Una herramienta para la Automatización de Procesos de Desarrollo de Software usando QVT: Transformación de Controles de Flujo SPEM a BPMN

Fabio Zorzan, Marcela Daniele, Mariana Frutos, Marcelo Uva

Dpto. de Computación, Facultad de Ciencias Exactas, Fco-Qcas y Naturales
Universidad Nacional de Río Cuarto, Río Cuarto, Córdoba, Argentina
{fzorzan, marcela, mfrutos, uva}@dc.exa.unrc.edu.ar

Abstract. Desde hace unos años Query/Views/Transformations (QVT), definido por la Object Management Group (OMG), se ha convertido en una herramienta muy utilizada a la hora de definir transformaciones de modelos. Este trabajo propone una herramienta QVT que permite la transformación de modelos que representan procesos de desarrollo de software, basados en el Software Process Engineering Metamodel versión 2 (SPEM), hacia modelos de procesos genéricos basados en la notación Business Process Modeling Notation (BPMN). Este artículo muestra una parte fundamental de la transformación definida en el Lenguaje Relations que forma parte de QVT, que define la traducción de los componentes SPE, que especifican el control de flujo de actividades de un proceso de desarrollo de software, a componentes BPMN que definen el flujo de actividades en un proceso de negocio. Para la implementación de la transformación se utilizó MediniQVT. La especificación BPMN resultante define parte de un proceso de negocio que representa un proceso de desarrollo de software, esta especificación es la entrada a un workflow estándar, de esta manera, se puede administrar con un motor de workflow la gestión de un proyecto de desarrollo de software.

Keywords: QVT, Relation, SPEM, Workflow, BPMN

1 Introducción

Como se mencionó en el resumen, desde hace algunos años ha tomado auge la utilización del estándar QVT [1] para la transformación de modelos, y se han desarrollado herramientas que lo implementan, hoy en día muy maduras, como lo es MediniQVT [2].

Los procesos de desarrollo de software se enfrentan a continuos cambios, lo que implica realizar esfuerzos importantes para adaptar las organizaciones y herramientas a estos cambios en los procesos. Para facilitar esto, es muy importante tener especificados los procesos de desarrollo en algún estándar, como lo es SPEM [3].

Un proceso de negocio es un conjunto de tareas lógicamente relacionadas, ejecutadas para obtener un resultado de negocio.

Los procesos de negocio pueden ser controlados y administrados por un sistema basado en software. Los procesos de negocio automatizados de esta manera se denominan workflow. Esta automatización resulta en una importante potenciación de las virtudes de dicho proceso. Se obtienen mejoras en cuanto a rendimiento, eficiencia y productividad de la organización.

En la industria del software se encuentran los procesos de negocios que tienen por finalidad la construcción o generación de un producto (software) de calidad en un tiempo determinado [4]. En este marco, el proceso de negocio más importante involucra la metodología de desarrollo utilizada para guiar la producción.

Para lograr parte de la automatización de la gestión de procesos de desarrollo de software, se propone una herramienta QVT [1]. Esta herramienta QVT implementa la traducción de los componentes WorkSequence del proceso de desarrollo de software especificado en SPEM a una especificación con componentes Gateway y Sequence Flow para un Workflow basado en el estándar BPMN [5] aceptado por la OMG. Esta traducción se obtiene a través de reglas de transformación definidas mediante el lenguaje Relations que forma parte de QVT. La dentición de dichas reglas es la principal contribución del trabajo. Las transformaciones se efectúan a nivel de metamodelo basadas en patrones de transformación. La transformación está definida entre el metamodelo SPEM y el metamodelo BPMN. La hipótesis de trabajo es plantear al proceso de desarrollo de software como un tipo proceso de negocio particular automatizándolos en todo o en parte a través de un motor de workflow.

En el ejemplo de aplicación se trabajó sobre la metodología de desarrollo de software denominada SmallRUP [6].

