ISSN 1409-2441

# Ingeniería

Revista Semestral de la Universidad de Costa Rica

Vol. 16 (2) Ago/Dic 2006

CONTENIDO

**Artículos**



**Nota técnica**



**Normas**



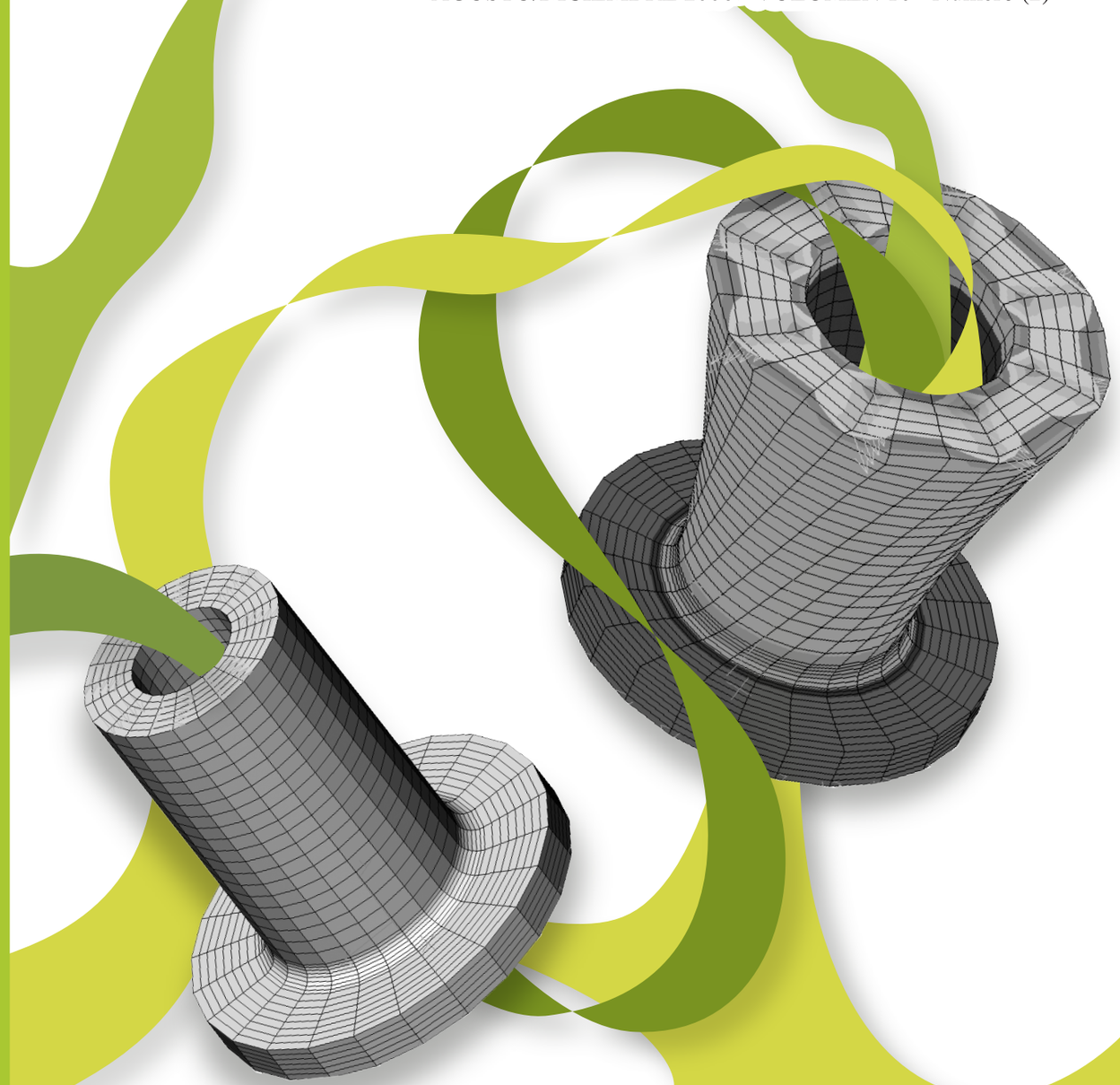
EDITORIAL DE LA UNIVERSIDAD DE COSTA RICA

http://www.revistaingenieria.inii.ucr.ac.cr
www.ucr.ac.cr
ISSN 1409-2441

# Ingeniería

Revista de la Universidad de Costa Rica

AGOSTO/DICIEMBRE 2006 - VOLUMEN 16 - Número (2)



Número (2)
Agosto / Diciembre 2006
Volumen 16
REVISTA INGENIERÍA

Ingeniería 16 (2): 85-101, ISSN: 1409-2441; 2006. San José, Costa Rica

# REPRESENTACIÓN DE LENGUAJES DE PATRONES DE ANÁLISIS DE DOMINIO

*Alan Calderón Castro*

**Resumen**

Se describe una forma de representar *lenguajes de patrones de análisis de dominio* para organizar el conocimiento específico del dominio que ingenieros de software elaboran en el contexto del desarrollo de una *familia de productos de software*. La forma de representación propuesta busca facilitar la comunicación entre los distintos actores, la evolución del conocimiento del dominio, el aprendizaje de nuevos desarrolladores sobre el dominio específico y el análisis de productos adaptados a clientes específicos en un proceso de desarrollo orientado a *familias de productos de software*. El aporte de este artículo consiste en extender el concepto de lexicón de patrones, propuesto en un trabajo previo, para que sea de utilidad en este contexto.

**Palabras clave:** lenguajes de patrones, patrones de software, familias de productos de software.

**Abstract**

A representation for *domain analysis pattern languages* is described aimed at organizing domain specific knowledge in the context of development of a **software product family**. The proposed representation is intended to facilitate the communication among stakeholders, the evolution of domain specific knowledge, the learning process of the domain specific knowledge by developers and the analysis of products personalized to specific clients in the development of a *software product family*. The contribution of this paper is to extend the pattern for pattern lexicons, proposed elsewhere, so that it can be useful in this context.

**Key words:** pattern languages, software patterns, software product lines.

**Recibido:** 22 de agosto del 2006 • **Aprobado:** 31 de enero del 2007

## 1. INTRODUCCIÓN

El problema que se analiza y para el cual se da una solución en este artículo consiste en ¿cómo representar *lenguajes de patrones* (LP) para el desarrollo de *familias de productos de software* (FPS)? Más específicamente ¿cómo representar *lenguajes de patrones de análisis de dominio* (LPAD) para facilitar el desarrollo de FPS?

Se parte del concepto de FPS de Clements, según el cual "...una FPS[1] es un conjunto de sistemas intensivos en software que comparten un conjunto administrado de características que satisfacen las necesidades específicas de un segmento particular de un mercado o de un tipo de misión crítica y que son construidos a partir de un conjunto básico de activos de una manera prescrita." (Clements y Northrop, 2002). Un ejemplo típico de una FPS es el conjunto de aplicaciones para ofimática de Microsoft que incluye MS-Word®, MS-Excel®, MS-Power Point®, etc.

Los lenguajes de patrones fueron propuestos por Alexander (1979) con la intención, en primer lugar, de que los arquitectos sistematizaran su conocimiento y lo compartieran eficientemente y, en segundo lugar, de que lo usaran con sus clientes en las actividades de diseño de sus casas de habitación, ciudades y desarrollos urbanos en general. Aunque algunos autores han publicado catálogos de patrones de análisis como Hay, 1996; Fowler, 1997; Silverston, Inmon y Graciano, 1996; Coad, 1992; Coad, 1997; Eriksson y Pender, 2000; además de numerosos artículos que describen colecciones pequeñas de patrones

de análisis, como por ejemplo Fernández, 1999; Fernández y Yuan, 1999; Fernández, Yuan y Brey, 2000; Gaertner y Trillón, 1999; Grand, 1999; Vaccare, Germano y Masiero, 1999; no se han realizado publicaciones que describan lenguajes de patrones de análisis para el desarrollo de FPS.

