Propuesta Metodológica y Herramientas de Soporte para Modelar y Validar Esquemas Conceptuales

Methodological proposal and support tools to model and validate conceptual schemes

Manuel Perez Cota
Facultad de Informática
Universidad de Vigo
Vigo, España
mpcota@uvigo.es

Marcelo M. Marciszack – Mario A. Groppo
Departamento de Ingeniería en Sistemas de Información
Universidad Tecnológica Nacional
Córdoba, Argentina
marciszack@gmail.com - sistemas@groppo.com.ar

Resumo **— El presente trabajo propone la definición de una metodología y la herramienta de soporte asociada, para la especificación y validación de un Modelo Conceptual, a través de la transformación automatizada de modelos a Autómatas Finitos y su validación correspondiente. Describe las características del Proceso de modelado, utilizando la "Notación para el Modelado de Procesos de Negocios" en la actividad de modelado, aplicando los conceptos del Desarrollo de Software dirigido por Modelos. Se propone un mapeo directo entre actividades del proceso de negocio a Casos de Uso, los cuales se transforman a Máquinas de estados a las que se somete a un proceso automatizado de validación, y de esta forma validar el conjunto de especificaciones funcionales en el modelo conceptual de un dominio bajo estudio.**

Palabras Clave - Modelado Conceptual; Especificaciones funcionales; Validación de modelos; BPMN; Autómatas Finitos; Desarrollo de software Dirigido por Modelos .

Abstract **— This paper proposes the definition of a methodology and associated support tool for the specification and validation of a Conceptual Model, through automated processing to Finite Automata models and corresponding validation. It describes the characteristics of modeling process, using the "Notation for Business Process Modeling" in the modeling activity, applying the concepts of Model-Driven Software Development. It's proposed a direct mapping between activities of the business process and Use Cases, which are transformed to state machines that are subjected to an automated validation process, and thus validate the set of functional specifications in the conceptual model of a domain under study.**

Keywords - Conceptual Modeling; Functional Specification; Validation of models; BPMN; Finite Automata; Model Driven Software Development.

I. INTRODUCCIÓN

Las debilidades de la mayoría de los métodos y metodologías utilizados para la obtención de esquemas conceptuales se reflejan en las primeras etapas del proceso de desarrollo de software. El principal problema derivado de estas debilidades metodológicas radica en la dificultad en determinar si el modelo conceptual refleja fiel y completamente la esencia del dominio [1].

Los errores que se cometen en la etapa de especificación de requerimientos para lo obtención de un esquema conceptual tienen un costo relativamente alto en relación a su reparación y crecerá en forma exponencial a medida que se avanza en las diferentes etapas del proceso de desarrollo [2]. La preocupación por definir los requisitos de manera adecuada está extensamente tratada en [3], donde el eje central es la definición de buenas prácticas en el establecimiento de los mismos, ya que plantea que "el éxito de cualquier proyecto de desarrollo está íntimamente relacionado con la calidad de los requisitos." y que "el proceso de establecimiento de requisitos es mucho menos homogéneo y bien entendido que el proceso de desarrollo de software en su conjunto".

Los esquemas conceptuales deben procurar establecer una definición sin ambigüedad de lo que se quiere representar.

Es así, que la presente propuesta abarca la definición de una metodología, un conjunto de herramientas de soporte, y la definición de transformaciones automatizadas entre los modelos intermedios, que posibilita validar y verificar si el modelo conceptual construido representa fielmente el sistema de información a construir [4] [5].

II. MODELADO CONCEPTUAL Y REQUERIMIENTOS FUNCIONALES.

Desde la óptica disciplinar de los Sistemas de Información y los sistemas de software asociados a estos, un Esquema Conceptual será definido como un modelo de representación de la realidad, sobre un dominio de problema determinado, el cual deberá incluir además, el lenguaje utilizado en su definición, de manera que no existan ambigüedades, de esta manera de reducir el "gap" semántico, entre el constructor del modelo y los usuarios del mismo.

