Análisis de Variabilidad con Modelos de Objetivos

Bruno González-Baixauli[1], Miguel A. Laguna[1], Julio Cesar Sampaio do Prado Leite[2]

[1]*Departamento de Informática. Universidad de Valladolid*
{bbaixauli, mlaguna}@infor.uva.es
[2]*Departamento de Informática. PUC do Rio de Janeiro*
www.inf.puc-rio.br/~julio/

Abstract. Dentro del campo de la variabilidad, o la habilidad de cambio o personalización de un sistema, no es común que dicha variabilidad se tenga en cuenta en la definición de requisitos. En las propuestas donde si se contemplan se utilizan directamente *features*, aunque no son una técnica de elicitación de requisitos, sino que más bien están a medio camino entre los requisitos y la definición de la arquitectura. También existen propuestas que usan casos de usos, más cercanos a la elicitación de requisitos, pero que tienen problemas para modelar la variabilidad. En este artículo se propone aplicar técnicas de la Ingeniería de Requisitos (IR) al estudio de variabilidad utilizando un enfoque basado en modelos de objetivos, escenarios (incluyendo casos de uso) y *features*, para la especificación y análisis de requisitos de sistemas con variabilidad. De esta manera, se espera mejorar la fase de requisitos en este tipo de sistemas. Además, los modelos de objetivos proporcionan un mecanismo de decisión de alto nivel (los objetivos del usuario) y permiten incorporar los requisitos no funcionales (RNF).

1. Introducción

Durante los últimos años los sistemas software cada vez tienden a soportar una mayor variabilidad. Esta variabilidad se debe principalmente a dos causas: a las demandas de los usuarios de sistemas más adaptables a sus necesidades (sistemas altamente configurables) y a la presión del mercado, que hace que los productos software se agrupen en familias para poder reutilizar la mayor parte posible de sus artefactos software (familias de productos software [14]). De esta manera, el campo de la variabilidad está obteniendo una creciente importancia. En cuanto a las técnicas utilizadas, el principal esfuerzo recae en encontrar las partes comunes y variables dentro de los sistemas.

Dentro de la Ingeniería de Requisitos, una rama de la Ingeniería del Software, los objetivos (*goals*) aparecen como una de las propuestas de modelado para mejorar el proceso de definición de requisitos. Los objetivos proporcionan una visión intencional con varias ventajas respecto a otros mecanismos como su mayor estabilidad [17] (es más difícil que cambien las razones por las que se necesita un sistema, que la forma en que se realizan sus funciones) o la posibilidad de analizar distintas alternativas y seleccionar la mejor a partir de los requisitos no funcionales [3]. Además, este análisis

puede ser realizado adaptando técnicas de Inteligencia Artificial, mediante un razonamiento formal sobre los modelos de objetivos [5].

Sin embargo, el principal esfuerzo dentro del estudio de variabilidad ha sido desde el diseño e implementación del sistema, dando poca importancia a la parte de requisitos. Así, la mayor parte de las propuestas no mencionan a los requisitos o hablan genéricamente de una fase de especificación de requisitos. Otras propuestas utilizan modelos de características o *features* [10], pero estas técnicas están demasiado orientadas hacia el diseño y suelen ser difíciles de aplicar directamente. También existen propuestas para utilizar casos de uso [9][8], aunque fallan en la presentación de la variabilidad al estar contenida en diversos documentos (en cada caso de uso) y en que no dan el punto de vista intencional del sistema.

Por estas razones, en este artículo se propone introducir técnicas de la IR en el estudio de variabilidad. En este sentido, se utilizan los modelos de objetivos, puesto que es sencillo aplicar el estudio de alternativas a la variabilidad del sistema y permiten mejorar el estudio de los requisitos no funcionales. Pero también proponemos utilizar escenarios para refinar los objetivos y para encontrar las *features* del sistema, que utilizaremos como punto de inicio hacia la obtención de la arquitectura en las subsiguientes fases.

