X CONGRESO INTERNACIONAL DE INGENIERIA DE PROYECTOS
VALENCIA, 13-15 Septiembre, 2006

VERIFICACIÓN DE LOS MODELOS EN LA INGENIERÍA DE SISTEMAS

A. Monzón Díaz[(p)], J.L. Fernández Sánchez

Abstract

One of the most important challenges of engineering practice is the increasing size and complexity of the systems developed. Modern systems include both mechanical and electronic components, as well as software, and their integration involves a higher complexity in the definition and verification phases.

In order to successfully afford this challenge, Systems Engineering is focusing to models instead of the traditional documental approach, for describing systems.

Models are partial representations of the systems, using concrete syntax and semantic clearly defined to describe rigorously certain aspects of the systems, in contrast to natural language, traditionally used to describe systems.

An intrinsic problem of natural language is its risk of subjective interpretation, and for this reason the descriptions of systems in natural language are essentially inconsistent, incomplete and difficult to understand. Nevertheless, even modelling languages are subject to inconsistencies. As model inconsistencies have a strong impact on the system verification activity, because inconsistent models derive in faulty or even dangerous systems, it is required to establish mechanisms to assure the correctness and consistency of models.

PPOOA architectural style [4] is a high-level approach to represent real-time systems at early stages of development. This modelling paradigm suits specially well in embedded systems produced in diverse industrial fields such as telecommunications, automotive, robotics and aerospace.

The objective of this paper is to identify the typical inconsistencies associated with models developed using PPOOA style and the strategies to solve them in order to improve the consistency of models created using this style.

Keywords: Systems Engineering, Architectural Models, Model Verification

Resumen

Uno de los mayores retos a los que se enfrentan los proyectos de ingeniería actuales es el creciente tamaño y complejidad de los sistemas. Los sistemas integran componentes mecánicos, electrónicos y cada vez más software, y esta integración supone también una mayor complejidad en la definición y en la verificación de los mismos.

La Ingeniería de Sistemas se está orientando a un enfoque basado en modelos frente al tradicional enfoque documental, a la hora de describir los sistemas.

Los modelos son representaciones parciales de los sistemas que utilizan una notación y una semántica claramente definidas para describir de forma rigurosa ciertos aspectos de los mismos, en contraposición al lenguaje natural empleado para las descripciones tradicionales de los sistemas.

Un problema inherente al lenguaje natural es que está sujeto a la interpretación de quien lo utiliza, y esto supone que las descripciones de sistemas realizadas en lenguaje natural son esencialmente inconsistentes, incompletas y difíciles de contrastar. Sin embargo, ni siquiera los lenguajes de modelado están exentos de inconsistencias. Las inconsistencias en los modelos tienen un grave impacto en la tarea de verificación de sistemas, dado que un diseño defectuoso puede provocar problemas de operación o de seguridad del sistema en operación y las consecuencias de esta concepción errónea podrían llegar a ser catastróficas.

El estilo arquitectónico PPOOA [4] es una aproximación de alto nivel para la representación de modelos, en las primeras etapas del desarrollo de sistemas con requisitos de tiempo real, de especial relevancia en sistemas embarcados de aplicación en industrias tan diversas como las telecomunicaciones, la automoción, la robótica o el sector aerospacial.

Con objeto de mejorar la consistencia de modelos basados en PPOOA, se mostrarán en el presente artículo las inconsistencias típicas relacionadas con este estilo arquitectónico y se plantearán las estrategias más adecuadas para su resolución.

Palabras clave: Ingeniería de Sistemas, Modelos Arquitectónicos, Verificación de Modelos

1. Introducción

Uno de los mayores retos a los que se enfrentan los proyectos de ingeniería actuales es el creciente tamaño y complejidad de los sistemas. Los sistemas integran componentes mecánicos, electrónicos y cada vez más software, y esta integración supone también una mayor complejidad en la definición y en la verificación de los mismos.

Para ser capaz de afrontar con éxito este reto se requiere una aproximación multidisciplinar dada por la Ingeniería de Sistemas. Ésta se está orientando a un enfoque basado en modelos frente al tradicional enfoque documental, a la hora de describir los sistemas.

Los modelos son representaciones parciales de los sistemas que utilizan una notación y una semántica claramente definidas para describir de forma rigurosa ciertos aspectos de los mismos, en contraposición al lenguaje natural empleado para las descripciones tradicionales de los sistemas.

Actualmente los modelos son parte esencial de un paradigma de desarrollo conocido como “Model Driven Engineering”, o ingeniería dirigida por modelos, y que es aplicado a sistemas complejos e intensivos en software [11]. En el MDE se parte de los requisitos del sistema y se desarrollan modelos de arquitectura y diseño desde niveles de abstracción más altos a niveles más detallados, que incluyen la definición de la plataforma de ejecución. En el

paradigma MDE el esfuerzo se pone en el desarrollo de los modelos y su consistencia, generando en muchos casos el código de forma automática.

