Lámpsakos | Nº 12 | pp. 101-109 | julio-diciembre | 2014 | ISSN: 2145-4086 | Medellín - Colombia

MODELOS DE REQUISITOS BASADOS EN I* PARA DETECTAR PROACTIVIDAD EN *DASHBOARDS*

REQUIREMENTS FOR I* BASED MODELS TO DETECT PROACTIVITY IN DASHBOARDS

Alain Pérez-Acosta, Ing.
Instituto Superior Politécnico José Antonio Echeverría (Cujae)
La Habana, Cuba
aperezac@ceis.cujae.edu.cu

Mailyn Moreno-Espino, PhD
Instituto Superior Politécnico José Antonio Echeverría (Cujae)
La Habana, Cuba,
my@ceis.cujae.edu.cu

(Recibido el 10-02-2014. Aprobado el 20-05-2014)

Resumen. El objetivo del trabajo es presentar modelos para la captura de los requisitos de un dashboard para detectar un comportamiento proactivo. Estos modelos siguen un enfoque orientado hacia metas y fueron creados con el marco de trabajo i*, que toma como base las premisas del modelado social. Para detectar el comportamiento proactivo se usaron patrones basados en modelos de i* para detectar proactividad en la etapa de requisitos de un sistema de software. Los modelos que se obtienen como resultado del trabajo tienen representados los actores, metas, intenciones, tareas y recursos que se necesitan para modelar los requisitos de un dashboard con un comportamiento proactivo y pueden, además, ser utilizados en distintos contextos de negocio.

Palabras clave: Dashboards; i*; Modelos de requisitos.

Abstract. This paper aims to present models for the capture of the requirements of a dashboard that allow detecting a proactive behavior. These models follow a goals oriented approach and were created using the framework i *, which is based on the premises of the social modeling. In order to detect the proactive behavior patterns based on models of i * that allow detecting proactivity in the stage of requirements of a system of software were used. In the models obtained as a result of the paper were represented the actors, goals, intentions, tasks and resources necessary to model the requirements of a dashboard with a proactive behavior, and these models can be used in addition in different contexts of business.

Keywords: Dashboards; i*; Requirements models.

Citación de artículo, estilo IEEE:
A. Pérez-Acosta, M. Moreno-Espino, "Modelos de Requisitos Basados en I* para Detectar Proactividad en Dashboards", Lámpsakos, N° 12, pp. 101-109, 2014

1. INTRODUCCIÓN

Los tableros de control (*dashboards*) son actualmente de los sistemas más usados en la Inteligencia de Negocios (BI de *Business Intelligence*) para medir el desempeño de las organizaciones [1]. La meta principal de un *dashboard* es proporcionar información relevante para los directivos acerca del estado de la organización y apoyar la toma de decisiones [1].

Para proporcionar la información relevante de una organización, los *dashboards* visualizan los Indicadores Claves de Rendimiento de la misma (KPIs por sus siglas en Inglés de *Key Performance Indicators*) [2], [3] y utilizan diferentes modelos para identificar, implementar y monitorear métricas para todos los niveles de la organización [3]. Según [1] el mayor reto que existe cuando se construye un *dashboard* es hallar la información que se le debe presentar al usuario para apoyar el proceso de toma de decisiones, o sea, los KPIs. Esto se da porque los KPIs que se muestran en el *dashboard* son estrictamente dependientes de las fuentes de datos y recursos de la organización, así como de las metas particulares de cada usuario. Por tanto, para identificar correctamente estas métricas, se debe tener un profundo conocimiento de la organización, de las metas individuales de cada usuario y de las relaciones que existan entre ellos, así como de los recursos disponibles [1], [2].

El diseño de un *dashboard* involucra múltiples actores, incluidos los usuarios finales y los analistas del negocio [4]. Si en la etapa de requisitos no se tienen en cuenta los diferentes actores que intervienen, las metas y tareas de cada uno, sus relaciones y los recursos que se necesitan para cumplir una determinada meta o tarea, se corre el riesgo de que el *dashboard* no cumpla con los propósitos deseados. Por tanto, esta etapa debe estar basada en un enfoque orientado a metas, puesto que un *dashboard* proporciona la información necesaria para apoyar la toma de decisiones de los directivos en una organización y, por consiguiente, cumplir las metas estratégicas de la misma [5]. De manera general, en la revisión bibliográfica analizada, en el diseño de un *dashboard* no se modelan los requisitos con un enfoque orientado a metas ni se modelan las relaciones de dependencia entre los actores que intervienen. En [3] se propone un marco de trabajo (*framework)* para el diseño de un *dashboard* con base en un conjunto de principios formulados a partir de un proyecto de caso. A pesar de los principios que se exponen en el *framework*, no existe un análisis orientado a metas ni se proporcionan elementos para modelar las dependencias entre los actores que intervienen en los requisitos del *dashboard* ni las tareas o recursos que se necesitan. Tampoco en [1], [2], [6], [7] se encontraron elementos para modelar los requisitos de un *dashboard* ni los actores que intervienen en el diseño, más bien, el eje en estos trabajos son los KPIs y el diseño de la interfaz visual.

