Un patrón para lexicones de patrones

Alan Calderón Castro

[1]Escuela de Computación e Informática – Universidad de Costa Rica (UCR)
Fax 506-207-5527 – San José – Costa Rica

calderon@ecci.ucr.ac.cr

Abstract. *Presently many pattern catalogs for software development have been published (design patterns, architecture patterns, analysis patterns, idioms and other). The pattern knowledge keeps growing day by day and it is essential its incorporation to software engineering teaching. On the other hand, the software industry also requires to systematize pattern knowledge. A system is required that allows to organize the pattern knowledge in a natural way that facilitates searches, the organization of pattern, pattern variants and pattern combinations, as well as the continuous appearance of new patterns for existent domains of experience and new ones. Seen from another angle, a system to support pattern languages development and to improve their expressiveness is required. To be consequent with the heuristic focus accustomed by those who specify patterns, this paper describes a pattern aimed at building systems to support pattern language evolution. Some principles to construct these type of systems are organized in a pattern form. An example of this pattern is described in [Calderón, 2003c].*

Resumen. *A la fecha se han publicado muchos catálogos de patrones asociados con el desarrollo de software (patrones de diseño, patrones de arquitectura, patrones de modelaje de requerimientos, modismos o estilos de programación—conocidos como "idioms" en inglés—y otros). El conocimiento de patrones sigue creciendo día a día y es esencial su incorporación a la enseñanza de la ingeniería de software. Por otro lado, la industria del software también requiere ir sistematizando el conocimiento sobre patrones. Se requiere un sistema que permita organizar el conocimiento de patrones de manera natural que facilite las búsquedas, la organización y el intercambio de conocimiento de patrones, de variantes de patrones y combinaciones de patrones, así como la continua aparición de patrones para los dominios de experiencia existentes y otros nuevos. Visto desde otro ángulo, se requiere un sistema que soporte el desarrollo evolutivo de lenguajes de patrones y mejore su expresividad. Para ser consecuente con el enfoque heurístico acostumbrado por quienes especifican patrones, en este artículo se propone un patrón para construir sistemas que soporten la evolución de lenguajes de patrones. Este patrón intenta proveer principios para construir sistemas de software que sistematicen el conocimiento de diseño de la industria del software y de las organizaciones dedicadas a la enseñanza de la ingeniería de software. En [Calderón, 2003c] se ejemplifica la aplicación del patrón propuesto.*

1. Nombre del Patrón: Lexicón de Patrones

2. Contexto

A la fecha se han publicado muchos catálogos de patrones asociados con el desarrollo de *software*. Podría comenzarse citando "Design Patterns" [Gamma, 95], denominado catálogo de GoF (por "Gang of Four" como se referencia a sus cuatro autores), que organiza el conocimiento heurístico a nivel de diseño detallado de objetos. Seguidamente podrían mencionarse las ediciones 1 y 2 de "Pattern-Oriented Software Architecture", de Buschmann y otros la primera, y de Schmidt y otros la segunda (ver [Buschmann, 96] y [Schmidt, 2000]), que enfatizan en la organización del conocimiento heurístico a nivel de diseño arquitectónico. Luego, se debería agregar "Data Model Patterns" [Hay, 96], "Object Models" [Coad, 97] que organizan conocimiento heurístico a nivel de modelaje de requerimientos en torno a distintas familias de aplicaciones. Finalmente, no pueden dejar de mencionarse "Core J2EE Patterns" [Alur, 2001], "Advanced C++" [Coplien, 92], "JAVA 2 Performance and Idiom Guide" [Larman, 99] y otros, que organizan conocimiento heurístico a nivel de programación y por ende se refieren al uso óptimo de tecnologías de programación específicas. Esto solo para mencionar algunos de los catálogos más conocidos, porque el estudio [Czichy, 01] identifica tres grandes categorías de documentos publicados con patrones: una que abarca las actividades típicas del proceso de desarrollo de *software* (modelaje de requerimientos, diseño, programación, prueba, etc.), otra que abarca a patrones de proceso de desarrollo de *software* y a patrones de organizaciones de desarrollo de *software* y una tercera categoría con otros tipos de patrones.

Se puede observar en algunos de estos catálogos una estructura de categorización recurrente orientada a facilitar la resolución de problemas de diseño. Los catálogos más reconocidos incluyen mapas de patrones que muestran cómo los patrones se complementan entre sí, y también recomendaciones generales para la guía del lector. Por estas características, el catálogo de GoF ha sido considerado por muchos autores como un ejemplo prototípico[1]. Sus características esenciales son:

1. Categorización de patrones en: patrones de estructuración, patrones de comportamiento y patrones de construcción de objetos o patrones construccionistas .
2. A cada patrón se le asigna un nombre significativo.
3. El uso de una plantilla derivada de la tripleta "Contexto", "Problema" y "Solución" propuesta en [Alexander, 79], para la especificación de cada patrón.
4. Como parte de la especificación de cada patrón se incluyen secciones para analizar las consecuencias de la aplicación de la solución y para hacer explícitas las relaciones con los demás patrones del catálogo.
5. Mapa de interrelaciones entre patrones.

[1] Cabe aclarar que sobre la categorización de patrones planteada en GoF no hay suficiente acuerdo como para afirmar que es la mejor, sin embargo sí es un ejemplo muy conocido, imitado por otros y por ende representativo.

Estas características organizativas esenciales también están presentes en otros catálogos, como por ejemplo [Buschmann, 96] conocido como POSA 1, [Schmidt, 2000] conocido como POSA 2 y [Alur, 2001] . La categorización es distinta a la de GoF, y la plantilla también difiere, pero los principios organizativos se mantienen, como cuando se aplica un patrón a un contexto específico. En otros casos, como [Hay, 96] y [Fowler, 97] se puede observar claramente la primera característica pero no las otras dos. Otros catálogos sin embargo no coinciden del todo con ninguna de las características citadas (por ejemplo, [Coad, 97] y [Pree, 95]). Es posible construir toda una explicación teórica para la coincidencia entre GoF, POSA 1, POSA 2 y [Alur, 2001]. A partir de esta explicación se puede derivar una estructura más amplia y adecuada que la que muestran estos catálogos para organizar el conocimiento de patrones.

Otro aspecto importante de destacar es que los autores de GoF señalan que el catálogo provee un vocabulario de diseño, con lo que establecen el vínculo con la noción de lenguaje de patrones:

"Los científicos de la computación nombran y catalogan algoritmos y estructuras de datos pero no nombramos con frecuencia otros tipos de patrones. Los patrones de diseño proveen un vocabulario para que los diseñadores lo usen al comunicarse, documentar y al explorar alternativas de diseño. Los patrones de diseño hacen que un sistema se vea menos complejo al permitirle a usted hablar a un nivel de abstracción más alto que el que provee la notación de diseño o el lenguaje de programación. Los patrones de diseño elevan el nivel al cual usted diseña y discute sobre diseño con sus colegas" (ver pág. 352 de [Gamma, 95]).

Es bueno aclarar que los autores de GoF no consideran que su libro sea un lenguaje de patrones y admiten ciertas limitantes en este sentido . La noción de lenguaje de patrones también se ha asociado con un esquema para organizar el cuerpo de conocimientos de patrones que evoluciona muy rápidamente. Esta rápida evolución se puede percibir en la frecuente aparición de variantes de patrones y combinaciones de patrones que eventualmente pueden llegar a considerarse patrones en sí mismas, así como en la identificación de nuevas asociaciones con otros patrones del mismo dominio y con dominios de experiencia nuevos. En este contexto, la pregunta clave es si la estructura de los mejores catálogos publicados hasta ahora puede soportar la evolución natural del conocimiento de patrones. El estado del arte sugiere que la aplicación intensiva, en contextos industriales y de procesos de enseñanza y aprendizaje, de este inmenso y cambiante cuerpo de conocimientos, naturalmente requiere de una estructura organizativa tal que, no solamente se facilite su consulta para asistir la resolución de problemas, sino también su reorganización continua dada su rápida evolución natural.

