Transformación de Transiciones de Procesos de Desarrollo de Software Basados en SPEM a Transiciones de un Workflow

Fabio Zorzan
Departamento de Computación
Universidad Nacional de Río Cuarto,
Ruta 36 Km 601, 5800, Río Cuarto, Córdoba, Argentina
fzorzan@dc.exa.unrc.edu.ar

y

Daniel Riesco
Departamento de Informática
Universidad Nacional de San Luís,
Ejercito de los Andes 950, 5700, San Luís, Argentina
driesco@unsl.edu.ar

Resumen

El modelado de los procesos de negocio es de vital importancia en el desarrollo de toda industria, en particular, en la industria del software. Una forma de optimizar la producción es mediante la automatización de los procesos de negocio, esto implica definir reglas de transición, recursos involucrados, entre otros. En este trabajo se propone una alternativa para lograr la automatización de los procesos de desarrollo de software especificados con el Software Process Engineering Metamodel(SPEM). Esto se alcanza mediante la utilización de motores workflow que son utilizados para automatizar procesos de negocio. Para lograr la automatización de los procesos de desarrollo de software basados en SPEM por medio de workflows, se define una transformación del metamodelo SPEM al metamodelo workflow de la Workflow Management Coalition(WFMC) por medio del lenguaje Relations que forma parte de Query/Views/Transformations (QVT). De esta forma diferentes procesos de desarrollo de software especificados en SPEM pueden ser transformados en procesos workflow que soportan el estándar de la WFMC. Al utilizar un estándar se tiene la ventaja de lograr la automatización con cualquier motor workflow que siga el estándar. Esta transformación puede ser, también, automatizada debido a que actualmente se están desarrollando herramientas que permiten la ejecución de transformaciones especificadas en QVT. Con esto se logra, tanto la automatización de procesos de desarrollo de software especificados en SPEM a través de su transformación a proceso workflow estándar, sino también, la automatización de la transformación.

Palabras claves: Workflow, SPEM, Automatización, QVT, Proceso de Desarrollo de Software, Ingeniería de Software.

1 INTRODUCCIÓN

Un proceso de negocio es un conjunto de tareas lógicamente relacionadas, ejecutadas para obtener un resultado de negocio. El modelado, análisis y administración de estos procesos, ha cobrado gran importancia ante la necesidad de una industria competitiva, dinámica, que se adapte rápidamente a los cambios, y donde se aprovechen al máximo los recursos disponibles. Toda esta situación ha provocado que uno de los objetivos principales de la industria actual sea la automatización de los procesos de producción.

Los procesos de negocio pueden ser controlados y administrados por un sistema basado en software, proceso de negocio automatizado de esta manera se denomina 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.

El caso particular de la industria del desarrollo de software, no es diferente al del resto de las industrias. Dentro de ella, se encuentran los procesos de negocios tendientes a la construcción o generación de un producto (software) de calidad en un tiempo determinado[3]. Actualmente, ingenieros de software trabajan para optimizar los procesos de desarrollo. Los desarrolladores de las herramientas de ingeniería de software pueden explotar la conexión entre la administración de proceso de desarrollo de software y workflow[2]. El proceso de negocio mas importante dentro de la industria de desarrollo de software es conocido como “metodologías de desarrollo”, encargadas de guiar la producción. Este trabajo aporta a la optimización del proceso de producción de software mediante la automatización de las metodologías de desarrollo. Para esto se trabajo sobre la hipótesis de que el proceso de desarrollo de software es un tipo proceso de negocio particular, y los procesos de negocio pueden ser automatizados en todo o en parte a través de un motor de workflow, la idea es transformar “el proceso” de desarrollo de software en un proceso de un workflow para poder lograr su “automatización” en todo o en parte. El paradigma workflow ofrece interoperabilidad con otros sistemas, ejecución en ambientes distribuidos, facilidades para el monitoreo y manejo de recursos humanos[1].