También, en el contexto de requisitos de un *dashboard,* es preciso incorporar un concepto adicional: la proactividad. En la proactividad se trata de tomar el control para hacer que las cosas sucedan y no esperar que ocurran eventualmente [8]. Un *software* con comportamiento proactivo en áreas como la gestión, la toma de decisiones [9], [10], los ambientes asistidos [11], [12] y los sistemas en tiempo real [13] representa un beneficio para el usuario final que lo utiliza. Los *dashboards*, por constituir sistemas que apoyan la toma de decisiones, deben incluir un comportamiento proactivo. En este sentido, se ha investigado sobre trabajos [14] que proponen patrones para detectar proactividad en sistemas informáticos desde la etapa de requisitos, pero ninguno ha abarcado el contexto de los *dashboards* en BI.

El objetivo de este trabajo es presentar modelos para la captura de requisitos de un *dashboard* que tengan incorporados elementos para detectar un comportamiento proactivo. Se usó el *framework* y lenguaje de modelado i* [15], [16] pues proporciona los artefactos para representar actores y sus dependencias, así como estructurar las metas que la organización pretende lograr [17]. Este *framework* toma como premisa el enfoque del modelado social [15], [16] para la captura de requisitos de un sistema de *software* [14]. Los modelos que se obtienen como resultado del trabajo pueden ser usados cuando se modelen los requisitos de un *dashboard* para diferentes contextos de negocio.

2. DESARROLLO

2.1. Dashboards

Según se plantea en [7] no existe una definición clara del concepto de *dashboard*, puesto que los vendedores de este tipo de *software* definen los *dashboards* desde una perspectiva de lo que el producto ofrece. Few en [6] define un *dashboard* como una representación visual de la información más importante

para lograr uno o más objetivos, consolidada en una pantalla de manera que pueda ser monitoreada de un vistazo. Una definición más exacta del concepto se expone en [7]: "Es una herramienta visual e interactiva de función administrativa que muestra en una pantalla la información más relevante para lograr uno o varios objetivos individuales o empresariales, permitiendo a los usuarios identificar, explorar y comunicar áreas de problema que necesitan acción correctiva". Aunque en este trabajo se toma como concepto el expuesto en [7], no se limita el concepto de *dashboard* al contexto empresarial.

2.2. Proactividad

Viktor Frankl en [18] plantea que la "Proactividad es una actitud en la que el sujeto asume el pleno control de su conducta vital de modo activo, lo que implica la toma de iniciativas en el desarrollo de acciones creativas y audaces para generar mejoras".

La proactividad en el contexto informático es cuando un *software* es capaz de exhibir un comportamiento dirigido a metas y toma la iniciativa con el fin de satisfacer sus metas de diseño [19].

2.3. Modelado social y lenguaje i*

La ingeniería de requisitos es un área de la ingeniería de software que va más allá de definir la funcionalidad esperada del sistema de *software* a desarrollar, puesto que establece la relación entre esta funcionalidad y los procesos de negocio de la empresa. La ingeniería de requisitos facilita el mecanismo apropiado para comprender lo que quiere el cliente, analizar necesidades, confirmar su viabilidad y negociar una solución razonable [20], [21].

La captura de requisitos es la actividad mediante la que el equipo de desarrollo de un sistema de *software* extrae, de cualquier fuente de información disponible, las necesidades que debe cubrir dicho sistema [22]. La captura de requisitos incluye varias actividades en las que la interacción con los clientes (*stakeholders*) tiene una importancia primordial [23].

En la ingeniería de software y de sistemas de información la construcción de modelos ha girado en torno a las relaciones estáticas y las propiedades dinámicas y de su comportamiento [16]. Este enfoque es obvio puesto que los modelos conceptuales, al final, se traducen en los datos y las operaciones que ejecutará la computadora. Sin embargo, un sistema para tener éxito debe funcionar dentro del contexto de su entorno [15].

Si se adopta una visión social del mundo, se puede ver que en éste existe una intencionalidad. La intencionalidad la originan los actores, como los seres humanos. Los actores intencionales tienen necesidades y deseos y realizan acciones para satisfacerlos. Los actores pueden elegir cuáles acciones tomar, lo que los hace autónomos. Los actores no existen de forma aislada, existen en algún entorno y comparten e interactúan con otros [15].

