilizar la técnica de programación combinada, por ejemplo, con lenguajes visuales o de texto. En efecto, la técnica de programación visual simplifica mucho el desarrollo de aplicaciones, ya que aporta una interfaz amigable y sencilla de utilizar por el usuario con un nivel alto de abstracción, ocultando la parte más tediosa de generación de código. No obstante, esto no quita para que a veces lleve a confusiones o errores de funcionamiento, motivados por el mal entendimiento de lo que realiza cada componente. De ahí que con frecuencia se utilice en combinación con la técnica de comandos para dar un soporte superior y simplificado de lo que el usuario realmente está haciendo. Esta línea de investigación –que se conoce como "lenguaje visual de información en cascada"– es utilizado por muchos, como en Yahoo! Pipes, donde proporciona un entorno capaz de agregar, manipular y mezclar el contenido desde la web.

Existe una gran cantidad de líneas de investigación a través de las cuales se intenta buscar una serie de recomendaciones y guías de diseño que ayuden al desarrollo de este tipo de herramientas. En [2], los autores REPENNING y IOANNIDOU afirman que, independientemente de la técnica usada para el EUD, ha de prestarse atención a lo siguiente:

- los diferentes tipos de habilidades del usuario y los desafíos con los que tienen que enfrentarse para generar un desarrollo y

- lograr o conseguir que un desarrollador usuario final pueda adquirir gradualmente las habilidades necesarias para abordar los desafíos del desarrollo.

También nos aportan unas guías de diseño usando una herramienta de simulación para usuarios finales llamada AgentSheets, dando sugerencias sobre la sintaxis, semántica y aspectos pragmáticos que hay que tener en consideración para el desarrollo efectivo de una herramienta para usuario final.

Adicionalmente hay que tener en cuenta las barreras que se le presentan al usuario final cuando usa este tipo de herramientas. Andrew Ko, Brad Myers y Htet Htet Aung en [3] proponen un desglose detallado de las mismas en caso concreto de unos nuevos programadores que aprenden a usar el Visual Basic.NET.

Identifican seis tipos:

- Barreras de Diseño: “No sé qué quiero que haga el ordenador...”
- Barreras de Selección: “Creo que sé lo que quiero que haga el ordenador, pero no sé qué utilizar para ello...”
- Barreras de Coordinación: “Creo que sé qué cosas usar, pero no conozco cómo hacer que funcionen juntas...”
- Barreras de Uso: “Sé qué puedo usar, pero no sé cómo usarlo...”
- Barreras de Comprensión: “Pensaba que sabía cómo usarlo, pero no realiza lo que yo esperaba...”
- Barreras de Información: “Creo que sé por qué no realiza lo que yo esperaba, pero no sé cómo comprobarlo...”

De ahí que, al realizar cualquier tipo de desarrollo para usuarios finales, haya que prestar atención a que todos estos tipos de barreras sean solventados o intentar, en la medida de lo posible, minimizar su impacto para que los usuarios no se estanquen en el desarrollo que desean hacer. De lo contrario, podrían frustrarse y dejar de utilizar la herramienta.

En cuanto a la interacción del usuario, existen diferentes recomendaciones de patrones de interfaces para solventar este tipo de barreras, tal y como muestra el estudio que realizaron Martijn van Welie y Hallvard Trætteberg en [4] sobre distintos tipos de interfaces con sus pros y contras detalladamente. Una lista de los mismos queda reflejada en su artículo y la transcribimos a continuación:

- Asistentes o instrucciones paso a paso (wizards): permite al usuario realizar tareas complejas siguiendo un camino ya determinado paso a paso (por ejemplo, el asistente de instalación de aplicaciones Installshield).
- Diseño en cuadricula (Grid layout): presenta las opciones que el usuario puede configurar en forma de cuadrícula (por ejemplo, el Microsoft Word Frame Options).
- Progreso (progress): panel que, mediante una barra dinámica, indica al usuario que está realizando las tareas deseadas (por ejemplo, Netscape's Download box).
- Shield: cuadro de diálogo que alerta al usuario de la realización de una tarea crítica o de un posible error (por ejemplo, las alertas de

Microsoft Explorer).

- Configuración (preferences): panel completo, usualmente con menú en forma de árbol, donde el usuario puede configurar todas las opciones que se ofrecen (por ejemplo, Configuración Internet Explorer).
- Menú Contextual (Contextual Menu): menú fácilmente accesible (por ejemplo, el botón derecho del ratón) en el que acceder a distintas opciones inmediatas. (por ejemplo, los menús del botón derecho del ratón).
- Focus: ofrecer algún tipo de marca en la interfaz para ayudar al usuario a saber en qué lugar está trabajando o el siguiente punto a realizar (por ejemplo, resaltar cuadro de texto con un marco de distinto color).
- Formato no ambiguo (Unambiguous Format): panel donde el usuario sólo puede insertar los datos necesarios (por ejemplo, el Calendario Microsoft Windows).
- Navegación entre espacios (Navigating between Spaces): ventana donde se permite al usuario navegar por distintos paneles para cambiar opciones o datos (por ejemplo, MS PowerPoint).
- Como en la vida real (Like in the real world): se muestra al usuario el objeto en el que se esté trabajando como en la vida real (por ejemplo, The Worldbeat system).
- Vista previa (Preview): permite al usuario ver cómo quedarían las tareas en las que está trabajando al terminar (por ejemplo, la Vista previa MS Word).
- Favoritos (Favourites): permite al usuario crear una lista de

opciones o datos favoritos para su posterior reutilización (por ejemplo, Favoritos Internet Explorer).

- Área de comandos (Command Area): zona habilitada donde el usuario pueda escribir los comandos a realizar.
- Contenedor de navegación (Container Navigation): ventana partida en tres o más áreas donde se muestra la localización de la información y, en la última, la información seleccionada (por ejemplo, Microsoft Outlook).
- Modo de cursor (Mode Cursor): muestra al usuario el modo de trabajo que está realizando, cambiando el icono del cursor (por ejemplo, Photoshop).
- Filtros (Continuous Filter): permite al usuario generar filtros para acceder más rápido a la información que está buscando. (por ejemplo, IntelliSense).

Todas estas interfaces recomendadas son muy simples para mejorar el aprendizaje, la memorización de los pasos a seguir y protege al usuario final de posibles errores ocasionados por razones tan diversas como olvidarse de introducir pasos importantes. Además, se recomienda añadir descripciones textuales de los movimientos del usuario e intentar que todas las interfaces posean una estructura similar para ayudar al usuario final a la interpretación adecuada de la información.

Teniendo en cuenta todos los aspectos y recomendaciones ya explicados que una herramienta para usuario final debería tener, se ha elegido para nuestra herramienta la interfaz apoyada en asistentes, por ser la más aceptada debido a la comunidad de desarrollo para usuarios finales. En

esta, los asistentes guiarán al usuario por un proceso particular con el objetivo de crear la composición de servicios.

### 2.2 Patrones de Cambio

Los patrones de cambio pueden ser definidos como las manipulaciones que el usuario puede realizar a las actividades y fragmentos. Estos son genéricos y con un alto nivel de abstracción, permitiendo así un acceso flexible y de fácil adaptación para los procesos de negocios. Han sido definidos y ordenados por Barbara Weber, Stefanie Rinderle y Manfred Reichert en [5] como sigue:

- Añadir/Borrar Fragmentos
  - AP2: Proceso de borrado de fragmentos
  - AP1: Proceso de añadido de fragmentos

- Desplazamiento/Reemplazamiento de fragmentos
  - AP5: Proceso de sustitución de fragmentos
  - AP14: Proceso de copiado de fragmentos
  - AP4: Proceso de Reemplazamiento de fragmentos
  - AP3: Proceso de Desplazamiento de fragmentos

- Añadido/Borrado de niveles
  - AP7: Proceso de cambio de nivel
  - AP6: Proceso de Extracción de subordinado

- Control de adaptación de dependencias
  - AP8: Proceso de Inclusión de fragmentos en bucles
  - AP9: Actividades en Paralelo
  - AP10: Proceso de Inclusión de fragmentos en rama condicional
  - AP11: Añadir control de dependencias
  - AP12: Eliminar control de dependencias

- Cambio de condiciones de transición
  - AP13: Actualizar Condición

Además, los patrones de cambio quedan separados por su grado de complejidad. Así, en primer lugar, los cambios de operaciones a alto nivel son cambios que requieren de varias acciones para ser completadas (por ejemplo, añadir una actividad en paralelo a otra), requiriendo un alto nivel de experiencia por parte del usuario final para llevarlas a cabo. En segundo lugar, los cambios primitivos son las acciones más elementales que se pueden realizar sobre los elementos (por ejemplo, añadir o borrar un elemento o eliminar un nodo como un fragmento paralelo). Gracias a la aplicación de este tipo de patrones se garantiza un resultado válido en los procesos de negocio.