Clements y Northrop (2002) usan patrones para representar y organizar conocimiento especializado en relación con la administración de FPS, es decir, se trata de patrones asociados al proceso de ingeniería de FPS. Jacobson, Griss y Jonson, 1997; Clements y Northrop, 2002; Gomaa, 2004, describen el uso de patrones de diseño y patrones arquitectónicos en el proceso de construcción de los artefactos reutilizables de la FPS y en la construcción de los productos de la FPS. En todos estos casos se recomienda la administración de artefactos reutilizables enfatizando su valor práctico en el proceso de construcción de software, pero no se profundiza en el conocimiento específico del dominio que va asociado a los artefactos. El conocimiento específico del dominio queda implícito y no se aborda el tema de la organización o administración de este conocimiento para facilitar el aprendizaje de la empresa desarrolladora de software. Tampoco se considera el valor práctico de preservar este conocimiento para la inducción de los desarrolladores novatos en una empresa desarrolladora de software que haya mantenido un enfoque de FPS durante varios años.

El objetivo de este trabajo es describir una forma de representar un LPAD como una estrategia para organizar (y por ende, preservar) el conocimiento específico del dominio que elaboran ingenieros de software en el contexto de una FPS. La forma de representación propuesta busca facilitar la comunicación entre los distintos actores (clientes, usuarios, expertos en el dominio y desarrolladores en general), la evolución del conocimiento del dominio, el aprendizaje de nuevos desarrolladores sobre el dominio específico y el análisis de producto adaptados a clientes específicos en un proceso de desarrollo orientado a familias de productos de software. Para esto se aplica el patrón para lexicones de patrones propuesto por Calderón (2003). El aporte específico de este artículo consiste en extender los conceptos de lenguaje de patrones y "lexicón de patrones" para que sean de utilidad en el desarrollo de FPS.

La estructura de este trabajo incluye en la segunda sección una descripción de la noción de lenguaje de patrones en Alexander (1979), así como relevancia y reinterpretación para el desarrollo de FPS. En la tercera sección se presentan sistemas alternativos para la organización de patrones. En la cuarta sección se caracteriza un *lexicón de patrones* como un tipo específico de sistema para la representación de LPAD. En la quinta sección se esquematiza, mediante un ejemplo, cómo se podría representar un LPAD mediante un *lexicón de patrones*. Finalmente, se concluye visualizando una colaboración futura entre empresas y universidades a fin de plasmar esta propuesta en la industria de software.

## 2. LENGUAJES DE PATRONES Y SU RELEVANCIA EN EL CONTEXTO DE FPS

El concepto de "lenguaje de patrones" propuesto por Alexander ha sido, en nuestra opinión, poco comprendido en la comunidad de ingenieros de software. Se ha reducido el concepto a un esquema de utilidad técnica, una herramienta para el desarrollador de software en su quehacer cotidiano y se ha perdido la dimensión de lenguaje, como herramienta para la sistematización de experiencias entre personas, que Alexander le ha atribuido originalmente. Por esto, Calderón (2003) ha intentado rescatar la propuesta original reinterpretándola en función de las necesidades de sistematización del conocimiento en la industria del software. Alexander define un lenguaje de patrones como "…un sistema finito de reglas que una persona puede usar para generar una variedad infinita de edificios diferentes -todos miembros de una familia- y que el uso del lenguaje permitirá a la gente de un pueblo o ciudad generar el balance exacto entre uniformidad y variedad que da vida a un lugar." (Alexander, 1979). Si se acepta la premisa de que el lenguaje natural *es*

conocimiento, lo que Alexander propone es que los lenguajes de patrones son sistematizaciones del conocimiento especializado de los arquitectos.

El conocimiento especializado de los arquitectos corresponde con el conocimiento del dominio de los ingenieros de software en el contexto de una FPS. "Un dominio es un cuerpo de conocimientos especializados, un área de pericia, o una colección de funcionalidad relacionada. Por ejemplo, el dominio de telecomunicaciones es un conjunto de funcionalidades de telecomunicación, que a su vez consiste de otros dominios tales como la conmutación (el "switching"), los protocolos, la telefonía y las redes. Una línea de productos de software es un conjunto específico de sistemas de software que provee parte de esta funcionalidad" (Clements & Northrop, 2002). Por tanto, se reinterpreta a Alexander afirmando que un lenguaje de patrones de análisis orientado a un dominio de aplicación específico (o LPAD) provee a cada ingeniero de software que lo usa, el poder de crear una infinita variedad de productos de software únicos y nuevos, pertenecientes a una misma FPS, de la misma forma en que su lenguaje ordinario le da el poder de crear una variedad infinita de oraciones.

Una adecuada representación de un LPAD puede ser útil para:

1. Preservar y organizar adecuadamente el conocimiento experto que adquieren y perfeccionan sus desarrolladores de software (arquitectos, analistas y programadores) en dominios específicos (informática hospitalaria, informática jurídica, sistemas de información empresarial, etc.) ante la frecuente imposibilidad de retener al personal que ha ido poco a poco, durante varios proyectos y años, adquiriendo una pericia invaluable, acumulando un capital de conocimientos sobre un dominio dado.

2. Potenciar este capital de conocimiento usándolo en:

   - La incorporación plena de los clientes, usuarios o expertos en el dominio al proceso de desarrollo, a través de un lenguaje mutuamente inteligible con los desarrolladores. Esto con el fin de reducir los riesgos asociados al análisis de requerimientos.

   - La inducción de ingenieros de software sobre el dominio de una FPS.

   - El análisis rápido de productos pertenecientes a una FPS, destinados a satisfacer necesidades de clientes específicos.

## 3. SISTEMAS PARA ORGANIZAR PATRONES

Un catálogo de documentos electrónicos de patrones con facilidades de búsqueda sobre los documentos que especifican los patrones no sería suficiente para representar un LPAD, de tal forma que sea útil en el desarrollo de una FPS. Buschmann, Meunier, Rohnert, Sommerland y Stal (1996) en su libro *Pattern-oriented software architecture (a system of patterns)*, conocido como POSA 1, en la comunidad de ingenieros de software que usan o investigan sobre patrones, proponen organizar y representar el conocimiento de patrones por medio de "sistemas de patrones": "Un sistema de patrones liga los patrones que lo constituyen. Describe cómo se conectan los patrones y cómo se complementan entre sí. Buschmann et al. (1996) indican que un sistema de patrones también da soporte efectivo al uso de patrones en el desarrollo de software". Luego agregan que: "Un sistema de patrones para el diseño arquitectónico de software es una colección de patrones de diseño arquitectónico, junto con los lineamientos para su aplicación en la programación, su combinación y uso práctico en el desarrollo de software". Para especificar aún más esta definición escueta, indican los requerimientos de un sistema de patrones:

- "Debe abarcar una base suficiente de patrones...

- Debe describir todos sus patrones uniformemente...

- Debe evidenciar las distintas conexiones entre los patrones...
- Debe organizar los patrones que lo constituyen...
- Debe soportar la construcción de sistemas de software...
- Debe soportar su propia evolución..." Buschmann et al. (1996).