Un Esquema Conceptual, se utiliza para abstraer la esencia de un dominio bajo estudio, sirviendo a la vez, para proveer de

una correcta y completa especificación de los requerimientos que él debe cumplir.

En este contexto el presente trabajo, se focaliza con la visión aportada por [6] en donde un Esquema Conceptual es interpretado como un refinamiento de los requerimientos de usuario a través de los requisitos funcionales que resultarán en especificaciones más detalladas que constituirán dicho esquema.

En este mismo sentido otro aporte es el desarrollado por [7] en donde el Modelo Conceptual, establece los requisitos funcionales del Software y es uno de los resultados principales de dichas actividades, constituyéndose en una pieza fundamental para posteriores actividades en el desarrollo del Software.

El Modelo Conceptual representando los requisitos funcionales de un sistema de información, es la pieza clave para establecer el vínculo entre el espacio del problema y el espacio de la solución.

III. TENDENCIAS ACTUALES EN LA CONSTRUCCIÓN DE MODELOS.

En los últimos años, el modelado de procesos de negocios, ha despertado especial interés por parte de la Ingeniería de Software, debido a que brinda un punto de partida para la captura de requisitos. Estos modelos se consideran esenciales para conocer las actividades de una organización, permitiendo establecer los fundamentos para la construcción de un sistema de información correcto.

El Object Management Group (OMG) ha utilizado para representar los modelos de negocio diferentes tipos de notaciones, pero le ha dado principal importancia a Business Process Modeling Notation (BPMN) [11], en español Notación para el Modelado de Procesos de Negocio, es una notación gráfica estandarizada que permite el modelado de procesos de negocio en un formato de flujo de trabajo y a Unified Modeling Language (UML) [8] (a través de los diagramas de actividad y diagrama de casos de uso). Ambas notaciones ofrecen soluciones similares para la mayoría de los patrones de flujo de trabajo que soportan. Esto es lógico debido a que ambos estándares fueron diseñadas para satisfacer las mismas necesidades de modelado, pero con objetivos diferentes en diferentes etapas del desarrollo.

El interés de la Ingeniería de Software en el Modelado del Negocio surge porque a partir del estudio de la transformación de modelos, es posible iniciar el modelado de sistemas de información (elicitación de requisitos) que se pueden integrar al proceso de desarrollo del software.

Para el desarrollo de esta propuesta nos centraremos en el Desarrollo de Software Dirigido por Modelos (MDD) [12], las definiciones y documentos emitidos por la OMG [9] [10] que es el organismo que se ha encargado del estudio y definición de los procesos de transformación de los modelos, en forma conjunta con el World Wide Web Consortium [13]. La transformación de modelos permite además de mejorar los tiempos en los procesos de elicitación de requisitos, otorgando confiabilidad a todo el proceso, brindando además la reducción de costos en el desarrollo e implementación de los sistemas.

IV. PROPUESTA METODOLÓGICA

Destacando el rol fundamental que actualmente desempeñan los modelos en el proceso de desarrollo del software, la propuesta metodológica inicia con el modelado del negocio a través de BPMN ver. 2.0. Una vez que se obtiene esta representación, se continúa identificando aquellas actividades del negocio que implican un manejo inherente de información para posteriormente transformarlas en Casos de Uso. Seguidamente se debe convertir cada caso de uso en una máquina abstracta. Ambos procesos, la gestión de cada Caso de Uso, y la transformación a Autómata Finito (AF) son soportados por la herramienta "Sistemas Integral Administración Requerimientos" (SIAR). Finalmente se procede a verificar la consistencia de cada Máquina abstracta ya sea que haya sido generada a partir de un Proceso de Negocio o un Caso de Uso, para de este modo detectar anomalías en las definiciones de los modelos.