La mayoría de las herramientas de creación de composición de servicios están basadas en la unión de múltiples cambios primitivos en la aplicación (por ejemplo, añadir/borrar actividades o fragmentos). Al utilizar cambios primitivos es necesario comprobar detalladamente el modelo de procesos después de cada transformación, ya que al no ser cambios de alto nivel pueden generar errores creando un modelo sin

garantías de uso. Como alternativa, se pueden usar los patrones de cambios puesto que trabajan a alto nivel y permiten, por lo tanto, un control sobre el modelo más garantista por parte de la herramienta (por ejemplo, se realizarían transformaciones o creaciones del tipo de fragmentos de procesos, agrupaciones en ramas o incrustamiento de actividades en fragmentos condicionales; si estos cambios tuviesen que ser realizados por cambios primitivos, se multiplicarían las acciones a realizar, por lo que la posibilidad de error iría en aumento).

Para la realización de este trabajo se escogerá una de las sub-listas de cambio de patrones presentada anteriormente. Estos cambios de patrones permitirán a los usuarios crear construcciones de control de flujo (por ejemplo, secuencias, distintas ramas de ejecución en paralelo, líneas condicionales o bucles), además de la inserción, borrado y edición de actividades y fragmentos. Los siguientes patrones constituyen el grupo seleccionado:

- AP1: Proceso de añadido de fragmentos (en dos variantes: para actividades y fragmentos)
- AP2: Proceso de borrado de fragmentos
- AP8: Proceso de Inclusión de fragmentos en bucles
- AP10: Proceso de Inclusión de fragmentos en rama condicional

A modo de aclaración, la Fig. 1 nos muestra los cambios de patrones de inclusión de actividades en fragmentos en bucle en un proceso original que hemos llamado S. Al realizar la transformación, la actividad llamada B queda dentro de un fragmento en bucle:

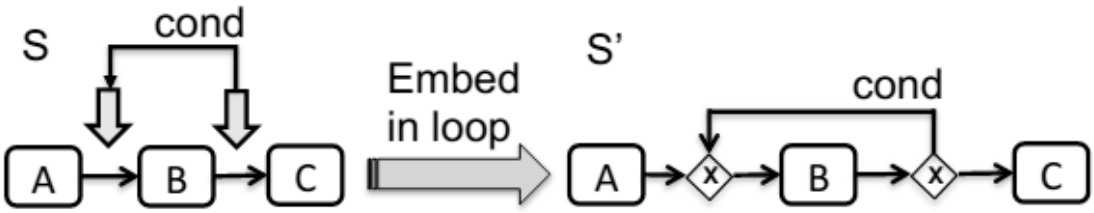


*Fig. 1– Patrón AP8 Proceso de Inclusión de fragmentos en bucles*

# 3 CASO DE ESTUDIO

Para llevar a cabo el desarrollo de la herramienta y la explicación de la misma se va a emplear un ejemplo ficticio cotidiano en el cual la herramienta pueda servir de ayuda al usuario. Este usuario ficticio, que llamaremos John, es un estudiante universitario que actualmente se encuentra realizando los estudios de cuarto año en una Escuela de Arquitectura. La universidad donde John estudia ofrece servicios en internet que proveen a los estudiantes de información actualizada sobre los servicios de la universidad como, por ejemplo, información relativa a las bibliotecas, las plazas de aparcamiento disponibles, los nuevos cursos, los eventos celebrados y su calendario concreto, etc.

Durante el periodo de exámenes, John acostumbra a ir a una de las bibliotecas de la universidad para estudiar y, a ser posible, ir con más compañeros. Para ello, John utiliza los servicios de la universidad en la nube para conocer la ubicación de las bibliotecas donde todavía tengan plazas de estudio disponibles y que estén próximas a plazas de aparcamiento disponibles para bicicletas. John se desplaza en bicicleta, huelga decir. Basándose en los resultados de la busca que realiza, John elige una biblioteca y un aparcamiento para bici, pudiendo reservar una plaza en ambos casos. Mientras realiza la reserva de las plazas, John notifica a sus compañeros vía una red social (podría ser Facebook) la información de las mismas, para que ellos a su vez puedan realizar la reserva y poder ir todos a la misma biblioteca.

Este ejemplo se puede simplificar mediante un proceso BPMN como en la Fig. 2, que describe las tareas que realiza John cada mañana durante

el periodo de exámenes para reservarse las plazas y notificar a sus compañeros su asignación. John desearía automatizarlo para poder definir una composición de servicios que enlace adecuadamente todas las actividades a realizar y los servicios de la universidad que tiene que usar.

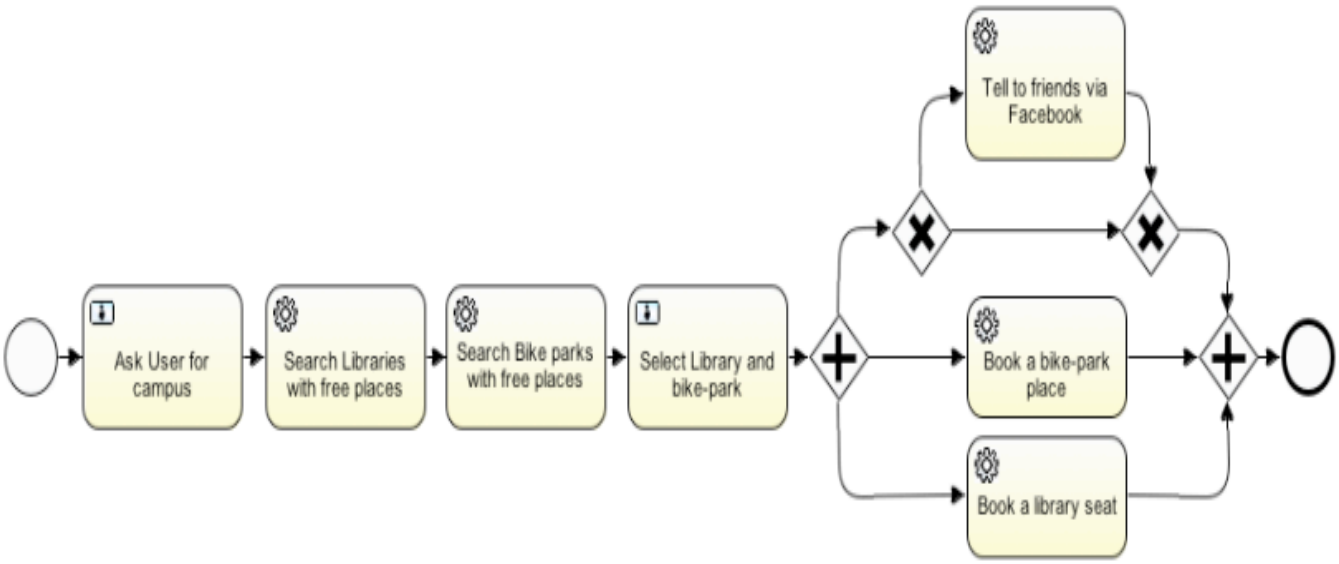


*Fig. 2 - Composición de servicios del caso de estudio en BPMN*

# 3. PROPUESTA

## 3.1 Ámbito Tecnológico

Para realizar la implementación de este trabajo se ha escogido utilizar la plataforma web, trabajando en HTML, con el fin de permitir a una gran cantidad de dispositivos utilizar esta herramienta, dado que se trata de un estándar ampliamente extendido.