El estilo arquitectónico PPOOA ("Processes Pipelines in Object Oriented Architectures") es una aproximación de arquitectura para la representación de sistemas intensivos en software, en las primeras etapas de su desarrollo, considerando los requisitos de tiempo real de especial relevancia en sistemas embarcados de aplicación en industrias tan diversas como las telecomunicaciones, la automoción, la robótica o el sector aerospacial.

Este artículo está elaborado en base a los resultados de los trabajos de desarrollo de una tesis doctoral sobre el tratamiento de las inconsistencias en modelos de sistemas basados en el estilo arquitectónico PPOOA [8][9].

Uno de los objetivos del análisis descrito en el presente artículo es el de reducir el coste de corrección de los errores de los sistemas. Cuanto antes pueda realizarse la tarea de **verificación** dentro del ciclo de vida de desarrollo de un sistema, antes podrán identificarse posibles errores, y menor será el impacto de éstos sobre el producto final. En las primeras etapas del diseño de un sistema, la cantidad de información disponible en el modelo es reducida y el nivel de detalle es bajo, con lo que no pueden plantearse mecanismos de verificación detallados. Sin embargo, con la ayuda de los aspectos estructurales y de comportamiento proporcionados por el estilo PPOOA, es posible plantear asuntos tan relevantes como si se van a producir bloqueos entre subsistemas, por el uso de recursos compartidos, que impidan al sistema cumplir con sus requisitos temporales.

2. Técnicas de Modelado

Existen infinidad de notaciones y técnicas de modelado de sistemas, pero la tendencia actual es la convergencia hacia unas técnicas de modelado unificadas. En particular, UML (Unified Modeling Language) [10] y SysML (Systems Modeling Language) [12] representan dos notaciones de modelado unificadas, la primera más implantada en la industria del software y la segunda específicamente concebida para la descripción de sistemas complejos.

Ambas notaciones comparten una serie de vistas para representar ciertos aspectos de los sistemas y un meta-lenguaje común. SysML supone una especialización basada en UML para la representación de aspectos particulares de los sistemas no tenidos en cuenta en la notación de UML.

Por su parte, PPOOA es un estilo arquitectónico que usa parte de la notación UML estándar y que extiende el metamodelo de esta última en ciertos aspectos relacionados con la representación de la concurrencia explícita dentro de sistemas con restricciones temporales.

En adelante centraremos las explicaciones en SysML y especialmente en PPOOA, dando por sentado que UML es la base sobre la que se asientan ambos perfiles de modelado.

2.1 SysML: la Notación de Modelado en la Ingeniería de Sistemas

SysML es una extensión de UML propuesta por la industria como estándar de modelado de sistemas en general. SysML, al contrario de UML, no sólo contempla los sistemas software sino otros sistemas más generales donde puede haber intercambios de masa, energía e información entre sus componentes. Estos sistemas podrían ser el motor de un automóvil o un sistema de control de una instalación industrial [6].

SysML soporta tres tipos de vistas o diagramas de un sistema: diagramas estructurales, diagramas de requisitos y diagramas de comportamiento. Cada uno de estos tipos de diagramas se subdivide en varios subtipos o modelos. Por ejemplo para los diagramas estructurales, SysML considera que éstos pueden ser diagramas de bloques, diagramas de

bloques internos (diagramas paramétricos) y diagramas de paquetes. Los diagramas de bloque son una modificación de los diagramas de clases de UML y los diagramas de requisitos y diagramas paramétricos son añadidos propios de SysML. Los diagramas paramétricos facilitan la descripción de modelos con expresiones matemáticas.

De la misma forma que UML, SysML proporciona los conceptos de puertos y conectores. Los puertos son puntos de interacción por donde un bloque suministra o requiere servicios de su entorno. Un conector enlaza dos puertos para soportar su interconexión. Ambos conceptos son importantes en las técnicas de verificación de modelos que se proponen en este artículo.

No es la intención de este artículo, profundizar en SysML. Se da al lector interesado la referencia al estándar [12], recientemente publicado para tal propósito.

2.2. PPOOA: la Notación y Metodología para la Arquitectura de Sistemas

PPOOA es una extensión de UML para el modelado de sistemas intensivos en software y con requisitos temporales. PPOOA no sólo suministra una notación extendida de UML, también suministra una metodología de arquitectura de sistemas y una herramienta de diseño de este tipo de sistemas [1].