A. Justificación de la propuesta

La ingeniería de software establece que el problema de construir software debe ser encarado de la misma forma en que los ingenieros construyen otros sistemas complejos, como puentes, edificios, barcos y aviones [12]. La idea básica consiste en observar el sistema de software a construir como un producto completo y a su proceso de construcción como un trabajo ingenieril. Es decir, un proceso planificado basado en metodologías formales apoyadas por el uso de herramientas.

B. Fundamentación.

Hacia finales de los años 70, Tom DeMarco [14] introdujo el concepto de desarrollo de software basado en modelos o MBD (por sus siglas en inglés "Model Based Development"). De Marco destacó que la construcción de un sistema de software debe ser precedida por la construcción de un modelo, tal como se realiza en otros sistemas ingenieriles.

La abstracción es una de las principales técnicas con la que la mente humana se enfrenta a la complejidad. Ocultando lo que es irrelevante, un sistema complejo se puede reducir a algo comprensible y manejable.

Actualmente casi todos los métodos de desarrollo de software utilizan modelos. Lo que varía de un método a otro es la clase de modelos que deben construirse, la forma de representarlos y manipularlos.

Esta metodología expresa la idea de que es posible modelar los procesos de negocio a través de BPMN y utilizar estos modelos como guías para la obtención del listado de casos de uso del sistema que darán soporte informático al negocio modelado.

El modelado del negocio es un punto fundamental para comprender el contexto del sistema que se está construyendo, y esto impacta directamente en el éxito o fracaso de un proyecto de software. Si no podemos entender el negocio, se pueden presumir conceptos erróneos sobre lo que debe hacer el software y cómo debe ser utilizado.

C. Etapas del proceso metodológico

La metodología que aquí se presenta, se conforma de las siguientes etapas, explicadas con mayor detalle a lo largo de este documento.

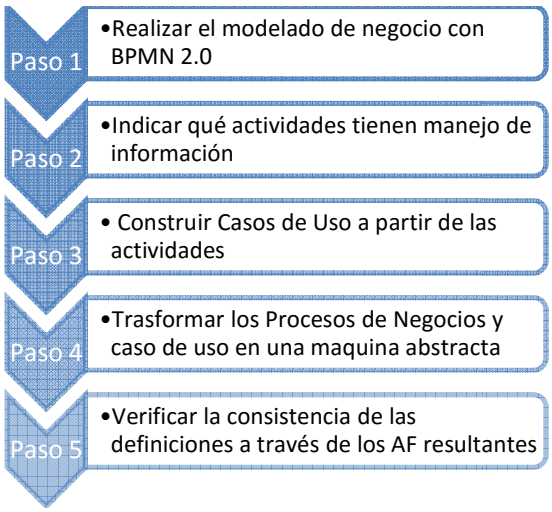

Figura 1 –Etapas de aplicación de la Metodologías

1) Realizar el Modelado de Negocio con BPMN 2.0

En BPMN, los Procesos de Negocio involucran la captura de una secuencia ordenada de las actividades e información que utiliza el proceso, el cuál implica representar cómo una empresa realiza sus objetivos centrales

BPMN es una notación basada en diagramas de flujo para definir procesos de negocio, desde los más simples hasta los más complejos y sofisticados, para dar soporte a la ejecución de procesos. BPMN es gráficamente más rico, con menos símbolos fundamentales, pero con más variaciones de éstos, lo que facilita su comprensión por parte de gente no experta. A continuación en la Fig.2 se visualiza un proceso de negocio.

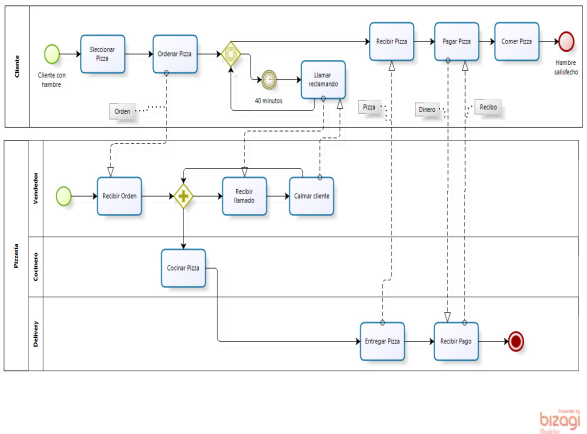

Figura 2. Ejemplo de un Modelo de Procesos de Negocio modelado con BPMN.