El artículo está organizado como sigue: primero se introducen los conceptos y elementos de la IR orientada a objetivos y, en el siguiente capítulo, los de la variabilidad. En la sección 4 se describe nuestra propuesta para la elicitación de requisitos en entornos con variabilidad basada en tres vistas: intencional, operacional y funcional. En la siguiente sección se muestra como dicho enfoque es útil también para mejorar el estudio de variantes. La sección 6 presenta algunas experiencias previas a partir de una herramienta que da soporte a una primera versión del análisis y los cambios necesarios para actualizar al nuevo enfoque. Por último, se termina con las conclusiones y trabajo futuro.

2. Ingeniería de Requisitos Orientada a Objetivos

La Ingeniería de Requisitos orientada a objetivos (*goal-oriented requirement engineering*) es un enfoque que promueve la utilización de objetivos como base para obtener los requisitos, incorporando de esta forma un punto de vista intencional, es decir, el propósito del sistema. Introducir un punto de vista intencional permite que los interesados expresen sus necesidades de una manera más natural, centrándose en lo que quieren (sus objetivos) frente a la manera de alcanzarlos (los requisitos convencionales). A partir de los objetivos, los requisitos se pueden derivar como maneras de alcanzar esos objetivos.

Los objetivos se suelen estructurar en árboles And-Or descomponiéndose en obligatorios u optativos. Así, es posible estudiar alternativas en los requisitos y verificar la completitud de un conjunto de requisitos con respecto a los objetivos planteados. Además este estudio puede hacerse de manera formal si los objetivos están suficientemente formalizados. Por tanto, los objetivos ayudan a identificar, justificar y organizar los requisitos. Otras ventajas de los objetivos son que los requisitos resultantes son más independientes de la tecnología, que el tratamiento de

requisitos no funcionales (RNF) y conflictos es más natural y que guían el desarrollo software permitiendo trazar los artefactos software hasta sus objetivos finales [17].

Existen diversos enfoques dentro de la IR orientada a objetivos, los cuales distan de ser homogéneos, si bien los conceptos básicos que utilizan son similares. Los enfoques se diferencian básicamente en dos factores: la actividad de IR a la que se enfocan (elicitación, modelado o análisis de requisitos) y el grado de formalismo que soportan. Por ejemplo, la propuesta de KAOS (*Knowledge Acquisition in Automated Specification of Software*) [4] está enfocada al modelado de requisitos con un grado de formalismo, lo que le permite realizar un proceso de análisis formal de los requisitos. En cambio, el *NFR Framework* [3], que también está enfocado al modelado, se centra en los RNF, y utiliza un enfoque cualitativo menos formal. Un enfoque que se basa en el *NFR Framework* es i* [16], centrado en la elicitación de requisitos en las fases iniciales de la IR, es decir en el modelado de negocio. Otras propuestas buscan integrar otras técnicas como los escenarios en el modelado de objetivos como en GBRAM (*Goal Based Requirements Analysis Method*) [1]. Su principal interés es que define un proceso en el cual los escenarios se crean a partir de los objetivos y, a su vez ayudan a descubrir objetivos ocultos.

En general, las distintas propuestas utilizan un mismo conjunto de elementos de análisis que se pueden resumir en:

- **Goal:** objetivo de alto nivel del negocio, de la organización, del sistema o, incluso, de los *stakeholders*. Capturan la finalidad o propósito del sistema, y guían el proceso de decisión entre las alternativas para su realización durante el proceso de desarrollo del software. Existen dos tipos diferenciados:
  - **Hardgoal:** objetivo funcional del sistema. Se caracteriza porque es posible verificar, de manera clara, si se ha cumplido el objetivo.
  - **Softgoal:** objetivo cuya satisfacción o cumplimiento no puede ser establecido de manera absoluta, es decir, que en algunos casos solamente se podrá afirmar que se ha cumplido (o incumplido) de manera parcial. Este tipo de objetivo se define en el *NFR Framework* para tratar los RNF.
- **Tarea:** manera de alcanzar un objetivo. Puede ser de bajo nivel como un algoritmo o un procedimiento, o de un mayor nivel como un dispositivo, entorno o GUI.
- **Agente:** entidad que existe en la organización que tiene objetivos y que puede realizar tareas para alcanzar dichos objetivos o ayudar a otros agentes a alcanzar sus objetivos. Ciertas propuestas están tan centradas en la idea del agente que se denominan RE orientada a agentes (como i*).
- **Recurso:** objeto que es necesario para realizar una tarea o para alcanzar un objetivo (que puede ser obtener ese recurso).