En la sección siguiente se presentan los trabajos relacionados, en la sección 3 se introducen los metamodelos SPEM y BPMN y en la sección 4 QVT. En la sección 5 se presenta el esquema general de la transformación, en la sección 6 la transformación propuesta, en la 7 un ejemplo de aplicación, y, finalmente, se exponen algunas conclusiones.

2 Trabajos Relacionados

Existen trabajos que abordan el problema de transformar modelos de procesos de alto nivel a procesos ejecutables, estos procesos ejecutables están definidos en un lenguaje de ejecución de procesos. En 2005, en [7] se presenta un trabajo que muestra como realizar una correspondencia de un modelo BPMN a una especificación de procesos BPEL4WS. En esta misma línea, en [8] se define una técnica para generar código BPEL4WS a partir de modelos de procesos especificados con un subconjunto de BPMN y diagramas de actividades UML. Estos dos trabajos están orientados a transformar modelos de procesos genéricos a modelos de procesos ejecutables, a diferencia del presente trabajo que está orientado a transformar modelos de procesos específicos para el desarrollo de software, especificados en SPEM, a modelos de procesos genéricos BPMN. Lo expuesto en estos artículos puede ser la base de algunos trabajos futuros planteados en esta tesis.

En 2006, en [4], los autores proponen una correspondencia entre dos metamodelos utilizados para la especificación de procesos de negocio. La correspondencia se define entre SPEM (versión 1), y el UML Extended Workflow Metamodel (UMLEWM),

que es una extensión de UML para modelar procesos workflow. El UMLEWM es un metamodelo workflow poco difundido comparado con el estándar BPMN. Además la versión 1 de SPEM ha tenido muy poca aceptación por parte de la industria.

En [9], se presenta una transformación formal de UML a BPMN, utilizando el lenguaje MOLA. El trabajo está orientado a la especificación de modelos de procesos genéricos orientados estrictamente al modelado sin considerar su ejecución. El presente trabajo contempla modelos de procesos específicos para el desarrollo de software, en particular los especificados con SPEM 2 y que puedan ser ejecutados por un motor de Workflow.

En 2010, en [10] se propone una transformación de componentes SPEM a BPMN. Este trabajo formaliza los metamodelos y su transformación usando RSL(RAISE). En el presente, además de proponer una transformación que tiene en cuenta aspectos dinámicos, se utiliza QVT, la cual es una tecnología que cuenta con herramientas muy maduras para especificar y ejecutar transformaciones.

3 Query/View/Transformation (QVT)

QVT [1] es un estándar definido por la OMG para la transformación de modelos y se basa principalmente en: la definición de un lenguaje para las consultas (Queries) sobre los modelos MOF, la búsqueda de un estándar para generar vistas (Views) que revelen aspectos específicos de los sistemas modelados, y finalmente, la definición de un lenguaje para la descripción de transformaciones (Transformations) de modelos MOF.

En este trabajo sólo se utiliza el componente de QVT que tiene como objetivo definir transformaciones, en particular el lenguaje Relations que es una especificación declarativa de relaciones entre metamodelos MOF. Una transformación QVT describe relaciones entre un meta-modelo fuente F y un meta-modelo objetivo O, ambos metamodelos deben estar especificados en MOF. Luego esta transformación definida se utiliza para obtener un modelo objetivo que es una instancia del metamodelo O a partir de un modelo fuente que es una instancia del metamodelo F.

3.1 MediniQVT

Esta herramienta implementa la especificación QVT/Relations de la OMG en un poderoso motor QVT. Está diseñada para transformaciones de modelos permitiendo un rápido desarrollo, mantenimiento y particularización de reglas de transformación. La herramienta está integrada a Eclipse y utiliza EMF para la representación de modelos.

4 Metamodelos

La transformación planteada involucra dos metamodelos: SPEM que permite el modelado de procesos de desarrollo de software, y el segundo BPMN, que está destinado al modelado de procesos de negocio.

