Generación de Interfaces de Usuario a partir de Modelos BPMN con Estereotipos

Eduardo Díaz[1], José Ignacio Panach[1], Silvia Rueda[1], Oscar Pastor[2]
[1] Escola Tècnica Superior d'Enginyeria, Departament d'Informàtica, Universidad de Valencia, Burjassot Valencia, España
[2] Centro de Investigación en Métodos de Producción de Software, Universidad Politécnica de Valencia, Valencia, España
[1]diazsua@alumni.uv.es, [1]joigpana@uv.es, [1]silvia.rueda@uv.es, [2]opastor@pros.upv.es

Resumen. La notación de modelo de procesos de negocio (BPMN – Business Process Model Notation) proporciona a las organizaciones un estándar que facilita una mayor compresión del proceso empresarial. BPMN se centra en los procesos funcionales, dejando el desarrollo de las interfaces a un lado. De este modo, el diseño de la interfaz generalmente depende de la experiencia subjetiva del analista y no existe un procedimiento para extraer la interfaz de los procesos. Este artículo propone un nuevo método para generar interfaces de usuario a partir de modelos BPMN y Diagramas de Clases. La propuesta se basa en la identificación de reglas de generación de procesos a interfaces. Se han definido estereotipos para extender la notación BPMN en aquellas reglas donde haya más de una posible transformación. Estos estereotipos permiten aplicar la regla de forma inequívoca. Las reglas se extrajeron de cinco proyectos, tres existentes de Bizagi y dos de empresas reales. Específicamente, la propuesta se basa en la extracción de reglas para generar interfaces de usuario basadas en tres patrones, Patrón de Secuencia, Patrón de Decisión Implícita y Patrón de Estructura de Unión Sincronizada. Como resultado de nuestra propuesta, se han agregado catorce nuevos estereotipos a la notación BPMN. Para ilustrar la propuesta, los estereotipos se aplicaron a un ejemplo ilustrativo. Los resultados muestran que este trabajo es un "paso adelante" para la generación automática de códigos a partir modelos.

Palabras Clave: BPMN, Método, Reglas, Estereotipos, Interfaces de usuario.

1 Introducción

La notación de modelo de proceso de negocio (BPMN) proporciona a las organizaciones la capacidad de comprender sus procedimientos empresariales internos en una notación gráfica de una manera estándar. Las primitivas conceptuales básicas en los modelos BPMN son eventos, tareas, compuertas, carriles y flujos [1]. Actualmente, se requiere que BPMN documente los procesos empresariales con el objetivo de reducir el tiempo y los costos empresariales [2]. Los modeladores BPMN existentes permiten diseñar diagramas fácilmente, proporcionando opciones para mejorar los procesos empresariales, entre las herramientas de modelado de procesos, podemos destacar: Bizagi [3], Adonis NP [4], Auraportal [5] y además WebRatio [6]

que cuenta con un modelador BPMN y también una plataforma de combinación de BPMN y IFML (Interaction Flow Modeling Language) [7], que define la interacción de los usuarios con la aplicación BPM en todas sus tareas, el diseño de la interfaz se pone en segundo plano, dejando esta tarea al diseñador, quien debe hacer el esfuerzo de implementar interfaces de acuerdo con el modelo BPMN. La forma en que las interfaces se derivan de los modelos BPMN se realiza actualmente de forma artesanal, sin ningún tipo de procedimiento, solo dependiendo de la experiencia del analista. Esto significa que todo el esfuerzo que se ha puesto en la construcción del modelo BPMN no es útil en el diseño de las interfaces. Además, normalmente, los analistas que crean los modelos BPMN no son los mismos diseñadores que implementan la interfaz de usuario, generando un espacio entre lo que se describe en los modelos BPMN y lo que realmente se implementa en la interfaz.

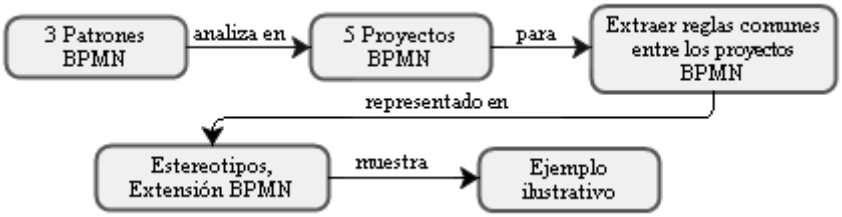


Fig. 1 Estructura de nuestra propuesta.

En este artículo, proponemos un método para generar interfaces a partir de modelos BPMN. El proceso se ha resumido en la Fig. 1. El enfoque se basa en el estudio de 3 patrones (Patrón de secuencia, Patrón de decisión implícita y Patrón de estructura de unión sincronizada) que son ampliamente utilizados en los modelos de procesos de negocios del conjunto de patrones como control básico de flujos, basado en eventos y para ramificicación y sincronización. Hemos buscado el uso de estos tres patrones en 3 proyectos de Bizagi [8] y 2 proyectos de empresas reales para extraer las reglas comunes de transformación de los modelos BPMN al código. Bizagi cuenta con un modelador BPM y además con un repositorio de modelos BPMN y la implementación de las interfaces, por lo que podemos analizar el mapeo entre ambos elementos. Al analizar cómo resulta cada modelo de BPMN en una interfaz, podemos extraer reglas de transformación que se pueden generalizar para cualquier proyecto que contengan estos patrones. Cuando las reglas tienen varias alternativas para generar las interfaces, necesitamos una semántica inequívoca para especificar qué alternativa vamos a usar. Para este objetivo, hemos ampliado el modelo BPMN con nuevos estereotipos que permiten especificar cómo generar interfaces en esas reglas con más de una alternativa. Los estereotipos del modelo BPMN también se han complementado con los estereotipos del Diagrama de Clase UML, ya que parte de la interfaz depende del modelo de persistencia.