PPOOA se basa en los paradigmas del desarrollo basado en componentes, y en el modelado de concurrencia explícito. Este último permite la evaluación del comportamiento de los sistemas de tiempo real en sus primeras etapas de desarrollo.

En PPOOA se suministra un vocabulario de elementos constructivos de los sistemas, formado por componentes y mecanismos de coordinación. Los componentes son lugares de cómputo, y los mecanismos de coordinación son una especialización del concepto de conector que se ha descrito anteriormente.

En PPOOA se describe la arquitectura de un sistema con dos tipos de diagramas: diagramas de componentes y diagramas de comportamiento o CFAs (“Causal Flow of Activities”). Estos últimos representan cada una de las respuestas del sistema a un evento externo, interno o temporizado.

El diagrama de componentes de una arquitectura en PPOOA tiene al menos dos niveles de jerarquía. En el primer nivel o nivel de sistema, se representan los principales subsistemas que componen el sistema a desarrollar. En el segundo nivel, o nivel de subsistema, se descompone cada subsistema en sus principales componentes y se detallan las relaciones de composición y dependencia entre componentes, y entre éstos y los mecanismos de coordinación.

En PPOOA se definen explícitamente las reglas de composición y dependencia que aplican a sus elementos constructivos utilizados en los modelos de un sistema, ya sean estos componentes o mecanismos de coordinación [4].

3. Definición de Consistencia entre Modelos

De acuerdo con el Diccionario de la Lengua Española, "consistencia" es la "coherencia entre las partículas de una masa o los elementos de un conjunto". Esta definición involucra la noción de "coherencia", según la cual dos o más elementos son "coherentes" siempre que exista cualquier tipo de relación entre ellos.

Hay muchas formas distintas de representar un sistema, y cada una de ellas debe ser considerada como un punto de vista del mismo. Dentro de la propia definición de UML está inmersa la metáfora del “punto de vista” de un sistema, que puede ser establecida como sigue: "Cualquier sistema complejo se puede entender mejor a través de un número

pequeño de vistas independientes del mismo. No es suficiente con una única vista". UML proporciona distintos modelos, junto con sus representaciones gráficas, para describir mejor diferentes aspectos de un sistema. UML puede ser considerado como un conjunto de modelos bien conocidos, cada uno de los cuales representa una vista particular del sistema a modelar.

La consistencia de los modelos UML podría ser definida como la coherencia entre sus modelos constitutivos. Usando de forma recurrente la definición general de coherencia, dos o más modelos UML son consistentes si sus elementos constitutivos son a su vez coherentes entre ellos. Los elementos constitutivos de un modelo UML se denominan "elementos de modelado". Como la definición general de coherencia es muy vaga (sólo se requiere que exista algún tipo de relación entre los elementos), se hace necesario proporcionar una definición más concreta acorde con las particularidades de la disciplina de la ingeniería de modelos.

Desde un punto de vista práctico dos modelos se podrán considerar consistentes si no existen ni redundancias ni contradicciones en las definiciones de los elementos y relaciones que los forman. Por el contrario, serán inconsistentes en cuanto aparezcan contradicciones o repeticiones no idénticas en las descripciones de los elementos y relaciones de modelado.

El concepto de consistencia lleva emparejada una tarea de ingeniería: la evaluación o el chequeo de la consistencia entre modelos. Esta tarea debe ser realizada una vez que los modelos han sido elaborados y requiere siempre de un juicio de valor sobre los aspectos de consistencia contemplados durante la evaluación.

Las evaluaciones de consistencia de modelos se realizan en base a una serie de reglas de consistencia preestablecidas, con unos criterios claros de aceptación o rechazo. La descripción de estas reglas de consistencia es parte del trabajo de preparación de las evaluaciones de la consistencia entre modelos.

4. Identificación de Inconsistencias entre Modelos

De acuerdo con lo enunciado en el apartado anterior, una primera tarea que se debe realizar es identificar los tipos de inconsistencias que se van a buscar entre diferentes modelos. Para ello es preciso identificar primero los modelos que se van a comparar, y después plantear las reglas que definirán las inconsistencias.

En el presente artículo nos centraremos en las inconsistencias que pueden aparecer entre los modelos estructurales del estilo arquitectónico PPOOA. Buscaremos por tanto las que llamaremos en adelante inconsistencias estructurales del estilo.

Cabe decir que en la vista de comportamiento también pueden aparecer inconsistencias, que deben ser tratadas de forma específica. Sin embargo, estas inconsistencias dinámicas quedan fuera del ámbito del presente artículo.

4.1 Inconsistencias Estructurales Intrínsecas de PPOOA

Para ilustrar la tarea de identificación de las inconsistencias estructurales de PPOOA vamos primero a mostrar algunos ejemplos de inconsistencias que se derivan de las restricciones impuestas por el propio metamodelo de PPOOA. Estas restricciones se traducen en una serie de reglas constructivas, que implican una serie de posibles inconsistencias que denominaremos intrínsecas del estilo arquitectónico.