Es posible construir un marco teórico para comprender mejor la evolución del cuerpo de conocimientos de patrones, pero además, a la luz de esta explicación, resulta que es conveniente ampliar la estructura recurrente de GoF, POSA 1, POSA 2 y [Alur, 2001]. Desde este enfoque, la función general de un sistema de patrones (ya no

solamente un catálogo) es la de soportar la evolución natural de lenguajes de patrones afines y facilitar su uso. En este artículo se especifica un patrón para sistemas de patrones que se basa no solamente en el patrón recurrente observado en los catálogos mencionados, sino también en la siguiente caracterización de lenguaje de patrones :

"Un lenguaje de patrones puede caracterizarse como una especialización de un lenguaje natural (es decir un subsistema simbólico empotrado en un sistema más amplio que es un lenguaje natural) cuyas categorías naturales han sido construidas para describir y organizar planes para resolver problemas de diseño de *software*, en forma heurística y en un dominio de diseño específico. Los planes son los patrones y constan a su vez de cinco aspectos básicos,"El Contexto" (C), "El Problema" (P), "La Solución" (S) y "Las Asociaciones con otros patrones" (A)—de acuerdo con [Alexander, 79]. Finalmente "Las consecuencias de aplicar la solución" (CS) que han agregado los ingenieros de *software*. Toda tupla (C, P, S, A, CS) tiene un nombre significativo que constituye una palabra de la especialización y por ende refleja las escogencias gramaticales preferidas por la comunidad de usuarios y autores de patrones." [Calderón, 2003b].

Como subconjunto de un lenguaje natural, un lenguaje de patrones no se puede reducir a un conjunto (o catálogo) de patrones interrelacionados, ni a un sistema de *software* que organice las especificaciones de los patrones y sus interrelaciones. El lenguaje como tal emerge en la intersubjetividad, en el uso cotidiano. Un lenguaje de patrones evoluciona como un lenguaje natural, solo que especializándose en un dominio muy limitado—de fronteras difusas—de experiencias de diseño, al que ha sido enfocado por la comunidad de sus gestores y usuarios cotidianos. Esto significa que continuamente van a emerger nuevas categorías de patrones y nuevos patrones producto de la combinación y variación del conjunto original, así como de la construcción de otros nuevos por parte de sus gestores y usuarios. Por el mismo proceso, un lenguaje de patrones entrelaza asociaciones con otros especializados en dominios de experiencia afines. De la misma forma en que existen diccionarios para los lenguajes naturales, es posible construir uno para un lenguaje de patrones. Un sistema de patrones es básicamente un diccionario "hipervinculado" que sistematiza el conocimiento sobre los términos de un lenguaje de patrones. Sobre esta base, se ha preferido denominar a los sistemas de patrones como lexicones de patrones .

No se discutirá en este documento la justificación teórica de esta caracterización; más bien, siguiendo al tradición heurística del movimiento de patrones, se usará una estructura inspirada en los aspectos "Contexto", "Problema", "Solución" y "Consecuencias" para describir un patrón para lexicones de patrones. En lugar de la sección de referencias a lo largo de todo el trabajo se mencionan catálogos que pueden considerarse ejemplos parciales del patrón propuesto.

3. Problema

Si la función general de un lexicón de patrones es dar soporte al proceso natural de evolución de lenguajes de patrones afines, se puede derivar que el problema central en el diseño de un lexicón de patrones consiste en idear una estructura, por un lado, suficientemente bien definida y natural como para facilitar la búsqueda de patrones y la

resolución de problemas de diseño, y por otro lado, suficientemente flexible como para acomodarse a la evolución natural de los lenguajes afines, lo cual conlleva la continua aparición de patrones nuevos, combinaciones y variaciones típicas, y dominios de experiencia de diseño nuevos pero afines. Cabe aclarar que el problema no consiste en definir una categorización de patrones específica, sino más bien un modelo de categorización de patrones que cada grupo de ingenieros de *software* debe adaptar a sus propias necesidades. Este patrón sí pretende en cambio presentar ciertas heurísticas generales para construir categorizaciones de patrones. Se parte del supuesto de que aunque se puede llegar a cierto grado de coincidencia en cuanto a la categorización de patrones, la evolución acelerada del conocimiento sobre patrones todavía hace muy difícil un acuerdo suficientemente sólido que sustente una categorización óptima.

4. Fuerzas

Para lograr la estabilidad y adaptabilidad requeridas se deben considerar varias tendencias o fuerzas que se analizan a continuación:

4.1 Cada dominio y equipo de ingenieros de *softwar*puede configurar su categorización idiosincrática.

De cada dominio de experiencias de diseño emerge un sistema de categorización idiosincrático. Cada sistema de categorías va a tener distinta anchura y niveles de subcategorización. Por ejemplo, la cantidad de categorías de patrones en GoF, POSA 1, POSA 2 y [Alur, 2001] es distinta y resulta de la construcción particular que cada grupo de autores ha hecho del dominio de experiencias que ha trabajado, aún cuando hayan trabajado a veces el mismo dominio. Más aún, en la actualidad el conocimiento sobre patrones se sigue incrementando día a día muy rápidamente, en estas circunstancias es difícil poder establecer una categorización de patrones universal. Por tanto, es relevante facilitar a cada equipo de ingenierios de *software* la definición de su propia categorización.

4.2 Las fronteras entre dominios de experiencia de diseño son difusas.

Los dominios de experiencia de diseño de *software* tienden a mezclarse en su evolución natural, debido a que las fronteras son difusas; por ejemplo POSA 1 propone un esquema matricial—es decir, categorías con subcategorías—de categorización. Las columnas de tal esquema pueden considerarse de hecho dominios de experiencia distinguibles. Existen patrones como “Singleton” cuya categorización es ambigua, de hecho ha sido incluido tanto en “Patrones de Diseño” como en “Patrones de Programación” (o “idioms”). El patrón “Composite Entity” de [Alur, 2001] se puede considerar una variante de “Whole-Part” de POSA 1, pero a la vez es de hecho un “Patrón de Programación”. Todo esto contribuye a que las fronteras entre dominios sean difusas. Ésta es una característica muy poderosa de los lenguajes naturales que

debe aprovecharse en un lenguaje de patrones y por tanto debe ser representada en un lexicón de patrones. Desde este punto de vista, no se trata de eliminar este tipo de ambigüedades, sino más bien de sacar provecho de estas, tal como sucede en un lenguaje natural. Por ejemplo, la categorización de un patrón en distintos dominios puede facilitar su búsqueda para distintos usuarios que de previo suponen que se ubica en distintas categorías.

4.3 Las categorías y dominios son parte esencial de los vocabularios de diseño.

Los nombres de las categorías de patrones y de los dominios de experiencia tienden a usarse en las referencias a los catálogos y en las búsquedas que hacen los ingenierios de *software* como parte de la resolución de problemas de diseño. De hecho estos nombres también son parte esencial de los vocabularios organizados por los catálogos publicados hasta ahora. La cantidad de categorías de patrones también se incrementa constantemente, aunque no al mismo ritmo que la de los patrones.

4.4 De cada patrón tienden a emerger variantes típicas.

De cada patrón propuesto van a emerger muchas variantes con el tiempo. Algunas de éstas se van a repetir al igual que los patrones que las originaron. Tanto las variantes típicas como la virtualmente infinita cantidad de aplicaciones va a estar implicada de alguna forma en el uso del nombre del patrón, de manera análoga a como usamos cualquier palabra de un lenguaje natural. Por ejemplo, en el contexto de una conversación entre diseñadores de *software* que están considerando la posibilidad de aplicar el patrón "EJB de Entidad Compuesta" (“Composite Entity” de [Alur,2001]) van a estar implicadas las variantes originadas por las distintas estrategias de programación que de hecho se documentan en [Alur, 2001]. En un momento dado, la necesidad de mantener un cierto nivel de abstracción evitará que los actores discutan sobre las variantes, pero en otro momento, cuando se haya llegado a un acuerdo sobre la conveniencia de aplicar el patrón por ejemplo, definitivamente las variantes se harán explícitas.

En este contexto son relevantes los casos en que las variaciones son típicas y por lo tanto pueden ser consideradas en sí mismas patrones. El surgimiento de variaciones típicas podría basarse por ejemplo en la adaptación de patrones a tecnologías de programación específicas. Este es el caso de "Composite Entity" de [Alur, 2001] que puede considerarse una variación típica de "Composite" de GoF adaptada a la tecnología J2EE . Como todo patrón, una variante que llegue a ser típica tendrá su propio nombre. Los nombres de las variantes típicas incluirán elementos de los nombres de los patrones base.