El modelado social, por enfocarse en la etapa temprana de la ingeniería de requisitos, se centra en la dimensión social de los sistemas informáticos y su entorno. En un enfoque social, los intereses estratégicos de los actores deben ser usados para guiar la búsqueda de concepciones alternativas para el nuevo sistema. Cada actor debe proponer sus intereses estratégicos [15]. El modelado social ve la ingeniería de requisitos de una forma orientada a metas. Un análisis de metas revela deseos para identificar conflictos o expectativas. Un modelo orientado a metas puede ayudar a gestionar cambios. Las metas proporcionan criterios y guías para producir y evaluar posibles soluciones [24].

El lenguaje de modelado i* introduce aspectos del modelado social y del razonamiento sobre los métodos de ingeniería de sistemas de información, especialmente en los requisitos [15]. i* reconoce la primacía de los actores sociales, quienes son vistos como intencionales, es decir, tienen metas, creencias, habilidades y compromisos. El análisis se enfoca hacia la captura de las metas de los distintos actores por la configuración de las relaciones entre los actores humanos y el sistema. La configuración de estas relaciones puede ayudar a plasmar los intereses estratégicos de los actores [16].

En i* se propone dividir la captura de requisitos en dos fases: Análisis de requisitos tempranos y Análisis de requisitos tardíos. La primera fase consiste en la identificación y análisis de los principales actores del dominio involucrado en el problema y de sus necesidades e intenciones. La segunda trata de modelar lo más claro posible "qué" debe hacer el futuro sistema [16]. Además i* hace uso de dos modelos, cada uno con un nivel diferente de abstracción: el nivel intencional, representado por el Modelo de Dependencia Estratégica (*Strategic Dependece Model*,

SD) y el nivel racional, representado por el Modelo Estratégico de Racionalidad (*Strategic Rational Model*, SR) [16]. En el modelo SD se representan los actores y las relaciones entre ellos, mientras que en el modelo SR se representan las dependencias entre los objetos dentro de un actor.

Tanto el modelado social como i* pueden aplicarse en áreas como los procesos de negocio de una organización [15]. El uso de modelos para describir y analizar los procesos de negocios de una organización permite comprender el negocio antes de considerar alguna solución tecnológica. Dentro de las características que incluyen la mayoría de las técnicas de modelado de este tipo se pueden mencionar los flujos de información y los diagramas de actividades [15]. Aunque estos modelos son fáciles de comprender y validar por los *stakeholders* y proporcionan un buen punto de partida para soluciones tecnológicas, no describen el por qué de cada cosa. Los modelos de i* pueden ser usados para relacionar los procesos de negocios con metas de negocios y proporcionan una visión más social de los procesos de negocio [15].

2.4. Patrones basados en modelos de i*

Para detectar la proactividad se necesita conocer las causas de las dependencias entre los actores (los por qué), es decir, la intencionalidad de las relaciones. A partir de los modelos de i* que se obtienen en la etapa de Requisitos tempranos, es posible detectar metas que se pueden complementar de forma proactiva [14]. Según Yu [25] la dependencia es intencional si el objeto del que se depende (*dependum*) está relacionado de alguna manera con la meta o deseo del actor que depende (depender). Las intenciones se pueden delegar al *software* y de esta forma que el mismo trabaje en pos de lograr metas e intenciones, lo que conlleva un comportamiento proactivo [14].

En [14] se presentan varios patrones de requisitos que utilizan como base los modelos de i* en las etapas de Requisitos tempranos y Requisitos tardíos. Todos estos patrones tratan con un problema común: actores en los requisitos tempranos con intenciones que denotan un comportamiento proactivo. También se describe la solución para delegar estas intenciones en el *software* que se desarrollará en los Requisitos tardíos.

Tabla 1. Actores sociales que intervienen en el diseño de un dashboard

Actor	Descripción
Stakeholder	Representa el cliente final que usará el *dashboard*. En caso de existir más de una persona que utilizará el *dashboard*, se puede cambiar el estereotipo de actor por el de rol.
Analista de negocio	Este actor representa la persona encargada de obtener los requisitos del *dashboard* y los KPIs que serán visualizados.
Diseñador de dashboard	Este actor representa la persona encargada de diseñar la interfaz visual del *dashboard* cuando se tengan los requisitos y los KPIs.

3. RESULTADOS Y DISCUSIÓN

En la Figura 1 se presentan los modelos obtenidos en la etapa de Requisitos tempranos.

En este modelo se representan tres actores sociales que se describen en la Tabla 1.