Schmidt, Stal, Rohnert y Buschmann (2002) en su libro *Pattern-oriented software architecture: patterns for concurrent and networked objects,* conocido en la comunidad de ingenieros de software que usan o investigan sobre patrones, como POSA 2, presentan una organización alternativa que busca destacar fuertemente las conexiones entre los patrones por medio de un diagrama de la red de patrones y que, a juicio de estos autores, es mejor que la representación lograda en POSA 1. En POSA 2 se hace mayor énfasis en la organización del conocimiento de patrones por medio de la representación de lenguajes de patrones (LP). El argumento que se da es que "La categorización de patrones de acuerdo con ciertas áreas específicas o propiedades no permite capturar las relaciones e interdependencias que existen en un conjunto particular de patrones…Estas relaciones e interdependencias influyen en la aplicabilidad de un patrón dada la presencia de otros patrones porque no todo patrón puede ser combinado con cualquier otro de manera significativa" (Schmidt et al., 2002).

En Calderón (2003) el concepto de *lexicón de patrones* se ha planteado en primera instancia como una síntesis de las propuestas de POSA 1 y POSA 2 incluyendo además características valiosas de los esquemas organizativos de otros catálogos como el de Gamma, Helm, Johnson y Vlissides, 1995; (denominado en adelante como GoF, tal como se le conoce en la comunidad de ingenieros de software que usan o investigan sobre patrones) Alur et al., 2001; Fowler, 1997; Hay, 1996; y Silverston et al., 1996. En términos generales, un *lexicón de patrones* es un sistema para organizar patrones mediante la representación de lenguajes de patrones.

La estructura de un *lexicón de patrones* incluye las características de los sistemas descritos y además: integra la representación de lenguajes de patrones (LP) complementarios; facilita la evolución de los LP y facilita el aprendizaje por parte de los ingenieros de software de los LP representados.

En la siguiente sección se extiende la propuesta de organización de Calderón (2003) caracterizando el concepto de "lexicón de patrones de análisis de dominio".

## 4. CARACTERIZACIÓN ESPECÍFICA DE UN LEXICÓN DE PATRONES DE ANÁLISIS DE DOMINIO

Un lexicón de patrones de análisis de dominio es un sistema para organizar patrones mediante la representación de LPAD complementarios desde la perspectiva del desarrollo de una FPS. Cada LPAD sistematiza conocimiento de un subdominio relevante. Por ejemplo, un lexicón de patrones para el análisis del dominio de la FPS "Sistemas de información empresarial para procesos administrativos" se puede dividir en tres subdominios "Arquitecturas alternativas", "Procesos del negocio", "Interacción humano-sistema". Para cada subdominio el lexicón incluiría la representación correspondiente de un LPAD. Se afirma que estos LPAD son complementarios en el sentido de que los patrones de cada uno se interconectan significativamente con los patrones de los demás LPAD.

Los objetivos de un lexicón de patrones de análisis de dominio son:

1. Proveer un lenguaje mutuamente inteligible entre desarrolladores y usuarios, clientes o expertos del dominio a fin de hacer efectiva su participación en la especificación de la funcionalidad, la definición de los detalles

esenciales de la interacción humano-sistema y en general, las características del producto de software.

2. Facilitar la evolución del conocimiento asociado a la FPS. La evolución del conocimiento del dominio es simplemente la contraparte de la evolución misma de una FPS. Si bien el esfuerzo de análisis de dominio supone ya de por sí una ventana de tiempo mucho más amplia que la que usualmente se considera al realizar un análisis tradicional, sin embargo, parte de la estrategia del análisis de dominio consiste en elaborar modelos que puedan adaptarse fácilmente a requerimientos no previstos.

3. Facilitar el aprendizaje, por parte de ingenieros de software, del conocimiento específico asociado a una FPS mediante la representación de LPAD complementarios para el desarrollo de la FPS.

4. Facilitar la construcción automatizada de los productos de una FPS, en consonancia con las tendencias modernas del desarrollo de software orientado por modelos, según las cuales, la construcción de productos de software debería concentrarse en la elaboración de modelos de análisis precisos de tal manera que luego se pueda escoger la tecnología de implementación y generar automáticamente software a partir de los modelos de análisis elaborados previamente.

Los principios organizativos para un lexicón de patrones de análisis de dominio son:

1. Para cada subdominio, los patrones del LPAD asociado se organizarán en tres niveles de categorización: nivel supraordinal (constituido por categorías de patrones), nivel básico o común que incluirá a los patrones propiamente y nivel subordinal que incluirá a las variantes y combinaciones típicas de patrones.

2. Las categorías de patrones son consustanciales a un LPAD, por tanto, son representadas en un "lexicón de patrones". Las categorías de patrones están asociadas con áreas de problemas de diseño de una FPS. En los distintos catálogos referidos se pueden encontrar categorías de patrones tales como "Patrones de construcción", "Patrones de comportamiento" y "Patrones de estructura" en GoF; "From mud to structure" y "Management" de POSA 1. Fowler, 1997; Hay, 1996 y Silverston et al., 1996 diferencian categorías de patrones tales como "La empresa y su entorno", "Cosas de la empresa", "Procedimientos y actividades", "Contabilidad", "Contratos".

3. Igualmente son consustanciales a los LPAD las categorías de conexiones o relaciones entre patrones. Algunos ejemplos de categorías de conexiones entre patrones que aparecen con frecuencia en catálogos de patrones (y que se pueden representar como estereotipos de UML para facilitar su representación gráfica) son <<soporta a>>, <<implementado con>>, <<usado con>>, <<contrasta con>>, <<similar a>>. Entre patrones de análisis de dominio aparecen con mayor frecuencia tipos de conexiones como <<evoluciona a>> (para representar cómo evolucionan los productos de software de una FPS desde una configuración a otra conforme los requerimientos cambian) o <<combinación de>> (para representar cómo los productos de una familia suelen combinar patrones de análisis básicos).

4. Las conexiones entre categorías de patrones y entre subdominios complementarios son fundamentales en un lenguaje de patrones y por tanto, se representan en un "lexicón de patrones".

5. A lo largo de la evolución de un LPAD surgirán de manera natural combinaciones de patrones y variantes de patrones que se repiten con frecuencia, es decir, que a su vez constituyen patrones. Por tanto, un lexicón de patrones asociado a una FPS representará y organizará variantes y combinaciones típicas de patrones en el nivel subordinal de categorización.

6. Las categorías de patrones de un LPAD tienen una estructura compleja, no se trata simplemente de listas de patrones. En las categorías aparecen patrones centrales y otros relativamente periféricos, buenos y no tan buenos ejemplos de la categoría. Con frecuencia, los autores de catálogos de patrones de análisis inician la exposición de una categoría de patrones muy simples, que luego se extienden paulatinamente. Estos patrones de análisis simples incluyen diferenciaciones que son fundamentales y por tanto, pueden considerarse buenos ejemplos de la categoría.

7. Los patrones centrales en una categoría de patrones se denominan "patrones prototípicos". Un patrón prototípico siempre será más fácil de aprender y de uso más común que sus comiembros de categoría. Al ser más simple que sus comiembros de categoría, un patrón prototípico deberá sugerir una imagen o metáfora que pueda ser usada para representar no solamente su propia intencionalidad, sino también la intencionalidad general de la categoría como un todo. Por tanto, identificar patrones prototípicos es muy útil para compartir conocimiento del dominio en la empresa desarrolladora de software y muy especialmente con desarrolladores que tienen poca experiencia en el dominio. Un patrón prototípico es un buen punto de inicio para una búsqueda de patrones si se complementa con conexiones del tipo <<similar a>> y <<contrasta con>> para facilitar la búsqueda de patrones apropiados a un contexto de análisis específico. Análogamente, la identificación clara de "categorías centrales de patrones" en el nivel supraordinal y la representación de conexiones con otras categorías, puede servir para fines similares.