Para este trabajo, el principal interés de los modelos de objetivos es que permiten el estudio de alternativas y, por tanto, son fácilmente aplicables al estudio de variabilidad. De esta forma, se pasa de estudiar las alternativas para obtener la mejor, que contendrá los requisitos del sistema final, a mantener las alternativas (ahora variantes) hasta el producto final o como base para la construcción de la familia de productos. Además, proporcionan un mecanismo de decisión de alto nivel (los objetivos de los usuarios) y permiten relacionar requisitos funcionales y no funcionales utilizando las técnicas del *NFR Framework*. Esto puede ser utilizado para la selección de la funcionalidad requerida para una determinada aplicación utilizando como entrada los objetivos del usuario, tanto funcionales como no funcionales.

3. Variabilidad

La variabilidad se define como la habilidad de cambio o de personalización de un sistema [15]. Existen dos grandes grupos en los que se dividen los sistemas con variabilidad: los sistemas personalizables, donde la variabilidad se debe principalmente a la selección por parte del usuario de la parte que más le interesa; y las familias de productos [14], donde una serie de productos más o menos similares se unen para permitir la reutilización de la parte común.

En estos sistemas con variabilidad, en los requisitos se deben definir las cualidades del sistema variable o de la familia de sistemas. La principal diferencia con los sistemas tradicionales es que se debe prestar un especial interés al análisis de la parte común y de la parte variable, estableciendo las dependencias entre ellas.

Para ello, los métodos más utilizados son los basados en *features* como FODA (*Feature Oriented Domain Analysis*) [10], FORM [11] o FeatureRSEB [7]. Estas *features*, definidas como características o conceptos destacados y distintivos visibles a varios *stakeholders*, capturan la parte común y variable organizándose por medio de diagramas jerárquicos And / Or. En estos diagramas los nodos And dan la parte común y los Or la variable. De esta forma, son utilizados para definir la arquitectura de referencia del sistema y los componentes reutilizables instanciables durante el desarrollo de aplicaciones con variabilidad [11]. Las *features* se suelen clasificar en cuatro niveles: de *capabilities* (servicios, funciones y no funcionales), de tecnologías del dominio, de tecnologías de implementación y de entornos de ejecución. El problema de esta técnica es que requiere un gran conocimiento previo del dominio y que está enfocada hacia la definición de la arquitectura y no tanto a la definición de requisitos.

Otro enfoque es utilizar casos de uso, que permiten representar la variabilidad esencial, es decir, la variabilidad que interesa a los clientes [8]. Por ejemplo, Jacobson [9] utiliza el mecanismo *extend* de los casos de uso tradicionales para representar gráficamente la variabilidad, describiendo los puntos de variación (tipo, cardinalidad, etc) en la descripción del caso de uso. En cambio, Halmans y Pohl [8] proponen indicar explícitamente los puntos de variación y las distintas variantes en el modelo de casos de uso para obtener una mejor visión de la variabilidad y mejorar la comunicación con el usuario. Para ello extienden la notación de los casos de uso indicando gráficamente los puntos de variación, el tipo de variabilidad (múltiple, opcional, etc.) y la cardinalidad. En ambos casos, las técnicas están enfocadas hacia la elicitación de requisitos desde el cliente, pero falla en la representación global de la variabilidad, ya que se presenta por separado en cada uno de los casos de uso, no globalmente como en el modelo de *features*. Por otro lado, da una visión operacional, sin tener en cuenta la intencionalidad del sistema o sistemas y, por tanto, sin beneficiarse de sus ventajas.

Otro elemento de análisis dentro de la variabilidad es el análisis de variantes, en donde a partir del modelo de variabilidad se analiza cual es la variante que mejor se adapta a las necesidades de un usuario o a un determinado producto de la familia. Por lo general, este análisis se realiza de manera manual, seleccionando en cada punto de variabilidad la parte que más convenga.