Posteriormente entraremos en la identificación de inconsistencias estructurales de carácter explícito, es decir, aquellas no directamente derivadas de las restricciones del metamodelo del estilo. El primer tipo de inconsistencias tiene un carácter apriorístico, porque si un modelo de un sistema las incumple, está incumpliendo las propias reglas definitorias del

estilo, con lo que deben detectarse a priori (antes de que se termine de construir el modelo). El segundo tipo de inconsistencias puede dejarse para la fase de análisis de un modelo ya construido, con lo que su tratamiento ocurrirá a posteriori.

Normalmente las primeras son detectadas por la herramienta que soporte la edición de modelos basados en PPOOA, ya que constituyen incumplimientos de la propia sintaxis del estilo. La forma de materializar esta detección será mediante mensajes de error de la herramienta, indicándonos que estamos intentando violar alguna regla del estilo, e impidiéndonos ésta cometer tales incorrecciones [7].

La forma de materializar las segundas será mediante un módulo adicional a la propia herramienta de creación de modelos en PPOOA, y deberá considerarse una auténtica herramienta de análisis de consistencia estructural.

En la siguiente tabla se muestran algunos ejemplos de inconsistencias intrínsecas de PPOOA.

Descripción	**Tipo**
Un componente de dominio puede estar compuesto por otros componentes de dominio, componentes de tipo estructura o componentes de tipo algoritmo.	Composición
Un Proceso no puede estar compuesto por componentes del tipo Objeto Controlador ni Subsistema.	Composición
Un Mecanismo de Coordinación de tipo Buffer no puede usar otros Mecanismos de Coordinación (ni siquiera de su mismo tipo).	Uso
Un Mecanismo de Coordinación de tipo Semáforo no puede usar otros Mecanismos de Coordinación (ni siquiera de su mismo tipo).	Uso
Un componente sólo puede heredar de otro componente del mismo estereotipo (de acuerdo con el metamodelo de PPOOA).	Herencia

Tabla 1. Inconsistencias intrínsecas del estilo PPOOA.

Por ejemplo, si un arquitecto software intenta descomponer un elemento constructivo de tipo proceso en otros de tipo subsistema, la herramienta no se lo permitirá, ya que por la propia semántica del estilo tal composición carece de sentido y, por tanto, debe ser considerada como un error de consistencia intrínseca.

Como las reglas de consistencia intrínseca se aplican antes de que el elemento de modelado sea almacenado en el repositorio de datos, no es necesario preocuparse más de este tipo de inconsistencias. No pasa lo mismo con las inconsistencias explícitas que tratamos a continuación.

4.2 Inconsistencias Estructurales Explícitas de PPOOA

De acuerdo con las conclusiones obtenidas en estudios previos [8], las **interfaces** entre componentes pueden ser el origen de inconsistencias, dado que una regla de carácter general, que debe cumplirse a nivel del modelo completo, es que toda interfaz requerida por un componente debe ser suministrada por algún otro componente dentro de algún subsistema. En la Figura 1 se muestra el metamodelo de consistencia de PPOOA, donde se ponen de manifiesto las potenciales fuentes de inconsistencias explícitas, y que ha sido usado como base para proponer un listado de las mismas.