2) Indicar qué actividades tienen manejo de información.

En la Fig. 3 se muestra como se procede para seleccionar las actividades de negocio que son automatizadas y formarán parte del Sistema de Información

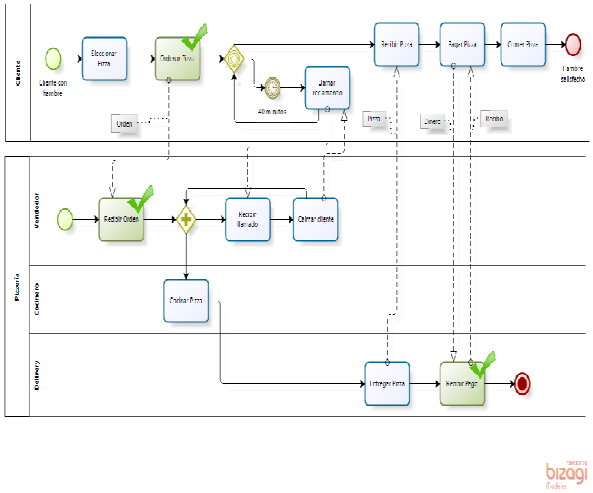

Figura 3. Se seleccionan las actividades automatizadas

Aquí, el analista deberá identificar en los diagramas de procesos aquellas actividades que utilicen / generen información, diferenciándolas de aquellas que son puramente manuales.

3) Construir Casos de Uso a partir de las actividades.

Un Caso de Uso es una descripción de los pasos o las actividades que deberán realizarse para llevar a cabo algún proceso. En el contexto de la ingeniería de software, un Caso de Uso es una secuencia de acciones que se desarrollarán entre un sistema y sus usuarios en respuesta a un evento sobre el propio sistema.

Los Diagramas de Casos de Uso sirven para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios y/u otros sistemas. O lo que es igual, un diagrama que muestra la relación entre los actores y los Casos de Uso en un sistema. Una relación es una conexión entre los elementos del modelo, por ejemplo, la especialización y la generalización son relaciones.

Como técnica de extracción de requerimiento un Diagrama de Casos de Uso permite que el analista se centre en las necesidades del usuario, qué espera éste lograr al utilizar el sistema, evitando que la gente especializada en informática dirija la funcionalidad del nuevo sistema basándose solamente en criterios tecnológicos. A su vez, durante la extracción (elicitation en inglés), el analista se concentra en las tareas centrales del usuario describiendo por lo tanto los Casos de Uso que mayor valor aportan al negocio. Esto facilita luego la priorización de los requerimientos. Los Casos de Uso evitan típicamente la jerga técnica, prefiriendo el léxico del usuario final.

Desde una perspectiva tradicional de la ingeniería de software, un Caso de Uso describe una característica del sistema.

En este paso, utilizando como guía las actividades marcadas como no manuales en el punto anterior, es preciso identificar los Casos de Uso del sistema que darían soporte "informático" a las actividades.

4) Trasformar los Procesos de Negocios y Casos de Uso en un Autómata Finito).

Las máquinas abstractas son usadas para modelar una gran variedad de sistemas en diversas áreas. En este caso utilizamos un tipo particular de máquina abstracta que son los Autómatas Finitos.

Entonces, una vez que se cuenta con una especificación detallada de los Procesos de Negocios y Casos de Usos que satisfacen las necesidades informáticas del negocio, se realiza la trasformación de los mismos a máquinas abstractas.