Para poder desarrollarlo en esta plataforma utilizaremos Jquery Mobile, que es un framework optimizado para dispositivos táctiles creado por el equipo de desarrollo de Jquery.

Este framework nos permite crear la interfaz necesaria para la herramienta sin tener que preocuparnos del dispositivo con el que se va a usar, ya que en un mercado tan heterogéneo de tabletas y smartphones es realmente complicado optimizar las interfaces para cada uno de ellos. Jquery Mobile trabaja con temas de CSS que no sólo permiten esta optimización, sino que genera una temática estándar que se mantiene en toda la aplicación y que ayuda a no desconcertar al usuario final. Al trabajar sobre la base de Javascript, podemos utilizarlo para crear el motor de la herramienta y generar así las distintas páginas dinámicas necesarias para que el usuario final pueda realizar, manipular y guardar en XML su composición de servicios sin problemas.

## 3.2 Descripción detallada

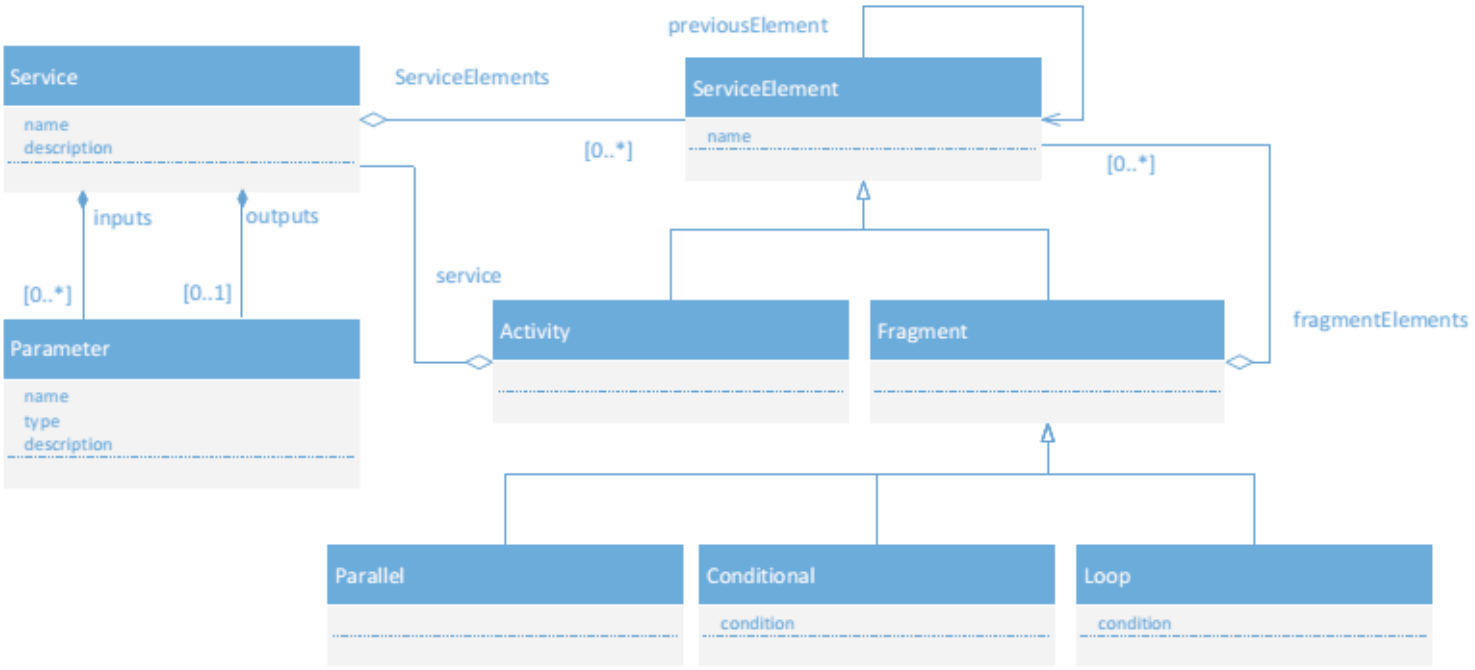


*Fig. 3 - Metamodelo*

Para crear la arquitectura de la herramienta de composición de servicios, nos hemos inspirado en los conjuntos de patrones ya presentados en la sect 2.2 creando el metamodelo de la Fig. 3 con un nivel alto de abstracción. El metamodelo posee dos conceptos principales: las actividades y los fragmentos. Al reducirse sólo a dos, permite que sea más entendible para los usuarios finales. Las actividades se definen como la tarea que el usuario especifica que hay que realizar y los fragmentos son actividades o conjunto de actividades englobados en grupos con características especiales. Paralelos serían un

conjunto de actividades que se realizarían en orden en distintas ramas de ejecución; condicionales serían un conjunto de actividades que se realizarían si se cumpliese la condición especificada; y, por último, los bucles serían un conjunto de actividades donde se realizarían hasta que se cumpliese la condición especificada.

Las actividades y los fragmentos son ServiceElements definidos como patrones compuestos. Esto facilita la creación de interfaces de usuario por parte del usuario final para permitir definir, manipular o borrar cada fragmento individualmente. El previousElement, que está relacionado con ServiceElement, permite establecer el orden de secuencia de entre las actividades y fragmentos para ayudar a los usuarios finales sin nociones de modelado de procesos. Habida cuenta la creación de un nuevo elemento, este definiría simplemente cuál es el elemento anterior y, así, no tendría que especificar elementos de flujo. Además, este orden de secuencias define el principio y el final de la composición, siendo el primero y el último una actividad o fragmento respectivamente. Finalmente quedaría por definir y enlazar los servicios, a los que la herramienta tendría acceso, a las actividades correspondientes, permitiendo a los usuarios finales definir los parámetros de los servicios para poder hacer uso de los mismos. Se complementará con una descripción textual para ayudar a entender a los usuarios finales de la semántica de los servicios o parámetros.

Para poder entender mejor el concepto del metamodelo, en la Fig. 4 se muestra el proceso en el caso de estudio. Este proceso está compuesto por una secuencia de cuatro actividades, seguido de un fragmento

paralelo que contiene dos actividades en distintas ramas de ejecución y, en otra rama, un fragmento condicional conteniendo otra actividad. Esto significa que al terminar la actividad número 4 se iniciarían tres líneas de ejecución diferentes, donde se ejecutarían simultáneamente las dos actividades y la actividad del fragmento condicional si cumple los requisitos establecidos por el usuario en la condición.