4.5 De cada conjunto de patrones tienden a emerger combinaciones típicas.

Los patrones siempre son usados en combinación con otros. Algunas de estas combinaciones se van a repetir y podrían llegar a ser consideradas patrones en sí mismas. Se puede citar como ejemplo "Controlador de Fachada" que se basa en "Controller" de [Larman, 98] y "Facade" de GoF. En términos generales, a un objeto "Controlador de Fachada" se le asigna la función de resolver los comandos solicitados por el usuario distribuyendo mensajes apropiados entre un subsistema de objetos subsidiarios. A la vez, el "Controlador de Fachada" provee una interfaz simplificada para el sistema de objetos subsidiarios, una interfaz de alto nivel que oculta la complejidad del conjunto de interfaces provistas por cada uno de los objetos subsidiarios. Combinaciones típicas como ésta, derivan su nombre de la combinación de los nombres de los patrones originales; típicamente el nombre de uno de los patrones funcionará como adjetivo del otro.

4.6 Cada dominio de experiencias configura sus propias plantillas.

De cada dominio de experiencias de diseño emerge una plantilla propia para la especificación de patrones. De hecho se conocen a la fecha varias plantillas . No parece conveniente estandarizar un esquema, más bien una parte importante de la comunidad de autores y usuarios de patrones asociados a la construcción de *software* ha optado por especificar con base en cinco aspectos primarios, cuatro tomados de [Alexander, 79] y otro que se ha agregado en catálogos como GoF y POSA 1, y que ha sido aceptado por la comunidad en general:

- Contexto (C).
- Problema (P).
- Solución (S).
- Consecuencias de aplicar la Solución (CS).
- Asociaciones con otros patrones (A).

4.7 Las categorías de asociaciones relevantes están cambiando.

"En esta red, las asociaciones entre patrones son parte del lenguaje tanto como los patrones mismos." (ver pág. 314 de [Alexander, 79]).

En la comunidad de usuarios y autores de patrones se está haciendo un gran esfuerzo por identificar asociaciones relevantes. Se considera que esto es esencial para lograr mayores grados de expresividad en los lenguajes de patrones. Hasta ahora, se ha puesto mucho énfasis en las asociaciones de la categoría de complementariedad, como cuando un patrón X se puede usar para aplicar otro patrón Y.

Se pueden considerar otras categorías de asociaciones relevantes en un sistema o catálogo de patrones. Por ejemplo, en [Salingaros, 2000] se consideran algunas como las siguientes:

· "Un patrón contiene o generaliza a otro de escala más pequeña.

· Dos patrones se complementan mutuamente.

· Dos patrones resuelven problemas distintos que se traslapan y coexisten en el mismo nivel.

· Dos patrones resuelven el mismo problema en formas igualmente válidas y —por ende—alternativas.

· Patrones distintos comparten una estructura de solución similar lo que implica una conexión a un nivel más alto." (ver páginas 153 y 154 de [Salingaros, 2000]).

Lo cierto es que la red de asociaciones relevantes entre patrones es muy dinámica y entrelaza a patrones de un mismo dominio, de diferentes dominios y de diferentes niveles de abstracción, esto en virtud de que "... la cualidad innombrada aparece, no cuando un patrón vive aislado, sino cuando un sistema completo de patrones, interdependientes a varios niveles, deviene completamente estable y vivo" (ver página 135 de [Alexander, 79]).

Más aún, desde un enfoque teórico como el propuesto en [Calderón, 2003b], se puede argumentar que existen otras categorías de asociaciones relevantes, por ejemplo, asociaciones de complementariedad entre categorías de patrones. Considérese la asociación de complementariedad entre "patrones de estructuración" y "patrones de comportamiento" de GoF. Ésta deriva, entre otras razones, de la complementariedad entre "Compuesto" e "Iterador". En términos más generales, este enfoque lleva a considerar otras categorías de asociaciones tales como:

· Extensión, denominada literalmente en [Lakoff, 87] “instanciación”. En el contexto de patrones se puede interpretar como que un patrón tiene asociado un vector de atributos Y que está contenido en el vector X de un segundo patrón, el énfasis se pone en que X agrega otros atributos a Y, o "X extiende a Y".

· Similaridad, denominada literalmente en [Lakoff, 87] “similarity”. Los vectores de atributos de X y Y correspondientes a dos patrones, tienen algunos atributos en común, pero también hay otros en que difieren. El énfasis se pone en el subconjunto de atributos en común.

· Transformacional, denominada literalmente en [Lakoff, 87] “transformational”. Los vectores de atributos de X y Y de dos patrones también tienen en este caso un subconjunto en común. Tienen atributos en los que difieren, pero aquí el énfasis se pone en el tipo de relación específica que existe entre los atributos en que difieren. Este tipo de asociación existe entre un patrón X y una de sus variantes Xv. Si X resulta de la combinación de Y y Z, también existe una asociación de tipo transformacional entre Y y X, y Z y X.

Finalmente cabe apuntar que identificar y caracterizar apropiadamente cualquier categoría de asociación que facilite la búsqueda de patrones, su aplicación en contextos específicos, la resolución de problemas de diseño, el aprendizaje o la enseñanza del diseño de *software* con base en patrones, va a mejorar la expresividad de los lenguajes de patrones. La sistematización de tales asociaciones es por lo tanto esencial.

4.8 La red de dominios de diseño es muy inestable

El concepto de dominio de diseño es una adaptación de lo que en [Lakoff, 87] se denomina el principio del dominio de experiencia:

"Si existe un dominio básico de experiencia asociado con A, entonces es natural que todas las entidades de este dominio estén en la misma categoría que A" (página 93 de [Lakoff, 87]).

Hasta la fecha, con base en los catálogos publicados, se pueden considerar cuatro dominios generales, es decir, independientes del dominio de aplicación:

4.8.1 Patrones de modelaje de requerimientos ([Hay, 96], [Fowler, 97] y [Coad, 97]).

4.8.2 Patrones arquitectónicos (POSA 1 y POSA 2, aunque también contienen patrones de diseño).

4.8.3 Patrones de diseño (GoF) y [Alur, 2001].

4.8.4 Patrones de programación, estilos o "idioms" en inglés ([Coplien, 92] y [Alur, 2001]).

La principal asociación entre estos dominios es de complementariedad. De acuerdo con el dominio de la aplicación se debe escoger un patrón arquitectónico. Luego los patrones de diseño complementan la aplicación del patrón arquitectónico escogido y a su vez, los modismos o patrones de programación complementan la aplicación de los patrones de diseño escogidos.

A estos dominios genéricos se deben agregar los dominios de aplicación. Un dominio de aplicación constituye una categoría difusa y compleja de aplicaciones, relacionadas entre sí por el tipo de problemas y contextos que direccionan, así como por los requerimientos funcionales genéricos y las restricciones de desempeño o de plataforma tecnológica que normalmente les acompañan.

Por ejemplo un dominio de aplicación o dominio de experiencias podría ser lo que los ingenierios de *software* denominan "sistemas de información

empresariales" o "sistemas de información para los negocios", y un buen ejemplo de aplicación o sistema de *software* propio de este dominio es un sistema de contabilidad. El siguiente diagrama presenta un panorama amplio de los distintos dominios de diseño de *software* que podrían ser relevantes (cada rectángulo representa un posible dominio).