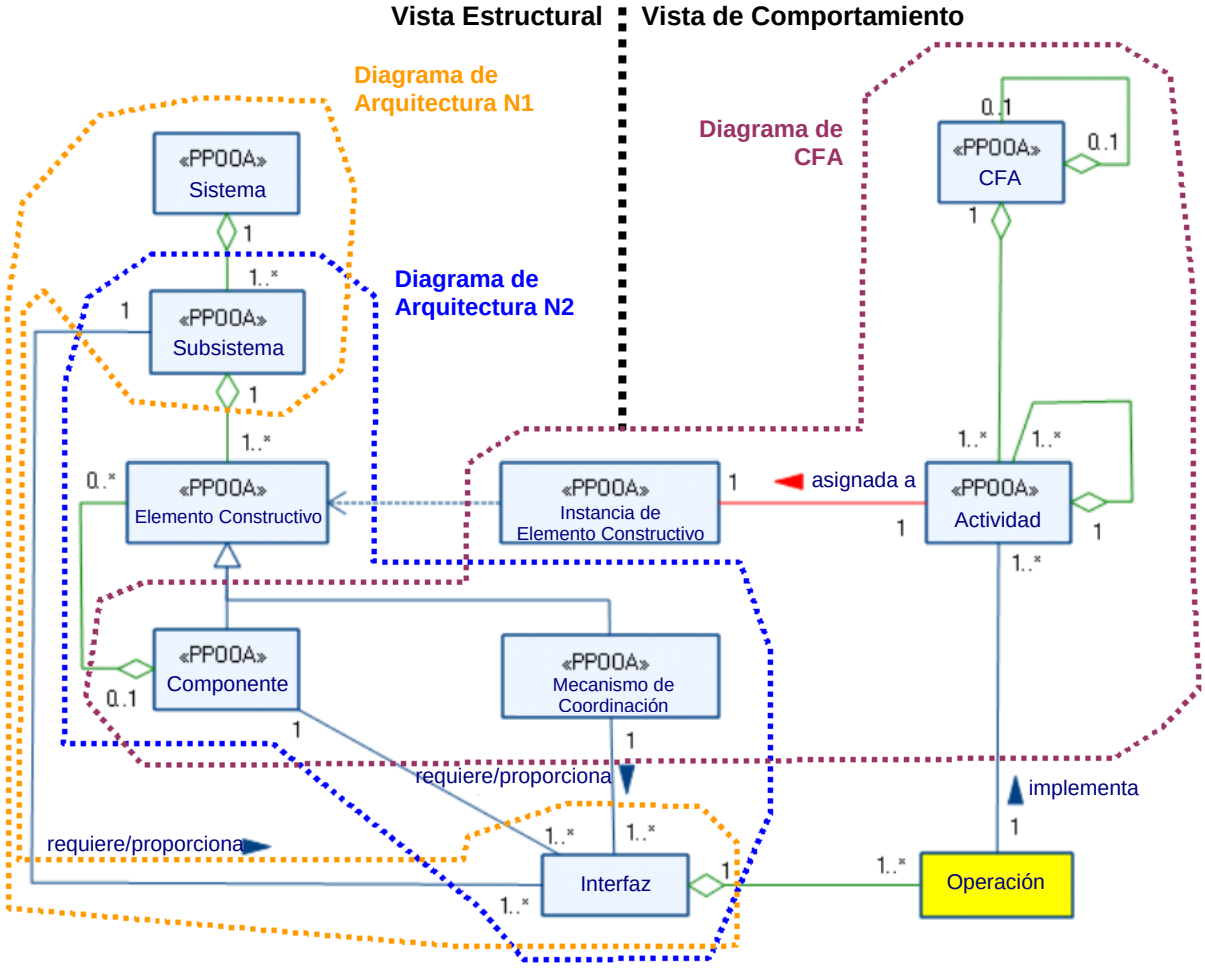

Figura 1. Metamodelo de consistencia de PPOOA

Los **mecanismos de coordinación** son otra fuente potencial de inconsistencias, dado que éstos pueden aparecer en los diferentes diagramas, que representan la descripción de la arquitectura de los diferentes subsistemas en los que se descompone el sistema a desarrollar. Aunque esto mismo es aplicable en general para todos los elementos constructivos de PPOOA, se ha considerado que los mecanismos de coordinación son especialmente sensibles a la aparición de inconsistencias por sus implicaciones respecto al tratamiento de la concurrencia.

Se ha considerado relevante una de las categorizaciones descritas en la referencia [2], por su similitud con el concepto abstracto-concreto de los dos niveles de diagramas estructurales de PPOOA. Los diagramas de arquitectura de primer nivel son más abstractos

que los de segundo nivel, siendo estos últimos más concretos (poseen mayor nivel de detalle).

Las inconsistencias se plantean implícitamente a través de sus reglas asociadas (por ejemplo, "las clases abstractas no pueden ser instanciadas"). Una inconsistencia ocurre cuando al revisar el modelo de acuerdo con la regla se produce una cierta respuesta (por ejemplo, un comprobador del modelo detecta durante una revisión un objeto de una clase que fue declarada como abstracta). Si la regla se describe de forma positiva (regla de consistencia) la inconsistencia se produce cuando se incumple la regla. Por el contrario, si la regla se describe de forma negativa (regla de inconsistencia) la inconsistencia aparece cuando se cumple la regla.

A continuación se muestran algunos ejemplos de inconsistencias estructurales explícitas de PPOOA, mediante sus reglas asociadas con formulación positiva o negativa, según el caso.