4. Objetivos, Escenarios y *Features* para la Elicitación de Requisitos en Variabilidad

Debido a las carencias de las técnicas analizadas, nosotros proponemos utilizar objetivos, escenarios y *features* para la elicitación de requisitos. De esta forma, los objetivos dan una visión intencional del sistema, tanto en su parte funcional (modelo de objetivos) como en la no funcional (modelo de softgoals) y son el primer paso en la elicitación de requisitos; los escenarios permiten la operacionalización de los objetivos, proporcionando una vista operacional que describe las maneras de alcanzar los objetivos y permite su refinamiento; y, por último, a partir de las operacionalizaciones se puede obtener la funcionalidad del sistema que puede ser descrita por medio de *features*, que además proporcionan un enlace a la arquitectura del sistema. También es posible, en algunos casos, obtener los distintos modelos sin pasar por los escenarios como se muestra en la figura 1.

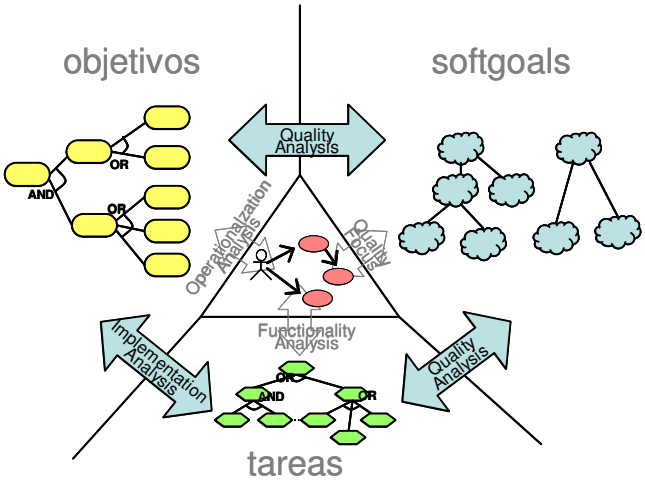

Figure 1. Relaciones entre modelos. Los objetivos son operacionalizados en escenarios, cuyas variantes están enfocadas hacia diferentes factores de calidad (softgoals). Las *tareas* implementan las acciones de los escenarios con diferentes contribuciones a las *softgoals*. También es posible encontrar los distintos elementos directamente (sin escenarios).

De esta forma, un análisis de calidad puede proporcionar nuevos softgoals a partir de los objetivos o de las tareas (*features* de bajo nivel) o viceversa (a partir de los softgoals, nuevos objetivos o tareas). También se puede llegar a nuevas tareas a partir de la descomposición de objetivos, o descubrir objetivos no considerados desde las tareas. Aunque, por lo general será necesario utilizar escenarios para obtener las tareas.

Un análisis de los distintos tipos de *features* nos lleva a la conclusión de que existen *features* de dos tipos, las de alto nivel, contenidas en el nivel de *capabilities* (servicios, operaciones y no funcionales) y las de más bajo nivel o maneras de

implementar las anteriores contenidas en los niveles de tecnologías de dominio, tecnologías de implementación y entornos de aplicación. Estos dos tipos pueden relacionarse fácilmente con los objetivos funcionales (servicios y operaciones) y no funcionales (*features* no funcionales) y con las tareas o maneras de alcanzar un objetivo a alto nivel, que pueden ir desde algoritmos (*features* de implementación o de dominio) a distintos interfaces de usuario o plataformas (*features* de entorno de aplicación). Por tanto, si damos al primer tipo de *features* un enfoque más intencional, lo que nos queda en nuestro modelo de *features* son las tareas de la orientación a objetivos y por tanto lo denominaremos modelo de tareas. Estas tareas, como maneras de alcanzar los objetivos, son las que definen las contribuciones (positivas o negativas) a los distintos objetivos no funcionales. Por otra parte, las relaciones en los modelos de *features* son ligeramente distintas a las relaciones en los modelos de objetivos, pero fácilmente convertibles. En la tabla 1 se muestra como se puede pasar de uno a otro. Ahora bien, es necesario incluir las relaciones necesita / mutuamente excluyentes en el modelo de tareas. Al no considerar estas relaciones en el modelo de objetivos estamos suponiendo que, si este tipo de relaciones se dan entre objetivos, es debido a las maneras de alcanzarlos o las tareas.