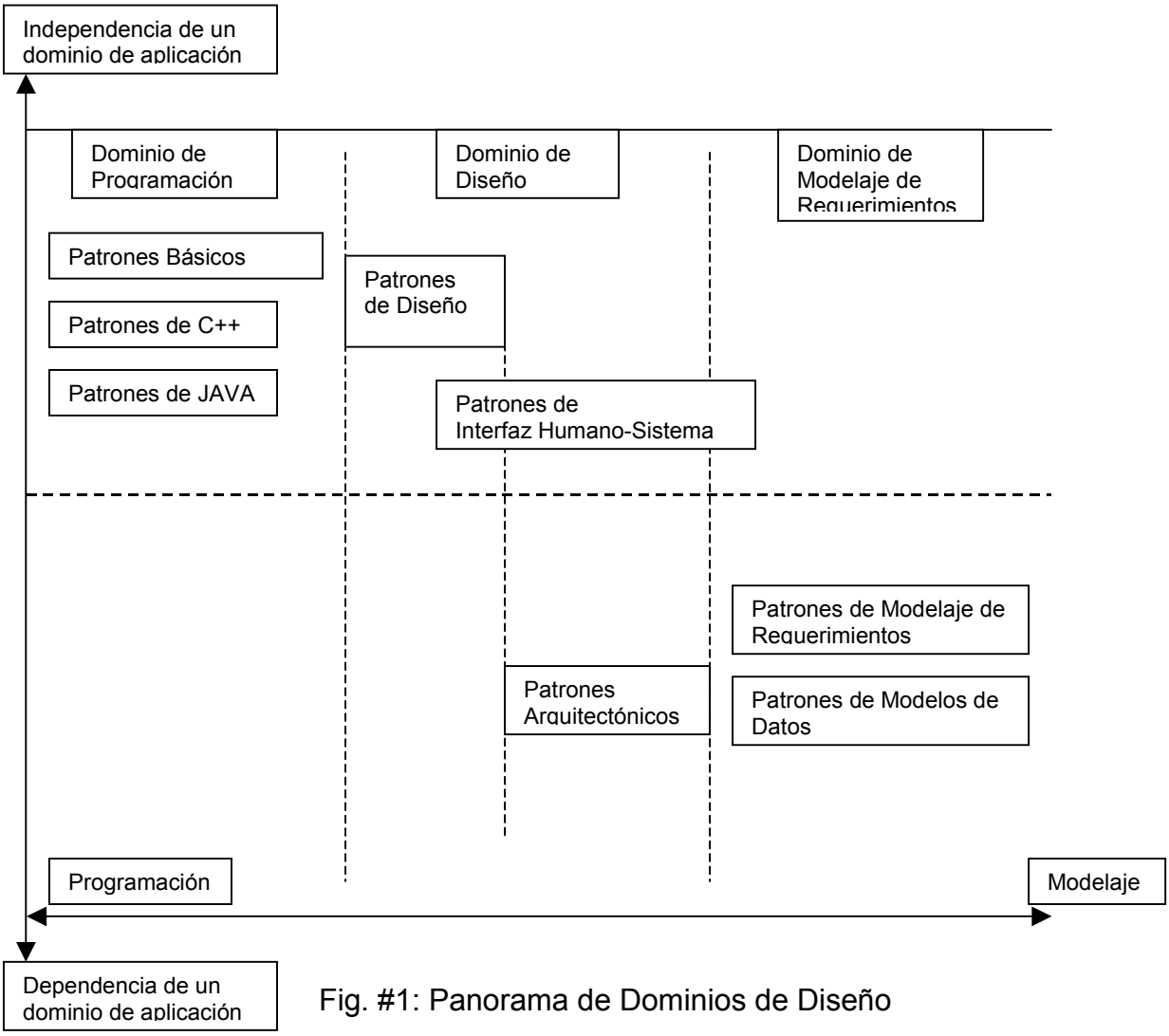

Fig. #1: Panorama de Dominios de Diseño

En la figura #1 se intenta presentar un continuo de dominios de diseño vinculados a lo que podría denominarse en términos generales el dominio de diseño de *software*. En el extremo superior del eje vertical del mapa se sitúan todos aquellos conocimientos que son suficientemente generales como para ser útiles en cualquier dominio de aplicación. En el extremo inferior del mismo eje, en cambio, se sitúan todos aquellos esfuerzos tendientes a la producción de conocimientos orientados a un dominio de aplicación específico. Entonces, la franja superior del mapa representa dominios genéricos de patrones, independientes de cualquier dominio de aplicación. La franja inferior representa, por el contrario, dominios de diseño orientados a un dominio de aplicación específico; del lado izquierdo aparecen los conocimientos asociados a la programación y del lado opuesto los asociados al modelaje. En el eje horizontal

se han representado tres de las áreas típicas del proceso de desarrollo de *software*: las tres columnas representan al análisis de requerimientos, diseño y programación, actividades del proceso de desarrollo de *software*. La columna de diseño incluye tanto el diseño arquitectónico como el diseño detallado de objetos.

Todas las fronteras son tenues y difusas, dependen, entre otras cosas, del grado de abstracción de cada patrón propuesto. Por ejemplo, [Hay, 96] propone patrones de modelos de datos orientados a dominios de aplicación específicos, pero también patrones de modelos de datos presumiblemente independientes de cualquier dominio de aplicación. "Singleton" de GoF podría verse como un patrón de programación independiente del lenguaje. Los patrones arquitectónicos están fuertemente orientados a dominios de aplicación ("Layers" de POSA 1 a aplicaciones empresariales distribuidas, "Pipes and Filtres" a la construcción de intérpretes o compiladores). Los patrones de interfaz humano-sistema como "Composite View" de [Alur, 2001], tienden a ser independientes del dominio de aplicación, pero pueden ser muy útiles en la actividad de modelaje de requerimientos para la elaboración de prototipos, y a la vez tienen implicaciones arquitectónicas que llegan a influir sobre el diseño de objetos.

Desde esta perspectiva se puede comprender que la red de asociaciones entre todos estos dominios es muy profusa y dinámica. Sin embargo su sistematización es esencial para facilitar la resolución de problemas de diseño basada en patrones. A la fecha no es posible anticipar la evolución que va a tener esta red de dominios. Un lexicón de patrones debería ser suficientemente flexible para adaptarse a la evolución natural de esta red de dominios sin prescribir un conjunto específico. Todos los dominios citados tan solo son ejemplos de los que podrían incluirse en un lexicón específico.

5. Solución

La solución abarca cuatro aspectos:

· Modelo de navegación de funciones: es un modelo de casos de uso que se propone como esquema genérico de las funciones que se puede considerar al construir un lexicón de patrones y de sus conexiones relevantes.

· Modelo de categorización de patrones: es un modelo conceptual de objetos que se propone como representación de la estructura de categorización de un lenguaje de patrones.

· Modelo de categorización de asociaciones: es un modelo conceptual de objetos que representa las principales relaciones entre categorías de asociaciones que se pueden considerar al construir un lexicón de patrones.

· Clases de plantillas de especificación: es un modelo conceptual de objetos que representa las distintas clases de plantillas para especificar los distintos tipos de elementos que conforman un lexicón de patrones.

Esta solución se sustenta en la perspectiva de los lenguajes de patrones como subconjuntos de los lenguajes naturales y en la teoría semántica cognoscitivista de Lakoff [Lakoff, 87]. La teoría de los Modelos Cognoscitivos Idealizados (MCIs) de Lakoff propone que cada categoría natural se puede modelar como un MCI, propone varias categorías de MCI y principios estructurantes, así como una organización de los MCI en tres niveles (supra-ordinario, ordinario y sub-ordinario) dado un dominio de experiencias. La teoría de Lakoff es relevante por cuanto los patrones pueden modelarse como MCI, lo que posibilita definir un modelo de categorización que permite construir lexicones de patrones estables y adaptables a la evolución natural de los lenguajes de patrones.

Para representar los distintos modelos que conforman la solución se ha optado por el UML ("Unified Modeling Language"), dado el uso tan extendido de este lenguaje gráfico entre ingenieros de *software*. Para una explicación completa de UML se refiere al lector a [Booch, 99].

5.1 Modelo de navegación de funciones

Aunque este patrón para lexicones de patrones también se puede aplicar a la construcción de catálogos de patrones, la estructura de funciones no se puede usar en tal caso. Las dos principales funciones de un lexicón de patrones están representadas por los casos de uso "Editar el Lexicón" y "Navegar por el Lexicón".