Nivel	**Descripción**	**Regla**	**OK**
1	No se permiten relaciones de dependencia de uso reflexivas.	¿Existe alguna relación de uso de un subsistema consigo mismo?	No
1 y 2	Si un diagrama de alto nivel muestra una dependencia de uso entre dos subsistemas, debe existir al menos una pareja de elementos constituyentes relacionados (uno en cada subsistema).	¿Existe al menos una pareja de componentes en diagramas de nivel 2, asociados a algún subsistema, cuando existe una relación de uso entre dos subsistemas en el diagrama de nivel 1?	Sí
2	Un subsistema debe contener al menos un componente constituyente (de bajo nivel).	¿Existe algún diagrama de nivel 2 que no contenga algún componente?	No
2	Toda interfaz requerida debe tener una interfaz suministrada por algún componente del sistema.	¿Existe alguna interfaz requerida por algún componente que no es suministrada por ningún otro componente del modelo?	No

Tabla 2. Inconsistencias explícitas de PPOOA.

Así por ejemplo, antes de que un arquitecto software vaya a dar por concluida la tarea de elaboración del modelo, debe asegurarse de que toda interfaz que sea requerida por algún componente sea proporcionada por algún otro dentro del modelo completo. Si esta circunstancia no se da, entonces estaremos ante una inconsistencia sistémica. En el caso de que esta inconsistencia no se haya detectado en esta fase temprana, podría ocurrir que los desarrolladores de dos componentes los creasen de acuerdo a lo especificado (en las fases posteriores de diseño detallado y de implementación), que funcionasen correctamente (estuviesen libres de errores) y que, sin embargo, no se comunicasen entre sí porque el arquitecto del sistema no previó que una de las interfaces requeridas por uno de ellos tuviese que ser proporcionada por el otro.

Éste es sólo un ejemplo, pero sirve para ilustrar la gravedad de las consecuencias de una inconsistencia no detectada y no corregida a tiempo.

5. Proceso de Resolución de Inconsistencias

Una vez identificadas las inconsistencias explícitas, debe comenzar el proceso de análisis propiamente dicho (ver Figura 2). La estrategia a seguir sería la de evaluar un modelo construido correctamente (de acuerdo con las reglas intrínsecas del estilo) con la lista de chequeo de las inconsistencias identificadas. Si el modelo pasa las reglas de chequeo, el modelo es consistente y no es necesario realizar ninguna tarea adicional.