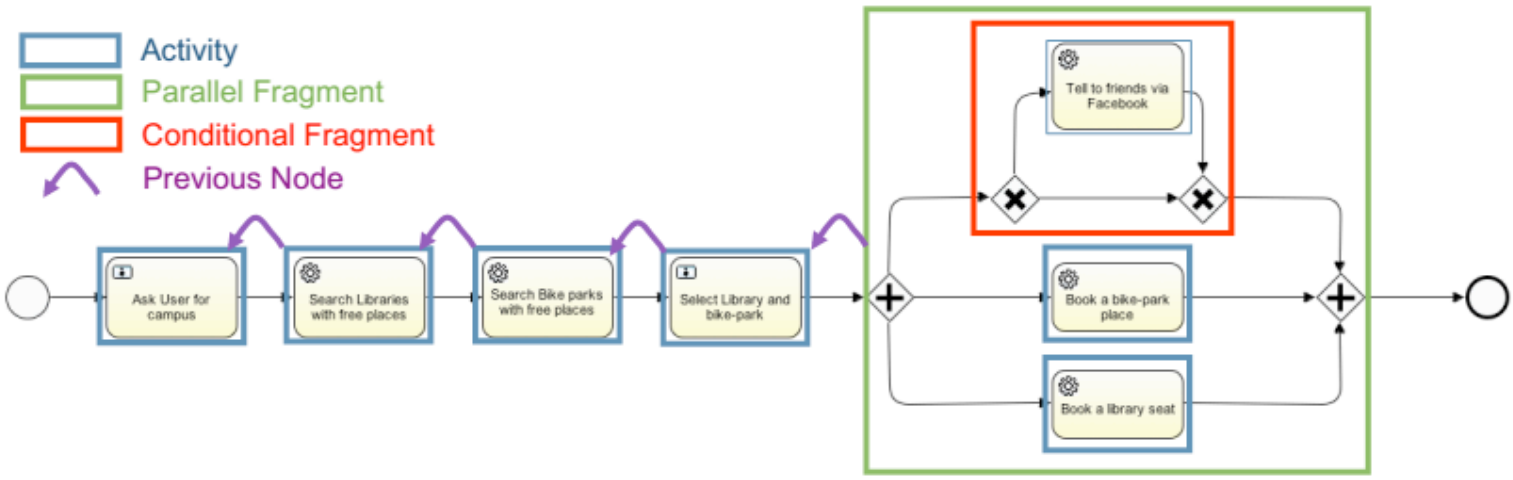


*Fig. 4 - proceso del caso de estudio en el metamodelo*

### 3.3 Herramienta de usuario final

En esta sección se explicará la sintaxis concreta de la herramienta para usuario final basándose en las tecnologías anteriormente mencionadas y centrado para su uso en entornos de smartphone y tabletas. En estos casos, la aplicación detecta el dispositivo que está en uso y, según cuál sea, utiliza los estilos y diseños más óptimos, generando así en la tableta un marco de trabajo más ágil y rápido para

la creación de composiciones. La interfaz utiliza para la interacción con el usuario final la utilización de asistentes, cuadros de diálogo para la creación de elementos en tabletas, un apoyo textual de ubicación del usuario en forma de migas de pan, ayuda textual para el apoyo a la creación de nuevas actividades y una sección de ayuda en cada panel con una descripción detallada de las opciones que pueden realizar; todo ello siguiendo las recomendaciones, conocimientos previos y patrones anteriormente explicados.

Catálogo de patrones: La herramienta ofrece un catálogo de patrones para dar una lista predefinida al usuario final con el fin de proveerle de las posibles tareas que puede realizar:

1. ofrece la posibilidad de crear nuevas composiciones de servicios creándolo desde la base o utilizando uno ya existente como base. Mostrado en Fig. 5A.
2. facilita la selección de una composición de servicios ya existente y poder manipularlo cambiando los parámetros o descripción. Mostrado en Fig. 5B.
3. ayuda a la selección de los elementos que ya existen para poder borrarlos o, si es un fragmento, para poder editarlos. Mostrado en Fig. 5C

Para mejorar el entendimiento por parte del usuario final cada tipo de elemento, ya sea actividad o fragmento, posee tres componentes gráficos mostrados en Fig. 6:

1. un icono identificativo del tipo de elemento del que se trata. Cada

uno de ellos posee uno distinto.

2. el nombre del elemento.
3. el número de elementos que incluye el fragmento. En caso de ser un fragmento paralelo, mostraría el número de ramas que posee.

Todos los elementos que se presentan en cada uno de los paneles estarán ordenados de arriba a abajo. Esto indica la secuencia en la que los elementos serán ejecutados, previamente definido por el usuario, dado que en la creación de los elementos debe indicar siempre el elemento previo.