El enfoque se ha aplicado en un ejemplo ilustrativo para mostrar cómo podemos trabajar con modelos BPMN extendidos junto con estereotipos y complementado con los atributos del Diagrama de Clases. El ejemplo también muestra claramente la correspondencia entre los modelos BPMN y la interfaz del usuario final.

El documento está estructurado de la siguiente manera: la sección 2 revisa la literatura relacionada con la generación de interfaces de usuario desde BPMN. La Sección 3 presenta patrones de BPMN. La Sección 4 define las reglas para generar interfaces a partir de patrones BPMN. La Sección 5 presenta reglas y estereotipos para extender la notación BPMN. La Sección 6 muestra un ejemplo ilustrativo. Finalmente, la Sección 7 concluye la propuesta.

2 Estado del Arte

Esta sección revisa los trabajos previos de la literatura relacionada con nuestra propuesta de generación de interfaz de usuario a partir de BPMN. Marco Brambilla et al. [9] definieron un modelado de procesos en aplicaciones web. Presentan nuevos métodos de ingeniería web para la especificación de alto nivel de aplicaciones que presentan procesos de negocios e innovación de servicios remotos, el proceso de negocio y las aplicaciones web se benefician del modelado de alto nivel y la generación automática de códigos. La investigación de Brambilla tiene como trabajo futuro abarcar una integración completa de modelo de procesos con BPML (Business Process Modelling Language) en el conjunto de herramientas WebRatio. Webratio cuenta con un lenguaje de modelado WebML [10] pemitidiendo agregar comportamiento funcional del sistema.

Lei Han et al. [11] definieron un enfoque de derivación de las interfaces de usuario de los modelos BPMN. Ese enfoque se basa en un modelo de proceso empresarial enriquecido con roles desarrollado con descripciones de tareas y datos asociados. El modelo se especifica en un BPMN extendido. Se identifica un conjunto de patrones de flujo de control y patrones de flujo de datos para la derivación de UI. Se especifica un conjunto completo de restricciones y recomendaciones para admitir la generación y la actualización de la interfaz de usuario. El problema principal con ese enfoque es que actualmente sólo existen reglas de generación a un modelo que represente interfaces abstractas. No hay reglas para generar interfaces finales todavía.

Chun Ouyang et al. [12] definieron reglas de transformación entre los modelos BPMN y las definiciones BPEL. Este enfoque incluye un conjunto integrado de técnicas para traducir los modelos capturados utilizando un subconjunto central de patrones BPMN para generar código BPEL. Esta técnica transforma diagramas de procesos de negocios en bloques BPEL mapeados. El enfoque se basa en la semántica de los componentes en términos de redes de Petri. El principal problema es que actualmente las técnicas de transformación de los modelos BPMN para BPEL, se traducen de un grafo no estructurado a lenguajes textuales similares, esta transformación solo funciona en modelos BPMN no complejos.

Javier Gonzales et al. [13] definieron un enfoque para transformar modelos de BPMN en artefactos software. El enfoque consiste en tres pasos: (1) refinar los modelos BPMN a través de la reingeniería, (2) automatizar patrones para derivar análisis de software, (3) diseñar artefactos (Diagramas de Clase UML o Casos de Uso de UML). Estos artefactos se usarán posteriormente para desarrollar los componentes de software que soportan las actividades del proceso empresarial. El principal problema de ese trabajo es que se centra en los primeros dos pasos del enfoque, donde todavía no hay generación de interfaces.

Como resumen, la mayoría de artículos estudiados están desarrollando enfoques para la generación de interfaces de usuario que implementan modelos BPMN. Las reglas de generación identificadas en los trabajos existentes anteriores no son adecuadas para la mayoría de los modelos BPMN reales, ya que la mayoría de estos trabajos se encuentran en una etapa inicial en la que aún no se han definido las reglas de transformación.

3 PATRONES BPMN

Los patrones BPMN son situaciones que comúnmente ocurren en cualquier modelo de proceso de negocio. [14]. Estos patrones ayudan a reutilizar soluciones para enfrentar los problemas sufridos en cualquier proceso empresarial, por lo que se utilizan con frecuencia. Esta es la razón por la que elegimos la generación de interfaz a partir de tales patrones utilizados con frecuencia. De esta forma podemos asegurar que en cualquier modelo de BPMN podamos generar algunas interfaces (al menos la parte de los modelos basada en patrones). Este trabajo se enfoca en 3 patrones: un patrón de control de flujo básico (1) Patrón de Secuencia, un patrón basado en eventos (2) Patrón de Decisión Implícita y un patrón avanzado para ramificación y sincronización (3) Patrón de Estructura de unión sincronizada. Esta elección está justificada para los diferentes modelos de BPMN que utilizan este tipo de patrones.