Para formalizar esta transformación se definieron un conjunto de reglas de conversión bidireccionales:

Partiendo de BPMN: Cada una de las actividades de Negocio identificadas tendrá un mapeo directo con cada estado identificado del Autómata finito. Lo mismo ocurrirá con los estados de Inicio y de finalización, ya sea por el éxito del procedimiento o por el fracaso del mismo. Los arcos del autómata finito surgirán a través de los flujos de trabajo que vinculan las Actividades del proceso de Negocio.
Partiendo de Plantilla de Caso de Uso: El estado inicial lo constituye las precondiciones, cada acción o respuesta del sistema son arcos del AF, siendo los episodios alternativos las transiciones a estados diferentes con diferentes entradas. Este proceso es soportado tanto para CU a nivel de trazo Grueso y Finos, lo único que varía es la cantidad de estados que la constituyen.

5) Verificar la consistencia de las definiciones a través de los Autómatas Finitos resultantes.

Al contar con la representación de cada Proceso de negocio o Caso de Uso, y tal lo desarrollado en [15] , sobre cada uno de estos se obtiene un Autómata Finito, partir de una transformación directa de la representación del modelo que debe ser validado, podemos efectuar distintas acciones:

- *Conjunto Conexo y accesibilidad de estados.*

Estas verificaciones resultan fundamentales para verificar que todas y cada una de las abstracciones de estados por los que transita el AF tienen correlación con el planteo del mismo, ya que si un estado definido en el AF no es accesible desde el estado inicial, significa que el modelo que está siendo representado por el autómata no está correctamente planteado y debe necesariamente ser reformulado.

Si un autómata no es conexo basta con eliminar los estados inaccesibles (estados no conexos) y todas sus transiciones (las de entrada y las de salida) para obtener un nuevo autómata conexo equivalente.

- *Autómata Finito Determinista.*

La forma de definir los modelos de procesos puede resultar en caminos o procesos paralelos o simultáneos, los que se traducen en no determinismo dentro de los Autómatas Finitos, los cuales merecen una especial atención de su conveniencia en mantenerlos en los modelos. Así, es necesario convertir el AF No Determinista en uno Determinista equivalente, de manera de brindar al analista la posibilidad de analizar si se reformula el modelo o se mantiene tal como está definido.

- *Minimización del Autómata Finito.*

Un AF no mínimo significa la presencia de estados equivalentes, los cuales pueden ser identificados y reemplazados, y de esta manera simplificar el Modelo que representa al Proceso de Negocio (en el proceso de Negocio dos estados equivalentes del AF equivalen a la existencia de una re invocación de una acción que puede ser eliminada).

- *Simulación de Ejecución de Autómatas Finitos.*

Para cada modelo de proceso de Negocio y su correspondiente representación del Autómata Finito, pueden establecerse un conjunto de entradas, que al ser simuladas, verifican si se producen los resultados esperados por el modelo.

Luego de hacer esto podemos realizar una trazabilidad inversa hacia los procesos y determinar si en el proceso hay actividades que no se realizan (a partir de los estados no conexos) y procesos que son irrelevantes o innecesarios desde la minimización del autómata relacionado.

V. HERRAMIENTAS DE SOPORTE

Para dar sustento y soporte a la metodología propuesta a través la transformación de modelos se ha desarrollado una herramienta que definiremos a continuación.

A. Sistema Integral de Administración de Requerimiento.

A partir de la definición de las actividades de negocios automatizadas identificadas en los Procesos de negocios definidas se realiza la construcción de los casos de usos del sistema, los cuales son administrados por una herramienta (SIAR) que agilice su registración, normalice su contenido y posibilite implementar validaciones funcionales a través de los Autómatas Finitos.

B. Descripción de la herramienta.

1) Construir Casos de Uso a partir de las actividades.