*Fig. 5 - Primeros pasos: creación de un nuevo modelo*

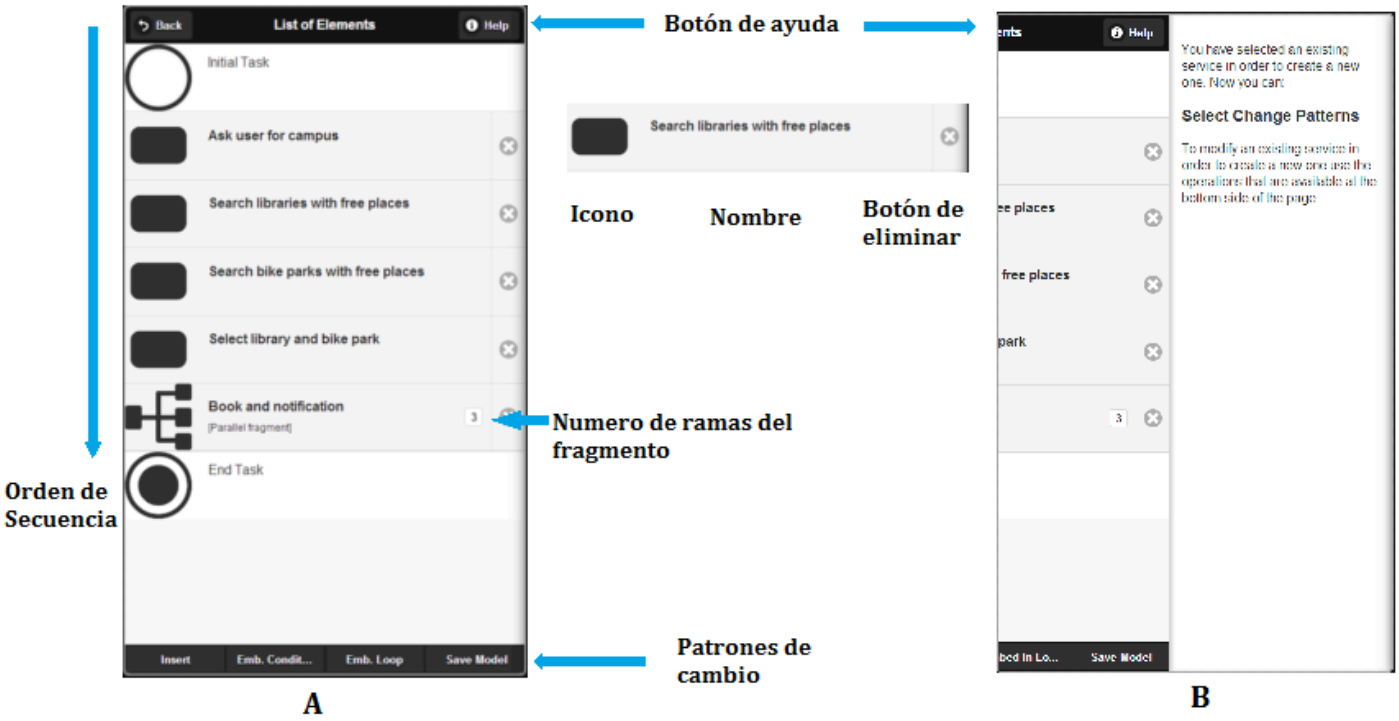


*Fig. 6 - Detalles de una composición de servicios en smartphone*

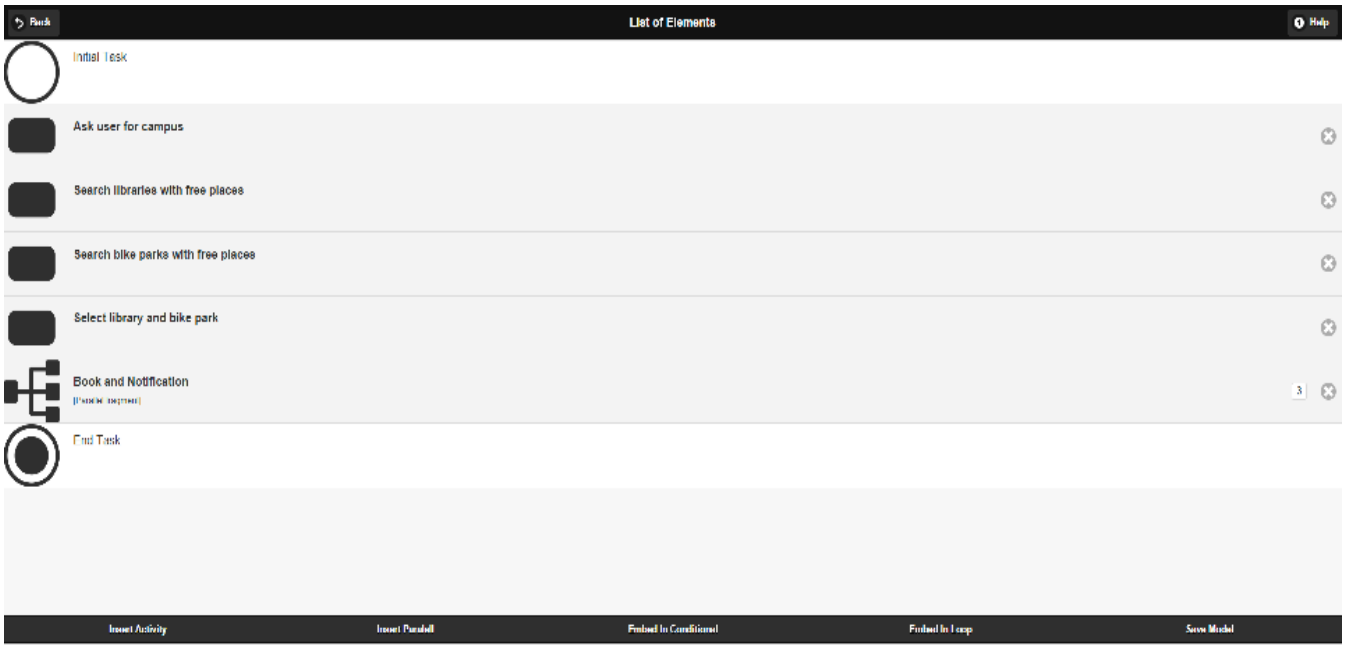


*Fig. 6C - Detalles de una composición de servicios en tabletas*

Paneles: El usuario está siempre guiado gracias a estos paneles. Se accede a estos mediante botones situados en la cabecera de la página. En la parte de la izquierda se encuentra el botón de Help (ayuda) donde el usuario puede saber las acciones que acaba de realizar para llegar donde está y las acciones que puede realizar en la página actual. Mostrado en Fig. 6B.

En la parte de la derecha se encuentra el botón de Map donde el usuario puede saber dónde se encuentra en todo momento con un simple vistazo y navegar por los elementos padre desde la posición en la que se encuentra. Mostrado en Fig. 7.

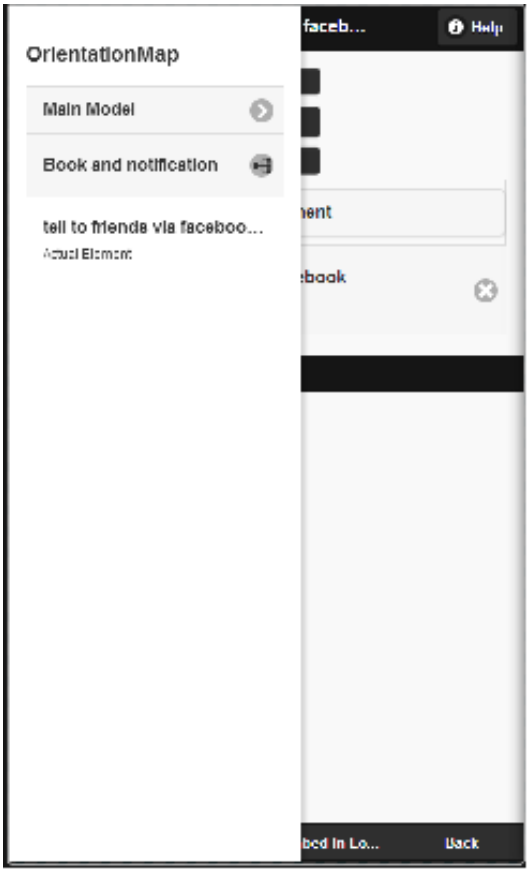


*Fig. 7 - Mapa de orientación*

Patrones de cambio. Para poder crear los usuarios finales su composición de servicios pueden hacer uso de cualquier patrón de cambio, tal y como expusimos anteriormente en la Sec. 2.2. El acceso a estos patrones de cambio se encuentra en distintas posiciones de la interfaz y son siempre fácilmente identificables. Ocupan prioritariamente la parte inferior de la interfaz, en forma de botones. Aquí es donde se encuentran siempre para la versión de smartphones de la herramienta, mostrados en la Fig. 6A. En cambio, en la versión para tabletas se amplía la cantidad de botones para el acceso más inmediato, como se muestra en la Fig. 6C. Se han posicionado en un cuadro de diálogo para agilizar su uso a la hora de crear fragmentos. Más adelante se darán cuenta de los motivos que lo justifican.

Desde estas posiciones, el usuario puede acceder a los patrones de cambio: Insertar (AP1: Proceso de añadido de fragmentos), Inclusión en Condicional (AP10: Proceso de Inclusión de fragmentos en rama condicional) e inclusión en bucles (AP8: Proceso de Inclusión de fragmentos en bucles). Para poder acceder al patrón de borrado (AP2: Proceso de borrado de fragmentos), a diferencia de los demás, este se encuentra en forma de botón X en la parte derecha de cada uno de los elementos, como se muestra en la Fig. 6.

Cuando el usuario selecciona uno de los botones de inclusión en rama condicional o inclusión en bucle accederá a una nueva página, tal y como se muestra en las Fig. 8A y Fig. 8B, para smartphone y tabletas respectivamente. En ambos casos, el usuario deberá seleccionar el primer y último elemento del grupo que desea agregar al fragmento, además de su nombre y la condición. En el caso de que sea para un

fragmento condicional, los elementos seleccionados serán ejecutados si la condición es cumplida. En caso de ser un fragmento en bucle, los elementos seleccionados serán procesados hasta que la condición sea cumplida. A su vez, al crearlos, los elementos seleccionados pasarán de donde se encuentren al interior del fragmento, borrándose de donde pertenezcan. El fragmento condicional o en bucle se posicionará en el lugar que ocupaba el primero de los elementos seleccionados. Ambas páginas muestran una descripción textual del resultado al final de la página.