En la Figura 1 se ha representado en el modelo SR del actor Stakeholder una jerarquía de metas. El actor Stakeholder tiene una (o varias) meta(s) estratégica(s), que se corresponde(n) con los intereses estratégicos de la organización. Las metas estratégicas se descomponen en varias metas de decisión, que se encargan de responder cómo lograr las metas estratégicas. Cada meta de decisión se descompone en metas de información, que se encargan de responder cómo lograr las metas de decisión. Todo este enfoque sigue la idea que se expone en [17] para la creación de Almacenes de Datos (*DataWarehouses*) [26] a partir de un análisis orientado a metas. En este trabajo se ha adaptado parte de la idea expuesta en [17] para el diseño de un *dashboard*. Para cumplir las metas de información se desarrollan planes que constituyen los medios para lograr la meta “Identificar los requisitos funcionales del dashboard”. Los requisitos no funcionales, por tratarse de restricciones de calidad que debe cumplir el sistema, se modelan como un recurso que pertenece al actor Stakeholder. La meta suave “Mantenerse informado sobre el estado del negocio” constituye una intención del Stakeholder que contribuye de forma positiva en el logro de las metas estratégicas y que permitirá denotar un comportamiento proactivo en los Requisitos tardíos.

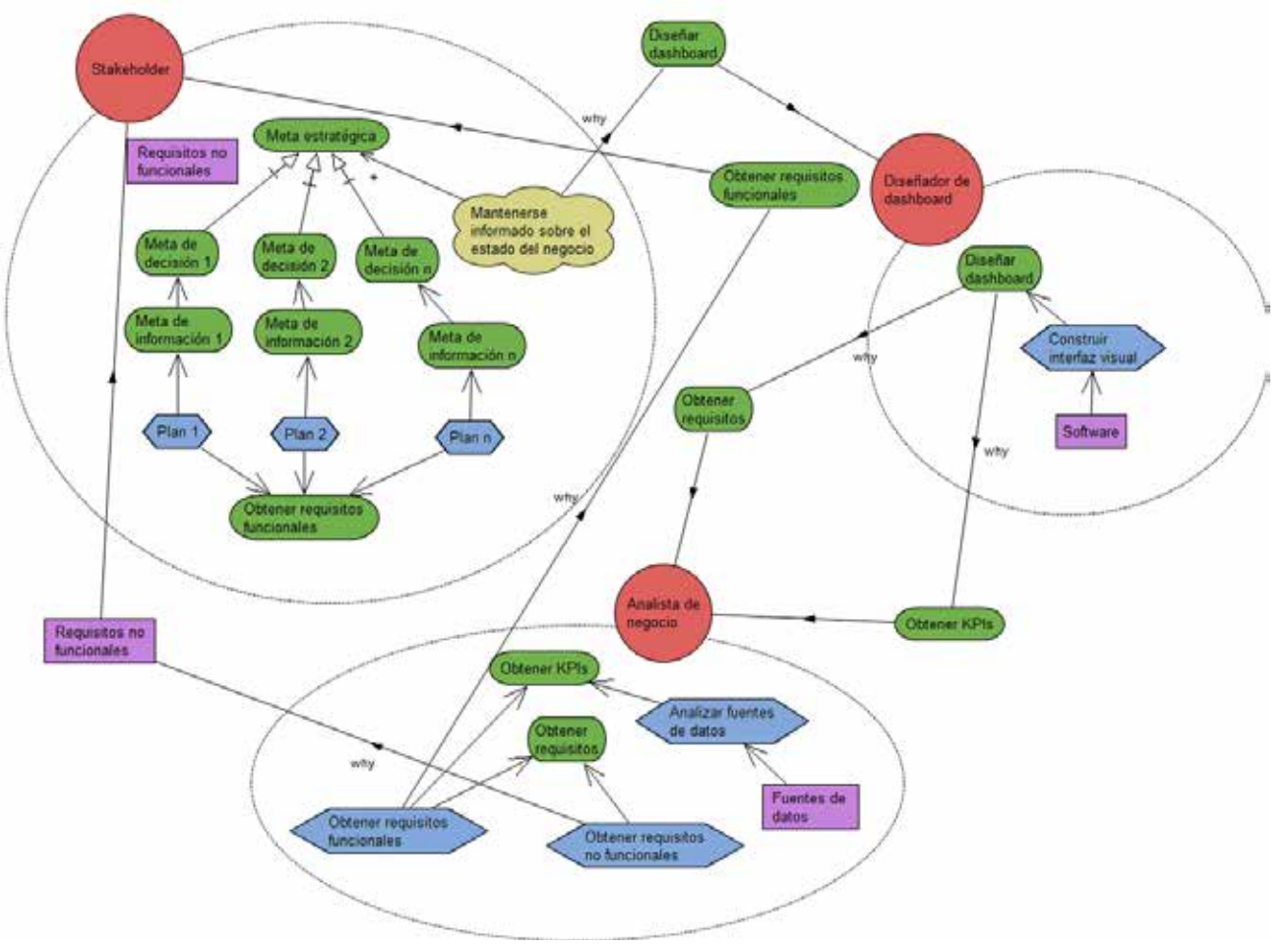

Figura 1. Modelo de Requisitos tempranos en i*