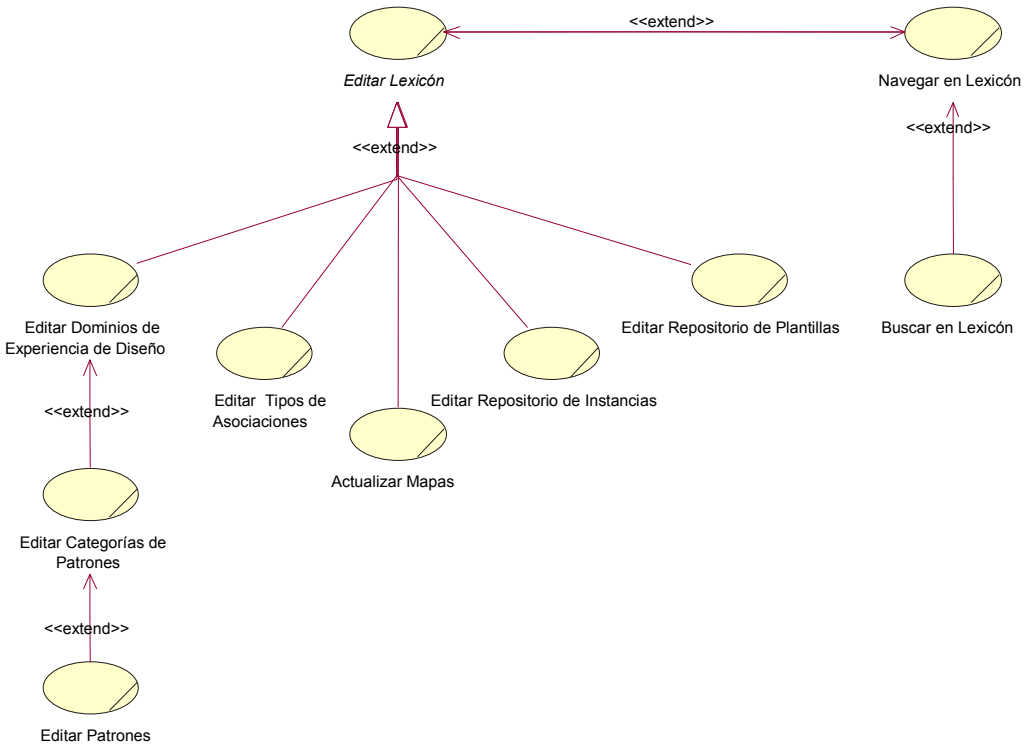

Fig. #2: Modelo de Navegación de Funciones

5.1.1 "Editar Lexicón"

Como caso de uso abstracto, representa la función general de actualizar las estructuras de categorización, de asociaciones y de especificación de los patrones y a la vez es un punto de acceso a otros casos de uso de un lexicón de patrones (en particular a "Navegar por el Lexicón", ver sección 5.1.2):

a. "Editar Dominios de Experiencias de Diseño (DED)": Permite editar la lista de términos usados para denominar los distintos dominios de experiencia de diseño (DED) abarcados por un lexicón de patrones. Esto significa que permite editar sus especificaciones y sus relaciones.

b. "Editar Categorías de Patrones (CP)": Permite editar la lista de términos usados para denominar las distintas categorías de patrones (CP) de un DED. Esto significa que permite editar sus especificaciones y sus relaciones.

c. "Editar Patrones (P)": Permite editar la lista de términos con que se conoce a los patrones mismos (P) de una CP de un DED. Esto significa que permite editar sus especificaciones y sus relaciones. Esta funcionalidad se aplica igual sobre patrones base, así como sobre patrones de variantes de patrones y patrones de combinaciones de patrones.

d. "Editar Tipos de Asociaciones": Permite editar la lista de términos con que se conocen los tipos o categorías de asociaciones (TA). Esto significa que permite editar sus especificaciones.

e. "Editar Repositorio de Instancias": Permite editar el conjunto de instancias de patrones disponibles en un lexicón. Cada patrón puede tener asociadas varias instancias. Una de éstas puede ser declarada instancia prototípica (asociación "prototipo de"). Esta función se aplica igual sobre patrones base, como sobre patrones de variantes de patrones y patrones de combinaciones de patrones.

f. "Editar Repositorio de Plantillas": Permite editar el conjunto de plantillas disponibles en un lexicón. Cada elemento de un lexicón de patrones tiene asociada una especificación que se basa en una plantilla. Esto significa que permite editar las plantillas mismas y sus especificaciones.

g. "Actualizar Mapas": Algunas de las plantillas para especificar elementos de un lexicón de patrones convenientemente incluirán mapas de asociaciones entre los elementos subsidiarios. Por ejemplo, una categoría de patrones incluye un mapa que muestra los patrones de la categoría y sus asociaciones. Estos mapas se desactualizarán fácilmente con la realización de operaciones de edición sobre un lexicón. Por esta razón es útil automatizar la actualización de mapas a partir de los cambios recientes en las especificaciones de los elementos de un lexicón de patrones.

5.1.2 "Navegar por Lexicón"

Es un caso de uso concreto y representa la función general que permite recorrer los documentos de un lexicón de patrones, a través de los hipervínculos gráficos de los

mapas y los hipervínculos textuales. A la vez es un punto de acceso a los demás casos de uso de un lexicón de patrones (en particular a "Editar el Lexicón", ver sección 5.1.1):

a. "Buscar en Lexicón": Permite hacer búsquedas de documentos basadas en texto libre, o en descriptores. Estas búsquedas se pueden aplicar a un dominio, a una categoría o a un patrón.

5.2 Modelo de categorización de patrones

La categorización de un lexicón de patrones se basará en la estructura de categorización típica de los lenguajes naturales, dado que un lenguaje de patrones es básicamente una especialización de un lenguaje natural. Por tanto, dado un dominio de experiencias de diseño, una de categorización de patrones se basará en tres niveles: básico u ordinario, supra-ordinario (más abstracto y general que el básico) y sub-ordinario (más concreto y específico que el básico) . El nivel básico u ordinario es el más usado, el más fácil de aprender, el más fácil de recordar, el que corresponde con la percepción gestáltica, el que corresponde con la mayor capacidad de formar imágenes (ver páginas de 31 a 56 de [Lakoff,87]). Para una explicación más detallada en el contexto de lenguajes de patrones se remite al lector a [Calderón, 2003b].

La figura #3 muestra los elementos esenciales del modelo de categorización para un lexicón de patrones, el nivel de categorización al que pertenecen y sus asociaciones. Para cada elemento se introduce un estereotipo, con el fin de construir representaciones en UML del diseño de un catálogo o lexicón de patrones:

5.2.1 Dominio de Experiencias de Diseño (<<DED>>):

Corresponde con el principio de dominio de experiencia descrito en [Lakoff, 87]. Ejemplos de dominios de experiencia que han sido elaborados en catálogos publicados son: "diseño detallado de objetos" en [Alur, 2001], GoF y POSA 1, "diseño arquitectónico" en [Alur, 2001], POSA 1 y POSA 2, "estilos de programación" (o "idioms" en inglés) en [Alur, 2001]. Tal como se analizó en la sección , existen principalmente asociaciones de complementariedad entre los DED, por ejemplo entre "diseño arquitectónico" y "diseño detallado de objetos".

5.2.2 Categoría de Patrones (<<CP>>):

Estos elementos pertenecen al nivel supra-ordinario descrito en [Lakoff, 87]. Todo DED se dividirá naturalmente en varias categorías de patrones (CPs). Por ejemplo, en GoF existen tres categorías básicas de patrones: "patrones de construcción", "patrones de estructuración" y "patrones de comportamiento" las cuales a su vez se subdividen en dos sub-categorías: "patrones de objetos" y "patrones de clases". En cada categoría se identificará el patrón prototípico, es decir el patrón que mejor representa a la categoría.

Esto proveerá a la categoría una estructura más natural que facilitará procesos de aprendizaje (ver [Calderón, 2003b] para una explicación más detallada).