4.1 SPEM

Los procesos en el desarrollo de software pueden ser vistos como productos, ya que están constantemente cambiando y evolucionando. También deben ser administrados y configurados para adaptarlos a las organizaciones y a las nuevas necesidades del entorno, agregando de esta forma la necesidad de un estándar unificado en esta área, esto debido a que cada una de estas técnicas y procesos definió sus propios estándares y terminologías usando incluso diferentes significados para la misma palabra.

Para especificar las actividades propuestas por un proceso de desarrollo particular y de esta forma proveer una solución a la necesidad antes planteada, la OMG definió un metamodelo para la Ingeniería de Procesos de Software (SPEM).

Para la definición de nuevos lenguajes, la OMG define una arquitectura basada en cuatro niveles de abstracción que van a permitir distinguir entre los distintos niveles conceptuales que intervienen en el modelado de un sistema. A esos niveles se les denomina M0, M1, M2 y M3. SPEM está dentro del nivel M2 y describe un metamodelo genérico para la descripción de procesos de software concretos que está basado en MOF [11] y utiliza UML como notación de modelado.

4.2 Workflow y BPMN

Un workflow se define como la automatización total o parcial de un proceso de negocio, durante la cual documentos, información o tareas son intercambiadas entre los participantes conforme a un conjunto de reglas procedimentales preestablecidas [12].

Un workflow comprende un número de pasos lógicos, conocidos como actividades. Una actividad puede involucrar la interacción manual o automática con el usuario.

Un motor workflow es un sistema de software que controla la ejecución de las actividades definidas en el workflow. La WfMC ha definido un Modelo de Referencia Workflow (Workflow Reference Model). Este modelo define 5 interfaces para la interoperabilidad de diferentes productos con un motor workflow.

En este trabajo interesa la interfaz 1 que especifica el formato de intercambio común para soportar la transferencia de definiciones de procesos entre productos diferentes, utilizando un lenguaje de definición de procesos como el XML Process Definition Language – (XPDL) [13] definido por la WfMC o el Business Process Execution Language for Web Services (BPEL4WS) [14] adoptado por OASIS.

Es importante a la hora de modelar un proceso de negocio poder utilizar una herramienta independiente de la implementación, así, de esta manera, poder utilizar la especificación del proceso de negocio para diferentes plataformas. Una herramienta de estas características que está siendo muy utilizada por grandes empresas es BPMN.

Los elementos de la notación están especificados en el metamodelo BPMN [15]. Este metamodelo está definido en el nivel M2 de la OMG y está basado en MOF.

5 ESQUEMA GENERAL DE LA TRANSFORMACIÓN

El esquema general de la transformación de procesos de desarrollo de software basados en SPEM a workflows puede ser visto en tres niveles: Metamodelo, Definición/Modelo y Ejecución, como lo muestra la Fig. 1.

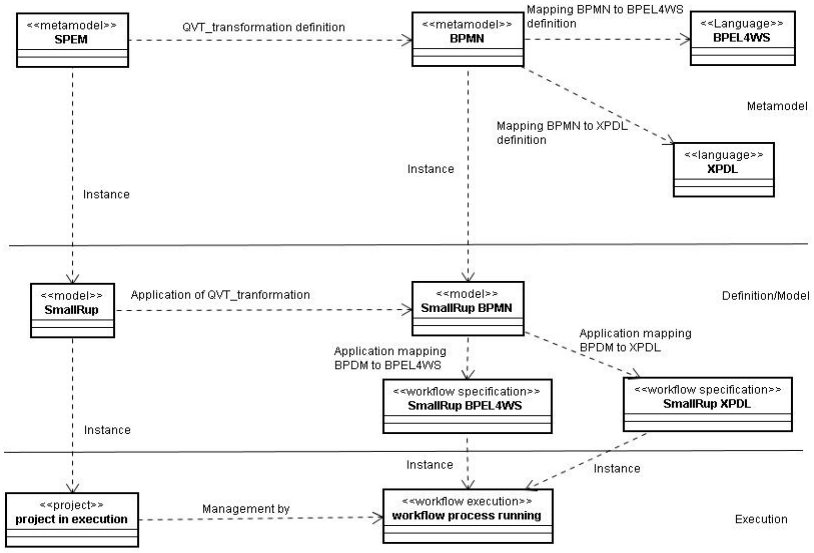

Fig. 1. : Vista general de la transformación.