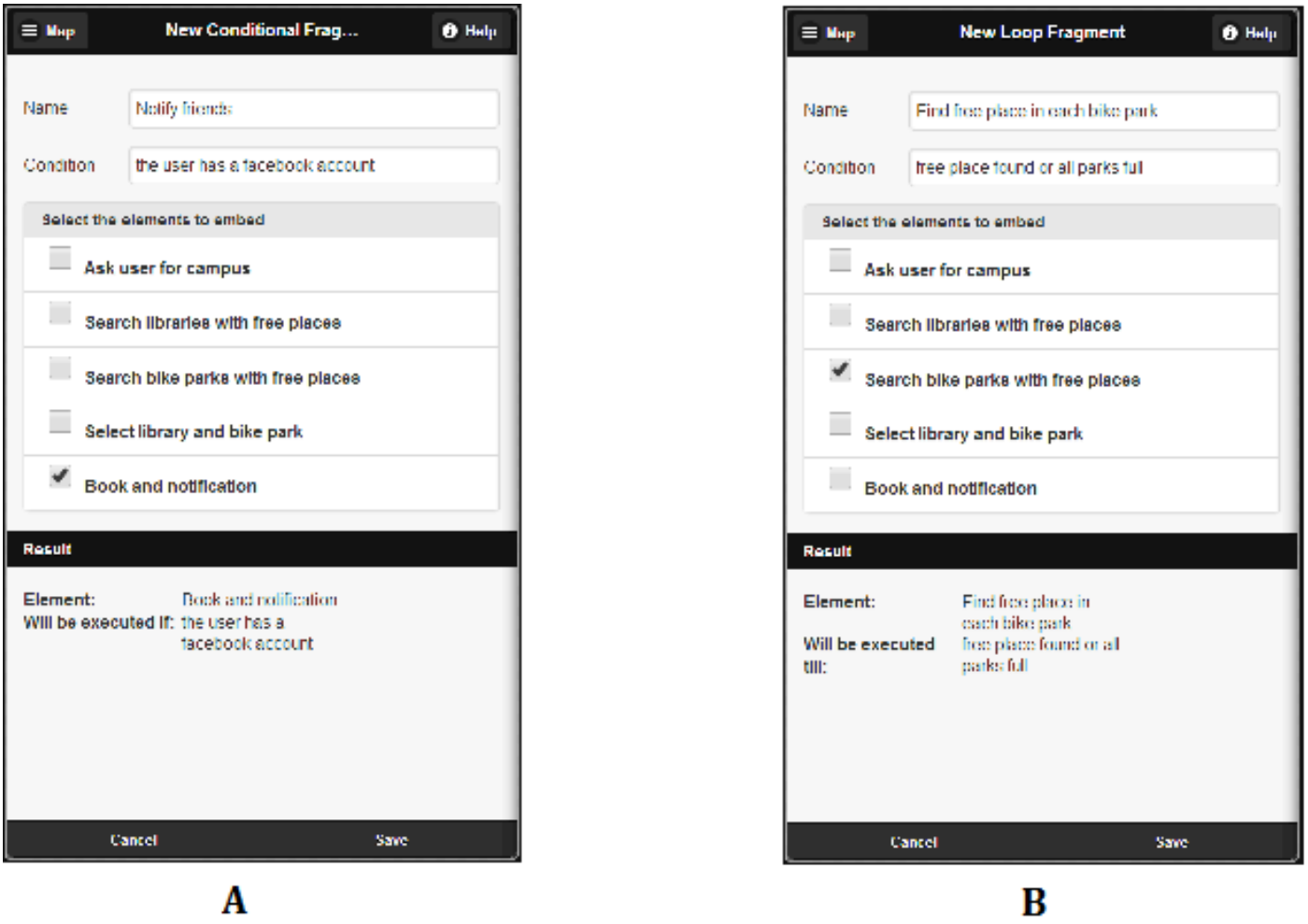


*Fig. 8 - Inclusión de elementos en rama condicional o bucle*

Si el usuario final desea insertar un nuevo elemento desde su móvil, puede hacerlo usando el botón de insertar y se abrirá la página de transición mostrada en la Fig. 9. Desde allí puede acceder a los distintos tipos de creación de actividades o de creación de cualquiera de los distintos fragmentos ofrecidos. Si selecciona la primera opción, accederá a la página de creación de la nueva actividad, tal y como se muestra en la Fig. 9B. Ha de indicar el nombre, el elemento previo y un servicio predefinido de los ya existentes soportados por la herramienta. Una vez cumplimentado esto, puede acceder a otra página, tal y como se muestra en la Fig. 9C, desde donde cambiar los parámetros del servicio o, si estuviesen definidos de antemano, elegir los datos externos del servicio que serán usados como datos internos. En las tabletas, el usuario final puede acceder a las distintas páginas de creación directamente desde los botones que están ubicados en la parte inferior de la interfaz sin necesidad de entrar en la página de transición, como se puede apreciar en la Fig. 6C.

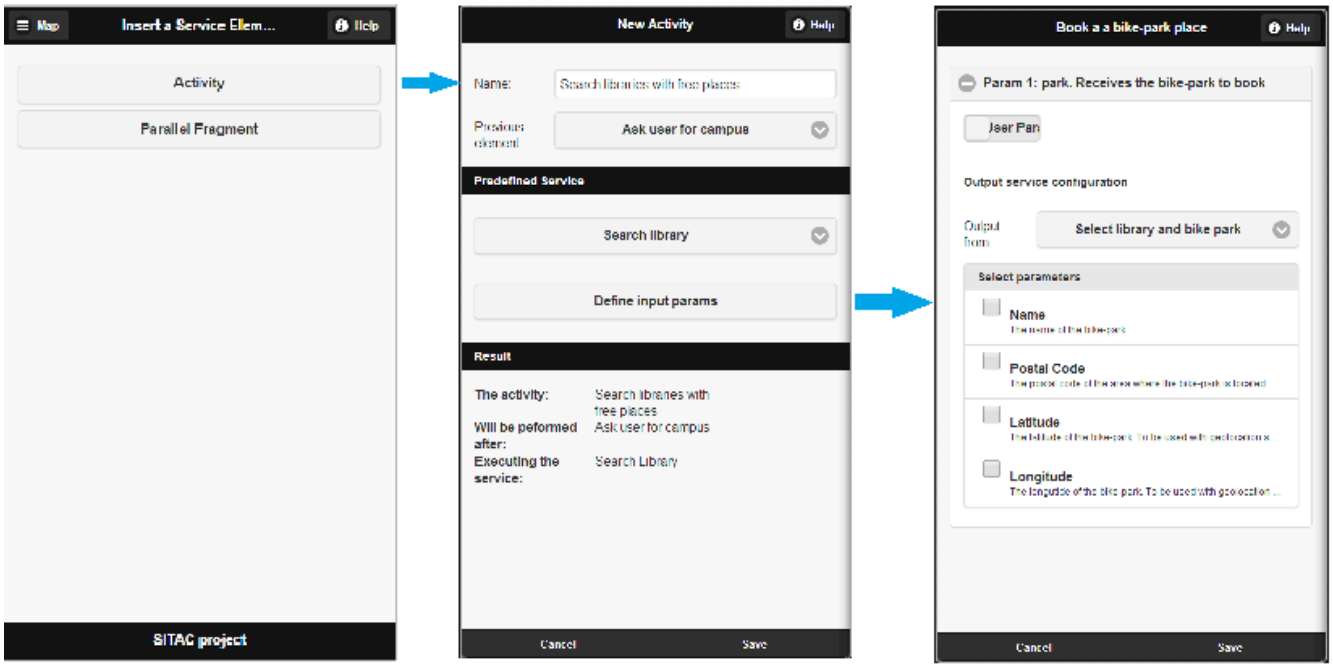


*Fig. 9 - Insertar nueva actividad en smartphone*

Si los usuarios seleccionasen la creación de un fragmento en paralelo se mostraría la página de la Fig. 10B. En este caso, debería indicar el nombre del fragmento y su elemento previo. Con la barra elegiría la cantidad de ramas que ha de asignarse al fragmento –con un máximo de 5– y automáticamente se mostrarían las ramas que posee el nuevo fragmento. El usuario puede acceder a editarlas para agregar nuevas actividades o fragmentos en su interior. Además se da la posibilidad de utilizar los cambios de patrones de inclusión en condicional o en bucle, tal y como se muestra en la Fig. 10C.

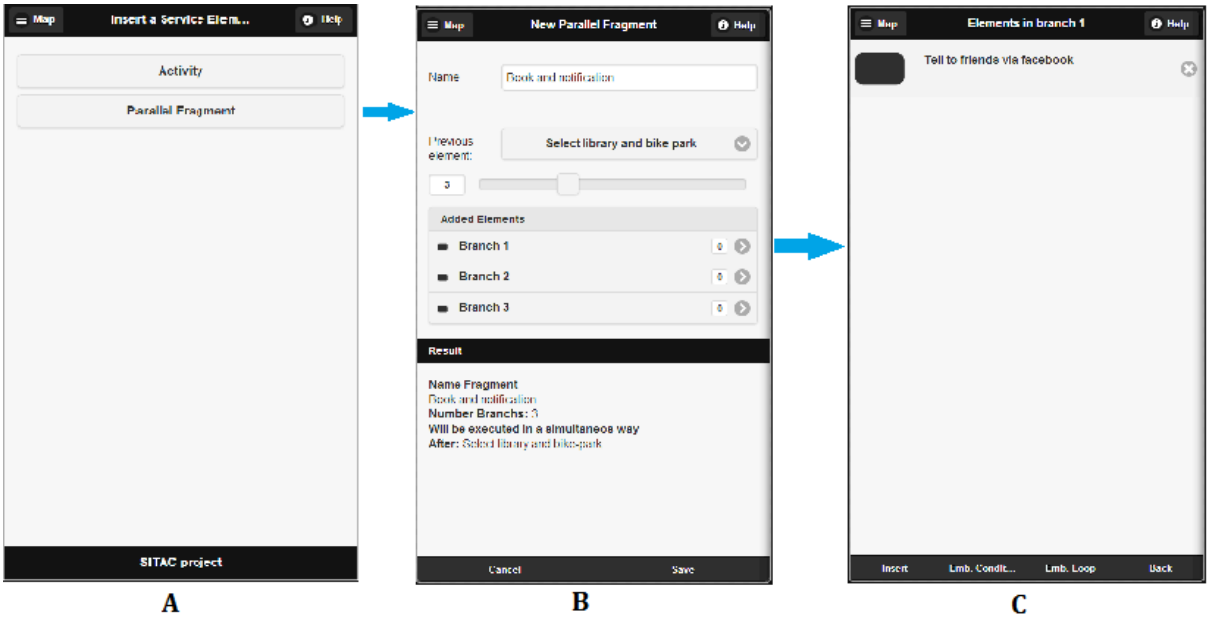


*Fig. 10 - Insertar un fragmento paralelo en smartphone*

En la versión para tabletas la creación de fragmentos paralelos, mostrado en Fig. 11, se vuelve más dinámica. Cada una de las ramas da acceso desde un botón en su parte superior al cuadro de diálogo, donde se podrán insertar cualquier tipo de elemento en esa rama, mostrando posteriormente la nueva inserción en el orden especificado. Posteriormente, el usuario tiene la posibilidad de acceder a cada una de ellas con el fin de editarlas y agregar los elementos necesarios.

