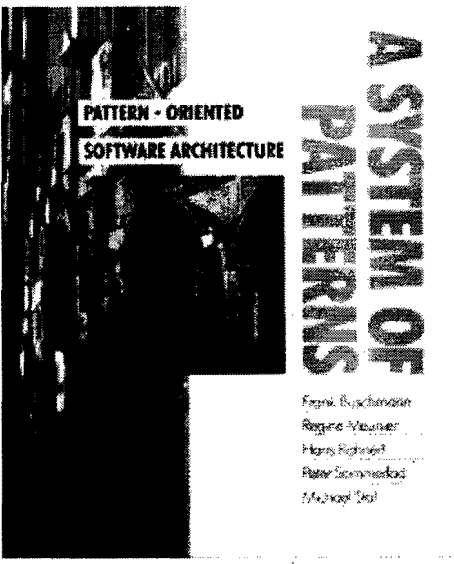
REVISIÓN DE LIBROS



Pattern-Oriented Software Architecture: A System of Patterns

Frank Buschmann
Regine Meunier
Hans Rohnert
Peter Sommerlad
Michael Stal

John Wiley & Sons Ltd, 1996

ISBN 0 471 95869 7

Por B. J. Ferro Castro
Centro de Investigación en Computación
Instituto Politécnico Nacional

"When you buy into the idea of developing personal computer software, you're buying into a way of life"

Jim McCarthy
Dynamics of Software Development,
Microsoft Press, 1995

Dos años han bastado para que la utilización de patrones forme parte de muchos métodos importantes para el desarrollo de software. Si bien, los patrones se han venido presentando desde hace mucho tiempo en casi todas las etapas del ciclo de vida de desarrollo de software, para encapsular abstracciones de análisis y diseño, no es menos cierto que su "fama" se produce después de la publicación de la obra de Gamma y sus colegas [*Gamma, 1995*] con la presentación descriptiva de 23 patrones de diseño, aplicables a casi todos los problemas imaginables que son modelados y resueltos con una computadora.

El enfoque hacia los patrones, responde a un cambio evolutivo en la forma de pensar de los analistas y diseñadores, que desean acercar cada vez más el dominio de la solución al dominio del problema a resolver. Si se analizan todos los métodos utilizados hasta la etapa actual, puede observarse que ese objetivo está presente en todos ellos, pues responde a una manera natural de resolver problemas; la de encontrar formas de disponer de entidades en el espacio de la computadora que se correspondan fielmente a las entidades o actores presentes en el problema. Los métodos orientados a objetos, que organizan tanto la información, como el procesamiento que manipula esa información de acuerdo con los objetos del mundo real que esa información describe, han contribuido más que los otros métodos conocidos a ese acercamiento.

En el contexto de la orientación de objetos, surgen los patrones como el paso siguiente en la organización de estructuras de clases y objetos, que en conjunto brindan la solución de problemas recurrentes, que la propia solución describe. Un sistema

de patrones brinda un conjunto de soluciones probadas y depuradas a muchos problemas recurrentes de diseño de diversos niveles de abstracción: de patrones arquitectónicos a patrones de bajo nivel que dependen del lenguaje que se usa para implementarlos. Esta diversidad brinda un mecanismo adecuado para el desarrollo constructivo de sistemas de software con un alto grado de reusabilidad.

El libro de Buschmann y sus colegas ofrece al lector una visión panorámica de los patrones con "alcance arquitectónico", de algunos patrones de diseño que representan microarquitecturas de propósito general y brevemente se refiere a los ya conocidas frases idiomáticas (del inglés idioms) para el diseño de cápsulas recurrentes de código.

El libro está organizado en 8 capítulos, que pueden leerse de forma secuencial o atendiendo a la propia Guía para el Lector que se ofrece, y que puede satisfacer tanto al lector novato como al experto en patrones.