En el nivel metamodelo se encuentran los metamodelos objetivos de la transformación definida en este trabajo, el metamodelo SPEM y el metamodelo BPMN, entre los cuales está definida la transformación mediante el lenguaje QVT. A su vez en este nivel se encuentran las definiciones de los mapping entre el metamodelo BPMN y los diferentes lenguajes de implementación de workflow, en este caso los lenguajes XPDL y BPEL4WS. Pasando al nivel de modelo/definición se encuentran los modelos específicos que definen un proceso de desarrollo de software concreto, como por ejemplo SmallRUP, y a partir de éste, por aplicación de la transformación definida en QVT a nivel metamodelo, se obtiene el modelo BPMN que define a SmallRUP como un proceso de negocio. También en este nivel se encuentran la aplicación de los mapping entre el metamodelo BPMN y los diferentes lenguajes de definición de procesos, que como resultado de la aplicación de estos mapping se obtiene la definición de SMALLRUP en un lenguaje(XPDL o BPEL4WS). Ésta definición se utiliza como entrada para la definición de procesos en un motor Workflow que implemente el lenguaje. Por último, en el nivel de ejecución, se encuentran los proyectos de desarrollo de software que siguen como metodología de desarrollo de software a SmallRUP y que son administrados automáticamente a través de motores de workflow que siguen como especificación de procesos de negocio a la definida en el nivel anterior.

6 TRANSFORMACIÓN DE SPEM AL METAMODELO BPMN

Como se mencionó anteriormente, el objetivo de este trabajo es hacer una contribución al mejoramiento de la gestión de los procesos software que están basados en el estándar SPEM. Para esto se propone automatizar la transformación de un proceso software basado en SPEM a un workflow estándar. De esta forma se puede utilizar una herramienta workflow que implemente el estándar de la WfMC(XPDL) o el estándar de OASIS(BPEL4WS) para asistir en la gestión de los procesos de desarrollo de software.

Para la definición de las reglas de transformación de metamodelos se adoptó el lenguaje Relations de QVT. Este lenguaje fue presentado en la sección 3 de este trabajo. En este caso la transformación sólo necesita ser definida en la dirección Metamodelo SPEM hacia Metamodelo BPMN. De esta manera los elementos del metamodelo SPEM están marcados como checkonly y los elementos del metamodelo BPMN están marcados como enforced, para que de esta forma, la ejecución de la transformación cree los elementos del modelo BPMN que se corresponden a los elementos del modelo fuente especificado en SPEM.

6.1 Transformación de WorkSequence SPEM a Gateway y SequenceFlow BPMN

En esta sección se presenta el núcleo del trabajo, éste consiste de la transformación de los elementos WorkSequence de SPEM a elementos SequenceFlow y Gateway del metamodelo BPMN. Los elementos WorkSequence SPEM permiten definir la dependencias entre actividades de los procesos de desarrollo de software. Los elementos SequenceFlow y Gateway BPMN permiten modelar el flujo de actividades en un modelo de procesos genérico, como lo son los procesos BPMN. Los objetos de la metaclase WorkSequence de SPEM son transformados de una manera genérica de acuerdo a su tipo, a continuación se presenta sólo la transformación de uno de los cuatro tipos de WorkSequence SPEM y la transformación genérica que tiene en cuenta todos los tipos de WorkSequence.

