Lenguajes Notacionales para Modelado de Procesos: un análisis comparativo

Pilar Vasquez [1], Roxana Giandini [2], Patricia Bazán [3],
[1] Facultad de Informática UNLP, [2] LIFIA Facultad de Informática UNLP, [3] LINTI Facultad de Informática UNLP

CONTEXTO

El presente es un trabajo de fin de carrera de Licenciatura en Informática de la Facultad de Informática de la UNLP, de la alumna Pilar Vasquez, dirigida por la Dra. Roxana Giandini y la Lic. Patricia Bazán.

RESUMEN

El Desarrollo de Software Dirigido por Modelos o MDD (*Model Driven Development*) [7] [8] [9], plantea una nueva forma de entender el desarrollo y mantenimiento de sistemas de software con el uso de modelos como principales artefactos del proceso de desarrollo; corriendo o compartiendo el foco de principal artefacto, con el código de lenguajes de programación. En MDD, los modelos son utilizados para dirigir las tareas de comprensión, diseño, construcción, pruebas, despliegue, operación, administración, mantenimiento y modificación de los sistemas [10].De este modo, tales modelos son considerados como entidades primordiales, permitiendo nuevas posibilidades de crear, analizar y manipular sistemas a través de diversos lenguajes y herramientas. El modelo de un sistema provee un medio de comunicación y negociación entre usuarios, analistas y desarrolladores que oculta o minimiza los aspectos relacionados con la tecnología de implementación.

En particular, el modelado de procesos es una rama fundamental en MDD.

Permite organizar y documentar la información del proceso facilitando su comprensión y administración, estas tareas suelen ser engorrosas debido a la gran complejidad de los procesos y subprocesos en una organización.

Al modelar un proceso se busca desarrollar una descripción lo más exacta posible de este así como de las actividades y demás elementos que lo conforman. Es una actividad mediante la cual un proceso es representado o descrito usando lenguajes apropiados que faciliten: la comunicación de la representación, su documentación y la comprensión del proceso. No basta con tener disponible un modelo de proceso sino también es necesario contar con las herramientas adecuadas para definirlo, modificarlo y analizarlo.

Este trabajo tiene como objetivo seleccionar un conjunto de criterios y en base a los mismos establecer una comparación de lenguajes notacionales para modelado de procesos. Este conjunto de criterios se volcarán en un cuadro que muestre esquemáticamente la comparación realizada.

Consideramos en este estudio los lenguajes notacionales que adhieren a estándares de la OMG (*Object Management Group*) [1] como lo son SPEM (*Software Process Engineering Metamodel*) [2] para modelado de procesos y BPMN (*Business Process Modeling Notation*) [3] para el modelado de procesos de negocio.

Consecuentemente con esta comparación se pueden alcanzar conclusiones que faciliten la elección de un lenguaje notacional de modelado de procesos de un tipo particular, ya sea de negocio o de software.

Palabras clave: proceso, modelo, modelo de proceso, MDD.

1. INTRODUCCION

El modelado de procesos es una actividad importante en donde se representa la estructura y el comportamiento deseado de un sistema permitiendo así identificar con facilidad las interrelaciones existentes entre las actividades, analizar cada uno de los elementos existentes y sus relaciones, identificar oportunidades de simplificación y reutilización o sacar a la luz problemas existentes dando oportunidad al inicio de acciones correctivas. Permite realizar un mejor análisis de los procesos existentes, en base al cual se puede realizar la descomposición de procesos de trabajo en actividades discretas así como la identificación de las actividades que aportan un valor añadido y las actividades que sirven de soporte a éstas. También se puede visualizar qué sucede en cada una de las etapas del proceso, cuándo sucede y porqué [4].

De este modo, los modelos son considerados como entidades primordiales a la hora de implementar procesos.

Al modelar un proceso mediante una representación gráfica (diagrama de proceso), se pueden visualizar las interrelaciones existentes entre las distintas actividades que lo conforman, posibles puntos de conexión con otros procesos o subprocesos, los roles o participantes encargados de la ejecución de las actividades, entre otros. Del mismo modo, permite identificar posibles problemas existentes así como oportunidades de mejora. Esto conlleva a que la organización pueda automatizar, integrar, monitorizar y optimizar de forma continua los procesos que administra.

Los lenguajes notacionales nos permiten representar gráficamente un modelo de proceso. Estos lenguajes definen su conjunto de reglas semánticas y sintácticas que establecen los elementos y las relaciones que son parte de los procesos.