Las metas del actor Analista de negocio son “Obtener KPIs” y “Obtener requisitos”. La meta “Obtener requisitos” se descompone en dos planes: “Obtener requisitos funcionales” y “Obtener requisitos no funcionales”. El plan “Obtener requisitos funcionales” constituye el por qué se necesita la meta “Identificar requisitos funcionales” del actor Stakeholder; mientras que el plan “Obtener requisitos no funcionales” constituye el por qué se necesita el recurso “Requisitos no funcionales” del actor Stakeholder. Se mencionó anteriormente que los KPIs dependen de las fuentes de datos de la organización, por lo que en el modelo SR del actor Analista de negocio se representa el plan “Analizar fuentes de datos” como un medio para lograr cumplir la meta “Obtener KPIs”. El recurso “Fuente de datos” (que puede ser una base de datos relacional, un fichero, un *DataWarehouse*, etc.) representa un medio para lograr el plan “Analizar fuentes de datos”.

La meta fundamental del diseñador de dashboard es “Diseñar dashboard”, por ello necesita que el actor Analista de negocio cumpla las metas “Obtener KPIs” y “Obtener requisitos”. El plan “Construir interfaz visual” representa un medio para lograr cumplir la meta “Diseñar dashboard”. Para realizar el plan “Construir interfaz visual” se necesita de un sistema de software, que se modeló como un recurso. La intención “Mantenerse informado sobre el estado del negocio” perteneciente al Stakeholder constituye el por qué se hace necesaria la meta “Diseñar dashboard”.

En la Figura 2 se presentan los modelos obtenidos en la etapa de Requisitos tardíos.

En la Figura 2 se modelaron dos actores sociales: el Stakeholder (con el mismo concepto que en la etapa de Requisitos tempranos) y un actor Dashboard, representado con el estereotipo de Sistema que pro-