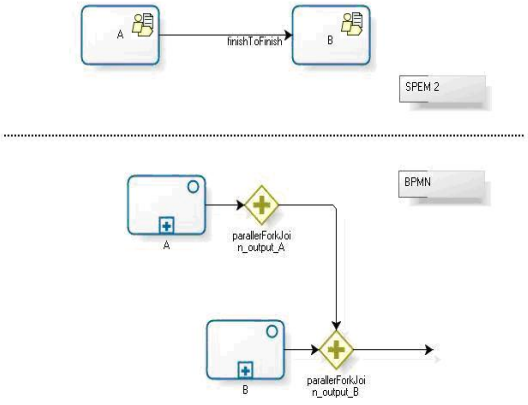

Fig. 2. Transformación de WorkSequence de tipo finishToFinish.

WorkSequence finishToFinish.

Como se muestra en la Fig. 2, el EmbeddedSubprocess BPMN “B” (generado por la Activity o TaskUse “B” SPEM), debe esperar a que finalice el EmbeddedSubprocess “A” (generado por la Activity o TaskUse “A” SPEM) para poder finalizar. La transformación se lleva a cabo agregando a la salida de “B”, un ParallelGateway BPMN que tiene como entrada a “B” y a un SequenceFlow que tiene como origen a “A” (en realidad al ParallelGateway que “A” tiene conectado en su finalización). Para la transformación de este tipo de WorkSequence SPEM, se considera que el EmbeddedSubprocess BPMN “B” termina cuando el token llega a él o los procesos que le siguen.

La relación que lleva a cabo esta transformación se denomina workSequenceFinishToFinishSPEMToSequenceFlowBPMN.

La definición de la relación en QVT/Relations se presenta a continuación:

```
relation workSequenceFinishToFinishSPEMToSequenceFlowBPMN
{
 nameActivity : String;
 nameWorkSequence :String;
 nameSuccessor: String;
 namePredecessor: String;
 checkonly domain spem2 activity: spem :: Activity
 {
  name = nameActivity,
  nestedBeakdownElement = workSequense:
     spem ::WorkSequence
  {
   name = nameWorkSequence,
   linkKind = spem::WorkSequenceKind ::finishToFinish,
   successor = successor:WorkBreakdownElement
            {name = nameSuccessor},
   predecessor=predecessor:WorkBreakdownElement
            {name = namePredecessor}
  }
 };
 enforce domain bpmn embeddedSubProcess:bpdm ::EmbeddedSubProcess
 {
  Id = 'id_' + nameActivity,
  Name = nameActivity,
  GraphicalElements =
     sequenceFlow...ToGategaySuccesor:bpdm::SequenceFlow{},
GraphicalElements=parallelGategayOutSuccessor:bpdm::ParallelGategay
  {
  Id ='id_'+'parallel_gategay_out_'+nameSuccessor
  },
GraphicalElements=parallelGategayOutPredecessor:bpdm::ParallelGategay
  {
```

```
    Id = 'id_'+'parallel_gategay_out_'+namePredecessor,
    Name = 'parallel_gategay_out_'+namePredecessor,
    GatewayType = bpdm::GatewayType::AND,
    Gates = gateOutParallerGategayPredecessor : bpdm::Gate
      {Id = 'id_gate_out_' + parallelGategayOutPredecessor.Name+
         '>' +parallelGategayOutSuccessor.Name,
      OutgoingSequenceFlow = sequenceFlowPaorToGategaySuccesor::
              SequenceFlow
          {id ='id'+'_sequenceFlow_gategay_out_'+ namePredecessor+
                     '->gategay_out_' + nameSuccessor,
               SourceRef = parallelGategayOutPredecessor,
               TargetRef = parallelGategayOutSuccessor
               }
    },
   GraphicalElements = parallelGategayOutSuccessor:
      bpdm::ParallelGategay
    {Id = 'id_'+'parallel_gategay_out_'+nameSuccessor,
     Name = 'parallel_gategay_out_'+nameSuccessor,
     GatewayType = bpdm::GatewayType::AND,
     Gates = gateInParallerGategaySuccessor: bpdm::Gate
      {Id = 'id_gate_in_'+ parallelGategayOutPredecessor.Name+
                '->'+parallelGategayOutSuccessor.Name,
      OutgoingSequenceFlow = sequenceFlowParToGategaySuccesor:
      SequenceFlow{}
        }
    }
   };
  when
  {
   activitySPEMToEmbeddedSubprocessBPMNGeneral
      (activity,embeddedSubProcess);
  }
 }
```