Para lograr esta automatización se propone una traducción “general” de un proceso de desarrollo de software especificado en SPEM[6] a un especificación de procesos Workflow basado en el estándar definido por la WFMC[4]. Esta traducción se obtiene a través de una transformación definida mediante el lenguaje Relations que forma parte de QVT[5]. La transformación esta definida entre el Metamodelo SPEM y el metamodelo Workflow definido por la WFMC. Por ejemplo, esta transformación aplicada a la especificación en SPEM del SmallRUP[9] da como resultado una especificación workflow que puede ser tomada por cualquier motor workflow que siga el estandar de la WFMC, y de esta manera poder administrar automáticamente por medio de un motor de workflow proyectos de desarrollo de software que utilicen como metodología de desarrollo a SmallRup.

Este trabajo esta organizado de la siguiente manera. En la sección 2 se presenta al SPEM con sus características. La sección 3 presenta la tecnología workflows. En la sección 4 se presenta el lenguaje de transformación parte de QVT. La sección 5 esta descripta la transformación de los metamodelos SPEM y worklflow. Por ultimo la sección 6, las conclusiones

2 SPEM (Software Process Engineering Metamodel)

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. Esos niveles se les denomina comúnmente con las iniciales M0, M1, M2 y M3 y se describen a continuación:

- **El nivel M0 – Las instancias.** El nivel M0 modela el sistema real, y sus elementos son las *instancias* que componen dicho sistema. Un elemento de este nivel es por ejemplo el cliente llamado Pablo Gonzalez.
- **El nivel M1 – El modelo del sistema.** Los elementos del nivel M1 son los *modelos* de los sistemas concretos. Existe una relación muy estrecha entre los niveles M0 y M1, los conceptos del nivel M1 definen las *clasificaciones* de los elementos del nivel M0. En este nivel esta definido el concepto cliente.
- **El nivel M2 – El modelo del modelo (el *metamodelo*)**. Los elementos del nivel M2 son los lenguajes de modelado. El nivel M2 define los elementos que intervienen a la hora de definir un modelo del nivel M1. Aquí también existe una gran relación entre los conceptos de los niveles M1 y M2 donde los elementos del nivel superior definen las *clases* de elementos válidos en un determinado modelo de nivel M1. Conceptos de este nivel son Clase, Atributo, etc.
- **El nivel M3 – El modelo de M2 (el *meta-metamodelo*)**. Finalmente, el nivel M3 define los elementos que constituyen los distintos lenguajes de modelado. La OMG ha definido un lenguaje para describir los elementos del M3 llamado Meta-Object Facility (MOF) [8], un elemento de este nivel en el clasificador, el concepto de clase definido en M2 es un clasificador.

SPEM describe un metamodelo genérico para la descripción de procesos software concretos que está basado en MOF y utiliza UML como notación de modelado. Por tanto, se basa en los principios de orientación a objetos.

El metamodelo SPEM sirve como plantilla para la creación de modelos de procesos concretos, como podrían ser el "Proceso Unificado de Desarrollo de software de Rational" (RUP) o el modelo de evaluación y mejora de procesos de ISO 15504. Por tanto, SPEM es un metamodelo del nivel M2 de MOF, mientras que estos procesos citados se definirían en base a SPEM en el nivel M1.

Este trabajo se centra en el paquete estructura de procesos de SPEM que define los elementos estructurales principales en la construcción de la descripción de un proceso y el paquete maquina de estados que define principalmente las transiciones entre las "actividades" del proceso.