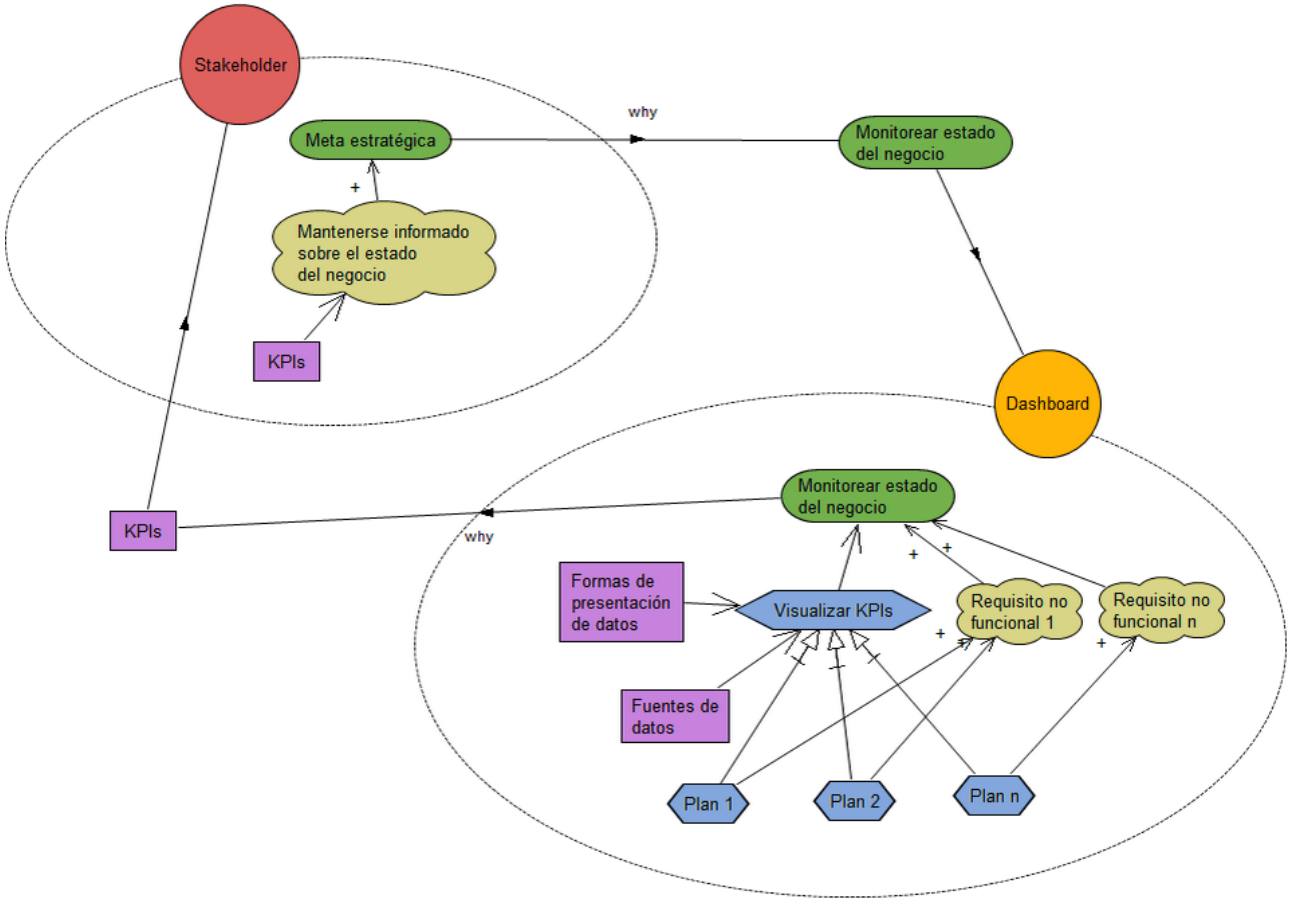

Figura 2. Modelo de Requisitos tardíos en i*

pone i*. En el modelo SR del actor Stakeholder no aparece la jerarquía de metas propuesta en los Requisitos tempranos porque solamente proporcionaba una forma para extraer los requisitos funcionales del *dashboard*, y luego los KPIs que respondían a esos requisitos, alineados con las metas estratégicas del Stakeholder. El recurso "KPIs" constituye un medio para complementar la meta suave "Mantenerse informado sobre el estado del negocio".

En el modelo SR del actor Dashboard se ha modelado la meta "Monitorear estado del negocio", necesaria para lograr la meta estratégica del actor Stakeholder. Los planes que estaban representados en los Requisitos tempranos como parte del modelo SR del Stakeholder y que constituían la base para extraer los requisitos funcionales del *dashboard*, se modelan en esta etapa como planes del actor Dashboard que deben ser implementados y constituyen los medios para cumplimentar el plan "Visualizar KPIs", que a su vez es el medio para cumplir la meta "Monitorear estado del negocio". El recurso "Fuente de datos" se incorpora también al modelo SR del actor Dashboard como medio para cumplir el plan "Visualizar KPIs". Además, se agrega el recurso "Forma de presentación de datos" (que pueden ser tablas, componentes gráficos, etc.) como medio para cumplir el plan "Visualizar KPIs". Los requisitos no funcionales del *dashboard* se representan como metas suaves y además se modela cómo influyen estos requisitos sobre la meta fuerte. También se modelan cómo contribuyen los planes en los que se descompone "Visualizar KPIs" en las metas suaves. Todo ello permite evaluar alternativas de modelos antes de considerar alguna solución tecnológica.

Los modelos presentados en la Figura 2 no permiten detectar un comportamiento proactivo en el *dashboard*, pues la intención "Mantenerse informado sobre el estado del negocio" permanece en el modelo