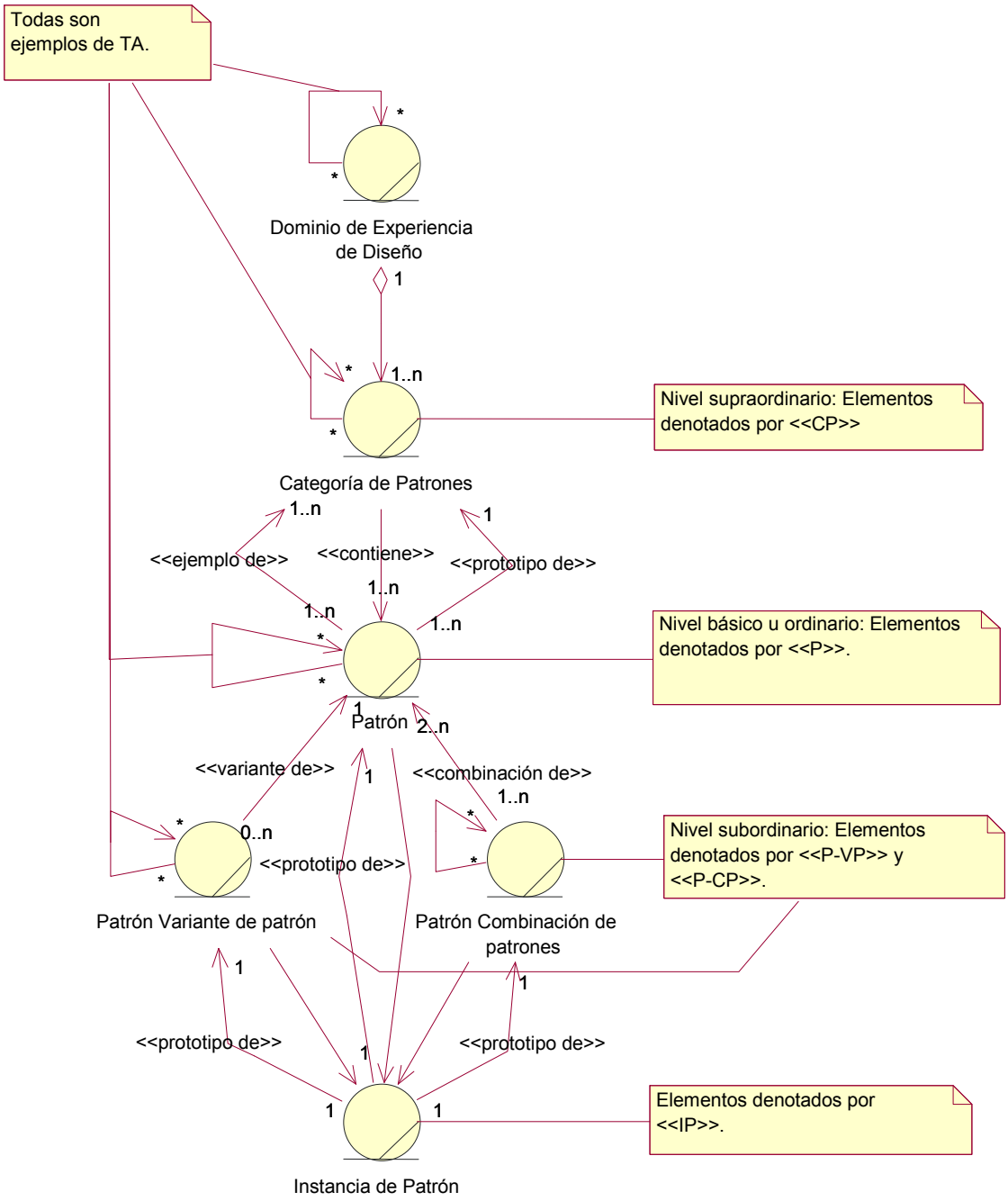

Fig. #3: Modelo de Categorización

5.2.3 Patrón (<<P>>):

Estos elementos pertenecen al nivel de categorización básico descrito en [Lakoff, 87] u básico u ordinario, pues son de hecho los que más se usan al resolver problemas de diseño. Para una explicación detallada de por qué conviene considerar a los patrones como categorías, se remite al lector a [Calderon, 2003b]. En GoF se presentan 23 patrones que muestran distintos tipos de asociaciones. Algunas de estas asociaciones se muestran en el mapa (ver solapa posterior de [Gamma, 95]), otras se analizan en el texto especificatorio de cada patrón.

5.2.4 Patrón de Variante de Patrón (<<P-VP>>):

Estos elementos pertenecen al nivel sub-ordinario de categorización descrito en [Lakoff, 87]. Son mucho más específicos que los del nivel básico u ordinario (los patrones base). Un buen ejemplo es el patrón "EJB de Entidad Compuesta" de [Alur, 2001] que resulta ser una adaptación de "Compuesto" de GoF a la tecnología JAVA .

5.2.5 Patrón de Combinación de Patrones (<<P-CP>>):

Estos elementos también pertenecen al nivel sub-ordinario de categorización descrito en [Lakoff, 87]. Al resultar de la combinación de patrones base, tienen asociados contextos de uso mucho más reducidos o específicos. Un buen ejemplo de patrón de combinación de patrones es "Controlador de Fachada" que se propone en [Larman, 98]).

El principio del continuo entre gramática y lexicón descrito en [Ellis, 93], opera naturalmente al darle nombre a las <<P-VP>> y <<P-CP>>. Por esta razón, se deberá poner especial atención en la función gramatical que se asigna a los nombres que componen el nombre de una variante o combinación. Por ejemplo, los nombres citados: "EJB de Entidad Compuesta" y "Controlador de Fachada", no podrían invertirse sin correr el riesgo de causar confusión en el lector de un catálogo o usuario de un sistema de patrones, con respecto a la intencionalidad de estos patrones de nivel sub-ordinario.

Los principios más importantes que subyacen este modelo de categorización con tres niveles de abstracción dado un dominio son, por un lado, estabilidad y flexibilidad ante cambios en la cantidad de patrones. Por otro lado adaptabilidad a las necesidades y preferencias de cada grupo de ingenieros de *software* específico. En última instancia se busca que los usuarios de un lexicón puedan organizar los patrones que consideren relevantes en la forma en que mejor les convenga, estableciendo sus propios dominios, categorías de patrones, patrones, variantes y combinaciones. Sin embargo, el presente patrón sí ofrece una serie de principios que buscan optimizar cada categorización idiosincrática. A través del modelo de categorización se establece que, dado un dominio de experiencias, debe incluirse tres niveles de abstracción (no cinco ni dos) y también se establece la funcionalidad de cada nivel. El sustento para estos principios deriva

teóricamente de la organización de los lenguajes naturales según la teoría de los modelos cognoscitivos idealizados propuesta en [Lakoff, 87].

Otro principio que cabe destacar es que no se busca definir fronteras precisas entre las categorías de los distintos niveles. Los estudios empíricos que reporta Lakoff sustentan el hecho de que los MCI no introducen fronteras precisas entre las categorías naturales. Esta característica de los lenguajes naturales no solo ha sido corroborada empíricamente sino que, como principio de diseño de lexicones de patrones, además es útil entre otras razones porque:

1. Facilita las búsquedas de patrones, pues se generan más caminos para llegar a cada patrón, categoría de patrones o dominio de experiencias de diseño.
2. Implica que se deben hacer explícitas relaciones de similitud y contraste entre patrones, lo cual facilita las búsquedas.
3. Implica que se deben hacer explícitas relaciones de complementariedad entre patrones, lo cual facilita el diseño basado en patrones.

5.3 Modelo de categorización de asociaciones

Cada categoría de asociaciones puede ser ejemplo de otra y a la vez pueden existir varios ejemplos de la misma. Idealmente el modelo de categorización de asociaciones no tendrá más de tres niveles. Para cada categoría de asociación se elaborará una especificación con base en una plantilla (ver sección de plantillas para los elementos de un lexicón de patrones).

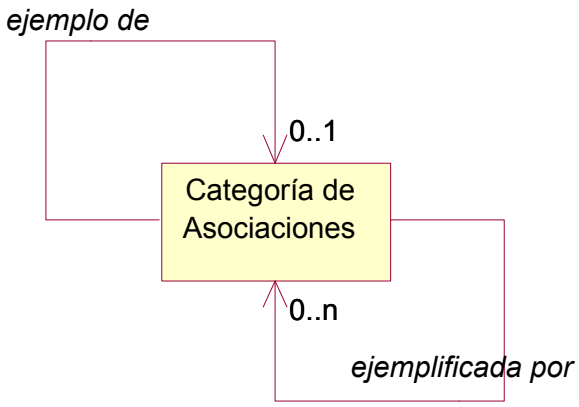

Fig. #4: Categorización de Asociaciones

La enumeración de las distintas categorías de asociaciones en un lexicón de patrones no es el objetivo de este trabajo (como tampoco la definición de una categorización específica de patrones); para efectos de esta propuesta lo importante es que debe considerarse a las asociaciones como elementos esenciales de un lenguaje de

patrones y por ende deben poder categorizarse y sistematizarse, al igual que los demás elementos analizados anteriormente. Esto implica que:

a. debe darse nombre a los tipos de asociaciones entre elementos de un lexicón de patrones, y

b. que los tipos o categorías naturales de asociaciones entre elementos de un lexicón de patrones deben ser visualizados como patrones de asociaciones. Por tanto debe existir una plantilla para especificar categorías de asociaciones.