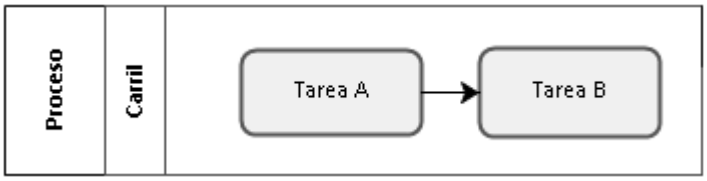


Fig. 2. Patrón de secuencia.

Patrón de secuencia, este patrón aparece cuando tiene que terminar una tarea antes de poder continuar con la siguiente de forma secuencial. La Fig. 2 muestra que la Tarea A debe finalizar antes de continuar con la Tarea B.

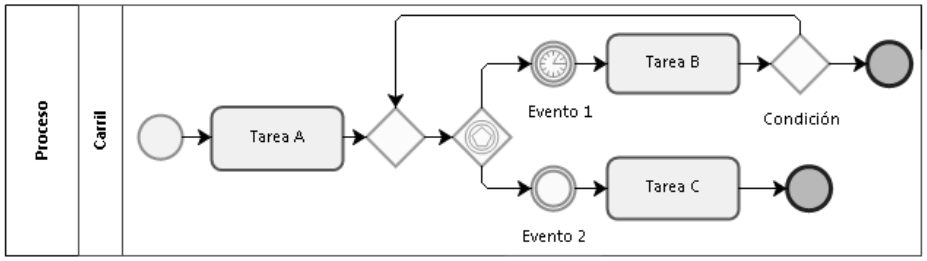


Fig. 3. Patrón de decisión implícita.

Patrón de Decisión Implícita, este patrón aparece cuando se debe escoger una rama de varias disponibles, la decisión se toma de acuerdo a los datos del proceso. Cuando se escoge una rama, las demás se deben deshabilitar. La Fig. 3 muestra una compuerta exclusiva basada en eventos (rombo con unos círculos en el interior), en la rama superior continúa el Evento 1 (tipo temporizador) con la Tarea B, en la parte inferior de la compuerta continúa el Evento 2 (tipo simple) que luego continúa la Tarea C.

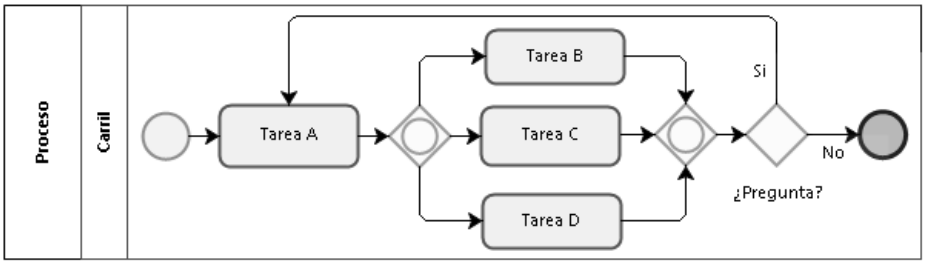


Fig. 4. Patrón de estructura de unión sincronizada.

Patrón de Estructura de unión sincronizada, El patrón aparece cuando un conjunto de tareas deben ser ejecutadas una a la vez, las tareas pueden ser realizadas en cualquier orden, sin embargo, no es posible realizar dos tareas al mismo tiempo, el proceso continúa hasta que todas las tareas de las ramas entrantes hayan terminado. La entrada está representada por una compuerta inclusiva, representada por un rombo y en su interior un círculo. La Fig. 4 muestra que utiliza una compuerta inclusiva, después continúa la Tarea B, Tarea C y Tarea D, luego se representa con una compuerta exclusiva preguntando si falta terminar alguna tarea.

Además de estos tres patrones, también es importante diferenciar entre el tipo de tareas y el tipo de eventos, ya que cada tipo tiene un rango diferente de posibilidades en la generación de interfaces. Este documento se centra en dos tipos de tareas: **tareas de tipo usuario** y **tareas de tipo servicio**. Las tareas de tipo usuario deben ser llevadas a cabo por una persona o usuario con la ayuda de un sistema software. Las tareas de tipo servicio son realizadas por un sistema sin intervención humana, por ejemplo, una tarea automática [15]. Por otro lado, un evento es algo que sucede durante el curso del proceso, afectando el flujo y generando un resultado. [16], estos se centran en dos tipos: **evento de tipo simple** y **evento de tipo temporizador**. Los eventos de tipo simple, indica que algo sucede en algún lugar entre el inicio y el final de un proceso, esto afectará el flujo del proceso. Los eventos de tipo temporizador indican un retraso dentro del proceso, este tipo de evento puede ser utilizado dentro de un flujo secuencial para indicar un tiempo de espera entre las tareas [16].

De acuerdo con el método de la Fig. 1, una vez que tenemos los patrones para analizar, luego estudiamos cómo de estos patrones se extraen reglas para generar componentes de interfaces de usuario. Para este objetivo, hemos utilizado tres proyectos de Bizagi [6] y dos proyectos de empresas reales. Bizagi es una suite con dos productos complementarios, un Process Modeler y una suite BPM. Desde 1989, Bizagi Process Modeler es un software freeware utilizado para diagramar, documentar

y simular procesos utilizando la notación BPMN estándar, cuenta con un repositorio de modelos BPMN e interfaces que implementan dichos modelos.