2. LINEAS DE INVESTIGACION y DESARROLLO

Se investigarán y analizarán en este trabajo los lenguajes notacionales para modelado de procesos más comúnmente usados, siendo además estándares de la OMG: la Notación BPMN y el Framework SPEM.

BPMN *(Business Process Modeling Notation)* es un estándar para modelar procesos de negocio, originalmente desarrollado por BPMI Notation Working Group, la cual se fusionó con la OMG para trabajar en conjunto sobre temas de BPM (*Business Process Management*). El desarrollo de BPMN es un importante paso para reducir la fragmentación que existe con las innumerables herramientas y notaciones de modelado de procesos [5].

El objetivo primario de BPMN fue proveer una notación que sea legible y entendible para todos los usuarios de negocios, desde los analistas que realizan el diseño inicial de los procesos, hasta los responsables de desarrollar la tecnología que ejecutará estos procesos, hasta los gerentes de negocios, encargadas de administrar y realizar el monitoreo de los procesos. Así, BPMN crea un puente estandarizado para cubrir el hueco provocado por las diferencias entre el diseño de los procesos de negocios y su implementación [6].El segundo objetivo, de igual importancia que el primero, es asegurar que los lenguajes XML diseñados para la ejecución de procesos de negocios, tales como BPEL4WS (*Business Process Execution Language for Web Services*) and BPML (*Business Process Modeling Language*), puedan ser expresados gráficamente con una notación común (una notación orientada al negocio).

La meta para los desarrolladores de BPMN es que la notación sea simple y adoptable para los analistas de negocio. También, hay un requerimiento potencialmente conflictivo, es el que BPMN provea el poder de describir procesos de negocio complejos y mapearlos con lenguajes de ejecución BPM. Para ayudar a entender como BPMN puede manejar ambos requerimientos, la lista de elementos gráficos BPMN se presenta en dos grupos [3]. Un primer grupo con los elementos básicos. Estos son los que definen el "look-and-feel" básico de BPMN. La mayoría de los procesos de negocio pueden ser modelados adecuadamente con estos elementos. Estos se pueden clasificar en cuatro categorías fundamentales:

- Flow objects: Eventos, Actividades y Gateway.
- Connecting Objects: Flujo de secuencia, Flujo de Mensaje y Asociación.
- Calles: Pool y Lane.
- Artefactos: Datos, Anotaciones y Grupo.

Un segundo grupo de elementos está compuesto por los elementos básicos sumándose variaciones de los mismos, lo cual ayudará a soportar requerimientos de una notación poderosa para manejar situaciones de modelado más avanzadas. Para los eventos existe una serie de desencadenadores para controlar el comportamiento del proceso, como: mensaje, error, temporizador, condicional. Los Gateways mediante sus variaciones pueden reflejar cualquier tipo de control de flujo de secuencia: OR exclusivo, OR inclusivo, paralelo, complejo.

Asimismo, los elementos gráficos de la notación soportan atributos no gráficos que proporcionan la información necesaria restante para mapear con un lenguaje de ejecución u otro propósito de modelado de negocio.

SPEM (*Software Process Engineering Metamodel*) es un estándar de la OMG, y es un framework que permite instanciar procesos. Es un marco de trabajo conceptual que provee los conceptos necesarios para modelar, documentar, presentar, publicar, gestionar, intercambiar y realizar métodos y procesos software. Por ello, está destinado a ingenieros de procesos, jefes de proyectos, gestores de proyectos y programas; que son responsables de mantener e implementar procesos para sus organizaciones o para proyectos concretos. El alcance de SPEM está limitado de manera premeditada, a la cantidad mínima de elementos necesarios para definir cualquier proceso de desarrollo de software o sistemas, sin agregar características específicas para dominios o disciplinas particulares. El objetivo es satisfacer un gran rango de métodos y procesos de desarrollo de diferentes estilos, contextos culturales, niveles de formalismo, modelos de ciclo de vida, y comunidades. SPEM 2.0 no proporciona sus propios conceptos de modelado de comportamiento. Pero define la capacidad para que el implementador elija el modelado de comportamiento genérico que más concuerde con sus necesidades y provee estructuras específicas para procesar tales modelos de comportamiento genéricos. SPEM 2.0 se concentra en proveer las estructuras de información adicionales que el desarrollador necesita para procesos modelados con actividades UML 2.0 o BPMN/BPDM para describir un verdadero proceso de desarrollo [2].

La idea central de SPEM 2.0 para representar procesos, como muestra la figura 1, está basada en tres elementos básicos: rol (process rol), producto de trabajo (work product) y actividades (activities), es que un proceso de desarrollo de software es una colaboración entre entidades abstractas activas llamadas roles que realizan operaciones llamadas actividades en entidades concretas y tangibles llamadas productos de trabajo.