FODA		Tareas
Mandatory		
Variant		
Optional		

Table 1. Conversión entre relaciones FODA a relaciones de descomposición de Tareas (modelos de objetivos)

En cuanto a los escenarios, son utilizados para el refinamiento de los objetivos (funcionales y no funcionales) y como punto intermedio entre dichos objetivos y su implementación a alto nivel (las tareas o *features*). Existen diversos trabajos que relacionan objetivos y escenarios como [2][12] y escenarios con *features* [7].

5. Modelo de Variabilidad

Los escenarios son utilizados como soporte en el proceso de elicitación de requisitos, para describir los objetivos y obtener las tareas, pero en el modelo de variabilidad, donde se representa la variabilidad del sistema, se puede prescindir de ellos y utilizar únicamente los otros tres modelos De esta manera se separan de manera clara los distintos elementos de variabilidad: la funcionalidad a alto nivel (modelo de objetivos funcionales), las cualidades del sistema (modelo de *softgoals*) y las maneras de alcanzar la funcionalidad (modelo de tareas). Ese modelo es similar al *V-Graph model* [18], pero aquí se hace explicita la inclusión de escenarios asociados al sub-grafo de tareas.

En cuanto a los modelos, son árboles And / Or aunque con ciertas peculiaridades: el primero, el modelo de objetivos funcionales se basa en los modelos de objetivos

utilizados en [13], pero sin incluir las tareas, proporcionando la funcionalidad a alto nivel del sistema y, mediante las relaciones Or, un primer factor de variabilidad. El segundo, o modelo de *softgoals* utiliza un enfoque basado en el *NFR Framework* [3], donde los objetivos son *softgoals* que se relacionan por descomposición And / Or o mediante las correlaciones del *NFR Framework*, resumidas en la tabla 2. Por último, el modelo de tareas también utiliza las relaciones de descomposición And / Or, y por medio de las relaciones de dependencia de los modelos de *features*, es decir necesita (*require*) y mutuamente excluyentes (*exclude*).

	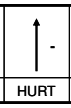	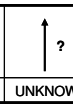		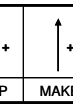
BREAK	HURT	UNKNOW	HELP	MAKE

Table 2. Enlaces de correlación / contribución, adaptado de [3]

En cuanto a las relaciones entre los modelos, los modelos de objetivos (funcionales y no funcionales) se relacionan por medio del modelo de tareas. De tal manera que los objetivos funcionales son operacionalizados por las tareas (la manera en que un objetivo se operacionaliza en tareas es el segundo factor de variabilidad) y las tareas contribuyen (positiva o negativamente) a la satisfacción de los *softgoals*. Para indicar estas contribuciones se utilizan como en el modelo de *softgoals* las correlaciones definidas por el *NFR Framework* [3]. Estas correlaciones son la base para el análisis de variantes a partir de los requisitos no funcionales. En la figura 3 se presenta el metamodelo que define los elementos de modelado y sus posibles interrelaciones.

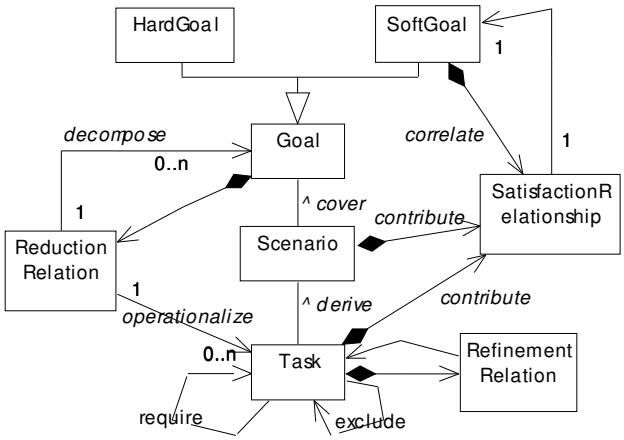

Figure 2. Metamodelo. Los objetivos (*Goal*) se clasifican en *Hardgoals* (funcionales) y *Softgoals* (no funcionales), y son reducidos (relaciones *And, Or o Xor*) en nuevos objetivos (descomposición) o en tareas (operacionalización), que a su vez se refinan en nuevas tareas (*And, Or, Xor*). Escenarios y tareas contribuyen a las *Softgoals* (relaciones *NFR Framework*), que se pueden correlacionar entre si (mismas relaciones).