En este artículo usaremos cinco proyectos BPMN para analizar cómo se han implementado las interfaces a partir de los tres patrones. Para cada proyecto hemos accedido a los modelos de BPMN y a las interfaces de usuario finales que implementan dichos procesos. De esta forma, podemos comparar modelos e interfaces para extraer reglas de generación. A continuación, describimos brevemente cada proyecto. Se han tomado proyectos con varios niveles de complejidad con aplicación en diferentes contextos para extraer conocimiento lo más general posible.

- Proyecto 1: Gestión de Oportunidades de Ventas, este proceso consiste en dar seguimiento y gestionar actividades necesarias para convertir oportunidades de negocio en negocios reales. Cuando una oportunidad ha sido detectada, un agente del área de marketing ingresa la información para crear el registro en el sistema. A partir de este momento, el agente de marketing podrá actualizar la información de la oportunidad a lo largo del proceso de evaluación y compra del cliente.
- Proyecto 2: Gestión de Peticiones, Quejas y Reclamos, el proceso inicia con una solicitud, es recibida por un agente de atención al cliente quien registra en peticiones, quejas, reclamos o sugerencias. Si la solicitud es una sugerencia, se registra y se responde inmediatamente, de lo contrario se registra en el sistema y se brinda una respuesta inmediata si es posible, sino, se solicita información al solicitante.
- Proyecto 3: Gestión de Cambios, el proceso responde a la necesidad de introducir cambios como parte de la estrategia de continuidad del negocio, nuevos requisitos y mejora continua, determinando y minimizando su impacto a través de una planificación y control adecuados de su ejecución. La información que debe registrarse en cada actividad y la rastreabilidad de cada cambio le permitirán priorizar y responder de manera eficiente a las solicitudes.
- Proyecto 4: Solicitud de crédito, el proceso consta básicamente de un registro de solicitud, donde el cliente manifiesta su interés de adquirir un crédito. En esta etapa se incluye la presentación de la solicitud y documentación requerida a la entidad, la documentación debe ser entregada en menos de cuatro días, pasado este tiempo harán seguimiento al cliente. Al presentar la documentación se realiza una verificación de la información, se realiza el análisis o estudio de la solicitud de crédito y por último encontramos las actividades referentes para hacer efectivo el crédito o informar el rechazo del cliente.
- Proyecto 5: Venta de pizza, el proceso se inicia cuando el cliente realiza el pedido de pizza, el vendedor verifica la pedido para que el cocinero pueda elaborarla, al terminar el pedido de pizza es recibida por el empaquetador, quien lleva la pizza al cliente. Por último, el cliente realiza el pago de la pizza. Se debe tener en cuenta que la entrega del pedido de pizza tiene una tolerancia de sesenta minutos, pasado ese tiempo el cliente puede hacer un reclamo.

Los que pertenecen a Bizagi son Proyecto 1, Proyecto 2 y Proyecto 3, y de las empresas reales son Proyecto 4 y Proyecto 5. Los proyectos más extensos y complejos son el Proyecto 1, el Proyecto 2 y el Proyecto 3, y los más simples son el Proyecto 4 y el Proyecto 5. El nivel de complejidad depende del número de primitivas conceptuales en el modelo de proceso comercial.

4 IDENTIFICACION DE REGLAS PARA GENERAR INTERFACES A PARTIR DE PATRONES BPMN

Cada uno de los cinco proyectos se analizó para identificar las reglas de transformación de los modelos BPMN a las interfaces de usuario. Estos proyectos BPMN ya cuentan con interfaces de usuario, a partir de las cuales hemos extraído las reglas de generación de componentes gráficos mediante la observación. Se ha intentado diferenciar varias opciones de transformación para cada patrón que generará diferentes opciones visuales en las interfaces.

El modelo BPMN tiene información sobre el comportamiento del proceso de negocio, pero carece de información importante para la generación de la interfaz. Por lo tanto, la generación de interfaz no puede abordarse solo con un modelo BPMN, sino que debe complementarse con un Diagrama de Clases UML [17]. Desde el modelo BPMN podemos extraer información sobre la navegación entre las interfaces y su comportamiento, mientras que el Diagrama de Clases se puede utilizar para identificar qué información se requiere en cada interfaz. Esa es la razón por la que también utilizamos elementos del Diagrama de Clases en nuestra propuesta.

A continuación describimos las reglas para generar interfaces a partir de modelos que fueron identificados en los cinco proyectos.

La regla 0 y la regla 1 se aplican en todos los tres patrones de BPMN.

R0: Hemos identificado que en todos los proyectos, un carril suele generar un formulario. A partir de los atributos de un diagrama de clases, podemos extraer los campos del formulario. En el formulario, hay campos de entrada de diferentes tipos: (1) campos de entrada que aceptan cualquier tipo de cadena de texto (Textbox) [18]; (2) campos de entrada que son una lista cerrada donde el usuario debe seleccionar un elemento (Combobox o Listbox) [19], (3) campos de entrada que representan un valor booleano (Radiobutton o Checkbox) [20].

R1: Hemos identificado que en todos los proyectos, las tareas de tipo usuario generalmente generan un formulario. Los campos de entrada del formulario se extraen de los atributos del diagrama de clases, cada atributo genera un campo de entrada y los nombres de los atributos generan etiquetas. R1 se complementa con R0.