6.2 Transformación genérica de WorkSequence SPEM

Teniendo en cuenta las transformaciones particulares de los cuatro tipos de WorkSequence SPEM, se presenta la siguiente propuesta general que puede ser vista de una manera gráfica en la Fig. 3. Para cada EmbeddedSubprocess BPMN generado, se genera un ParallelGateway a la entrada y otro a la salida, de manera genérica. El ParallelGateway de entrada a un EmbeddedSubprocess “A” tiene como entradas a conectores desde todos los EmbeddedSubprocess (en realidad al ParallelGateway de entrada o salida, según corresponda) de las que depende el comienzo del EmbeddedSubprocess “A”, y como salida tiene al EmbeddedSubprocess “A” y a los Gateway de entrada o salida de los subprocesos que dependen del comienzo de “A”.

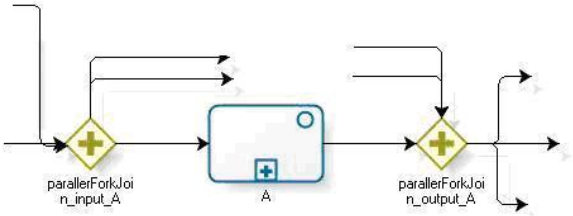

Fig. 3. Transformación genérica de WorkSequence SPEM.

El ParallelGateway de salida de un EmbeddedSubprocess “A” tiene como entradas a conectores desde todos los EmbeddedSubprocess de los que depende la finalización del EmbeddedSubprocess “A” y el propio EmbeddedSubprocess “A”, y como salida tiene a los ParallelGateway de entrada o salida de los EmbeddedSubprocess que dependen de la finalización de “A”.

7 Ejemplo de Aplicación

Como ejemplo de aplicación se utilizó una instanciación particular del Rational Unified Process (RUP) definida para pequeños proyectos denominada SmallRUP [6]. La elección del RUP se debió a que es una metodología que presenta buenas prácticas en el desarrollo de software moderno para una amplia gama de proyectos y organizaciones, está embebido en técnicas orientadas a objetos, utiliza UML como notación principal, permite a organizaciones del software ajustar el proceso a su necesidad específica y cubre diferentes dominios particulares.

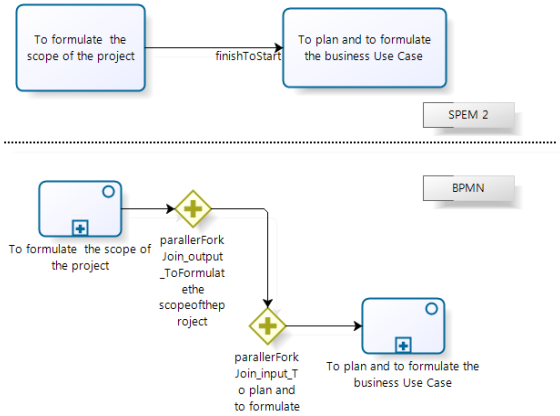

Fig. 4. Vista gráfica de la transformación propuesta para parte del ejemplo de aplicación.