Por tanto, la variabilidad es representada por un lado, por las relaciones Or en el modelo de objetivos funcionales y, por el otro, por la selección de tareas para

implementar los objetivos, que difieren en las distintas maneras que afectan a las cualidades del sistema o softgoals.

Un ejemplo simplificado de un modelo de variabilidad para el dominio de herramientas para discapacitados se presenta en la figura 3.

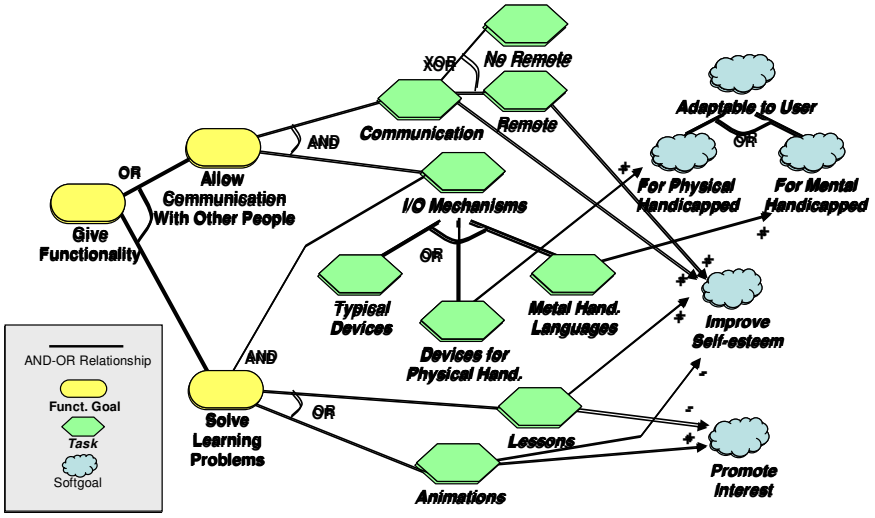

Figure 3. Objetivos (elipses), softgoals (nubes) y tareas una para parte del dominio de aplicaciones para personas discapacitadas (simplificado).

En este ejemplo, uno de los objetivos del dominio es dar cierta funcionalidad, que se descompone en permitir la comunicación con otras personas y en resolver problemas de aprendizaje. A su vez estos son implementados por distintas tareas, que contribuyen de distintas maneras a los softgoals del dominio.

6. Herramientas de Soporte

Como en todo análisis complejo, para el éxito de esta propuesta es necesario proporcionar una herramienta que de soporte a dicho análisis. En una primera aproximación [6] se ha propuesto una herramienta para el análisis visual de variantes, pero que utiliza únicamente dos modelos (el modelo de objetivos funcionales incluye el de tareas) y como metáfora, la de las tablas de decisión. De esta manera, mediante el modelo de objetivos funcional se especifican las posibles variantes (acciones de la tabla de decisión) y con el de *softgoals*, el criterio abstracto para elegir entre las variantes (condiciones). La herramienta permite introducir los dos modelos, las contribuciones de los elementos funcionales sobre los no funcionales y una serie de prioridades a partir de las cuales se presentan varias gráficas que permiten seleccionar la mejor variante utilizando varios criterios de selección, entre ellos el de la satisfacción global (una media de los softgoals según las prioridades), como se muestra en la figura 4. Para calcular dicho valor se utiliza una modificación del

algoritmo de propagación del grado de satisfacción (*satisficing*) presentado en [5]. Además, es capaz de mostrar los valores de satisfacción de cada *softgoal* para cada objetivo funcional de una variante, permitiendo la trazabilidad de dichos valores.