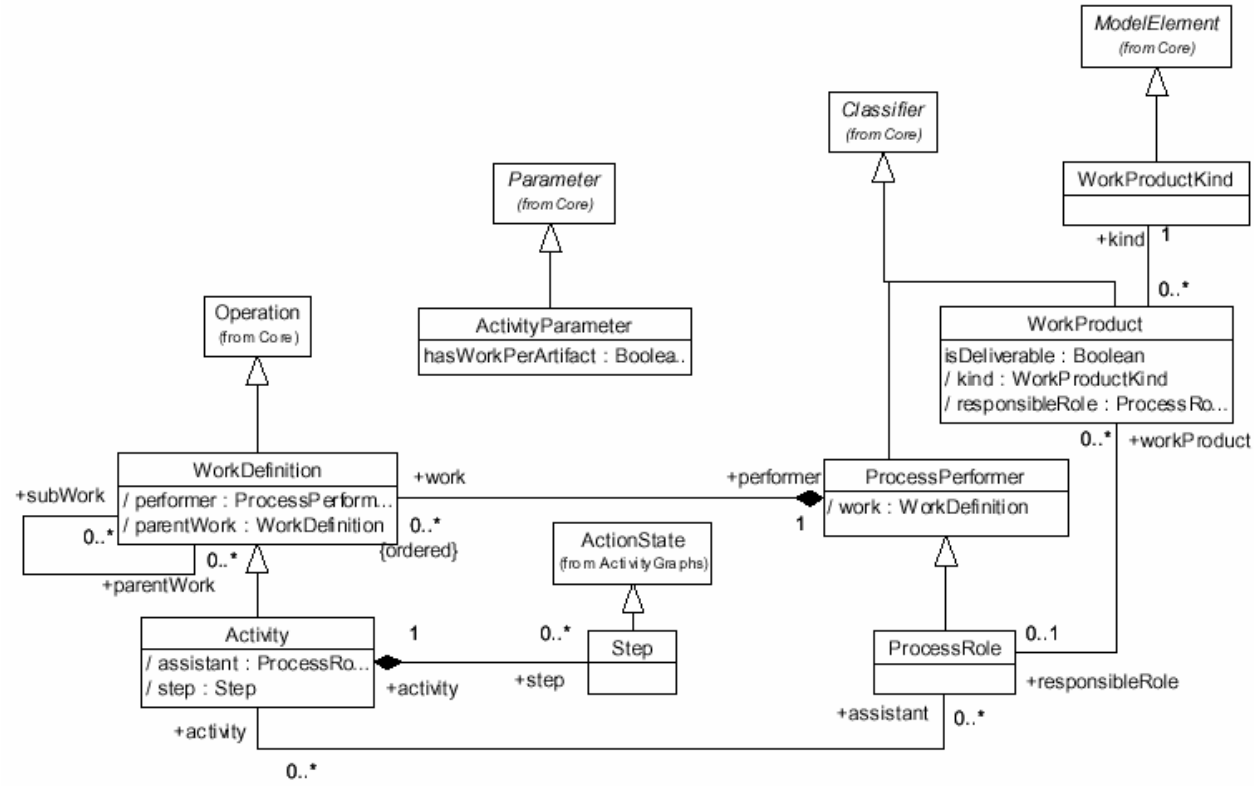

Figura 1: Paquete Estructura del Proceso de SPEM

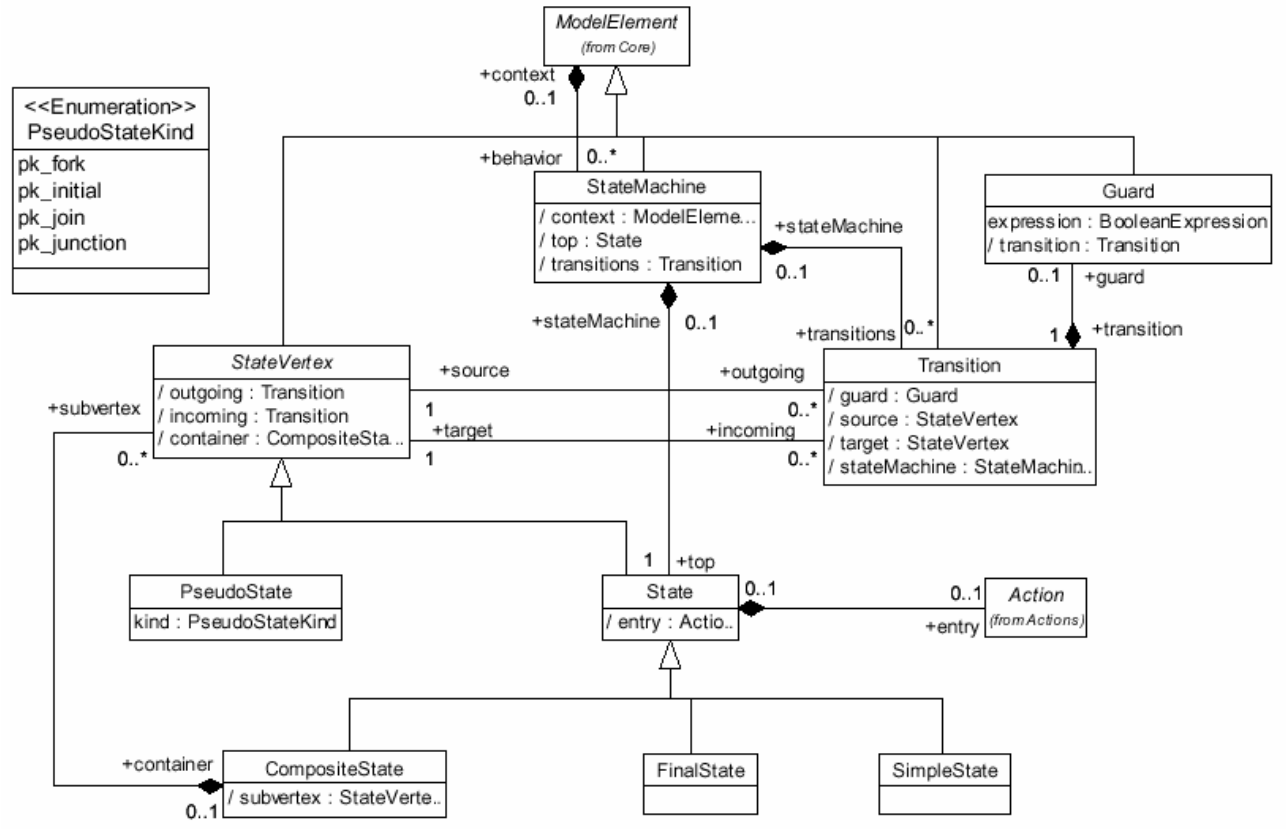

Figura 2: Paquete Maquina de Estados de SPEM

3 WORKFLOW

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 [7].

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.

Para especificar procesos workflow, a nivel general se debe hablar del metamodelo workflow donde se describen las clases que intervienen a la hora de definir un workflow, ahora bien, cuando se tiene que especificar detalladamente un procesos workflow se debe recurrir a un lenguaje basado en XML.

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 (XML Process Definition Language - XPDL)[4]

XPDL permite escribir especificaciones de procesos workflow de manera estandarizada. Esto significa que cualquier definición de proceso que cumpla con todos los requisitos establecidos en la interfaz 1 podrá ser tomada como entrada por cualquier motor workflow que respete el estándar establecido por la WfMC.

El metamodelo workflow está compuesto por entidades y relaciones. Las entidades principales del metamodelo Workflow son las siguientes:

- Workflow Process Activity: Representa una actividad dentro de un proceso.
- Transition Information: Describe las transiciones entre actividades y las condiciones que la habilitan o inhabilitan, durante la ejecución del workflow.
- Workflow Participant Specification: Representa los participantes intervinientes en una actividad. Puede ser un “rol”, un “humano”, u “otro sistema”.
- Workflow Application Declaration: Es representada como una lista de las aplicaciones o herramientas requeridas por el proceso workflow.
- Workflow Relevant Data: Representa a la información que circula dentro de unproceso workflow.