Esta herramienta es una aplicación web que permite registrar en forma normalizada los casos de uso que comprende:

- Administración de los atributos de un proyecto (de sistemas) y sus versiones.
- Gestión de los alcances de cada versión del proyecto y los casos de uso asignados.
- Administración de los artefactos de un caso de uso, incluyendo actores, pre-condiciones, post-condiciones, escenario principal y escenarios alternativos, y su versionado.
- Clasificación, priorización y trazabilidad de los casos de uso.
- Visualización de consultas y generación de reportes en distintos formatos, inclusive XPDL, para comunicarse con otras aplicaciones.

- Gestión de atributos de procesos de negocio, de actividades de negocio que los componen y los casos de uso asociados a estas actividades.

2) Trasformar un caso de uso en una máquina abstracta

Una vez completa la versión de un caso de uso y utilizando el conjunto de reglas de conversión del caso de uso en un grafo de estados, definidas en este paso de la metodología, SIAR genera el grafo de estados.

El grafo de estados asociado al caso de uso tiene un alfabeto de tres símbolos para indicar qué evento lo cambia de un estado/nodo a otro:

- A = Por medio de una Acción determinada.
- S = Cuando Si se cumple una condición que bifurca a un escenario alternativo.
- N = Cuando No se cumple una condición que bifurca a un escenario alternativo.

Partiendo de un estado/nodo origen, en la función de transición puede estar asociado solamente uno de los símbolos: A, N o S. Con esto se cumple la condición necesaria de un autómata finito determinista. De esta manera, si la transición entre dos estados/nodos se da dentro de un mismo camino, se asocia el símbolo A. Si en cambio interviene una bifurcación, la función de transición hacia el estado/nodo destino por cumplimiento de la condición de bifurcación, se asocia el símbolo S. Por el otro camino de la bifurcación, se asocia el símbolo N. Los estados y sur relaciones (arcos pueden verse en la próxima figura.

Tabla De Estados Casos de Uso: 1 - 1 - 1 - 1 CONSULTAR CURSOS Versión: 6

Estado / Paso Origen	Estado / Paso Destino	Transición	Estado Final por Error	Tipo
1	2	A		S
2	3	A		S
3	4	A		S
4	4 - A	N		C
4 - A	4 - A - 1	A		S
4 - A - 1	4 - A - 2	A	SI	S
4	5	S		C
5	6	A		S
6	6 - A	S		C
6 - A	6 - A - 1	A		S
6	7	N		C

Figura 4. Tabla de estados de un caso de uso. Transformación automática por SIAR.

Una vez generado el grafo de estados se expresa en protocolo XPDL, por ser el más adecuado para intercambiar modelos de procesos entre distintas herramientas. Este lenguaje da soporte a la definición y a la importación/exportación de procesos, con el objetivo de que, aunque se modele un proceso en una aplicación, este modelo pueda ser usado por otras aplicaciones de modelado y/o por otras aplicación es que trabajen en el entorno de ejecución.

A continuación en la Fig.5 se muestra el resultado de la transformación a través de la configuración de una herramienta XLST a XML aceptado por el módulo Validador de Autómatas Finitos.

```
<transicion
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="java:dominio.automatas.TransicionFinita">
    <simbolo>
        <simbolo>S</simbolo>
    </simbolo>
    <estado-inicio>
        <denominacion>4</denominacion>
    </estado-inicio>
    <estado-fin>
        <denominacion>5</denominacion>
    </estado-fin>
    <denominacion>f( 4, S) = 5</denominacion>
</transicion>
```

Figura 5. Fragmento de archivo XML generado por SIAR.

C. Simulador de Autómata Finito.

A partir de este punto es posible simular el comportamiento que tendrá el sistema y llevar a cabo la última etapa del proceso metodológico que es verificar la consistencia secuencial de los escenarios de los procesos de negocios y de los casos de uso.

Lo que se hace es ingresar el archivo XML (salida transformada de BPMN o SIAR), que representa al proceso de Negocio o caso de uso como grafo de estados, al sistema de validación de Máquinas Abstractas "Autómata Finito".