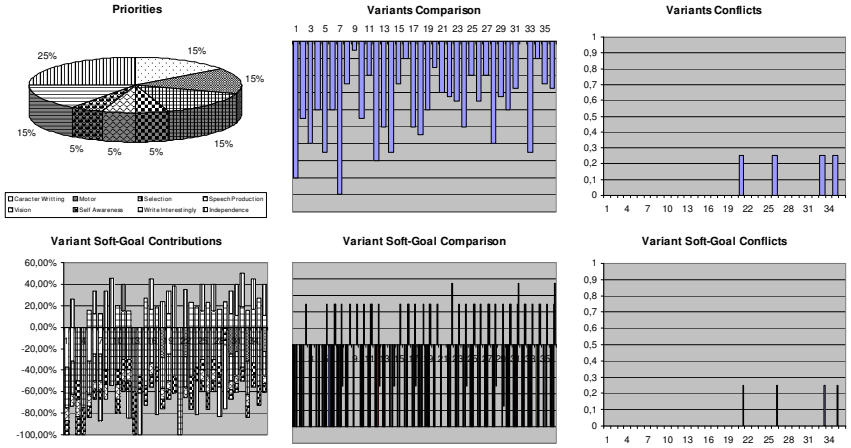

Figure 4. Ejemplo del resultado de la herramienta presentada en [6]. Se muestran varias gráficas con las prioridades seleccionadas (1), el grado de la satisfacción: global (2), contribuciones al global (4) y para cada *softgoal* (5); o los conflictos: global (3), y para cada softgoal (6).

El problema de dicha herramienta es la escalabilidad, debido al crecimiento exponencial del número de variantes. Aplicando el modelo de variabilidad presentado es posible mitigar dicho problema, ya que al separar objetivos funcionales y tareas, el número variantes se reduce a las posibles combinaciones de tareas (dadas por los objetivos funcionales). Además, permite la selección de la funcionalidad deseada en el modelo de objetivos, lo que reduce aún más el espacio de variabilidad (las posibles tareas). Por otro lado, es posible obtener relaciones entre objetivos funcionales a partir de las relaciones entre tareas observando las relaciones de dependencia entre tareas y los objetivos que implementan.

Una herramienta que de soporte a este análisis también debe dar soporte al proceso de elicitación, permitiendo la creación de modelos de objetivos funcionales y no funcionales, de escenarios y de tareas. Además debe trazar las relaciones entre los distintos elementos para comprobar su corrección (un escenario siempre debe responder a, al menos un objetivo y una tarea a, al menos, una operación de un escenario) y para obtener el modelo de variabilidad a partir de ellos.

7. Conclusiones y Trabajo Futuro

Este trabajo propone una estrategia de requisitos que considera la variabilidad explícitamente. Para ello se introduce un punto de vista intencional por medio de la Orientación a Objetivos, campo muy activo dentro de la IR. Además, se propone la

utilización de escenarios para, mediante técnicas ampliamente estudiadas relacionar la vista intencional (objetivos), con la operacional (dichos escenarios) y ésta a su vez con la funcional, representada por las tareas. Por último, este enfoque es compatible con el enfoque tradicional de modelos de *features*, puesto que se puede asimilar los objetivos a las *features* de alto nivel (nivel de *capabilities*) y las tareas al resto de *features* de menor nivel de abstracción. Esta asimilación es importante puesto que permite relacionar el modelo de variabilidad con dichas *features* y, en consecuencia, con la definición de la arquitectura.

La introducción de un punto de vista intencional permite mejorar la elicitación de requisitos, pero también permite realizar el análisis de variantes desde un mayor nivel de abstracción (los objetivos del usuario) y utilizar técnicas de análisis formal para ello. Además, el uso de *softgoals* mejora el tratamiento de los requisitos no funcionales o cualidades del sistema.

En cuanto al soporte mediante herramientas, en un trabajo inicial [6] se inició el estudio del problema del análisis de variantes aunque utilizando únicamente dos modelos. De este trabajo se obtuvieron las primeras conclusiones, que mostraban la validez de la propuesta. El enfoque mostrado en este artículo mejora dicha propuesta, disminuyendo el espacio de variabilidad y, por tanto, mejorando el rendimiento.

A partir de ahora, el trabajo a realizar consiste en profundizar en las ideas mostradas, implementando una herramienta que permita realizar la elicitación de requisitos en sus tres vistas (intencional, operacional y funcional) y el análisis de variantes. Dicha herramienta posibilitará la aplicación de las técnicas mostradas a un ejemplo real. Por último, es necesario realizar un estudio detallado de las maneras en que se relacionan objetivos y escenarios entre si, y estos últimos con las tareas, para seleccionar las mejores técnicas y aplicarlas en la elicitación.

Reconocimientos

El primer autor de este artículo disfruta de una beca concedida por la Consejería de Educación de la Junta de Castilla y León y el Fondo Social Europeo.