8. La especificación de patrones de análisis de dominio debe incluir varios niveles de abstracción, desde la identificación de requerimientos y características del producto en construcción hasta la implementación del patrón en términos de un "framework" orientado a una tecnología de desarrollo específica, pasando por el análisis detallado de objetos independiente de la tecnología de implantación. El nivel de los requerimientos y las características está orientado a soportar un lenguaje mutuamente inteligible entre desarrolladores y usuarios, clientes y expertos del dominio. El nivel del análisis detallado de objetos independiente de la tecnología de implementación, busca precisamente una representación técnica de los requerimientos y las características, es decir, una representación orientada a la construcción automática que no se comprometa con una tecnología específica. El nivel de la implementación busca automatizar la construcción de productos de tal manera que se reutilicen componentes de software desde los modelos de análisis. La idea es facilitar la evolución independiente de los tres niveles.

9. Se incluye en un lexicón de patrones de análisis de dominio la identificación explícita de patrones que sirven de "puntos de entrada" para el proceso de análisis y diseño de un producto de la FPS, es decir, que representan las cadenas de patrones que se pueden seguir para completar la construcción de un producto. Estos puntos de entrada a cadenas de patrones son muy apreciados por los ingenieros de software, pues definen grandes rutas típicas en el proceso de construcción, por esto mismo han recibido especial atención por parte de los autores de catálogos de patrones. En un lexicón de patrones de análisis de dominio, los patrones arquitectónicos serán los "puntos de entrada" por excelencia para la construcción de un producto de una FPS. Sin embargo, los patrones prototípicos de las categorías de patrones también pueden ser puntos de entrada válidos cuando el producto a construir en realidad conlleva una variante arquitectónica imprevista con respecto a las arquitecturas típicas de la FPS.

10. En un lexicón de patrones de análisis de dominio, un patrón puede aparecer como miembro de varias categorías, esto no solo refleja las fronteras difusas entre las

categorías relevantes para una FPS, sino también facilita distintas formas de acceder un mismo patrón, dependiendo del análisis que un desarrollador esté llevando a cabo en la construcción de un producto específico de la FPS.

11. Un lexicón de patrones de análisis de dominio permite integrar varios "framework" de implementación para la construcción automática de los productos. Un lexicón incluye los mecanismos que conectan los artefactos de análisis independientes de la tecnología de implementación (que aparecen "encapsulados" en los patrones de análisis de dominio) con los componentes de los "frameworks".

La siguiente sección presenta un ejemplo, en forma obligadamente esquemática, de un lexicón de patrones para una familia de productos de software de sistemas de información empresarial para el soporte de procesos administrativos. La fuente principal para la especificación de estos patrones y categorías de patrones han sido los catálogos de Hay, 1996 y de Silverston et al., 1996, que se centran en patrones de modelos de datos.

## 5. EJEMPLO DE LEXICÓN DE PATRONES PARA UNA FPS

Dado que evidentemente la mejor forma de representar un lexicón de patrones es a través de un hipertexto, aquí solamente se tratará de mostrar cómo se podrían organizar los distintos niveles del lexicón y algunos de los tipos de conexiones entre los distintos elementos. La FPS usada en este ejemplo está constituida por sistemas de información empresarial para procesos administrativos. El dominio de conocimientos específico se subdivide en tres subdominios[2]:

- "Arquitecturas alternativas" que incluye patrones arquitectónicos o estilos arquitectónicos alternativos para la FPS. Cada arquitectura típica se documenta mediante un patrón arquitectónico que sirve como "punto de entrada" en el diseño arquitectónico de un producto con características particulares para un tipo de cliente específico. El tema del diseño arquitectónico es central en el desarrollo de software orientado a FPS, según indican Jacobson et al., 1997; Clements y Northrop, 2002 y los autores de GoF, de ahí se deriva la importancia de representar este dominio en el lexicón.

- "Procesos del negocio" que incluye categorías de patrones y patrones orientados a sistematizar conocimiento sobre los procesos del negocio soportados por los productos de la FPS en cuestión.

- "Interacción humano-sistema" que incluye categorías de patrones y patrones orientados a sistematizar conocimiento sobre los esquemas típicos de interacción humano-sistema en los productos de la FPS escogida.

Se ha escogido presentar una "vista transversal" del lexicón centrada en el dominio "Procesos del negocio" y específicamente en la subcategoría de patrones "Productos" de la categoría "Venta y compra de productos" que se ha considerado una categoría central del dominio "Procesos del negocio de los sistemas de información empresarial para procesos administrativos". Se ilustrará esta vista transversal a través de figuras en las que, tanto las categorías de patrones como los patrones mismos, son representados por medio de paquetes de UML. La Figura 1 muestra la organización general del lexicón de patrones de análisis de dominio para la FPS indicada.

La Figura 2 muestra las categorías principales del dominio "Procesos del negocio". En este caso se ha considerado que la categoría central del dominio es "Venta y compra de productos", el argumento para dicha escogencia es que el análisis de esta área del dominio es prioritario, puesto que está asociada con los procesos que motivan la existencia misma de la empresa. Dicho de otro modo, las diferenciaciones que se hacen en esta categoría de patrones son fundamentales para que los productos de la FPS se logren alinear con los objetivos del