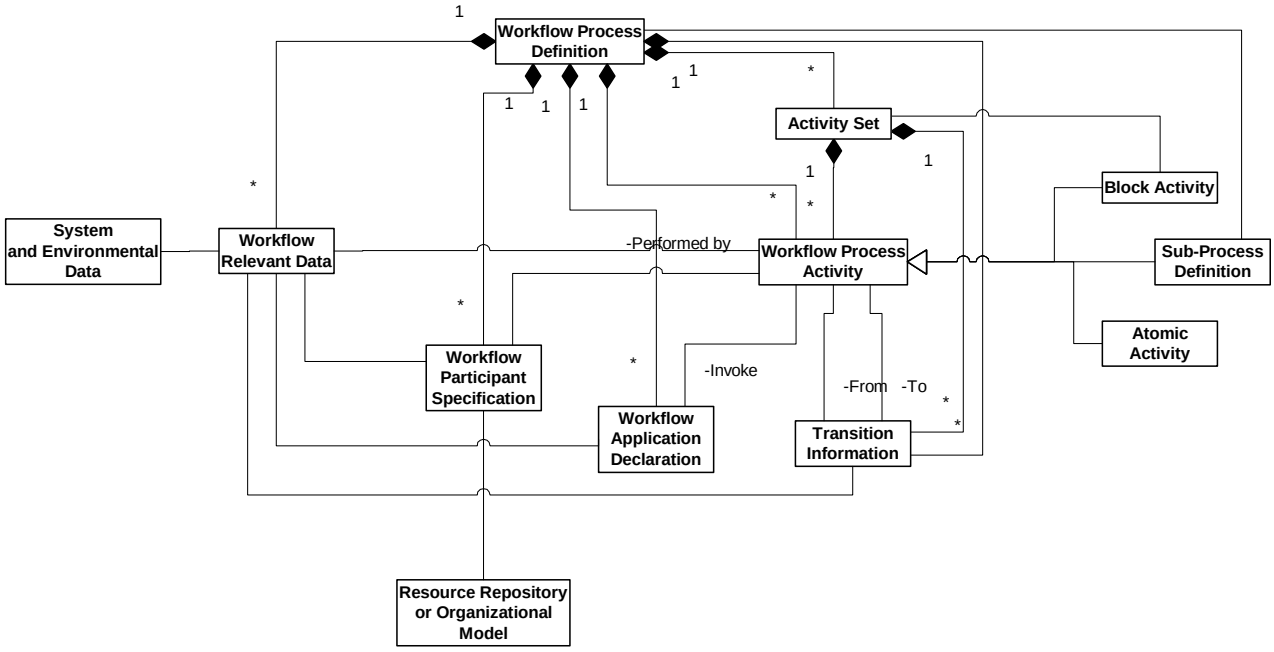

Figura 3: Metamodelo de Definición de Procesos Workflow

4 QVT (Query/Views/Transformations)

El planteamiento QVT 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 solo se presentara el componente de transformaciones de QVT que tiene como objetivo definir transformaciones. Estas transformaciones describen 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. Una característica muy importante de estas transformaciones es que pueden ser bidireccionales (multidimencionales también).

La especificación de QVT[5] que se ha utilizado en este trabajo tiene una naturaleza híbrida declarativa/imperativa, teniendo en cuenta lo declarativo se divide la arquitectura en dos niveles:

- Un Metamodelo y un lenguaje llamado Relations que soporta pattern matching de objetos complejos y un template de creación de objetos. El trace de los elementos de los modelos involucrados en las transformaciones son creados explícitamente.
- Un metamodelo y un lenguaje Core definido utilizando mínimas extensiones al EMOF y OCL.