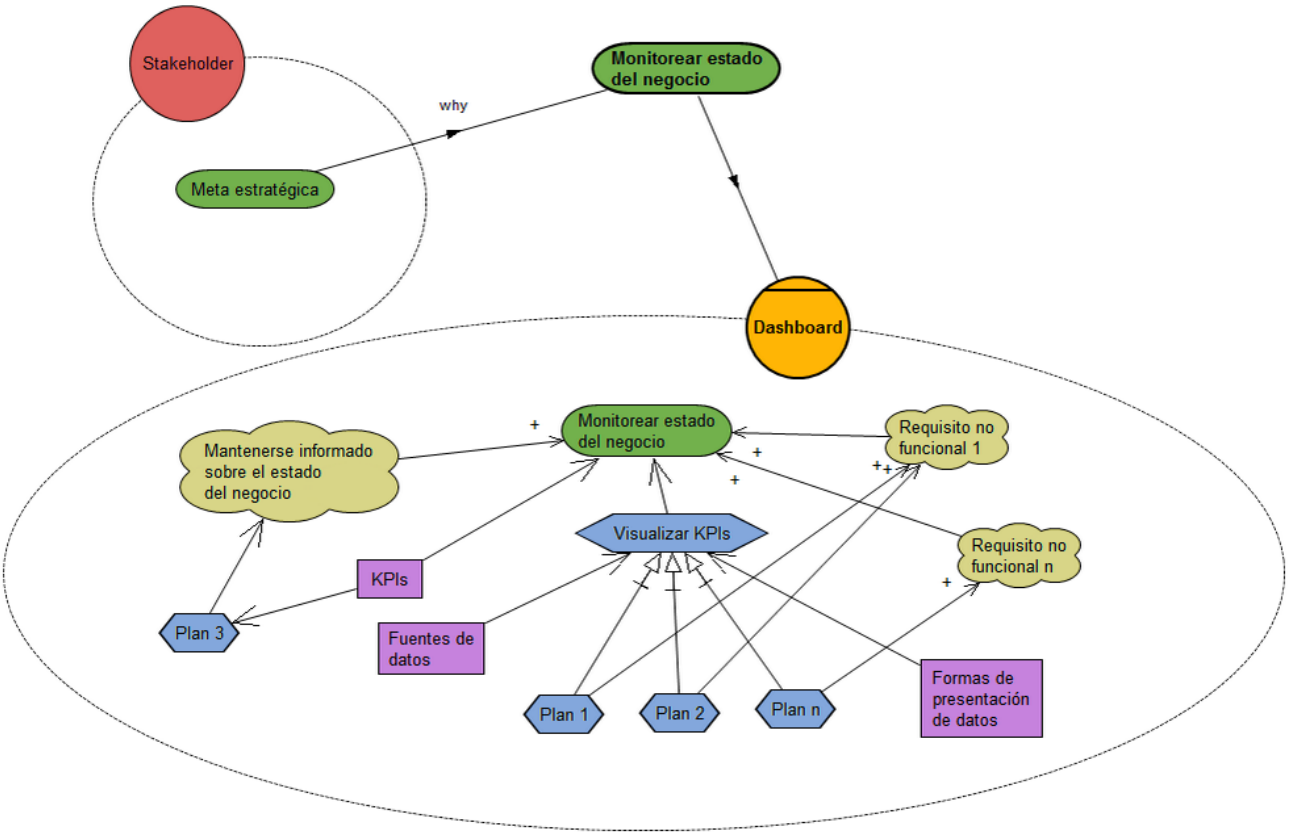

Figura 3. Modelo de Requisitos tardíos en i* con el uso del patrón.

SR del actor Stakeholder. Por tanto, para modelar un comportamiento proactivo se utilizó el patrón Resource why Dependency, descrito en [14], debido a las características que presentaban los modelos. La entrada de este patrón lo constituyen los modelos de la Figura 2, por lo que en la Figura 3 se muestran los modelos de Requisitos tardíos modificados con un comportamiento proactivo.

En la Figura 3, la intención "Mantenerse informado sobre el estado del negocio" se le delega al actor Dashboard (representado esta vez con el estereotipo de agente [27] propuesto por i*) lo que denota un comportamiento proactivo. Para cumplir esa intención se hace necesario implementar el plan "Plan 3", en el que el recurso "KPIs" constituye un medio para lograr dicho plan y también la meta "Monitorear estado del negocio". El resto del modelo SR del actor Dashboard permanece igual a como se muestra en la Figura 2.

Se hace necesario señalar que los modelos propuestos pueden contener menos o más actores, elementos y relaciones que las expuestas en este trabajo, en dependencia del contexto en el que se apliquen y de lo que se desee modelar. En ese caso, habrá que valorar otros patrones, además del Resource why Dependency si se desea modelar un comportamiento proactivo en el *dashboard*.

4. TRABAJOS FUTUROS

Se pretende desarrollar un *dashboard* con comportamiento proactivo que complemente el Sistema de soporte a la toma de decisiones del proceso docente de una universidad cubana. Los modelos propuestos servirán para capturar los requisitos del *dashboard* con un enfoque orientado a metas.

5. CONCLUSIONES

Los modelos obtenidos como resultado de este trabajo permiten capturar los requisitos de un *dashboard* con un enfoque orientado a metas, se usó el *framework* i* y las bases del modelado social, lo que permite detectar un comportamiento proactivo desde la etapa de requisitos.

Con la jerarquía de metas representadas en el actor *Stakeholder*, es posible obtener los requisitos funcionales del dashboard a partir de las metas estratégicas del actor para lograr una independencia entre los modelos y el contexto de negocio (empresarial y docente) en el que se apliquen. No obstante, estos modelos no tienen por qué contener todos los elementos o actores existentes en cada organización, pues esto depende de las características propias de cada entorno. Se sugiere utilizar los modelos como plantillas y modificarlos en dependencia de lo que se desee modelar.

REFERENCIAS

[1]. Zagorecki, A., et al., *Executive Dashboard Systems for Emergency Management.* Communications, 2012. **4**(2): p. 82-89.

[2]. Gröger, C., et al., *The Operational Process Dashboard for Manufacturing.* 2013: p. 6.