Para mostrar en detalle la transformación de SmallRUP especificado en SPEM a su correspondiente modelo de procesos BPMN se seleccionó un subconjunto del proceso SmalRUP, esto incluye la dependencia entre dos actividades SMALLRUP a

su correspondiente modelo BPMN que representa el mismo flujo de secuencia. La dependencia seleccionada es la que define la secuencia entre las actividades de la primera iteración de SmallRUP, “Formular el alcance del proyecto”, “Planear y formular los casos de uso de negocio”. Esta dependencia está representada por un objeto WorkSequence de tipo finishToStart. que conecta ambas actividades, como se muestra en la parte superior de la Fig. 4. El submodelo BPMN obtenido luego de aplicar la transformación es el mostrado en la parte inferior de la Fig. 4.

En la figura 5 se muestra, dentro de la herramienta MediniQVT, el modelo SmallRup basado en BPMN, resultado de la aplicación de la transformación al modelo SPEM.

El modelo BPMN obtenido por la transformación fue ingresado al motor de Workflow Bonita [16], esta incorporación fue realizada en forma manual por medio de su editor de procesos BPMN, esto debido a que bonita no reconoce el formato Ecore generado por la transformación. Luego se ejecutó en el motor de workflow, durante la ejecución del proceso se pudo observar como la herramienta va guiando el proceso preestablecido. Bonita en la ejecución del proceso, además de guiar el flujo de trabajo por medio de su motor de workflow, provee una herramienta Web, denominada “User XP”, que utilizan los participantes del proceso para interactuar con el motor de workflow.

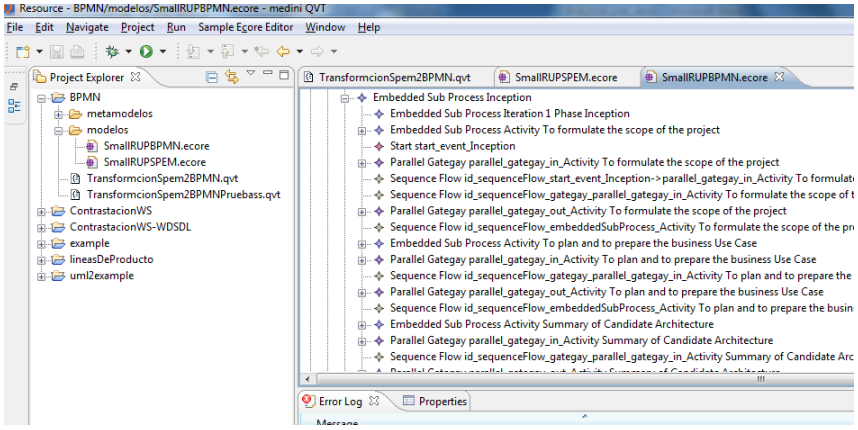

Fig. 5. Modelo SmallRUP basada en el Metamodelo BPMN en formato ecore, vista generada por el ecore model editor de MediniQVT

8 Conclusiones

Este trabajo tuvo como objetivo realizar una herramienta QVT que permita la traducción de elementos SPEM, que determinan el flujo de secuencia entre actividades, a elementos BPMN que representan la misma dependencia entre actividades.

Se logró, basado en las transformaciones de cada tipo de Work Sequence SPEM, una transformación genérica de elemento Work Sequence SPEM a Sequence Flow y Gateway BPMN. Los elementos BPMN que se obtienen respetan las mismas restricciones en el flujo de actividades que las originalmente definidas en SPEM.

Otro aspecto importante a destacar fue la utilización de una herramienta (MediniQVT), tanto para la definición de la transformación como para su ejecución. Esto era una limitación muy fuerte hace sólo un par de años atrás a la hora de utilizar QVT/Relation como lenguaje para la especificación de transformaciones de modelos.

El beneficio de esta transformación se reflejará también en el dinamismo de los cambios en los procesos de desarrollo de software: cualquier cambio en la especificación del proceso podrá ser propagado automáticamente a la especificación Workflow de dicho proceso.