Patrón de Secuencia:

R2: Hemos identificado que en el Proyecto 1, Proyecto 2 y Proyecto 3, si las tareas de tipo de usuario están en el mismo carril y existe una dependencia entre ellas, este carril genera interfaces para guiar al usuario a través de un proceso de varios pasos. Esto se muestra de diferentes maneras: (1) Asistente Wizard: la navegación a través de diferentes formularios se realiza en un orden específico. El número de formularios depende del número de tareas de tipo usuario y cada formulario es una tarea de tipo usuario en orden secuencial [21]; (2) TabControl: contiene múltiples pestañas de tal manera que cada tarea de tipo de usuario genera una pestaña [22]; (3) Groupbox: control que muestra un marco alrededor de un grupo de componentes [23], cada tarea de tipo usuario en orden secuencial genera un groupbox. Para las opciones (2) y (3) se crea un formulario que contenga tabcontrol o groupbox. Para todas las opciones mencionadas, los campos de entrada del formulario se extraen de los atributos de las clases, cada atributo genera un campo de entrada. Esta regla se complementa con R0.

R3: Hemos identificado que en el Proyecto 1 y en el Proyecto 2, si las tareas de tipo de usuario están en diferentes carriles y existe una dependencia entre ellos, cada

carril genera un formulario. La creación del formulario se realiza de acuerdo con el orden de secuencia de cada tarea; los campos de entrada del formulario se extraen de los atributos de las clases, cada atributo genera un campo de entrada. Esta regla se complementa con R0.

R4: Hemos identificado que en el Proyecto 2 y el Proyecto 3, si la tarea A de tipo usuario, la tarea B de tipo servicio y existe una dependencia entre ellos, entonces la tarea A conduce a generar un formulario. Los datos registrados en el formulario de tarea A pueden procesarse en la tarea B mediante un servicio web o una tarea automática, los resultados pueden mostrarse en: (1) informe: diseñador de informes y gráficos [24], (2) datagrid: muestra datos en una cuadrícula personalizable. [25]

Patrón de Decisión Implícita:

R5: Hemos identificado que en el Proyecto 1, Proyecto 3, Proyecto 4 y Proyecto 5, si después de la compuerta basada en eventos continúa un evento de tipo simple y si se encuentra asociado con atributos del diagrama de clases, da lugar a generar un hipervínculo que lleva el nombre del evento, este hipervínculo envía a un formulario que se genera con los atributos asociados al evento. Este hipervínculo está asociado con una tarea de tipo de usuario anterior. R5 se complementa con R0.

R6: Hemos identificado que en el Proyecto 4 y Proyecto 5, si después de la compuerta basada en eventos continúa un evento de tipo temporizador que utiliza una variable de tiempo (t), este evento se encuentra asociado a una tarea de tipo usuario anterior que ha registrado un campo de tipo datetime [26]. Esto puede generar los siguientes controles: (1) Timer, control que genera una acción programada después de finalizar un intervalo de tiempo [27], este control tiene una variable de inicio que empieza en cero y una variable de término que en este caso es la variable (t), solo se usa cuando el tiempo del evento de tipo temporizador del modelo BPMN son segundos, minutos y horas; se activa cuando es registrado el campo datetime. Por ejemplo registrar pedido de pizza, si no está elaborada en 60 minutos, avisará a través de un mensaje. (2) Cuadro de mensaje, este control contiene mensaje y un título definido por el usuario. Se usa cuando el evento de tipo temporizador del modelo BPMN son días, semanas y meses, el valor del campo datetime debe restarse con el valor de la hora actual del sistema. Este resultado tiene que ser del mismo tipo que la variable (t) para comparar, si el resultado es mayor o igual al valor de variable (t), genera un Cuadro de mensaje, indicando que ya se encuentra en el tiempo establecido por el evento, por ejemplo el registro de solicitud de crédito de un cliente, si el cliente no presenta la documentación requerida en cuatro días, el sistema avisará a través de un cuadro de mensaje que el analista tiene que contactar con el cliente.

Patrón de Estructura de Unión Sincronizada

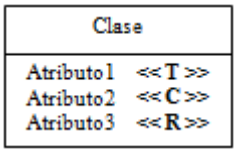
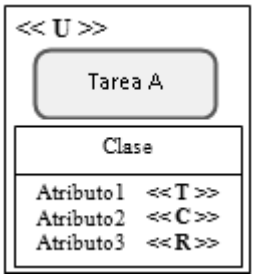
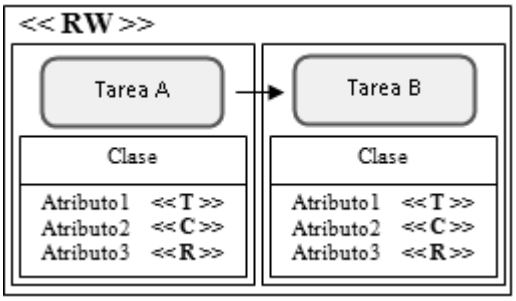
R7: Hemos identificado que en el Proyecto 1, Proyecto 2 y Proyecto 3, si todas las ramas de la compuerta inclusiva se encuentran en el mismo carril, se generan los siguientes controles: (1) Combobox, control que muestra lista desplegable de ítems, el número de ítems es el número de tareas de tipo usuario que se encuentran dentro del patrón. Los ítems de este control son los nombres de las tareas de tipo usuario. Cada ítem seleccionado muestra un formulario de cada tarea de tipo usuario, (2) Hipervínculo, el número de hipervínculo depende del número de tareas de tipo usuario que se encuentran dentro del patrón, cada hipervínculo tiene el nombre de cada tarea de tipo usuario, cada hipervínculo envía a una tarea de tipo usuario que es un formulario con los atributos del diagrama de clases. Cada formulario generado crea