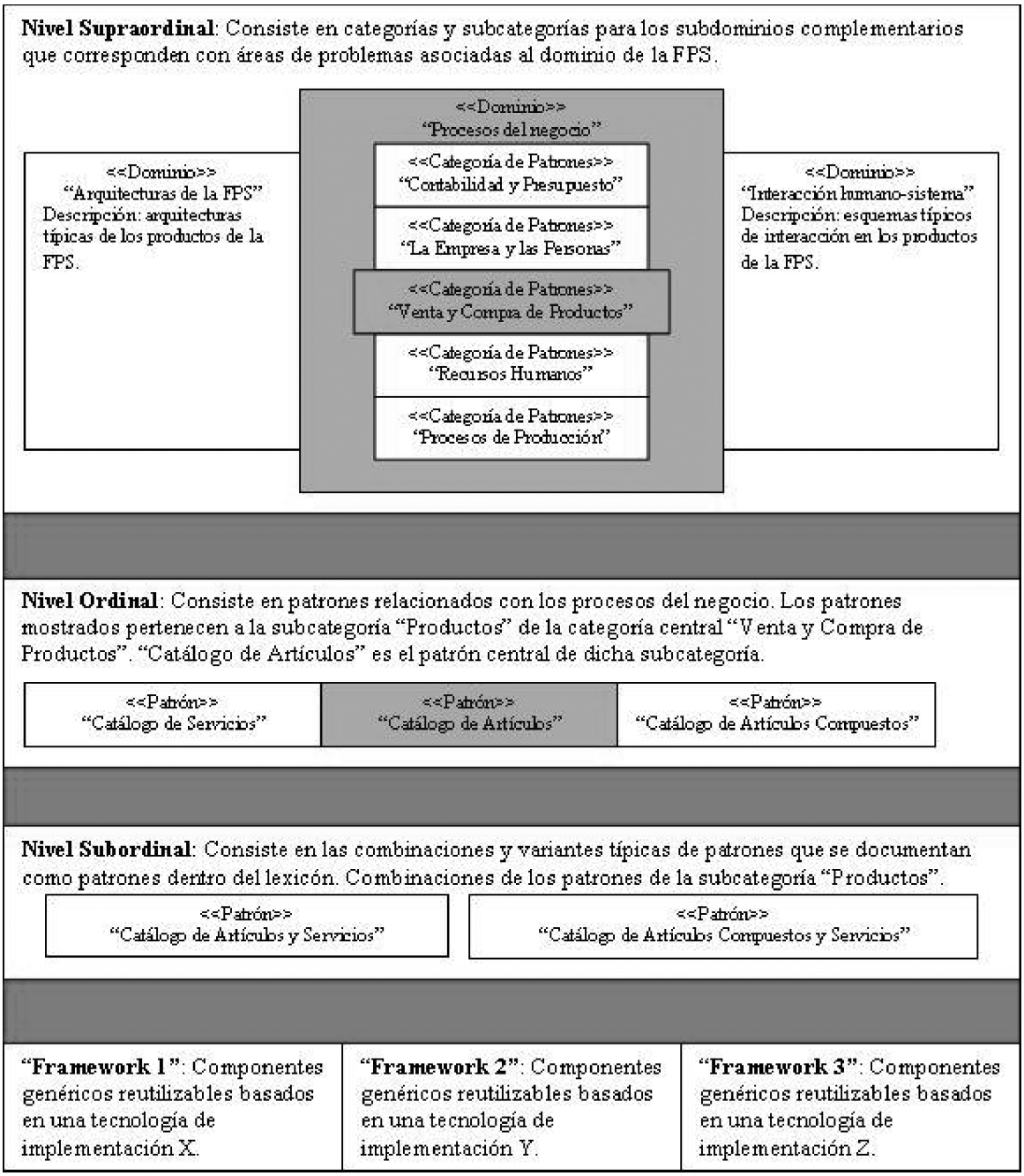



**Figura 1.** Estructura de un lexicón para el análisis del dominio "Sistemas de información empresarial para procesos administrativos".

Fuente: (El autor)

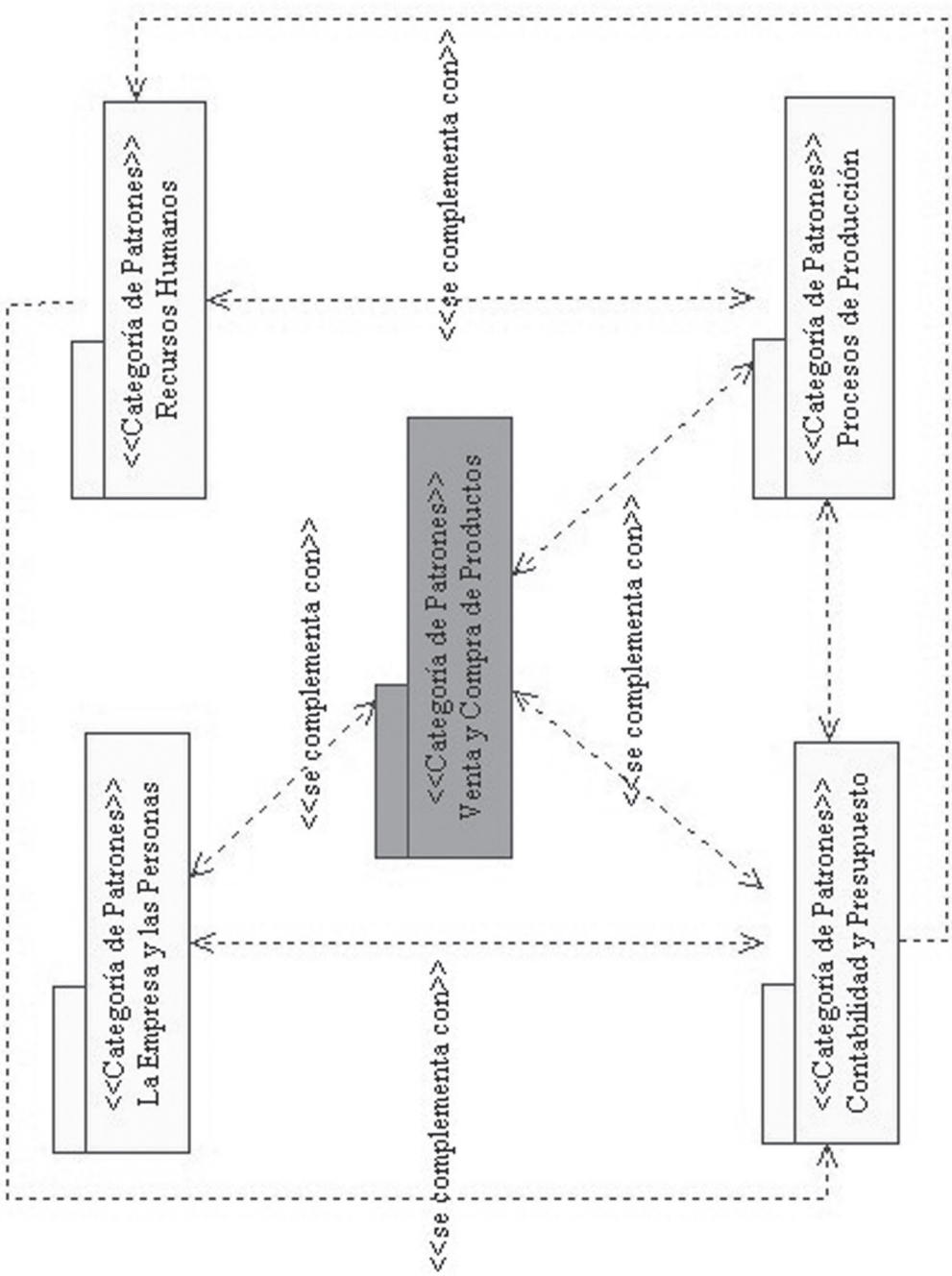



**Figura 2.** Mapa del nivel supraordinal del lexicón con categorías de patrones del dominio.
Fuente**:** (El autor)

**Intencionalidad de la categoría**

Los patrones de esta categoría generalizan artefactos del análisis que modelan requerimientos de información que tienen las organizaciones sobre sus productos y los procesos de venta y compra de dichos productos.

**Requerimientos de información**

1. "¿Cómo se comparan en calidad y precio los productos de la organización con los de la competencia?.
2. ¿Qué nivel de existencias se requiere en cada localización para poder satisfacer las necesidades de los clientes?.
3. ¿Cuál es el precio, el costo y la ganancia de los productos ofrecidos?.
4. ¿Dónde pueden comprarse los mejores servicios y productos a los mejores precios?" (Silverston et al., 1996)

**Terminología**

1. **Producto:** es un producto tangible o un producto intangible. Los productos intangibles se conocen típicamente como servicios.
2. **Proceso de venta:** proceso mediante el cual la organización provee productos o servicios a sus clientes obteniendo a cambio alguna retribución económica o de otro tipo.
3. **Proceso de compra:** proceso mediante el cual la organización se aprovisiona de bienes para cumplir con los procesos de venta a sus clientes.

**Figura 3.** Especificación de la categoría de patrones "Venta y compra de productos".
Fuente: (El autor)

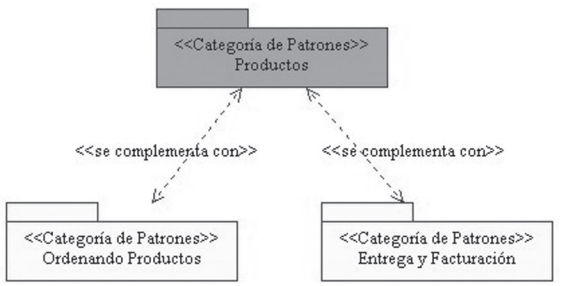



**Figura 4.** Subcategorías de la categoría central "Venta y Compra de Productos".
Fuente: (El autor)

negocio. La Figura 3 describe la categoría con base en la "Intencionalidad de la categoría", los "Requerimientos de información" y la "Terminología", mientras que la Figura 4 muestra un mapa de subcategorías que incluye "Venta y compra de productos".