5.4 Clases de Plantillas de especificación

En un lexicón de patrones, todas las instancias de los tipos de elementos analizados hasta ahora (los tipos de elementos estructurales de un lenguaje de patrones, a saber: <<DED>>, <<CP>>, <<P>> y <<TA>>, así como <<P-VP>> y <<P-CP>>) deben ser especificados con base en una plantilla. Una plantilla estructura los atributos relevantes de una instancia de un tipo de elemento.

Aunque tienen muchas características en común, los catálogos que han servido de referencia a este trabajo (GoF, POSA 1, POSA 2 y [Alur, 2001]) no muestran un acuerdo en torno a la plantilla para especificar patrones (con excepción de POSA 1 y POSA 2). Por otro lado, a manera de ejemplo, los patrones del dominio de modelaje de requerimientos, requieren una sección de "estructura" o "solución" más compleja que la que usan los catálogos referidos . Inclusive la especificación de patrones atípica observada en [Hay, 96], [Coad, 97], [Coplien, 92], [Fowler, 97] y [Pree, 95] dudosamente podría atribuirse a una decisión arbitraria de sus autores. Es factible que los mismos dominios de experiencia tratados por ellos hayan inducido otro tipo de estructuraciones. Estos autores se aproximan más a la conocida “Forma de Portland” que a la más rígida propuesta por los catálogos mencionados.

La figura #5 muestra las distintas clases de plantillas que se deben incluir en un lexicón de patrones. La “PE de P” aparece como una clase abstracta para representar el hecho de que existen diferentes plantillas para patrones. En [Calderón, 2003c] se amplía el diagrama incluyendo los esquemas propuestos por GoF, POSA 1 y POSA 2 como especializaciones de la clase genérica de plantillas para patrones. Las plantillas para variantes de patrones (“PE de VP”) y para combinaciones de patrones (“PE de CP”) aparecen como ejemplos de esta misma por cuanto básicamente se orientan a especificar patrones más especializados. Al aplicar el patrón de lexicones de patrones habrá que escoger cuál de las plantillas para especificar patrones se usará para especificar variantes y combinaciones de patrones.

Tal como se aprecia en la figura 5, se proponen atributos específicos para los demás elementos que tradicionalmente no son considerados en forma explícita en los catálogos de patrones (dominios de diseño, categorías de patrones y categorías de asociaciones). Con base en la práctica de cada grupo de ingenieros de *software*, estas

plantillas podrán adaptarse a sus propias necesidades, el principio que subyace es que se debe hacer explícito cómo es que el grupo de trabajo está organizando sus patrones, cuáles son los dominios, las categorías de patrones, las variantes y combinaciones, así como las categorías de asociaciones. Este también es conocimiento esencial de un lenguaje de patrones que no puede seguirse dejando implícito porque de hecho evoluciona constantemente con base en la experiencia de los diseñadores.

Se ha incluido una plantilla para especificar aplicaciones de patrones ("PE de IP"). Todos los catálogos mencionados incluyen al menos un ejemplo de aplicación de cada patrón. Esta sección típicamente no muestra una estructuración muy rígida, por esta razón solo se han incluido tres atributos que se han considerado esenciales con base en lo observado de los catálogos estudiados.

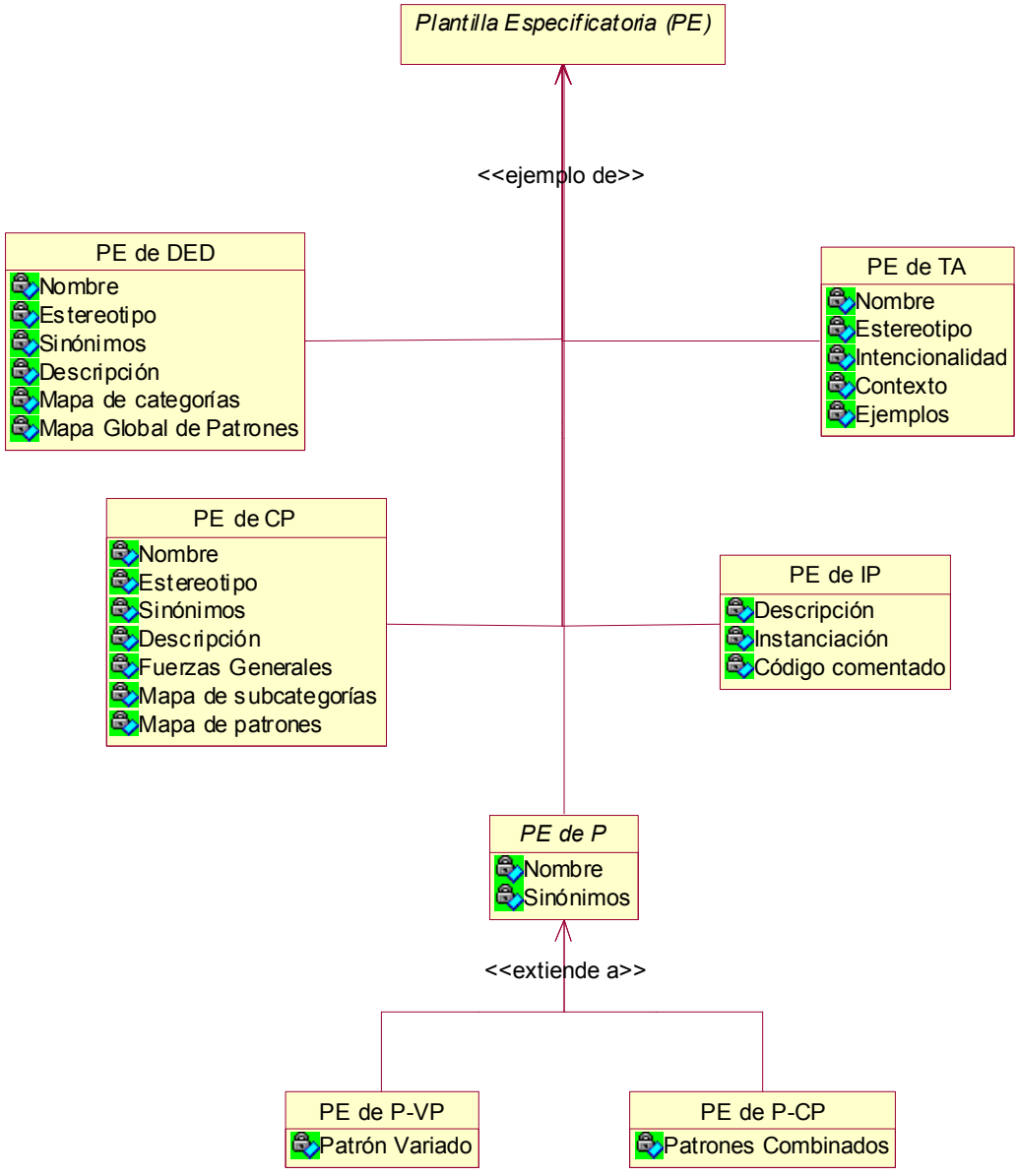

Fig. #5: Clases de Plantillas de Especificación

Al igual que en las secciones anteriores de la solución, no se trata de proveer plantillas específicas, sino más clases de plantillas que deben considerarse, así como la estructura básica de cada plantilla que deberá adaptarse. Por tanto, esta propuesta debe acomodarse a la evolución natural de un lexicón de patrones específico.