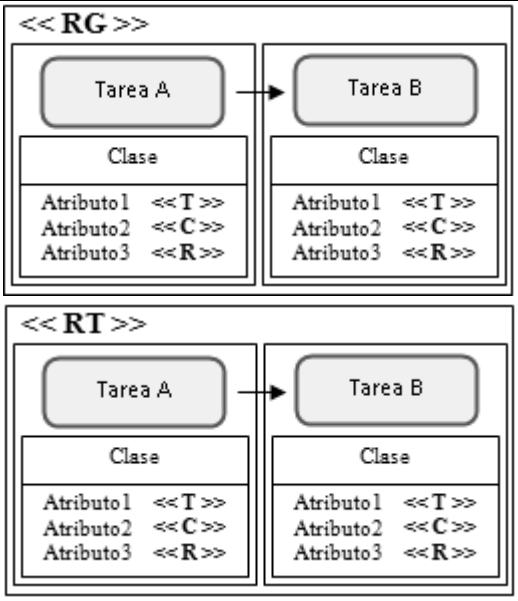
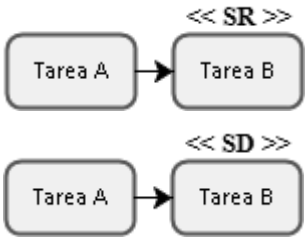
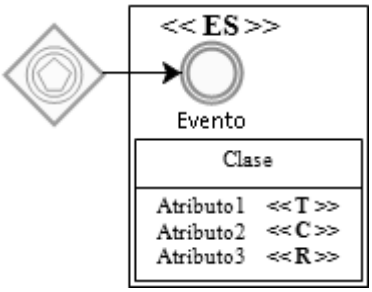
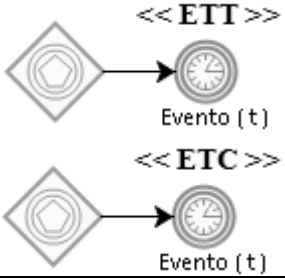
un botón que al reaccionar con el evento clic, muestra una ventana emergente. Esta ventana se puede diseñar de dos formas: (1) Una pregunta, es el texto de la compuerta exclusiva, esta compuerta se encuentra después del cierre de la compuerta inclusiva del patrón, y (2) las opciones, que son 2 botones (Sí / No) que llevan el nombre de las ramas de la compuerta exclusiva.

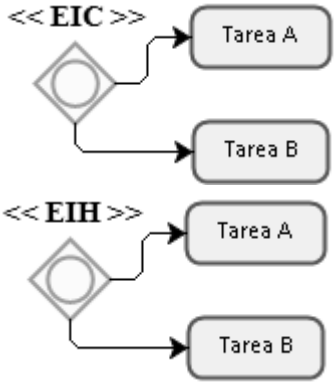
5 REGLAS Y ESTEREOTIPOS DE BPMN

Hemos visto en la sección anterior que el mismo patrón puede generar diferentes representaciones gráficas. Para cada alternativa de representación gráfica vamos a definir un estereotipo para que las reglas de generación se puedan aplicar de manera inequívoca. La Tabla 1, pretende resumir cómo generar interfaces utilizando estereotipos en las reglas de transformación previamente definidas. A continuación explicamos cada estereotipo

Tabla 1. Reglas, Estereotipos y Controles generados de interfaz de usuario.

Reglas	Estereotipos	Controles de Interfaz de Usuario
R0		<< T >> Textbox << C >> Combobox o Listbox << R >> Radiobutton o Checkbox
R1 R3		<< U >> Formulario
R2		<< RW >> Wizard < RT >> Tabcontrol < RG >> Groupbox

		
R4		<< SR >> Informe << SD >> Datagrid
R5		<< ES >> Hipervínculo - Formulario
R6		<< ETT >> Timer << ETC >> Cuadro de mensaje

R7		<< EIC >> Combobox – Formulario. << EIH >> Hipervínculo - Formulario

Las clases que aparecen son estereotipos que representan un subconjunto del modelo de clase que se usa en la tarea de tipo usuario, los estereotipos << T >>, << C >> y << R >> se pueden utilizar en los atributos del estereotipo de clase, << T >> representa la generación de Textbox, << C >> representa la generación de Combobox, o Listbox, << R >> representa la generación de Radiobutton o Checkbox. Estos estereotipos de clase están asociados con la tarea de tipo de usuario y se pueden usar para la Regla 0.

Los estereotipos << U >>, << RW >>, << RT >> y << RG >> se representan a través de una tabla que contiene tareas de tipo usuario. Estos estereotipos se complementan con los atributos del diagrama de clases usando la Regla 0. << U >> representa la generación de un formulario, este estereotipo se puede usar para las Reglas 1 y 3. << RW >> representa la generación de un asistente wizard << RT >> representa la generación de un tabcontrol. << RG >> representa la generación de un groupbox. Estos estereotipos se pueden usar para la Regla 2. << SR >> representa la generación de un informe. << SD >> representa la generación de un datagrid. Tanto los informes y datagrids se crean dentro de un nuevo formulario. Estos estereotipos se pueden usar para la Regla 4. << ES >> se puede usar en evento de tipo simple que pertenece al patrón de decisión implícita, puede generar hipervínculo y formularios.