Referencias

[1]. Antón, A.I., and Potts, C. *"The Use of Goals to Surface Requirements for Evolving Systems"*, in proc. of International Conference on Software Engineering (ICSE '98), Kyoto, Japan, April 1998, pp:157-166

[2]. Antón, A.I., Carter, R., Dagnino, A., Dempster, J., and Siege D. "*Deriving Goals from a Use-Case Based Requirements Specification*." Requirements Engineering Journal, Springer-Verlag, Volume 6, pp. 63-73, May 2001.

[3]. Chung, L., Nixon, B., Yu, E. and Mylopoulos, J. *"Non-Functional Requirements in Software Engineering*" Kluwer Academic Publishers 2000.

[4]. Dardenne, A., van Lamsweerde, A., *"Goal-Directed Requirements Acquisition"*, Science of Computer Programming, vol. 20, 1993, 179-190.

[5]. Giorgini, P., Mylopoulos, J., Nicchiarelli, E., and Sebastián, R. *"Reasoning with goal models"* In Proceedings of the 21st International Conference on Conceptual Modeling (ER 2002), Tampere, Finland, October 2002.

[6]. González-Baixauli, B., Leite J.C.S.P., and Mylopoulos, J. "*Visual Variability Analysis with Goal Models*". Proc. of the RE'2004. Sept. 2004. Kyoto, Japan. IEEE Computer Society, 2004. pp: 198-207.

[7]. Griss, M., Favaro, J., and d' Alessandro, M. "*Integrating feature modeling with the RSEB*". In Proceedings of the Fifth International Conference on Software Reuse, pages 76--85. IEEE Computer Society Press, 1998.

[8]. Halmans, G., and Pohl, K., "*Communicating the Variability of a Software-Product Family to Customers*". Journal of Software and Systems Modeling 2, 1 2003, 15--36.

[9]. Jacobson I., Griss M., and Jonsson P.: Software Reuse. Architecture, Process and Organization for Business Success. ACM Press. Addison Wesley Longman. 1997.

[10]. Kang, K. C., Cohen, S., Hess, J., Nowak, W. and Peterson, S. *"Feature-Oriented Domain Analysis (FODA) Feasibility Study"*. Technical Report, CMU/SEI-90-TR-21, Software Engineering Institute (Carnegie Mellon), Pittsburgh, PA 15213. 1990.

[11]. Kang, K. C., Kim, S., Lee, J., and Kim, K. "*FORM: A Feature-Oriented Reuse Method with Domain-Specific Reference Architectures*". Annals of Software Engineering, 5:143-168 (1998)

[12]. Kavakli, E., Loucopoulos, P., and Filippidou, D. "*Using Scenarios to Systematically Support Goal-Directed Elaboration for Information System Requirements*". In Proceedings of IEEE Symposium and Workshop on ECBS'96, Germany:, 1996. pp. 308-314

[13]. Mylopoulos, J., Chung, L., and Yu, *"From Object-Oriented to Goal-Oriented Requirements Analysis"*, Communications of the ACM, 42(1), Jan. 1999 pp:31-37.

[14]. Parnas, D.L. "On the Design and Development of Program Families," IEEE Trans. on Software Eng. 2(1), March 1976, pp: 1-9

[15]. Svahnberg, M., van Gurp, J., and Bosch, J., "*On the Notion of Variability in Software Product Lines*", In Proc. of the Working IEEE/IFIP Conference on Software Architecture. IEEE Computer Society Press, 2001. pp: 45-54.

[16]. Yu, E. *"Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering"*, Proceedings of the 3rd IEEE Int. Symp. on Requirements Engineering (RE'97) Jan. 6-8, 1997, Washington D.C., USA. pp:226-235.

[17]. Yu, E., Mylopoulos, J., *"Why goal-oriented requirements engineering"*, Proceedings of the Fourth International Workshop on Requirements Engineering: Foundations of Software Quality, Pisa, Italy, June 1998, pp. 15-22.

[18]. Yu, Y., Leite J.C.S.P., and Mylopoulos, J. "*From Goals to Aspects: Discovering Aspects from Requirements Goal Models*". Proc. of the RE'2004. Sept. 2004. Kyoto, Japan. IEEE Computer Society, 2004.