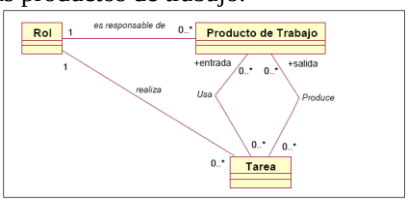

Figura 1 Idea básica de proceso en SPEM 2.0.

Este estándar se describe de dos maneras:

- Como un Metamodelo. Éste define todas las estructuras y reglas de estructuración para representar contenidos de métodos y procesos. Es

completo en sí mismo, es decir, no necesita de otro metamodelo.

- Como Perfil de UML 2, es decir, como un conjunto de estereotipos UML 2 que permite representar métodos y procesos usando UML 2. En este caso, la definición sólo abarca la presentación, mientras que las definiciones semánticas y restricciones están en el metamodelo anterior.

El metamodelo de SPEM 2.0 está organizado en 7 paquetes. Cada paquete es una unidad lógica que extiende los paquetes de los que depende, proveyendo estructuras y capacidades adicionales.

Estos paquetes proporcionan las siguientes capacidades:

- Core: Contiene todas las clases y abstracciones que constituyen la base para el resto de los paquetes del metamodelo.
- Process Structure: Define la estructura de desglose de trabajo estática mediante anidamiento de actividades y dependencias de precedencia entre ellas.
- Process Behavior: Permite extender las estructuras del paquete anterior con modelos de comportamiento externo: diagramas de actividad de UML 2 (comportamiento de proceso), máquina de estados (ciclos de vida de un producto de trabajo), etc.
- Managed Content: en el proceso de desarrollo de software muchos casos no son representados por modelos, son documentados y manejados con descripciones del lenguaje natural. Este paquete introduce conceptos para manejar el contenido textual de estas descripciones.
- Method Content: Contiene los conceptos de SPEM 2.0 relacionados con los usuarios y la organización. Estos conceptos son necesarios para construir una base de conocimiento sobre desarrollo que pueda ser utilizada independientemente del proceso o proyecto específico. Los elementos de método permiten describir, con el detalle necesario, cómo se alcanzan los objetivos del proceso. Los procesos reutilizan y relacionan entre sí los elementos de método de distintas maneras para diferentes tipos de proyectos.
- Process WithMethods: Contiene los elementos necesarios para integrar los conceptos del paquete Process Structure con los conceptos y elementos del paquete Content Method. Al asociar elementos de método a partes específicas de procesos, se crean nuevas clases (tarea en uso, rol en uso, etc.) que heredan de los elementos de método pero pueden tener cambios individualizados.
- Method Plug-In: Introduce los conceptos para diseñar, gestionar y mantener repositorios y librerías de Methods Content y Procesos, que sean mantenibles a gran escala, reutilizables y configurables. Con estos conceptos los ingenieros de procesos pueden definir una o varias Configuraciones de Método, de cada proceso. Esto permite tener vistas diferentes de un mismo proceso adaptadas a distintas audiencias o tipos de usuarios.

Utilizando unos u otros paquetes, un ingeniero de procesos puede disponer de diferentes capacidades, conjuntos de conceptos y niveles de formalismo para expresar sus procesos.

SPEM 2.0 se define como un perfil UML. Con la creación del perfil los creadores pretenden que se puedan seguir utilizando para SPEM la gran cantidad de herramientas que han sido creadas para trabajar con UML. De esta manera SPEM, el perfil de SPEM y UML se relacionan dentro de los niveles de abstracción de la OMG.

3. RESULTADOS OBTENIDOS/ESPERADOS

Definiremos los criterios que creemos necesarios, para comparar los lenguajes notacionales seleccionados. Luego aplicaremos los criterios definidos, a cada lenguaje, para realizar la comparación. A continuación se establecen los criterios de comparación que se considerarán para los lenguajes notacionales seleccionados y la fundamentación de su elección.