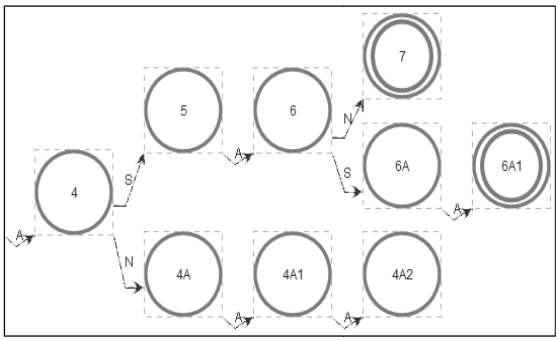

Fig. 6. Grafo de estados en el simulador de autómatas finitos

D. Validaciones sobre el Autómata finito generado

El conjunto de validaciones que se efectúan, sobre cada uno de los AF construidos a partir de los CU Derivados y de Soporte son los siguientes:

- **Visualización de Autómatas Finitos en la Herramienta Validación**: Definición Formal, Grafo y Tabla de Estados/Entradas
- **Conjunto Conexo – Accesibilidad de Estados**: Se verificará si todos los estados definidos son accesibles desde el estado inicial, con lo cual garantiza una correctitud en la definición de los mismos.

- **Construcción de Autómata Finito Determinista**: Para cada uno de los Autómatas Finitos construidos se informará si el mismo en caso de ser No determinista (Procesos en Paralelo) un AF Determinista equivalente, con procesos secuenciales.
- **Minimización del Autómata Finito Determinista**: Al Igual que el punto anterior, para cada uno de los Autómatas Finitos construidos se informará si el mismo es factible de ser minimizado, esto es hay estados equivalentes y la herramienta propone una nueva solución.
- **Simulación de Ejecución**: Este proceso resulta imprescindible para validar si los Procesos de Negocios representados a través del Autómata Finito respectivos están construidos de manera correcta. Este punto, se necesita una ejemplificación sobre su aplicación, ya que hay realizar una interpretación sobre las acciones descriptas en los casos de uso, y las transiciones entre estados de los AF.

El simulador posibilita probar el grafo de estados y comprobar si es aceptado o rechazado. Si es aceptado, significa que la consistencia de los escenarios del caso de uso es correcta. Si no, el rechazo, indica que habrá que revisar e introducir modificaciones en el modelo de origen al AF.

A continuación en la Fig 7 se muestra una pantalla de salida del módulo de la herramienta de Validación en donde se muestra la secuencia de entradas y la transición de estados.

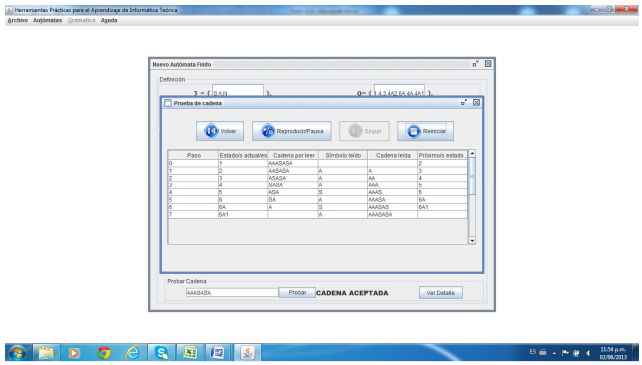

Fig.7. Pantalla informe resultado ejecución

De esta manera se logra control y trazabilidad de los cambios en los escenarios de los requerimientos funcionales.

VI. CONCLUSIÓN.

A través de esta propuesta metodológica y sus herramientas de soporte, que se sintetizan como un conjunto de transformaciones aplicadas sobre el modelo conceptual primario, es posible generar nuevos modelos que sirvan para representar las máquinas abstractas necesarias para la validación de los requerimientos funcionales iniciales, garantizando de esta forma que los modelos reflejen fielmente la realidad, sin ambigüedades, manteniendo la coherencia y asegurando la trazabilidad a lo largo de todo el proceso de gestión de requerimientos.