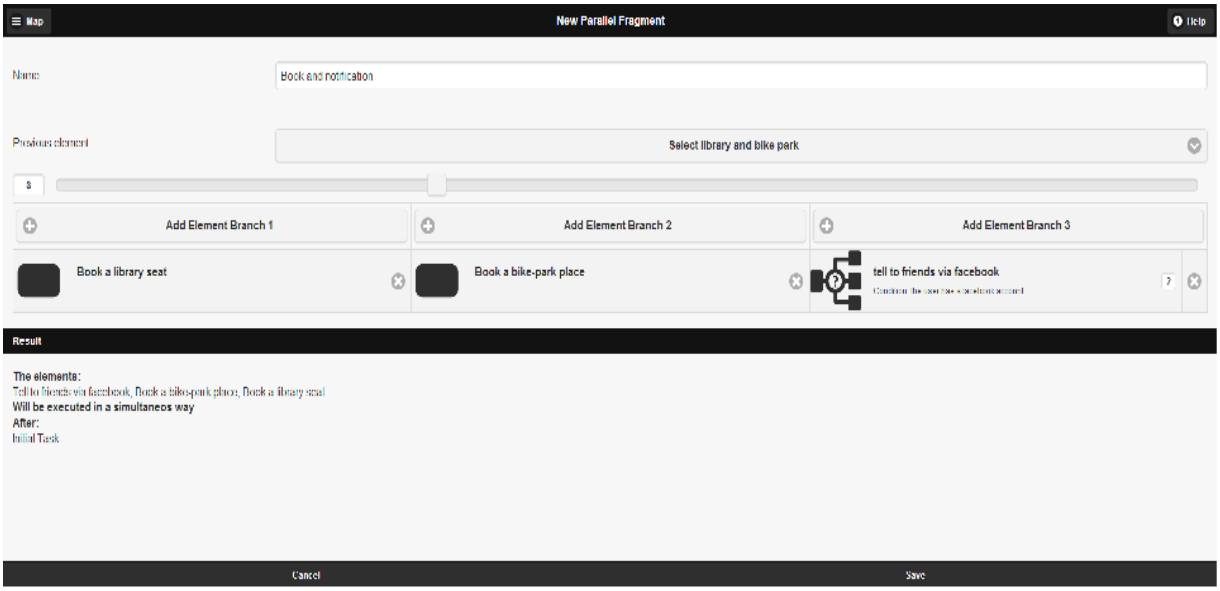


*Fig. 11 - Insertar un fragmento paralelo en tabletas*

La herramienta permite al usuario final agregar dentro de un fragmento nuevos fragmentos por lo que se pueden realizar composiciones realmente complejas, concediéndole así múltiples posibilidades. Los fragmentos en paralelo pueden crearse en el interior de otros o fragmentos condicionales que, a su vez, contienen otros fragmentos condicionales en su interior.

Descripciones textuales: Todas las acciones realizadas por el usuario final están complementadas con descripciones textuales que muestran el resultado de la inserción para facilitar su entendimiento.

Estructuras similares.  Todas las páginas de la herramienta comparten una estructura similar que facilita la interpretación y el uso por parte del usuario final de las mismas. Por razones ya mentadas, ambas

versiones de la herramienta difieren en la manera cómo se insertan los elementos en los fragmentos, pero mantienen la coherencia dentro de la misma versión y la coherencia a nivel de aplicación, como las posiciones de los botones de ayuda, mapa y cambio de patrones.

Cuadro de diálogo. Como hemos explicado anteriormente, la inserción en fragmentos para la versión para tabletas es distinta a la forma de inserción para smartphone. Para ello se utilizan cuadros de diálogos accesibles mediante los botones que se encuentran encima de las listas de elementos. El objetivo que se persigue con ello es aprovechar la mayor resolución de estos dispositivos para lograr una mayor agilidad en cuanto a la inserción de los elementos en fragmentos. Cuando se presiona el botón de "add element" se abre un cuadro de diálogo en el centro de la pantalla donde se deberá insertar el nombre del elemento y el tipo. Al cambiar el tipo, la interfaz cambiará dinámicamente, añadiendo los huecos necesarios para que el usuario pueda agregar la información específica para cada tipo de elemento escogido.
Como se muestra en la Fig. 16, al elegir el tipo de actividad el usuario puede escoger el servicio entre los ya predefinidos (Fig. 12A); al elegir el tipo paralelo podrá introducir la cantidad de ramas de las que consta el fragmento (Fig. 12B); y, por último, el tipo inclusión en condicional (Fig. 12C) o inclusión en bucle (Fig. 12D) permitirá crear un fragmento de este tipo eligiendo el primer y ultimo elemento de un grupo en la rama paralela que formarán el nuevo fragmento. En el momento en el que son creados, se mostrarán automáticamente en la lista de elementos. Así, si es un fragmento, se podrá crear su propia lista de elementos internos.