a. **Formalismo del lenguaje**: El formalismo de un lenguaje, nos provee de una sintaxis y semántica mediante la cual podemos analizar, desarrollar y comunicar nuestros proyectos, nuestros procesos; hablando en un mismo lenguaje (un conjunto de reglas de representación y entendimiento) entre los distintos actores: desarrolladores, operadores, clientes, etc. El lenguaje puede ser: Formal tiene sintaxis y semántica formal, Semi-Formal tiene notación formal pero no tiene semántica formal, Informal sin sintaxis y semántica formal (lenguaje natural). Cuanto más formal es el lenguaje, se reducen los errores de comprensión y ambigüedad del mismo. Los lenguajes formales de modelado tienen poco uso en la industria, debido a la complejidad de sus formalismos matemáticos. Contrariamente los lenguajes gráficos de modelado tienen mucho éxito basado principalmente en el uso de iconos gráficos que transmiten un significado intuitivo. Sin embargo, la falta de precisión en la definición de su semántica puede originar malas interpretaciones de los modelos, inconsistencia entre los diferentes sub-modelos del sistema y discusiones acerca del significado del lenguaje. Por esto creemos que es importante analizar esta característica, así como su capacidad para transmitir significado a través de sus iconos gráficos.
b. **Objetivos de modelado:** Conociendo los principios para los cuales fueron creados, hacia qué tipos de procesos están dirigidos, público al que están dirigidos; podemos encastrar los lenguajes en los diferentes perfiles para los cuales pueden ser usados.
c. **Capacidad para representar las diferentes vistas del proceso:**

i. Funcional: Capacidad para representar que funciones realiza el proceso.
ii. Estructural: cuales son los elementos que componen el proceso. El lenguaje nos debe dar la posibilidad de representar todos los objetos del proceso, para poder modelarlo en completitud y contar con diferentes vistas del mismo.
iii. Comportamiento: como opera el proceso.

Es esencial que el lenguaje permita representar las diferentes vistas del proceso. Para poder tener una vista general del mismo.

d. **Capacidad de abstracción y modularidad:** se evalúa en este criterio la independencia de la tecnología utilizada para su ejecución y también la facilidad para poder llevar a cabo una división modular que ayude a manejar la complejidad de los procesos
e. **Capacidad de expresividad de la notación:** Posibilidad, a través de sus elementos, de poder modelar el proceso con la mayor claridad y expresión posible. De esta manera los diferentes actores pueden desarrollar el modelo, observarlo y aportar sus diferentes perspectivas.
f. **Facilidad de aplicación de la notación:** Grado de dificultad para poder aplicar los elementos y sus significados en el modelado de procesos. Esta característica está ligada con el éxito del lenguaje, ya que el mismo debería facilitar la tarea de los desarrolladores del modelo y no por el contrario entorpecerla. Cuanto más natural resulte aplicar la notación, los resultados obtenidos pueden reflejar con más claridad la realidad que se necesita modelar.
g. **Facilidad de comprensión de la notación:** Nivel de dificultad para leer y comprender los modelos de procesos para un actor que no estuvo dentro del desarrollo del mismo. En el modelado de un proceso no sólo actúan personas idóneas con la tecnología informática también participan actores que forman parte del proceso desde la parte intelectual u operativa. Por esto es muy útil que el lenguaje permita a estos actores comprender el modelo sin mayores dificultades.
h. **Posibilidad de automatización del proceso:** La automatización de los procesos reduce errores, asegurando que los mismos se comporten siempre de la misma manera y dando elementos que permitan visualizar el estado de los mismos. La administración de los procesos permite asegurar que los mismos se ejecuten eficientemente, y la obtención de información que luego puede ser usada para mejorarlos. A través de la información que se obtiene de la ejecución diaria de los procesos, se puede identificar posibles ineficiencias en los mismos, y actuar sobre estas para optimizarlos [11].

En la tabla 1 se muestra un cuadro comparativo donde se aplican los criterios antes mencionados.

Desde el punto de vista de la definición de procesos de negocio, podemos observar que el desarrollo de procesos software puede ser considerado como un tipo particular de los mismos, ya que la industria del desarrollo de software no es diferente al resto de las industrias. Desde este punto de vista podemos concluir que BPMN cubre el dominio de tipos de procesos de SPEM.

Por otro lado SPEM brinda un framework para definir procesos software, lo cual permite crear metamodelos derivados de él heredando toda su semántica. Es decir permite metamodelar procesos más específicos, según el proyecto a encarar. BPMN es un lenguaje notacional y no nos da esta posibilidad. Con esto podemos deducir que SPEM se encuentra en un nivel de abstracción mayor que BPMN.

Con respecto a la notación, SPEM no tiene una notación específica, como ya mencionamos, pero se puede definir como perfil de UML 2 para obtener una representación gráfica del modelado de los procesos. El empleo del perfil UML tiene la ventaja de que no es necesario desarrollar herramientas CASE especiales para crear y mantener los métodos y procesos, bastando con usar una herramienta genérica de modelado UML. Pero esta opción tiene una importante desventaja: las reglas específicas de estructuración de SPEM, que sí están en el metamodelo pero no en el perfil, no pueden ser manejadas con herramientas genéricas basadas en UML, salvo que se empleen restricciones OCL, lo que supone aumentar significativamente el nivel de dificultad de UML. Por otro lado BPMN tiene una notación clara, precisa y fácil de aplicar.