Se ha considerado la subcategoría "Productos" como central para "Venta y compra de productos" dado que los patrones de esta subcategoría abarcan diferenciaciones básicas para que se puedan comprender los procesos de venta y compra cuyos patrones de análisis estarían incluidos en las otras dos subcategorías de la Figura 4.

En la Figura 5 se especifica la subcategoría central "Productos" escogida en esta vista transversal del lexicón. Esta subcategoría sólo

**Intencionalidad de la categoría**

Los patrones de esta categoría generalizan artefactos del análisis que modelan requerimientos de información que tienen las organizaciones sobre sus productos, específicamente tipos y categorías de productos..

**Requerimientos de información**

**1.** ¿Cómo organizar los artículos y servicios en tipos que representen sus atributos relevantes?.

**2.** ¿Se requiere la definición de categorías de productos? ¿Qué tan estables son estas categorías?.

**3**. ¿Se venden artículos? ¿Simples y compuestos?.

**4.** ¿Se venden servicios?.

**5.** ¿Se venden tanto artículos como servicios? En tal caso, ¿qué relaciones relevantes se presentan?

**Terminología**

1. **Producto:** es un producto tangible o un producto intangible. Los productos intangibles se conocen típicamente como servicios.
2. **Tipo de artículo:** representa un conjunto de artículos que poseen características similares, en particular un mismo número de modelo, o marca o identificador generado por los proveedores de la empresa.
3. **Tipo de artículo simple:** es un tipo de artículo que, desde la perspectiva de los procesos del negocio, se maneja como una unidad indivisible.
4. **Tipo de artículo compuesto:** es un tipo de artículo que, desde la perspectiva de los procesos del negocio, se maneja como una composición de otros tipos de artículo que también aparecen dentro de los catálogos de la empresa o simplemente son relevantes para los procesos de producción.
5. **Tipo de servicio:** es un tipo de actividad que la empresa vende a sus clientes y por cuya realización obtiene algún tipo de beneficio que constituye una meta central en el negocio de la empresa.
6. **Categoría de tipos de producto:** es una categoría de tipos de artículos o de tipos de servicio que es relevante para los procesos del negocio y que puede tener una existencia temporal o indefinida. En general, se supone que un tipo de producto (artículo o servicio) solo puede pertenecer a una categoría, sin embargo en ciertos contextos organizacionales este supuesto no será válido. Es tarea del analista determinar la aplicabilidad de este supuesto fundamental.

**Figura 5.** Especificación de la categoría de patrones "Productos".

Fuente: (El autor)

contiene patrones de análisis, por tanto, el mapa correspondiente (ver Figura 6) incluye patrones (paquetes UML con estereotipo <<patrón>>) en lugar de subcategorías.

Tal como se muestra en la Figura 6, se han considerado como patrones prototípicos de la categoría "Productos" a "Catálogo de Artículos" y a "Catálogo de Servicios" por cuanto se ha supuesto que representan las dos situaciones más simples que se pueden encontrar en una empresa. Estos dos patrones se pueden combinar para satisfacer necesidades de empresas que abarcan tanto artículos como servicios dentro de sus ventas.

En la Figura 7 aparece el mapa de características funcionales[3] que son consideradas en los distintos patrones de la figura anterior. El mapa de características funcionales también es útil para categorías que sólo contienen subcategorías (como "Venta y Compra de Productos") pero, por razones de espacio sólo se muestra para la categoría "Productos". Los patrones de esta categoría abarcarán las combinaciones más comunes de características funcionales. Las combinaciones típicas de características que son adoptadas por cada patrón constituyen la sección de "fuerzas" de la especificación de cada patrón. A su vez, los patrones básicos de una categoría se combinan en nuevos patrones para responder a requerimientos típicos más sofisticados que evolucionan en formas anticipadas por el análisis de características. Esto da origen a conexiones relevantes entre los patrones de una categoría de patrones.

Las características se agrupan en subconjuntos e xcluyentes o no excluyentes. En los subconjuntos excluyentes se puede identificar una característica