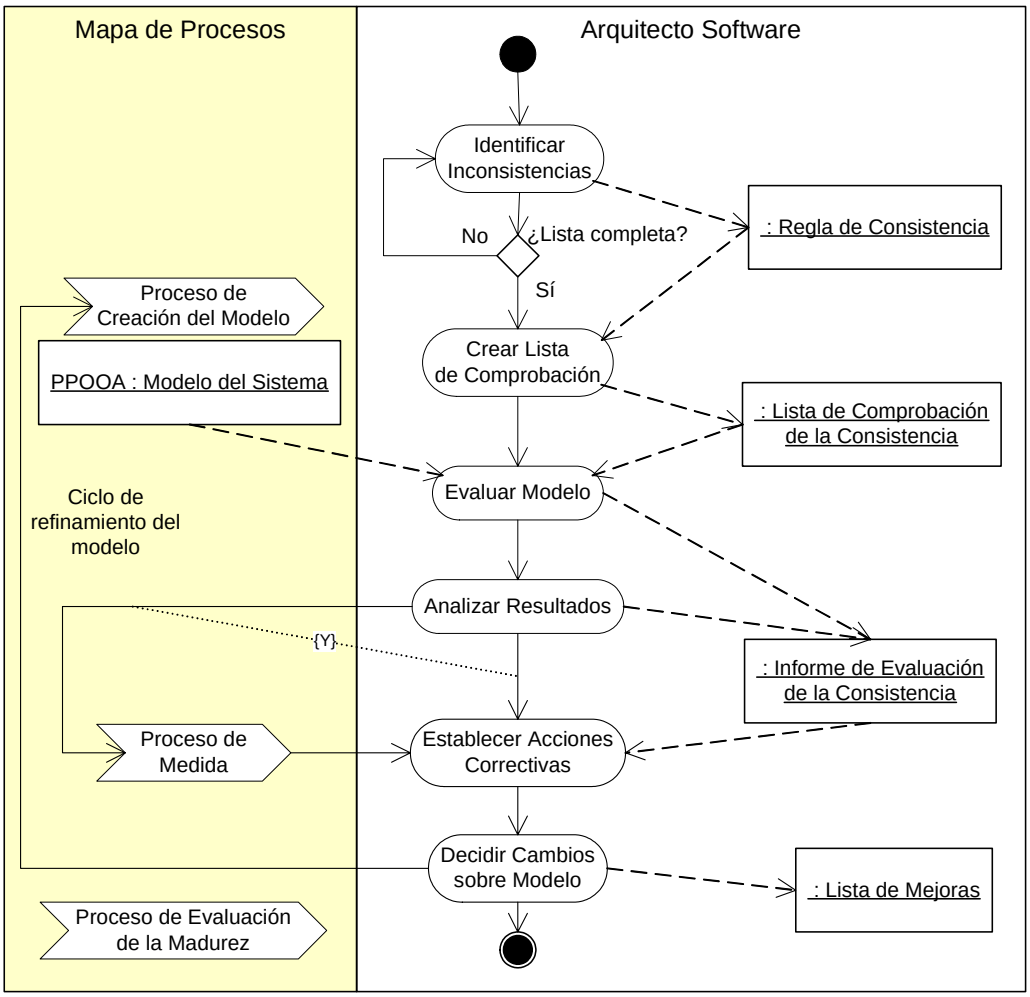

Figura 2. Proceso de Evaluación de la Consistencia

Por el contrario, si alguna de las reglas de consistencia es violada, entonces es preciso modificar el modelo para alinearlo con la regla violada.

Las tareas concretas de corrección del modelo para que éste cumpla con las reglas de consistencia dependerán de cada regla, pero en general van a consistir en la adición o eliminación de elementos del modelo que lo hagan más consistente.

Aunque la tarea de evaluación de la consistencia de un modelo puede realizarse de forma manual, parece lógico que se automatice mediante el uso de una herramienta de análisis. El desarrollo de tal herramienta se considera de crucial importancia para asegurar que todas las inconsistencias son detectadas y corregidas a tiempo. Una herramienta de análisis debe además proporcionar indicadores de situación (métricas de consistencia) y sugerencias de modificación del modelo para su alineamiento con los criterios de consistencia.

5.1 Inconsistencias Temporales

Todo modelo sufre un proceso de maduración evolutivo por un hecho evidente: nadie describe de una ver un modelo completo, sino que se va describiendo por partes y además cada parte sufre una maduración con el tiempo.

Esta característica de la construcción de modelos provoca que aparezcan inconsistencias temporalmente hasta que se completa o se refina el modelo. Un modelo debe estar suficientemente maduro como para ser aceptado como una solución de ingeniería que pueda ser entregada para el diseño detallado.

La actividad de análisis de la consistencia puede verse como una herramienta más para valorar la madurez de un modelo. Sin embargo no debe ser la única. Es posible llegar a dos modelos alternativos que resuelvan un cierto problema, que sean correctos de acuerdo con los criterios de consistencia establecidos y que sin embargo no sean igual de buenas bajo otras perspectivas, como puedan ser la viabilidad económica o la consecución de un cierto requisito no funcional no directamente relacionado con la capacidad de respuesta del sistema.

Es importante aclarar que el hecho de que un modelo sea consistente no asegura que esté suficientemente maduro, y será necesario proporcionar otras herramientas de ingeniería y otros criterios para asegurar la madurez. Sin embargo, la consistencia es una condición necesaria para la madurez de un modelo.

5.2 Métricas de Consistencia

Aunque estamos en una etapa muy preliminar de la concepción de un sistema, parece razonable que se cuantifique la consistencia global de un modelo en base a unas reglas predefinidas. Existen diversas estrategias de cuantificación, pero está claro que para poder valorar si un modelo es suficientemente consistente hay que establecer algún mecanismo de medida y fijar unos límites para los valores aceptables de las métricas.

Una primera aproximación es considerar cada regla de consistencia como un factor de igual peso que las demás y con la calificación de “pasa-no-pasa”. Un primer valor de la consistencia global sería la suma del número de criterios pasados comparado con el número total de criterios. Con esto tendríamos un porcentaje de cumplimiento de criterios que sería suficiente para un primer análisis.

El algoritmo de consistencia puede enriquecerse asociándole pesos a las reglas de consistencia. Una forma de hacerlo es valorar la importancia de la regla en función de la criticidad de la misma para la consecución de los requisitos temporales que se le van a requerir al sistema.

El siguiente paso a la decisión del algoritmo de medida sería decidir el valor aceptable. Una vez más, y empleando una opción simple, sería decir que no se considera el modelo suficientemente consistente hasta que el 100% de las reglas de consistencia han sido satisfechas.

Para valorar si este criterio estricto es razonable o no habrá que tener en consideración los otros criterios que impactan sobre la madurez de un modelo y que no tienen que ver estrictamente con la consistencia estructural del mismo.

5.3 Diseños Alternativos

Una práctica habitual en ingeniería es la de plantear diferentes alternativas para resolver el mismo problema. Desde el punto de vista práctico, las soluciones alternativas han de satisfacer los requisitos funcionales exigidos, así como los no funcionales considerados como críticos. Una vez planteadas las soluciones (diferentes), puede realizarse una

comparación (trade-off) de ambas que permita visualizar las ventajas e inconvenientes de cada una y tomar una de las dos como la que va a construirse mediante un criterio razonable.