A nivel de expresión de modelado, las diferentes vistas y objetos del proceso que se pueden representar, tanto BPMN y SPEM cubren todas las características. Variando esto en los diferentes dominios (dominios de proceso) objetivos.

Obteniendo una transformación completa desde SPEM a BPMN podríamos obtener una notación clara, legible y fácil de aplicar para el desarrollo de procesos software manteniendo en nuestro modelo toda la formalidad que nos proporciona SPEM. Así como también la posibilidad de automatizar estos procesos.

Tabla 1. Comparación de Lenguajes

CRITERIOS		BPMN	SPEM
Formalismo		Semi-formal	Formal
Objetivos de modelado		Proveer notación legible y entendible para usuarios de negocios, estandarizar notación de procesos de negocio.	Proporcionar un marco formal para la definición de procesos de desarrollo de sistemas y de software.
Representa las diferentes vistas del proceso:	**Funcional**	Si	Si
	Estructural	Si	Si
	Comportamiento	Si	No
Abstracción y Modularidad		Si	Si
Capacidad de Expresión		Si	No posee notación propia.
Facilidad de aplicación		Si	No posee notación propia.
Facilidad de comprensión		Si	No posee notación propia.
Posibilidad de automatización		Si	No

4. FORMACION DE RECURSOS HUMANOS

El desarrollo de software dirigido por modelos (MDD) es un nuevo paradigma que enfatiza a los modelos como conductores primarios del proceso, automatizando las transformaciones entre modelos hasta llegar a la generación de código. En particular, el modelado de procesos es una rama fundamental en esta nueva disciplina que se plantea como una línea de formación de recursos humanos necesarios para el área de Desarrollo de Software. El presente trabajo se enmarca en una línea de investigación en MDD donde tanto alumnos de grado como de posgrado pueden formarse y desarrollar su trabajo de tesis interactuando con docentes investigadores formados que están incorporando esta metodología como línea de acción.

5. BIBLIOGRAFIA

[1] *The Object Management Group (OMG)*

(http://www.omg.org)

[2] *Software & Systems Process Engineering Meta-Model Specification* - Version 2.0 with change bars

(http://www.omg.org/spec/SPEM/2.0/PDF)

[3] *Business Process Modeling Notation (BPMN) - Version 1.2*

(www.omg.org/spec/BPMN/1.2/PDF)

[4] *A Novel Approach for Modeling Business Process Definitions* –

Jean-Jacques Dubray, Eigner 200 Fifth Ave, Waltham, MA 02451

(http://www.google.com.ar/search?sourceid=navclient&hl=es&ie=UTF-8&rlz=1T4ADBF_esAR302AR303&q=A+Novel+Approach+for+Modeling+Business+Process+Definitions)

[5] *Introduction to BPMN- Stephen A. White, IBM Corporation*

(www.**bpmn**.org/Documents/**Introduction**%20to%20**BPMN**.pdf)

[6] *Business Process Modeling Notation (BPMN)- PNMSoft*

(http://www.pnmsoft.com)

[7] Stahl, T. and Völter, M. Model-Driven Software Development. John Wiley & Sons, Ltd. (2006)

[8] Claudia Pons, Roxana Giandini, Gabriela Pérez. "Desarrollo de Software Dirigido por Modelos. Conceptos teóricos y su aplicación práctica". Editorial: EDULP and McGraw-Hill Education. (2010).

[9] Kleppe, Anneke G. and Warmer Jos, and Bast, Wim. MDA Explained: The Model Driven Architecture: Practice and Promise. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA. (2003)

[10] Herramientas de soporte al proceso de desarrollo dirigido por modelos y su implementación con DSL Tools - L. Cuaderno, E. Di Lorenzo, A. Gaig, D. García, R. Giandini, L. Nahuel, L. Ocaranza, M. Pinasco, C. Pons, F. Salvatierra –

Universidad Tecnológica Nacional, Facultad Regional La Plata. La Plata, Buenos Aires, Argentina - proyectopampa@frlp.utn.edu.ar

(http://www.jidis.frc.utn.edu.ar/papers/45308ce78aafdf2092719a59d01c.pdf)

[11] *Business Process Management(BPM)*

(http://www.bpmi.org/)