Se adquirió una experiencia muy valiosa en la implementación de transformaciones QVT, esta experiencia será volcada en cátedras de Ingeniería de Software donde se aborden temas como Model Driven Architecture (MDA).

Al trabajar con estándares o especificaciones como SPEM, BPMN y QVT adoptadas a nivel mundial, proporciona facilidad de entendimiento por parte de los lectores y flexibilidad para poder interactuar con diferentes herramientas, permitiendo también una fácil extensión del trabajo.

Referencias

1. Object Management Group, “Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification”, OMG Document Number: formal/2008-04-03, Standard Document URL: http://www.omg.org/spec/QVT/1.0/PS/, último acceso Julio 2014.
2. ikv++: medini QVT. http://www.ikv.de/, último acceso Julio 2014.
3. Object Management Group, “ Software & Systems Process Engineering Metamodel Specification”; Version 2.0 with change bars; http://www.omg.org/docs/formal/08-04-01.pdf, último acceso Agosto 2014.
4. N. Debnath, D. Riesco, G. Montejano, et al, “Supporting the SPEM with a UML Extended Workflow Metamodel”, ACS/IEEE International Conference on Computer Systems and Applications (AICCSA'06), Dubai/Sharjah. March 8-11, 2006, www.ieee.org.
5. Object Management Group “Business Process Modeling Notation (BPMN) Specification”. Final Adopted Specification dtc/06-02-01, http://www.bpmn.org/Documents/OMG Final Adopted BPMN 1-0 Spec 06-02-01.pdf, último acceso Mayo 2014.
6. Gary Pollice “Using the RUP for small projects: Expanding upon Extreme Programming”, A Rational Software White Paper – 04/08/15, ftp://ftp.software.ibm.com/software /rational/ web/whitepapers/2003/tp183.pdf, último acceso Diciembre 2014.
7. Stephen A. White, “Using BPMN to Model a BPEL Process”, BPTrends, 3(3), March 2005, 1-18.
8. Chun Ouyang, Marlon Dumas, Stephan Breutel, and Arthur H.M. ter Hofstede, BPM, 2005, Translating Standard Process Models to BPEL, BPM Center Report BPM-05-27, BPMcenter.org, 2005.
9. Audris Kalnins, Valdis Vitolins “Use of UML and Model Transformations for Workflow Process Definitions” Databases and Information Systems, BalticDB&IS'2006, edited by Olegas Vasilecas, Johann Eder, Albertas Caplinskas, Vilnius, Technika, 2006, pp. 3.-15.
10. M. Perez Cota, D. Riesco, I. Lee, N. Debnath, G. Montejano,”Transformations from SPEM work sequences to BPMN sequence flows for the automation of software development

process” Journal of Computational Methods in Science and Engineering Volume 10, Supplement 1/ 2010, pages 61-72
11. Object Management Group “Meta Object Facility (MOF) Core Specification” OMG Available Specification. Version 2.0. formal/06-01-01, http://www.omg.org/docs/formal/06-01-01.pdf, último acceso Julio 2014.
12. Workflow Management Coalition, Workflow Standard – Workflow Process Definition Interface -XML Process Definition Language, Workflow Management Coalition , WfMC-TC-1025, 2002, http://www.wfmc.org/standards/docs/TC-025_10_xpdl_102502.pdf ,
13. BEA, IBM, Microsoft, SAP and Siebel, “Business Process Execution Language for Web Services Version 1.1” , S. Thatte, et al., May 2003, ftp://www6.software.ibm.com/software/developer/library/ws-bpel.pdf, último acceso Abril 2014.
14. Object Management Group, BPMN Documents “BPMNModel UML Documentation”. Draft Specificaction,
15. http://www.bpmn.org/Documents/BPMNMetaModel.zip , último acceso Octubre 2006.
16. Bonita Open Solution, http://www.bonitasoft.org/, ultimo acceso Agosto 2014.