En la siguiente figura se muestra la relación entre los tres metamodelos QVT:

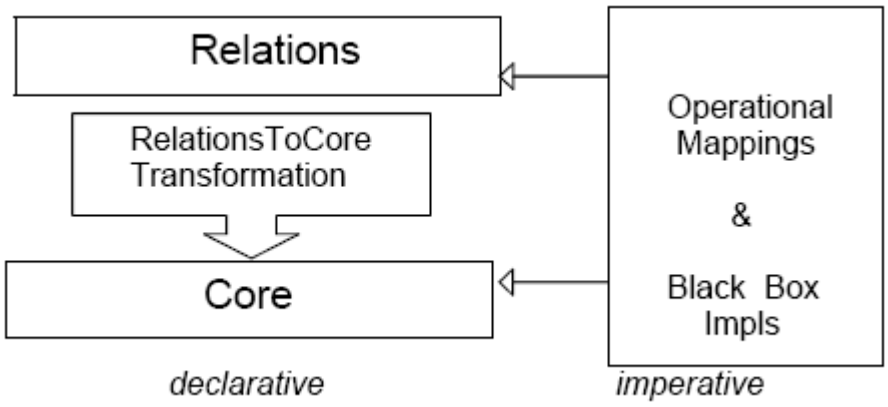

Figura 4 Relación entre los metamodelos QVT

4.1 El lenguaje Relations

El lenguaje relations es una especificación declarativa de las relaciones entre metamodelos MOF. Este lenguaje permite realizar pattern matching de objetos complejos y definir templates de creación de objetos. El trace de los elementos de los modelos involucrados en las transformaciones son creados explícitamente.

La semántica del lenguaje permite la ejecución sobre los siguientes escenarios:

- Transformaciones de solo chequeo para verificar que dos o más modelos estén relacionados de una determinada forma.
- Transformaciones en una sola dirección (modelo fuente a modelo objetivo).
- Transformaciones en bidireccionales. (también es posible transformaciones multidireccionales)
- Capacidad de establecer relaciones entre modelo existentes, desarrollados a mano o con alguna herramienta.
- Actualizaciones incrementales en cualquier dirección, cuando los modelos son cambiados.
- La habilidad de crear o eliminar objetos y valores, además de poder definir que objetos y valores no pueden ser modificados

Una transformación especifica un conjunto de relaciones que deben cumplir los elementos de los modelos involucrados. Una relación especifica una relación entre elementos de los modelos candidatos y consiste de dos o mas dominios, y dos restricciones denominadas cláusula guard (o cláusula when) y cláusula Where. Cada dominio define a un modelo del candidato. Cada dominio tiene patrones, los cuales pueden ser vistos como grafos, donde los nodos son objetos y las propiedades y relaciones son arcos. Un patrón de dominio puede ser considerado un template para los objetos y sus propiedades que deben ser encontrados, modificados o creados en los modelos involucrados para satisfacer la relación entre los modelos.

La cláusula guard de la relación especifica la condición bajo la cual la relación debe ser requerida. La cláusula where de la relación define la condición que deben cumplir los elementos que intervienen en la relación.

Una relación puede ser declarada como de solo chequeo(chech only) o Forzada (enforced). Si un domino se define como chequeado, cuando se ejecute la transformación solo será chequeado para ver si existe un "macheo" valido con otro dominio perteneciente al otro modelo; en cambio si el modelo esta marcado como forzado cuando una relación no se satisface, los elementos del modelo pueden ser creados, borrados o modificados en el modelo objetivo para hacer que la relación se satisfaga.

5 TRANSFORMACIÓN DEL METAMODELO SPEM AL METAMODELO WORKFLOW

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 basada en el estándar de la WFMC para asistir en la gestión de los procesos de desarrollo de software.

El trabajo muestra las relaciones entre las Clases ProcessRole, Transition, StateVertex y Step del metamodelo SPEM a sus correspondientes clases en el metamodelo Workflow. El objetivo de la definición de estas transformaciones de metamodelos es lograr una automatización en la "traducción" de un proceso de desarrollo de software específico a un proceso workflow.