Estas validaciones y simulaciones a las Máquinas Abstractas generadas a través de un proceso automatizado de transformaciones, ya sean sobre: Procesos de Negocios, o Plantillas de Casos de Uso, nos permiten confirmar las características deseables sobre las especificaciones de los requisitos funcionales del sistema a construir.

Por tal motivo, este proceso de validación propuesto resulta sumamente útil para validar el modelo conceptual propuesto para luego construir el sistema de software que de soporte al sistema de información.

REFERENCIAS BIBLIOGRAFICAS

[1] E. Insfrán, I. Díaz, M. Burbano, Modelado de Requisitos para la Obtención de esquemas conceptuales. Disponible en: http://www.dsic.upv.es/~einsfran/papers/39-ideas2002.pdf Fecha Consulta: 02/10/06

[2] B. Boehm, V.R. Basili, Software defect reduction top 10 list. IEEE Computer, 01/01/01

[3] I. Sommerville, Ingeniería del Software. ISBN 9788478290741, Pearson Educación.

[4] Ian Sommerville Ingeniería de Software Editado por Pearson Educación – México 2011 Versión impresa ISBN 978-607-32-0603-7

[5] F. Sesé Muniátegui, Tesis Doctoral: Propuesta de un método de validación de esquemas conceptuales y análisis comparativo de la noción de información en los métodos de desarrollo de Sistemas de información Disponible en: www.tesisenxarxa.net/TDX-0517107-131929/ Fecha consulta: 20/05/08

[6] E. Insfrán, E. Tejadillos, S. Marti, M. Burbano, Transformación de Especificación de requisitos en esquemas conceptuales usando Diagramas de Interacción. Disponible en: www.inf.puc-rio.br/~wer02/zip/Transformacion_Espec(7).pdf Fecha Consulta: 04/12/07

[7] [Letelier 1999] P. Letelier, P. Sanchez, I. Ramos. http://www.researchgate.net/publication/36720988_Un_ambiente_para_especificaciones_incremental_y_validacin_de_modelos_conceptuales/file/d912f50ca20c33f5e5.pdf. Fecha de consulta web: 12 de marzo de 2013.

[8] Object Management Group. Unified Modelling Language: Superstructure Version 2.0 (online), Julio 2005, http://www.omg.org

[9] Object Management Group: XML Metadata Interchange (XMI). version 2.1.1, 1 December 2007. http://www.omg.org/technology/documents/modeling_spec_catalog.htm #XMI.

[10] Object Management Group: MOF Query / Views / Transformations. Version 1.0, April 2008. http://www.omg.org/technology/documents/modeling_spec_catalog.htm

[11] Object Management Group. Business Process Modeling Notation (BPMN). http://www.omg.org/technology/documents/br_pm_spec_catalog.htm, version 1.2, 3 January 2009.

[12] C. Pons, R. Giandini, G. Pérez. Desarrollo de Software dirigido por modelos – Conceptos Teóricos y su aplicación práctica Editorial Universidad Nacional de la Plata 1ra edición 2010.

[13] World Wide Web Consortium: Extensible Markup Language (XML). Version 1.0(fifth edition), 26 November 2008. http://www.w3.org/XML/

[14] DeMarco, T., Structured Analysis and System Specification, Yourdon Press, 1978.

[15] Manuel Perez Cota, Mario Groppo y Marcelo Marciszack. Validación de Especificaciones Funcionales en el modelado de Esquemas Conceptuales a través de Máquinas Abstractas. CoNaIISI 2013..

[16] F. Branco, R. Gonçalves, J. Martíns, M. Pérez Cota, Decision Support System for the Agri-food Sector-The Sousacamp Group Case. New Contributions in Information Systems and Technologies, 553-563, 2015

[17] J. Martins, Ro. Gonçalves, J. Pereira, M. Pérez Cota. Iberia 2.0: A way to leverage Web 2.0 in Organizations. Information Systems and technologies (CISTI), 2011 7th. Iberican conference.