<< ETT >> se puede usar en el evento de tipo temporizador que pertenece al patrón de decisión implícita, puede generar un timer. << ETC >> se puede usar en el evento de tipo temporizador que pertenece al patrón de decisión implícita, puede generar un cuadro de mensaje. << EIC >> se puede usar en la compuerta implícita que pertenece al patrón de estructura de unión sincronizada, puede generar comboboxes y formularios. << EIH >> se puede usar en la compuerta implícita que pertenece al patrón de estructura de unión sincronizada, puede generar hipervínculos y formularios

6 EJEMPLO ILUSTRATIVO

Para ilustrar el uso de reglas de transformación y estereotipos, usaremos el Proyecto 4 Solicitud de Crédito. La Fig. 5 muestra el modelo tal como fue definido originalmente y luego mejoramos el modelo con estereotipos propuestos para generar su interfaz.

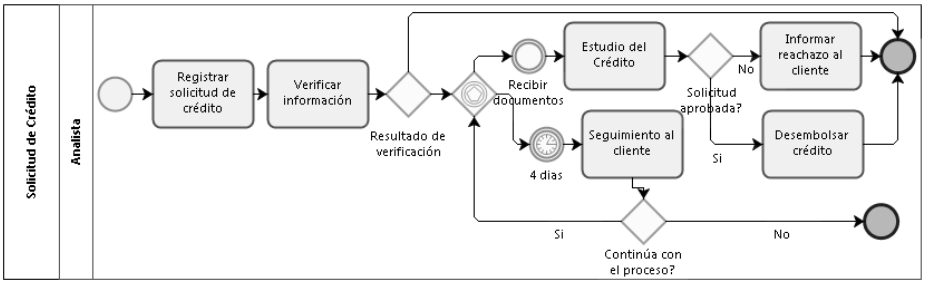


Fig. 5. Proyecto 4, Solicitud de crédito.

La Fig. 6 muestra un evento de tipo simple con estereotipo << ES >> aplicando la regla R5, que se complementa con los atributos del diagrama de clases. Estos atributos son asociados con estereotipos de la regla R0, la clase Cliente muestra Nombres, Apellidos y DNI asignados con << T >>, la clase Docs_Cliente muestra Recibo_pago, Boleta_pago, Copia_Dni y Otros_documentos asignados con << R >>, estos dos últimos estereotipos corresponden a R0.

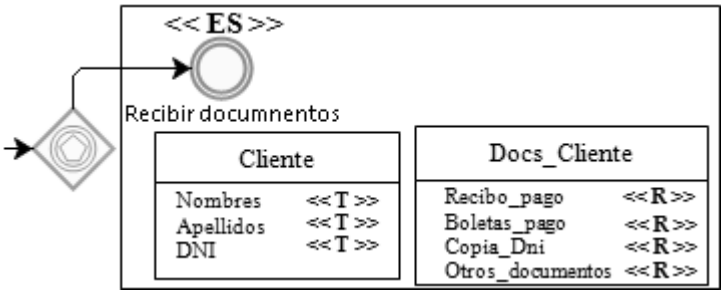


Fig. 6. Asignación de estereotipo << ES >> a evento de tipo simple Recibir documentos.

La Fig. 7 muestra la generación de un hipervínculo que se encuentra asociado al formulario que corresponde a la tarea de tipo usuario anterior. Este hipervínculo se encuentra enlazado con el formulario de Recibir Documentos – Solicitud de Crédito.

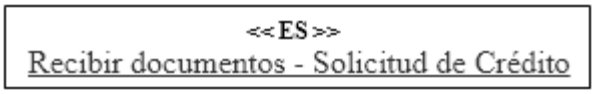


Fig. 7. Hipervínculo generado.

La Fig. 8 muestra formulario generado. Los estereotipos se han colocado para ilustrar de qué estereotipo procede cada uno de los elementos gráficos. Estos estereotipos no se verían en la aplicación final. Los elementos generados son: textbox (<< T >>), checkbox (<< R >>) y etiquetas (nombres de atributos del diagrama de clases).

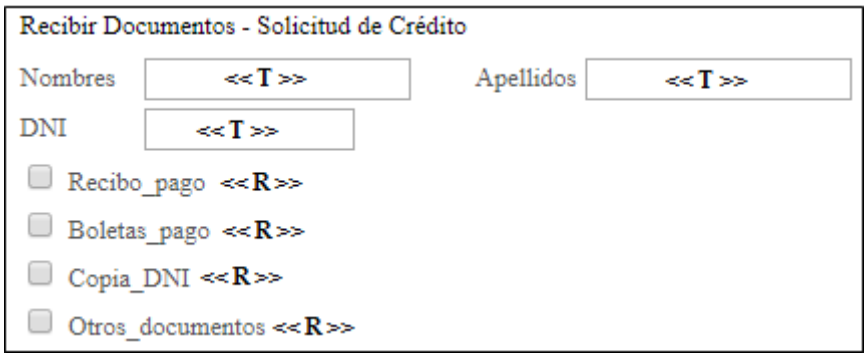


Fig. 8. Formulario Recibir Documentos – Solicitud de Crédito.