6. Ejemplos

En la medida en que catálogos como GoF, POSA 1, POSA 2 y [Alur, 2001] muestran categorizaciones que coinciden parcialmente con el modelo de categorización propuesto en la sección 5.2, puede afirmarse que son ejemplos parciales de este patrón de lexicones de patrones. Todos estos catálogos, como se ha visto, incluyen al menos el nivel supra-ordinario y básico u ordinario en sus clasificaciones. En algunos casos se hace referencia a variantes. Particularmente se puede citar la referencia a las variantes de "Proxy" en POSA 1 y las denominadas "estrategias" ("strategies") en [Alur, 2001]. En estos casos se empieza a vislumbrar un nivel sub-ordinario.

Estos catálogos sin embargo no incluyen explícitamente una especificación del dominio o dominios que abarcan, ni de las categorías de asociaciones, ni de distintas clases de plantillas. Por esta razón sólo pueden considerarse ejemplos parciales de este patrón.

En [Calderón, 2003c] se describe en detalle cómo se podrían integrar los patrones de GoF y POSA 1 utilizando la solución propuesta en este patrón. [Calderón, 2003c] es de hecho el ejemplo más completo de este patrón que por razones de espacio se elaboró en un documento aparte.

7. Consecuencias

Ventajas de usar el patrón propuesto para construir lexicones de patrones:

1. Se facilita la integración de lenguajes de patrones afines, pues la cantidad de dominios de diseño (<<DED>>) queda abierta, se pueden trazar asociaciones entre éstos y además es posible incorporar nuevas plantillas especificatorias de patrones que se adapten a los nuevos dominios.
2. Se facilita la reorganización continua de un lenguaje de patrones pues la red de categorías de patrones (<<CP>>) para cada <<DED>> también queda abierta y se pueden trazar asociaciones entre éstas.
3. Se facilita la integración del conocimiento a todos los niveles de categorización y dominios de diseño, pues la cantidad de tipos de asociaciones (<<TA>>) queda

abierta para que se adapte fácilmente a la evolución natural de los lenguajes de patrones afines.

4. Se facilita la sistematización del conocimiento sobre variantes y combinaciones de patrones puesto que es posible integrar los que surjan nuevos, trazar asociaciones entre ellos, además de categorizarlos y representarlos en mapas.

5. Se facilita la sistematización del conocimiento sobre el uso de patrones, pues el repositorio de instancias de patrones se puede incrementar y reorganizar continuamente.

6. Se facilita la búsqueda de patrones, pues se representan esquemas de categorización propios del lenguaje natural y se pueden introducir nuevos tipos de asociaciones. Además es posible introducir asociaciones útiles específicamente para la búsqueda de patrones (ver [Calderón, 2003c]).

7. Se facilita la resolución de problemas de diseño basada en patrones, pues es posible recorrer secuencias de patrones pertenecientes a distintos dominios complementarios. La resolución de problemas no solo está asociada con la identificación oportuna de patrones complementarios, sino también alternativos. Esto se puede lograr introduciendo categorías de asociaciones adecuadas (esto se muestra en [Calderón, 2003c]).

8. Se facilita el aprendizaje de diseño, lo cual puede ser relevante no solo para aprendices, sino también para los expertos que están incursionando en nuevos dominios de diseño. Esto por cuanto se propone una estructura de categorización cercana a la natural, se identifican prototipos de patrones en las categorías de patrones y se pueden introducir además categorías de asociaciones que faciliten la identificación de patrones similares y contrastantes.

9. Como representación de la estructura de categorización de un lenguaje de patrones, el esquema propuesto enriquece la representación de los lenguajes de patrones en comparación con las propuestas actuales que no toman en cuenta los distintos niveles de categorización propios del lenguaje natural, ni la estructura interna de las categorías, ni hacen explícito el uso de las distintas categorías de asociaciones.

La principal desventaja de usar el esquema propuesto es que probablemente implica mucho esfuerzo de actualización continua. Aunque un lexicón basado en este patrón tendrá una operación automática como "Actualizar mapas" para facilitar esta tarea, el uso de un sistema basado en este patrón en un ambiente real (sea industrial o académico) indudablemente permitirá identificar variantes de este patrón que incorporen nuevas funciones orientadas a disminuir el esfuerzo de actualización. Otras desventajas son:

1. Los lexicones basados en este patrón no promueven una unificación de la categorización de patrones. En el largo plazo, esto podría verse como desventaja, pues sí parece útil buscar, poco a poco, una unificación de criterios. Sin embargo, al hacerse explícitos los criterios de una organización específica de patrones (en términos de dominios, categorías etc.) es posible que se simplifiquen las discusiones tendientes a la definición de una categorización universal, pues cada grupo de ingenieros de *software* puede comparar su esquema de categorización con el de otros grupos más fácilmente.

2. Una categorización de patrones basada en tres niveles de abstracción para cada dominio de experiencias, como la que resultaría de aplicar este patrón, siempre va a ser más compleja que cualquier otra basada en menos niveles, sin embargo, el costo en términos de complejidad se compensa con el beneficio en términos de estabilidad, flexibilidad y adaptabilidad.
3. La construcción de herramientas para administrar el conocimiento de patrones es probablemente más difícil si se adopta este patrón, pero se reitera la ventaja en términos de mayor estabilidad, flexibilidad y adaptabilidad.

Finalmente, cabe destacar que el sistema propuesto permite imaginar el desarrollo ulterior de una nueva generación de asistentes inteligentes de diseño de *software* basados en patrones. Una implantación adecuada de la solución propuesta, serviría como base de conocimiento para un sistema que, utilizando técnicas apropiadas de inteligencia artificial, podría asistir al usuario de un lexicón en la búsqueda de patrones útiles para la resolución de su problema de diseño.

Referencias

[Alexander, 79] Alexander, Christopher. The Timeless Way of Building. Oxford University Press, Nueva York, 1979.

[Alur, 2001] Alur, D., Crupi, J., Malks, D. Core J2EE Patterns (Best Practices and Design Strategies). Sun Microsystems Press, E.E.U.U., 2001.

[Booch, 99] Booch, Grady; Rumbaugh, J.; Jacobson, Ian. El Lenguaje Unificado de Modelado. Addison-Wesley, primera edición en español, Madrid, 1999.

[Buschmann, 96] Buschmann, F., Meunier, R., Rohnert, H., Sommerland, P., Stal, M. Pattern-Oriented Software Architecture (A System of Patterns). John Wiley & Sons. West Sussex, Inglaterra, Octubre 1996. Conocido como POSA 1.

[Calderón, 2003c] Calderón, Alan. Unifying pattern catalogs (towards a pattern for pattern lexicons).

[Coad, 92] Coad, Peter. Object-Oriented Patterns. Communications of the ACM vol. 35(9), setiembre 1992.

[Coad, 97] Coad, Peter. Object Models (Strategies, Patterns and applications). Yourdon Press, New Jersey, 1997.

[Coplien, 92] Coplien, James. Advanced C++: Programming styles and idioms. Addison Wesley, 1992.

[Ellis, 93] Ellis, John M. Language, Thought and Logic. Northwestern University Press. Evanston, Illinois, 1993.

[Fowler, 97] Fowler, Martin. Analysis Patterns: Reusable Object Models. Addison-Wesley, E.E.U.U., 1997.

[Gamma, 95] Gamma, E., Helm R., Johnson, R. y Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Publishing Company, 1995. Conocido como GoF.

[Hay, 96] Hay, David C. Data Model Patterns (Conventions of Thought). Dorset House Pub. New York, 1996.

[Larman,98] Larman, Craig. Applying UML and Patterns (An Introduction to Object-Oriented Analysis and Design). Prentice Hall, New Jersey, 1998.

[Lakoff, 87] Lakoff, George. Women, Fire, and Dangerous Things (What categories reveal about the mind). The University of Chicago Press, Chicago, 1987.

[Pree, 95] Pree, Wolfgang. Design Patterns for Object-Oriented Software Development. Addison-Wesley Pub., E.E.U.U., 1995.

[Salingaros, 2000] Salingaros, Nikos A. The Structure of Patterns Languages. Architectural Research Quarterly, vol. 4, 2000 (páginas 149-161). Cambridge University Press.

[Schmidt, 2000] Schmidt, D., Stal, M., Rohnert, H., Buschmann, F. Pattern-Oriented Software Architecture: Patterns for Concurrent and Networked Objects. John Wiley & Sons. West Sussex, Inglaterra, January, 2000. Conocido como POSA 2.