Para poder definir las reglas de la transformación se definió una correspondencia entre algunas clases del metamodelo SPEM y el Metamodelo Workflow

Para la definición de las reglas de transformación de metamodelos se adopto el lenguaje Relations de QVT, QVT permite hacer transformación de diferentes formas (unidireccionales, bidireccionales, solamente chequeo de correspondencias entre modelos, entre otras). En este caso la transformación solo necesita ser definida en la dirección Metamodelo SPEM hacia Metamodelo Workflow. De esta manera los elementos del metamodelo SPEM están marcados como Checkonly y los elemento del metamodelo workflow están marcados como enforced, para que de esta forma la ejecución de la transformación cree los elementos del modelo workflow que se corresponden a los elementos del modelo fuente especificado en SPEM.

5.1 Transformación SPEM-WORKFLOW

A continuación se presentan la relaciones definidas para la transformación SPEM-Workflow, en general para todas las relaciones el identificador de las clases Workflow se construyen concatenando el string "id" mas el nombre del componente SPEM correspondiente. La transformación entre metamodelo SPEM y el Metamodelo Workflow se define de la siguiente manera:

Transformation SpemWorkflow (spem: Spem, workflow: Workflow)

Seguidamente se comenzara con la presentación de las relaciones definidas en la transformación. En primer lugar la relación **ProcessRoleToParticipant** establece el mapping entre las clases ProcessRole y Participant de los metamodelos SPEM y Workflow respectivamente donde se especifica que el tipo de participante del workflow(Participant) es "Role"

relation ProcessRoleToParticipant

```
        {nombre: String;
         ckeckonly domain spem processRole: ProcessRole
        {
                name = nombre;
        }

Enforce domain workflow participant: Participant
        {
                id = "id"+nombre;
                name = nombre;
                participantType = pt : ParticipantType ¨{type = 'Role'};
        }
} // Fin relación ProcessRoleToParticipant
```

TransitionSpemToTransitionWorkflow es la relación entre las transiciones de Spem y las transiciones del metamodelo workflow, las transiciones tiene un vértice de estado(stateVertex) fuente y otro vértice de estado como objetivo, ambos vértices son relacionados con la actividad(WorkflowProcessActivity) desde(from) y con la actividad a (to) de la transición workflow respectivamente. La guarda(Guard) de la transición de SPEM se transforma en la condición de la transición del workflow.

relation transitionSpemToTransitionWorkflow

```
{
nombre: String;
ckeckonly domain spem transitionSpem: Transition
        {
                source = stateVertexSource : StateVertex {};
                target = stateVertexTarget : StateVertex {};
                guard = guarda : Guard{};
                name = nombre;
        }
enforce domain workflow transitionWorkflow: Transition
        {
                 id = "id"+nombre;
                 from = activityFrom : WorkflowProcessActivity{};
                 to = activityTo : WorkflowProcessActivity{};
                 condition = condicion:Condition{};
        }
        Where
        {
                 stateVertexToActivity(stepSource, activityFrom);
                 stateVertexToActivity (stepTarget, activityTo);
                GuardToCondition(guarda,condicion);
        }

}
```

La relación **stateVertexToActivity** define la correspondencia entre la un vértice de SPEM y una actividad Workflow, para esto se consideraron 5 subtipos de vértices, paso (Step) y 4 tipos de pseudoestados(PseudoState), fork, join, inicial y junction. Un paso se mapea a una actividad atómica y los 4 tipo de pseudo estados a actividades de ruteo del metamodelo Workflow.

```
relation stateVertexToActivity
{
    nombre: String;
    checkonly domain spem stateVertex: StateVertex
          {
                Name: nombre;
          }

   enforce domain workflow activity: WorkflowProcessActivity
          {
                Name: nombre;
          }

    Where
       {
                stepToAtomicActivity(stateVertex,activity);
                forkActivityToRouteActivity(stateVertex,activity);
                joinActivityToRouteActivity(stateVertex,activity);
                intialActivityToRouteActivity(stateVertex,activity);
                juntionActivityToRouteActivity(stateVertex,activity);
       }
}
```