Como conclusión, tenemos un conjunto de reglas y estereotipos para generar interfaces. Se debe tener en cuenta que la funcionalidad detrás de estas interfaces no se aborda en este trabajo. Además, aún no hemos implementado reglas de transformación en un compilador de modelo real capaz de generar código de forma automática.

7 Conclusiones y trabajos futuros

Este artículo presenta un método para generar componentes de interfaces de usuario desde BPMN. Primero, se analizó 5 proyectos reales para identificar reglas de transformación de modelos BPMN a interfaces. Estas transformaciones se han centrado en el uso de 3 patrones BPMN ampliamente utilizados: de secuencia, de decisión implícita y de estructura de unión sincronizada. Segundo, hemos definido una lista de estereotipos para esas reglas de generación con más de una alternativa. Tercero, hemos ilustrado la aplicabilidad de la propuesta con un ejemplo que muestra cómo las reglas de transformación y los estereotipos podrían generar interfaces. El enfoque tiene algunas limitaciones, una de ellas es que existe una fuerte dependencia entre BPMN y diagramas de clases. No todos los modelos de BPMN tienen un diagrama de clase para representar su persistencia, lo que reduce la aplicabilidad de nuestro enfoque. El objetivo del enfoque a largo plazo es obtener un método holístico de desarrollo de software a partir de modelos BPMN. Sin embargo, se debe tener en cuenta que este trabajo solo se centra en la generación de interfaces. La forma en que la interfaz desencadena las funciones y cómo estas funciones se extraen de los modelos es parte del trabajo futuro. Otro trabajo futuro es la implementación de estas reglas de transformación en un compilador de modelos. De esta manera, podemos generar interfaces reales de acuerdo a reglas de generación, esto permitirá realizar una validación real a partir del código generado con las reglas.

Agradecimientos. Este trabajo se ha hecho con el soporte de la Generalitat Valenciana a través del proyecto IDEO (PROMETEOII/2014/039), Ministerio de Ciencia e Innovación Español a través del proyecto DataME (ref: TIN2016-80811-P) y Ministerio de Educación del Perú, PRONABEC – Beca Presidente de la República.

Referencias

1. BPMN: Business Process Modeling Notation. http://www.bpmn.org
2. Allweyer, T.: BPMN 2.0: Introduction to the Standard for Business Process Modeling (2016)
3. Bizagi: Bizagi. (2002) https://www.bizagi.com/es
4. Group, B.: Adonis NP. (2017) https://es.boc-group.com/adonis-E
5. IBM: Auraportal. (2017) https://www.auraportal.com/es/
6. WebRatio: WebRatio. (2018) https://www.webratio.com/site/content/es/home
7. IFML: Interaction FLow Modeling Language. (2018) http://www.ifml.org/
8. Bizagi: Examples of BPMN Projects. Vol. 2017 (2017) https://www.bizagi.com/es/comunidad/process-xchange
9. Brambilla, M., Ceri, S., Fraternali, P., Manolescu, I.: Process modeling in Web applications. ACM Trans. Softw. Eng. Methodol. **15** (2006) 360-409
10. Granada, D., Vara, J.M., Brambilla, M., Bollati, V., Marcos, E.: Analysing the cognitive effectiveness of the WebML visual notation. Software and Systems Modeling **16** (2017) 195-227
11. Han, L., Zhao, W., Yang, J.: An approach towards user interface derivation from business process model. Communications in Computer and Information Science **602** (2016) 19-28
12. Ouyang, C., Dumas, M., Aalst, W.M.P.V.D., Hofstede, A.H.M.T., Mendling, J.: From business process models to process-oriented software systems. ACM Trans. Softw. Eng. Methodol. **19** (2009) 1-37
13. Gonzalez-Huerta, J., Boubaker, A., Mili, H.: A business process re-engineering approach to transform BPMN models to software artifacts. Lecture Notes in Business Information Processing **289** (2017) 170-184
14. Bizagi: Process modelling patterns. (2014) http://resources.bizagi.com/docs/Workflow_Patterns_using_BizAgi_Process_Modeler_Esp.pdf
15. A., S.: BPMN Modeling and reference guide (2008)
16. Bizagi: Bizagi Event. (2002) http://help.bizagi.com/process-modeler/es/index.html?eventos.htm
17. UML: Unified Modeling Language. (2017) http://www.uml.org/
18. Perry, G.: Aprendiendo Visual Basic 6 (1999)
19. Ramírez, J.F.: Aprenda Visual Basic practicando (2001)
20. Chapman, D.: Visual C++ .NET (2002)
21. MacDonald, M.: Pro .Net 2.0 Windows Forms and Custom Control in VB 2005 (2006)
22. Rubin, E.: Microsoft Net Compack Framework Kick Start (2004)
23. Bronson, G.: Introduction of Programming with Visual Basic Net (2005)
24. Bischof, B.: Crystal Reports Net Programming (2004)
25. Schmidt, M.: Microsoft Visual C# .Net 2003 Developers CookBook (2004)
26. Weyl, E.: Mobile HTML5 (2014)
27. Network, M.D.: Class Timer Windows forms. (2016) https://msdn.microsoft.com/es-es/library/system.windows.forms.timer(v=vs.110).aspx