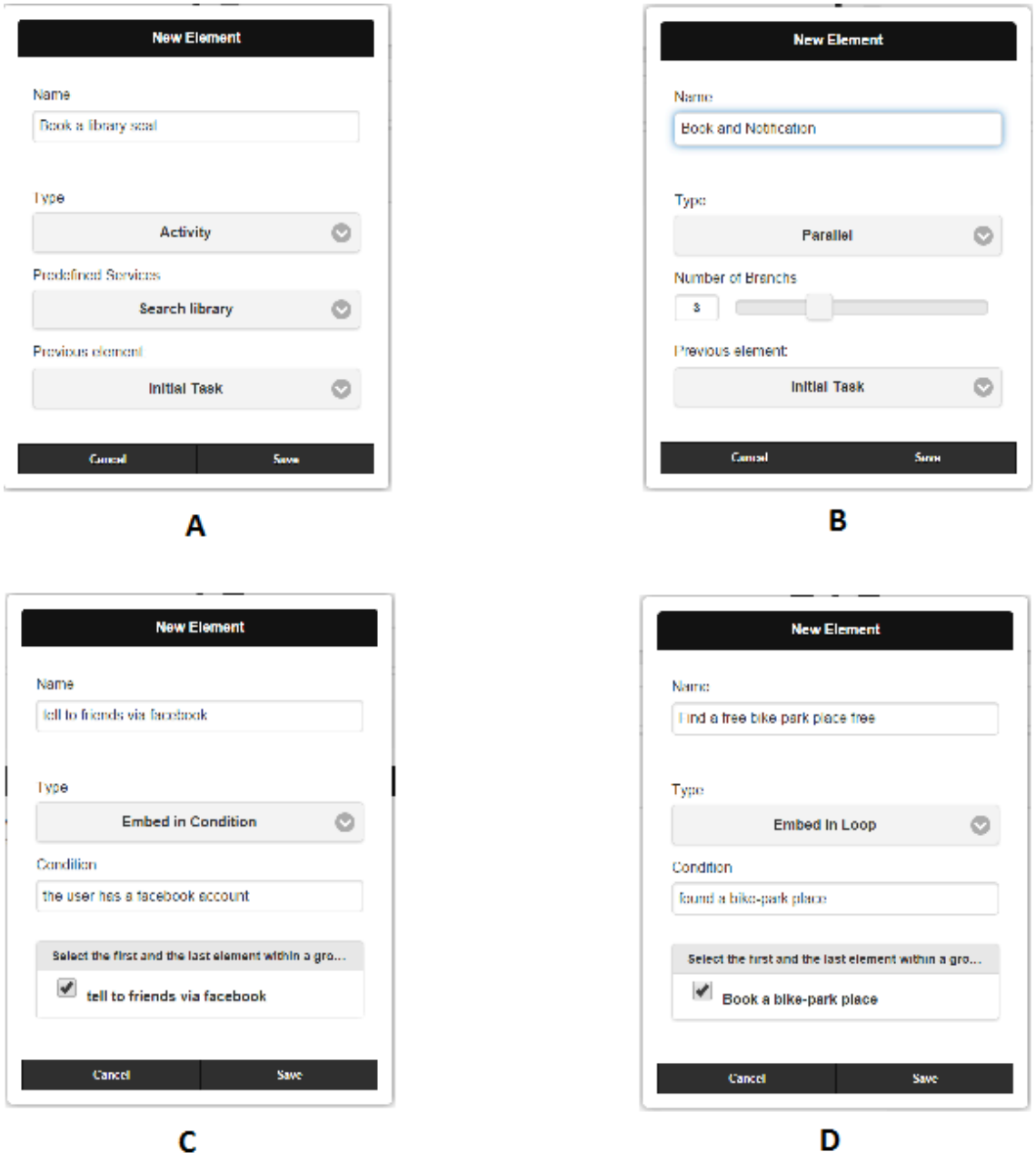


*Fig. 12 – Cuadro de diálogo de inserción en fragmentos para tabletas*

Persistencia. Al finalizar el usuario el modelo, la herramienta permite guardarlo para poder posteriormente retomarlo y continuar trabajando

en él. El usuario puede acceder desde la página principal del modelo a la "página de salvar", utilizando el botón Save Model de la parte inferior, tal y como se muestra en la Fig. 6A para smartphones y Fig. 6C para tabletas. En ella se deberá introducir el nombre del servicio y su descripción, quedando así guardado en un xml en el servidor. Véase la Fig. 17.

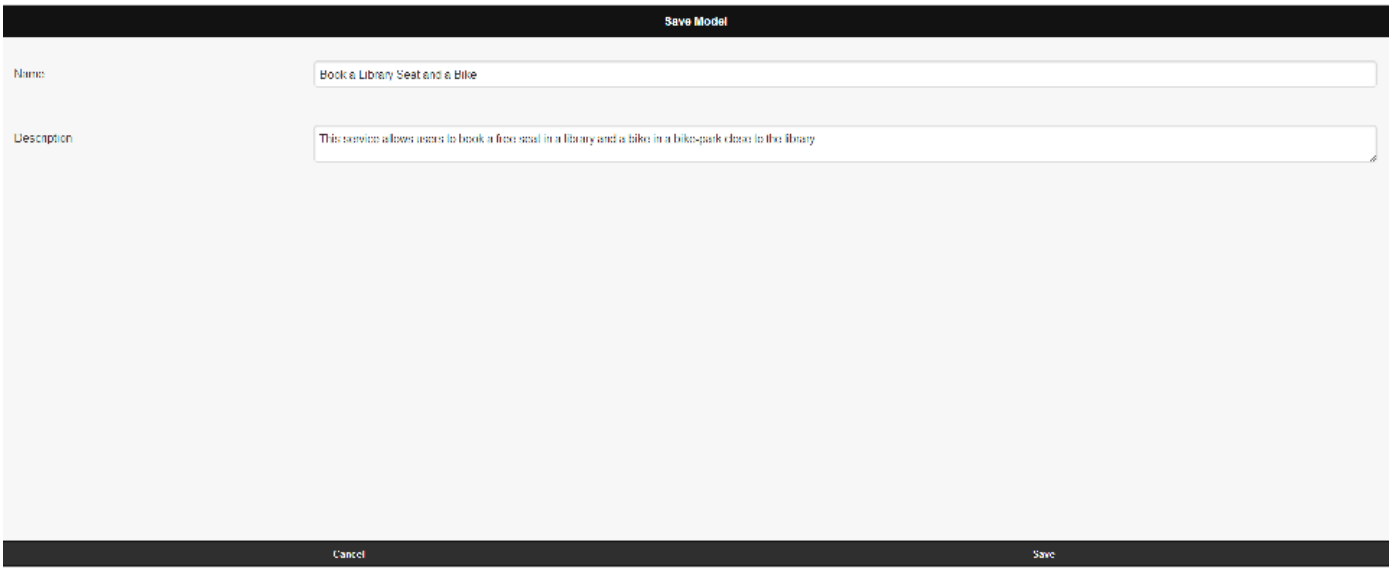


*Fig. 17 – Página de persistencia*

# 4 CONCLUSIONES

Con este trabajo se ha querido mostrar que es posible acercar, mediante una herramienta de desarrollo, la posibilidad de la creación de composiciones propias de servicios utilizado dispositivos móviles como smartphones o tabletas, servicios que serán suministrados para su utilización por la plataforma SITAC. La herramienta ha sido diseñada gracias a los conocimientos adquiridos para el desarrollo de usuarios finales como cambios de patrones y recomendaciones para la creación de una interfaz suficientemente amigable para que cualquier usuario, sea cual sea su bagaje de conocimientos previos, pueda utilizarla sin problemas.

Quedan muchos detalles que abordar para que los usuarios finales puedan procesar sus composiciones de servicios. Para ello se requiere la creación de las distintas partes tecnológicas (por ejemplo, BPMN, REST, XML) necesarias para su interpretación y ejecución. De ahí que la herramienta de usuario final sea completada con una lista de componentes como data mappings and transformations, invocación de servicios REST, etc.., que son necesarios para su ejecución en una máquina de procesos con el objetivo final de automatizar lo más posible las tareas necesarias para la ejecución de los procesos creados por el usuario final y permitir transparencia en la forma de ejecutarlos. Se ha puesto especial atención en no descuidar la herramienta para su creación, intentando mejorar las interfaces de una manera que permita al usuario interpretar mejor lo que está realizando e intentar reducir al máximo, sin mermar la facilidad de uso de la interfaz, las interacciones requeridas para realizar sus propias composiciones de servicios.

# 5 REFERENCIAS

Este trabajo está basado principalmente en el artículo “End-User Service Compositions in the Iot”, escrito por Victoria Torres Bosch, Pedro Valderas, Ignacio Mansanet y Vicente Pelechano Ferragud, todos ellos pertenecientes a la Pros Research Center perteneciente a la Universidad Politécnica de Valencia, España


**1** Lieberman, H., Paternò, F., Klann, M., and Wulf, V. (2006). End-User Development: An Emerging Paradigm. In: End-User Development, Lieberman, H., Paternò, F., and Wulf, V. (eds.), Springer Netherlands, 2006, ser. Human-Computer Interaction Series, vol. 9, Chapter 1, pp. 1-7

**2** Repenning, A., Ioannidou, A.: What makes end-user development tick? 13 design guidelines. End User Development 9 (2006) 1-41

**3** Ko, A.J., Myers, B.A., Aung, H.H.: Six learning barriers in end-user programming systems. In: Proceedings of the 2004 IEEE Symposium on Visual Languages-Human Centric Computing. VLHCC '04, Washington, DC, USA, IEEE Computer Society (2004) 199-206

**4** van Welie, M., Traelig;tteberg, H., Trtteberg, H.: Interaction patterns in user interfaces. In: Proc. Seventh Pattern Languages of Programs Conference: PloP 2000. (2000) 13-16

**5** Weber, B., Reichert, M., Rinderle, S.: Change Patterns and Change Support Features - Enhancing Flexibility in Process-Aware Information Systems. Data and Knowledge Engineering 66 (2008) 438-466

**6** Cuccurullo, S., Francese, R., Risi, M., Tortora, G.: Microapps development on mobile phones. In Costabile, M., Dittrich, Y., Fischer, G., Piccinno, A., eds.: EndUser Development. Volume 6654 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2011) 289-294