La última relación presentada en este trabajo es **stepToAtomicActivity** que especifica la correspondencia entre un paso de SPEM y una actividad atómica del metamodelo workflow

```
relation stepToAtomicActivity
{
nombre: String;
ckeckonly domain spem step: Step
    {
        name = nombre;
    }

Enforce domain workflow atomicActivity: AtomicActivity
     {
        id = "id"+nombre;
        name = nombre;
     }
}
```

6 CONCLUSIONES

Este trabajo tiene como objetivo hacer una contribución a la mejora de los procesos de desarrollo de software, siguiendo la hipótesis que el proceso de desarrollo de software es un proceso de negocio particular, y los procesos de negocio pueden ser automatizados en todo o en parte a través de un motor de workflow. Basado en esa hipótesis se define una transformación de "el proceso" de desarrollo de software a un workflow para poder lograr su "automatización" en todo o en parte. Teniendo en cuenta esto, el proceso de desarrollo de software se puede transformar en una especificación de procesos workflow que sigan el estándar de la WFMC, y de esta forma poder utilizar motores de workflow estándar que asistan a la gestión "automática" de los procesos de desarrollo de software especificados con el estándar de la OMG denominado SPEM. Al lograr definir una transformación genérica, especificada en QVT, de procesos de desarrollo basados en SPEM a un Modelo de procesos Workflow, también se está logrando la automatización de esta transformación, ya que en la actualidad se están desarrollando herramientas que permiten la ejecución de transformaciones especificadas en QVT. Las relaciones presentadas en este trabajo representan una parte central de la transformación entre los metamodelos SPEM y Workflow.

El beneficio de esta automatización también se aprecia teniendo en cuenta el dinamismo de los cambios en los procesos de desarrollo de software, con lo cual cualquier cambio en la especificación del proceso de desarrollo de software puede ser "propagado" a la especificación Workflow de dicho proceso y así adaptar "rápidamente" la especificación del workflow para la automatización del proceso de desarrollo de software.

La validez práctica de la propuesta esta dada en la aplicación de la transformación a una metodología de desarrollo concreta especificada en SPEM, por ejemplo, aplicar la transformación a la especificación en SPEM del SmallRUP da como resultado una especificación workflow que puede ser tomada por cualquier motor workflow que siga el estándar de la WFMC, y así poder administrar automáticamente, por medio de un motor de workflow, proyectos de desarrollo de software que utilicen como metodología de desarrollo a SmallRup.

Esta transformación optimiza la construcción del software debido a que se dispone de un sistema automatizado (motor workflow) que administrará los recursos y organizará a un equipo de ingenieros de software en el transcurso del desarrollo de un proyecto en particular. El proceso de desarrollo adopta todas las ventajas propias de un proceso.

REFERENCIAS

[1] Daniel K.C. Chan, Karl R.P.H. Leung, "Software development as a workflow process", IEEE 1997

[2] Anthony Barnes, Jonathan Gray, "COTS, Workflow, and Software Process Management: an exploration of software engineering tool development", 2000 IEEE

[3] Daniel Romero, Marcelo Uva, "De los procesos de desarrollo a la definición de procesos workflow", Asis 2005.

[4] Workflow Management Coalition, Workflow Standard - Workflow Process Definition Interface – XML Process Definition Language, Workflow Management Coalition , WfMC-TC-1025, 2002.

[5] Revised Submission for MOF 2.0, Query/View/Transformation RFP(ad/2002-04-10), QVT-Merge Group, version 2.0, ad/2005-03-02, 2005

[6] Object Management Group; "Software Process Engineering Metamodel Specification"; An Adopted Specification of the Object Management Group, Inc; Version1.1 formal/05-01-06; January 2005.

[7] Rob Allen, Open Image Systems Inc., United Kingdom Chair, WfMC External Relations Committee; "The Workflow Handbook 2001"; Workflow Management Coalition; October 2001.

[8] Object Management Group "Meta Object Facility (MOF) Core Specification" OMG Available Specification. Version 2.0. formal/06-01-01

[9] Gary Pollice "Using the RUP for small projects: Expanding upon Extreme Programming", A Rational Software White Paper – 17/06/2003.