Los chequeos de consistencia pueden ayudar a tomar decisiones sobre la solución más adecuada según los criterios establecidos que normalmente estarán ligados a atributos de calidad y requisitos no funcionales del sistema.

6. Conclusiones y Trabajos Futuros

En el presente artículo se ha puesto de manifiesto la importancia que tiene la identificación temprana de las inconsistencias en los modelos de sistemas, y que la eliminación de éstas supone un primer nivel de verificación sobre el producto final.

Se ha limitado el ámbito de estudio a aquellas inconsistencias que tienen que ver con la vista estructural del estilo arquitectónico PPOOA y específicamente las que van más allá de las propias reglas del estilo.

Se han mostrado una serie de ejemplos de inconsistencias específicas de PPOOA, y se han planteado una serie de estrategias de resolución de las mismas.

Quedan pendientes una serie de tareas que permitirán complementar la actual herramienta de modelado con PPOOA para darle capacidades de evaluación de modelos y de toma de decisiones de modelos alternativos. Las próximas tareas previstas son las siguientes:

- Elaborar una guía de ingeniería para la toma de decisiones en relación con la eliminación de inconsistencias de modelos basados en PPOOA.
- Evaluar la consistencia entre modelos de arquitectura y los diagramas de descripción de protocolos de puertos.
- Evaluar consistencia entre modelos PPOOA y nuevos diagramas de estructuras compuestas y de definición de protocolos de puertos.
- Evaluar el impacto que puede tener la introducción de los nuevos diagramas sobre la definición de la propia arquitectura PPOOA.
- Traducir las reglas identificadas a un lenguaje mecanizable.
- Crear una aplicación que permita ejecutar las reglas de consistencia sobre modelos creados con el estilo PPOOA.

Referencias

[1] PPOOA (Processes Pipelines in Object Oriented Architectures) Página web oficial: http://www.ppooa.com.es/)

[2] Egyed, A. "*Heterogeneus View Integration and its Automation*". Partial fulfilment work for PhD. University of Southern California. 2000.

[3] Fernández, J.L. and Monzón, A. "Extending UML for Real-Time Component Based Architectures". *14 th International Conference on Software and Systems Engineering (ICSSEA)*. Paris, December 2001.

[4] Fernández, J.L. "*Arquitectura software genérica para sistemas de tiempo real*". Tesis Doctoral. Universidad Politécnica de Madrid. 1997.

[5] Fernández, J.L. "Los Modelos de la Ingeniería de Sistemas". *IX Congreso Internacional de Ingeniería de Proyectos*, Málaga (Spain), Junio 2005.

[6] Fernandez, J.L. and Mason, W.M. “A Process for Architecting Real-Time Systems”, *15th International Conference on Software and Systems Engineering (ICSSEA)*, Paris, December 2002.

[7] Fernandez J.L. and Martinez J.C. “Implementing a Real-Time Architecting Method in a Commercial CASE Tool”. *16th International Conference on Software and Systems Engineering (ICSSEA)*, Paris, December 2003.

[8] Monzón, A. “*Técnicas para el Análisis de la Consistencia de Modelos en el Desarrollo de Software Embarcado - Memoria Trabajos Tesis y Suficiencia Investigadora*”, Escuela Técnica Superior de Ingenieros de Telecomunicación. Universidad Politécnica de Madrid, Septiembre 2005.

[9] Monzón, A. “*Técnicas para el Análisis de la Consistencia de Modelos en el Desarrollo de Software Embarcado - Propuesta de Tesis Doctoral*”, Escuela Técnica Superior de Ingenieros de Telecomunicación. Universidad Politécnica de Madrid, Abril 2003.

[10] OMG, "*UML 2.0 Superstructure*", OMG, ad/00-09-02, 2000.

[11] Schmidt, D.C. “Model driven Engineering”. *IEEE Computer*. February 2006.

[12] SysML Merge Team. “*Systems Modelling Language Specification*”. OMG ad/2006-03-01, April 2006.

Correspondencia

Antonio Monzón Díaz, Integración de Sistemas, División de Aviones de Transporte Militar, EADS-CASA, Av. John Lennon, s/n, 28906 Getafe, Spain.

Phone: +34 91 624 14 92, Fax: +34 91 624 27 05, E-mail: antonio.monzon@casa.eads.net

José Luis Fernández Sánchez, Área de Proyectos de Ingeniería, E.T.S. Ingenieros Industriales, Universidad Politécnica de Madrid, C/ José Gutiérrez Abascal, 2, 28006 Madrid, Spain.

Phone: +34 91 336 31 44, Fax: +34 91 336 31 46, E-mail: jlfdez@etsii.upm.es

URL: http://www.ppooa.com.es