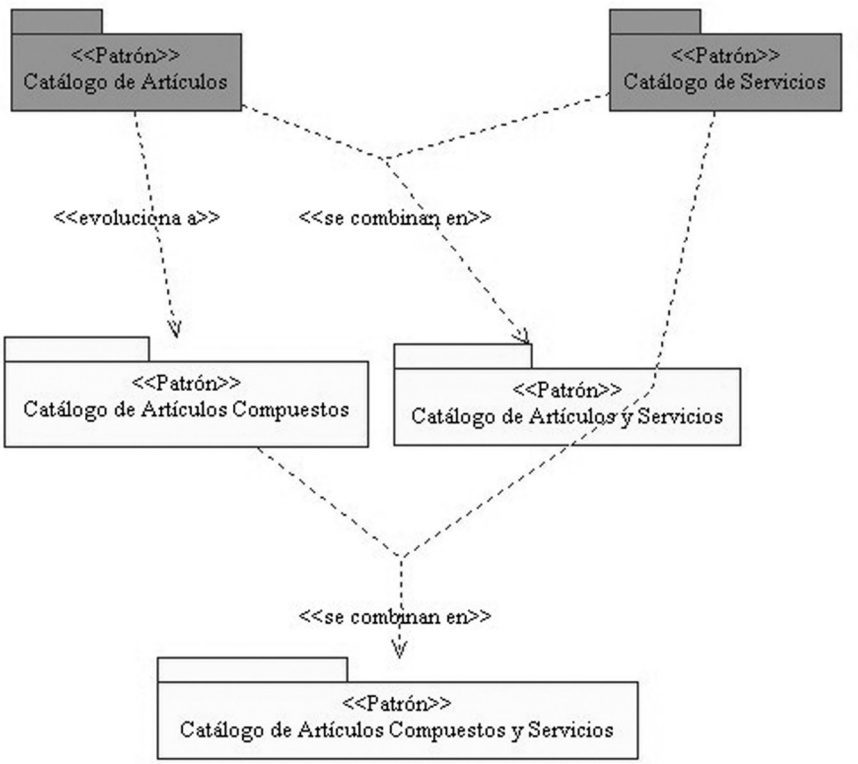



**Figura 6.** Patrones de la subcategoría central "Productos".
Fuente: (El autor)

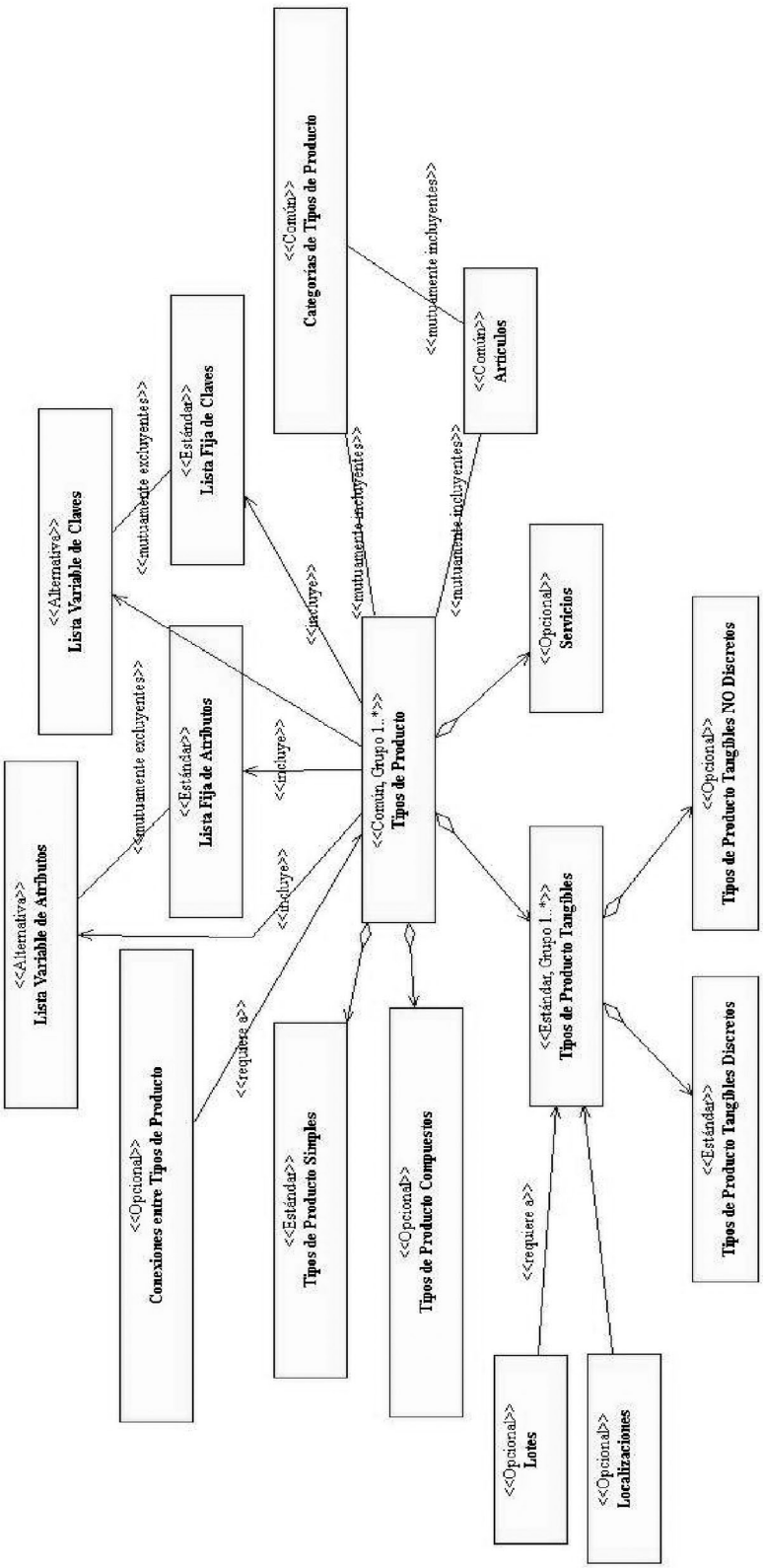



**Figura 7.** Características funcionales de la subcategoría central "Productos".
Fuente: (El autor)

estándar o típica y otras menos típicas que se denominan alternativas. En los conjuntos no excluyentes, igualmente se puede identificar una característica estándar o típica y otras menos típicas que se denominan opcionales. Un producto de una FPS sólo puede incluir una característica de un subconjunto excluyente. La inclusión de algunas características en un producto puede requerir o inducir la inclusión de otras, en cuyo caso se establece que son características mutuamente incluyentes. El análisis de características pretende responder a la variabilidad en el tiempo y en el espacio de las características de los productos de la FPS que se instalen a los distintos clientes. Así, una categoría en un lexicón de patrones de análisis de dominio representa un conjunto de características que típicamente aparecen asociadas como requerimientos de las organizaciones cliente.

Un "Catálogo de Artículos" (simples) puede fácilmente evolucionar hacia un "Catálogo de Artículos Compuestos". Por otro lado, la combinación de características más sofisticada que se considera típica en esta categoría de patrones consiste en una empresa que pueda requerir soportar no sólo la venta de artículos simples y servicios, sino también la venta de artículos compuestos. De esta forma, en un mismo diagrama (Figura 6) se visualizan el nivel ordinal del lexicón (patrones básicos) y el nivel subordinal (combinaciones de patrones y variaciones de patrones). Otras combinaciones de características no han sido consideradas típicas y por tanto no se han representado como patrones. Igualmente, algunas características de la categoría no aparecen en ninguno de los patrones, sin embargo, sí se han incluido en el análisis de características. De esta manera, a través de los patrones se construyen los artefactos (específicamente modelos) de utilidad inmediata en la construcción de productos de la FPS, lo cual permite inducir cuáles son los componentes de software que son de utilidad inmediata para la construcción de productos. A la vez, con el análisis más amplio de características se especifican los alcances de la FPS desde el punto de vista de posibles productos que se podrían construir en el futuro. Así, el lexicón de patrones para una FPS evolucionará hacia la inclusión de nuevos patrones conforme nuevas combinaciones típicas de características se consideren necesarias.

Por razones de espacio, no es posible describir la estructura de especificación de los patrones de análisis. Al respecto cabe señalar que los autores que han publicado patrones de análisis han incluido como solución del patrón ya sea modelos de datos Hay, 1996; Silverston et al., 1996, o modelos conceptuales Fowler, 1997. En algunos casos se han incluido adicionalmente modelos de objetos flujos de trabajo, o diagramas de secuencia o de colaboración Fernández, 1999; Fernández y Yuan, 1999 y Fernández et al., 2000. Una combinación de todos estos modelos junto con modelos, de casos de uso, es probablemente lo más adecuado.

### 5.1 Relaciones entre patrones

Ningún patrón es una isla, nos recuerda Alexander (1979). Por esta razón, las conexiones entre patrones reciben especial atención en la construcción de un lexicón de patrones. Por la misma causa, las conexiones entre categorías de patrones y dominios son muy importantes en la construcción de un lexicón de patrones. En Calderón (2003) se describe una plantilla para especificar conexiones. Las conexiones se representan por medio de relaciones estereotipadas en UML. Cada uno de estos estereotipos tiene un significado preciso en el contexto en que aparece. Por ejemplo, en el caso de <<se complementa con>> entre las categorías de patrones del nivel supraordinal del lexicón en la Figura 2, "La empresa y las personas" y "Venta y compra de productos", significa que al analizarse la administración de los productos que una empresa ofrece, con frecuencia surgirá la necesidad de analizar la administración de sus clientes y proveedores, razón por la cual el desarrollador podrá complementar su análisis mediante los patrones de la categoría "La empresa y las personas" que incluyen conceptos y diferenciaciones típicas en relación con los clientes, proveedores y socios de una empresa. En la Figura 6, la intención del tipo de conexión

<<evoluciona a>> es anticipar la forma en que evolucionan las necesidades de cada empresa en relación con la administración de sus productos, esto es, la variabilidad en el tiempo. Por otro lado, la variabilidad de necesidades en el espacio de empresas se logra representar en el lexicón mediante conexiones del tipo <<se combinan en>>.

### 5.2 "Frameworks" y lexicones para LPAD

El nivel inferior del lexicón de patrones incluye componentes de software asociados a los patrones de los niveles ordinal y subordinal. Los modelos o artefactos de los patrones deben basarse en elementos abstractos (casos de uso abstractos, objetos de frontera y clases de análisis abstractas). Por otro lado, los mecanismos para la construcción automática de un producto deben explotar las variaciones implícitas en los modelos abstractos de los patrones. La descripción detallada de estos mecanismos es imposible de abordar en este trabajo por limitaciones de espacio, pero algunos han sido descritos por Jacobson et al., 1997 y Greenfield, Short, Cook y Kent, 2004.