[3]. Lempinen, H., *Constructing a Design Framework for Performance Dashboards*, in *Nordic Contributions in IS Research*, C. Keller, et al., Editors. 2012, Springer Berlin Heidelberg. p. 109-130.

[4]. Elias, M. and A. Bezerianos, *Exploration Views: Understanding Dashboard Creation and Customization for Visualization Novices*, in *Human-Computer Interaction – Interact 2011*, P. Campos, et al., Editors. 2011, Springer Berlin Heidelberg. p. 274-291.

[5]. Eckerson, W.W., *What Are Performance Dashboards?*, in *Performance Dashboards: Measuring, Monitoring, and Managing Your*

Business, John Wiley & Sons.

[6]. Few, S., *Information Dashboard Design: The Effective Visual Communication of Data* 2006: O'Reilly.

[7]. Yigitbasioglu, O.M. and O. Velcu, *A review of dashboards in performance management: Implications for design and research.* International Journal of Accounting Information Systems, 2011. **13**: p. 19.

[8]. Grant, A.M. and S.J. Ashford, *The dynamics of proactivity at work.* Research in Organizational Behavior, 2008. **28**(-): p. 3-34.

[9]. Li, J. and Y. Feng, *Construction of Multi-Agent System for Decision Support of Online Shopping*, in *Emerging Research in Artificial Intelligence and Computational Intelligence*, H. Deng, et al., Editors. 2011, Springer Berlin Heidelberg. p. 318-324.

[10]. Srinivasan, S., J. Singh, and V. Kumar, *Multi-agent based decision Support System using Data Mining and Case Based Reasoning.* International Journal of Computer Science Issues, 2011. **8**(4): p. 340-349.

[11]. Lindgren, H., D. Surie, and I. Nilsson, *Agent-Supported Assessment for Adaptive and Personalized Ambient Assisted Livi ng*, in *Trends in Practical Applications of Agents and Multiagent Systems*, J. Corchado, et al., Editors. 2011, Springer Berlin Heidelberg. p. 25-32.

[12]. McNaull, J., et al., *Multi-agent System Feedback and Support for Ambient Assisted Living*, in *8th International Conference on Intelligent Environments*. 2012: Guanajuato p. 319 - 322.

[13]. Ghorbani, J., M.A. Choudhry, and A. Feliachi, *Real-time multi agent system modeling for fault detection in power distribution systems* in *North American Power Symposium*. 2012: Champaign. p. 1-6.

[14]. Moreno, M., *Patrones para incorporar proactividad en sistemas informáticos*. 2013, Instituto Superior Politécnico José Antonio Echeverría: La Habana. p. 172.

[15]. Yu, E., *Social Modeling and i**, in *Conceptual Modeling: Foundations and Applications*, T.B. Alexander, et al., Editors. 2009, Springer-Verlag. p. 99-121.

[16]. Yu, E., et al., *Social Modeling for Requirements Engineering*. 1st ed. 2011, Cambridge: The MIT Press. 760.

[17].Mazón, J.N., J. Pardillo, and J. Trujillo, *A Model-Driven Goal-Oriented Requirement Engineering Approach for Data Warehouses*, in *Advances in Conceptual Modeling – Foundations and Applications*, J.-L. Hainaut, et al., Editors. 2007, Springer Berlin Heidelberg: Berlin. p. 255-264.

[18].Frankl, V., *Man's Search for Meaning*. 2006: Beacon Press.

[19].Jennings, N.R., *An agent-based approach for building complex software systems.* Communications of the ACM, 2001. **44**(4): p. 35-41.

[20].Sommerville, I., *Integrated Requirements Engineering: A Tutorial.* IEEE Software, 2005. **22**(1): p. 16-23.

[21].Sommerville, I. and P. Sawyer, *Requirements Engineering: A Good Practice Guide*. 1st ed. 1997: John Wiley & Sons. 404.

[22].Jacobson, I., G. Booch, and J. Rumbaugh, *The Unified Software Development Process*. reprint ed. 2012: Prentice Hall.

[23].Fuentes-Fernández, R., J. Gómez-Sánz, and J. Pavón, *Understanding the human context in requirements elicitation.* Requirements Engineering, 2010. **15**(3): p. 267-283.

[24].Horkoff, J. and E. Yu, *Comparison and evaluation of goal-oriented satisfaction analysis techniques.* Requirements Engineering, 2013. **18**(3): p. 199-222.

[25].Yu, E., *Modelling Strategic Relationships for Process Reengineering*, in *Department of Computer Science*. 1995, University of Toronto: Toronto, Canadá.

[26].Inmon, W.H., *Building the Data Warehouse*. 4th ed. 2005, Chichester: Wiley.

[27].Burg, B., *Foundation for Intelligent Physical Agents*. 2004.