## 6. CONCLUSIONES

El concepto de lenguaje de patrones de análisis de dominio (LPAD) se ha definido como una extensión del concepto lenguaje de patrones de Alexander. Se ha adaptado la estructura de lexicón de patrones propuesta en Calderón (2003) para representar LPAD de manera que se facilite el desarrollo de FPS. Las características esenciales de un lexicón de patrones para representar LPAD facilitan:

- La comunicación entre los distintos actores en el proceso de desarrollo de FPS, dado que los LPAD sistematizan el conocimiento compartido y unifican los términos para referirse a los conceptos del dominio específico.

- La evolución del conocimiento del dominio mediante:

  - La representación integrada de subdominios relevantes.

  - Los tres niveles de categorización que estructuran los subdominios de conocimiento y las categorías de patrones de manera flexible.

  - La estandarización de conexiones entre subdominios, categorías de patrones y patrones que permite integrar el conocimiento conforme evoluciona.

- El aprendizaje de desarrolladores nuevos sobre el dominio de conocimientos específicos de una FPS mediante:

  - Los tres niveles de categorización que identifican subdominios de conocimiento y categorías de patrones que estructuran el dominio de manera natural.

  - La identificación de patrones prototípicos y categorías centrales que sugieren a los nuevos desarrolladores por dónde empezar a conocer el dominio.

  - La estandarización de conexiones entre subdominios, categorías de patrones y patrones que facilita un aprendizaje integrado.

- El análisis para construir productos de una FPS adaptados a clientes específicos mediante:

  - Los "puntos de entrada" que son patrones arquitectónicos que representan productos típicos de la FPS y por ende, representan tipos comunes de clientes posibles.

  - La subdivisión del modelo de características orientado a representar la variabilidad en un análisis de dominio, tal como lo propone Gomaa (2004), a través de los mapas de características de las categorías de patrones de análisis de dominio (como el de la Figura 7). De

esta manera se reorganiza de forma muy natural dicho modelo y se usa para facilitar la búsqueda de patrones, según las necesidades de cada cliente específico.

La propuesta de usar un lexicón de patrones para representar el conocimiento específico de dominio construido a lo largo de un proceso centrado en FPS, permite vislumbrar una sinergia entre centros de educación superior e industria para plasmar en la práctica el uso sistemático e intensivo de conocimiento de dominios específicos en la construcción de software. En términos generales, la estrategia consistiría en asignar a los centros de educación superior y de investigación, la responsabilidad de elaborar lexicones de patrones para diversos dominios de aplicación relevantes para las industrias locales de software; y a las empresas, dejar la posibilidad de desarrollar los repositorios de artefactos reutilizables y particularmente los "frameworks" de objetos que les permitirían desarrollar productos de una FPS, previamente acordada como relevante. De esta manera, los costos y los riesgos asociados con la sistematización del conocimiento del dominio (el análisis de dominio), que podría resultar difícil de afrontar para las empresas, serían absorbidos por las instituciones de educación superior. Por la diversidad de su personal y su vocación para la investigación, las universidades están mejor preparadas para hacer frente al análisis de dominio. Por otro lado, el desarrollo de artefactos reutilizables y particularmente de "frameworks" de objetos, requiere un conocimiento técnico y una perspectiva del mercado meta que concierne más a las empresas desarrolladoras de software.

## NOTAS

1. Se ha traducido "software product line" como "familia de productos de software".

2. En adelante se usa dominio como sinónimo de subdominio.

3. Respecto de la relación entre requerimientos funcionales y características funcionales, Gomaa (2004) plantea que "…El término requerimiento es usado para denominar las necesidades que una línea de productos de software, y por ende al menos uno de los miembros de la línea de productos, debe ser capaz de satisfacer. Una vez que la línea de productos de software ha incluido y especificado este requerimiento, el requerimiento se denomina una característica provista por la línea de productos de software" (Gomaa, 2004).

## REFERENCIAS BIBLIOGRÁFICAS

Alexander, C. (1979). *The timeless way of building*. (pp.191), New York: Oxford University Press.

Alur, D., Crupi, J. & Malks, D. (2001). *Core J2EE patterns (Best practices and design strategies)*. (pp. 360), New Jersey: Sun Microsystems Press.

Buschmann, F., Meunier, R., Rohnert, H., Sommerland, P. & Stal, M. (1996). *Pattern-oriented software architecture (a system of patterns)*. West Sussex: John Wiley & Sons. Conocido como POSA 1.

Calderón, A. (2003). Un patrón para lexicones de patrones. En: *The third latin american conference on pattern languages of programming sugarloaf PLoP, 2003*. Pernambuco, (pp. 5-14), Brasil.

Clements, P. & Northrop, L (2002). *Software product lines: practices and patterns*. Boston: Addison-Wesley.

Coad, P. (1992). Object-oriented patterns, *Communications of the ACM*, *35* (9), 152-159.

Coad, P. (1997). *Object models (Strategies, patterns and applications)*. New Jersey: Yourdon Press.

Eriksson, H. & Penker, M. (2000). *Business modeling with UML: business patterns at work*. New York: John Wiley & Sons.

Fernández, E. B. (2000). Stock manager: an analysis pattern for inventories. En: *Proceedings PLoP 2000*. Monticello, Indiana.

Fernández, E. B. & Yuan, X. (1999). An analysis pattern for reservation and use of reusable entities. En: *Proceedings PLoP 1999*. Monticello, Illinois.

Fernandez, E. B., Yuan, X. & Brey, S. (2000). Analysis patterns for the order and shipment of a product. En: *Proceedings PLoP 2000*. Monticello, Illinois.

Fowler, M. (1997). *Analysis patterns: reusable object models*. Boston: Addison-Wesley.

Gamma, E., Helm, R., Johnson, R. & Vlissides, J. (1995). *Design patterns: elements of reusable object-oriented software*. Boston: Addison-Wesley Publishing Company. Conocido como GoF.

Gaertner, N. & Thirion, B. (1999). Grafcet: an analysis pattern for event driven real-time systems. En: *Proceedings PLoP 1999*. Monticello, Illinois.

Gomaa, H. (2004). *Designing software product lines with UML: from use cases to pattern-based software architectures*. (pp. 95) Boston: Addison-Wesley.

Grand, M. (1999). Transaction patterns: a collection of four transaction related patterns. En: *Proceedings PLoP 1999*. Monticello, Illinois.

Greenfield, J., Short, K., Cook, S. & Kent, S. (2004). *Software factories (Assembling applications with patterns, models, frameworks, and tools)*. Indiana: Wiley Pub.

Hay, D. C. (1996). *Data model patterns (Conventions of thought)*. New York: Dorset House Pub.

Jacobson, I., Griss, M. & Jonsson, P. (1997). *Software reuse: architecture, process and organization for business success*. Boston: Addison-Wesley.

Schmidt, D., Stal, M., Rohnert, H. & Buschmann, F. (2002). *Pattern-oriented software architecture: patterns for concurrent and networked objects*. (pp. 525), West Sussex: John Wiley & Sons. Conocido como POSA 2.

Silverston, L; Inmon, W. H. & Graziano, K. (1996). *The data model resource book (a library of logical data models and data warehouse designs)*. New York: John Wiley & Sons, Inc.

Vaccare Braga, Rosana T., Germano, F. S. R. & Masiero, P. (1999). A pattern language for business resource management. En: *Proceedings PLoP 1999*. Monticello, Illinois.

## SOBRE EL AUTOR

**Alan Calderón Castro**
Ingeniero de Software
Profesor de la Escuela de Ciencias de la Computación e Informática de la Universidad de Costa Rica
San José, Costa Rica.
Teléfono: 207-4020
Facsímil: 207-5227
Correo electrónico: calderon@ecci.ucr.ac.cr