En el capítulo I se presenta una introducción del tema y se brinda una clasificación de los patrones atendiendo al grado de abstracción que desean modelar. Aunque no se presenta una descripción formal de los patrones, el lector tendrá un conocimiento suficiente de lo qué es y lo qué no es un patrón, después de concluir este capítulo.

Para ilustrar el concepto de patrones arquitectónicos, los autores utilizan el patrón Modelo-Vista-Controlador y logran el objetivo. Este patrón, descrito en 1988 por Krasner, para ilustrar el paradigma de interface de usuario del sistema Smalltalk-80, es usado por muchos autores puesto que resulta muy intuitivo para describir la estructura fundamental de un subsistema parte de un sistema interactivo.

El capítulo II de este libro, conjuntamente con los capítulos V, VI y VIII son los más interesantes para aquellos que ya han entrado a la comunidad de los patrones, a través de los patrones de diseño de Gamma. Aquí se presentan a los patrones arquitectónicos como esquemas de una organización estructural global, que brindan subsistemas con responsabilidades propias y reglas de uso para su conexión con otros subsistemas. Los autores presentan patrones representativos para diferentes dominios; los sistemas poco formalizados, los sistemas distribuidos, los sistemas interactivos y los sistemas adaptables. Para cada uno de estos dominios se presentan uno o más patrones que resuelven problemas recurrentes de diseño arquitectónico.

La descripción del patrón Pizarrón (del inglés Blackboard) presenta –con la misma descripción que el resto de los patrones- uno de los primeros métodos utilizados para la especificación de sistemas para los que no se encuentran estrategias determinísticas de solución. Para el lector familiarizado con las técnicas de inteligencia artificial que se usaron en sistemas de reconocimiento de voz en los años 70 (los sistemas HearSay y Harpy), la descripción del método del Pizarrón con un enfoque de patrones resulta muy atractiva.

El patrón Broker, descrito dentro del dominio de sistemas distribuidos brinda una disertación muy clara y concisa de la importancia de agentes de middleware en sistemas compuestos por componentes individuales que brindan y solicitan servicios mediante invocaciones remotas. Esta descripción puede acompañarse de la del patrón Proxy que aparece en el capítulo III y que describe el problema y la solución de un cliente conectándose con un representante de una componente remota, en lugar de conectarse directamente a ella misma. Muchos de los servicios que se especifican por CORBA siguen estas reglas.

El capítulo III describe 8 patrones de diseño aplicables a casi todos los dominios. Cada uno de esos patrones refleja por su propia estructura, el carácter de microarquitectura de diseño. De particular interés resultan los patrones que ofrecen variantes a los descritos por Gamma, como es el caso del patrón creacional Todo-Parte, comparado con el patrón Composite descrito por Gamma. En ambos casos se persigue el mismo objetivo; el de crear estructuras compuestas que permitan un tratamiento uniforme del todo y sus partes, pero los contextos en los que se aplican son diferentes.

Los lenguajes de patrones y el desarrollo basado en ellos se estudian en los capítulos V y VI. Los autores utilizan el término sistema de patrones para describir lo que en la literatura ya se conoce y discute como lenguaje de patrones. Esta distinción de términos se justifica atendiendo a la completitud computacional que un lenguaje formal debe poseer, que para el caso de los lenguajes de patrones no se cumple. Aunque puede aceptarse ese argumento, lo cierto es que modificar un término acuñado dentro de la comunidad científica no es prudente. A la larga, los propios patrones se describen por nombres que para algunos pueden resultar inapropiados, pero uno de los objetivos que se persiguen es que una vez que el término se hace familiar y se introduce en el vocabulario, debe usarse para

de patrones brinda un conjunto de soluciones probadas y depuradas a muchos problemas recurrentes de diseño de diversos niveles de abstracción: de patrones arquitectónicos a patrones de bajo nivel que dependen del lenguaje que se usa para implementarlos. Esta diversidad brinda un mecanismo adecuado para el desarrollo constructivo de sistemas de software con un alto grado de reusabilidad.

El libro de Buschmann y sus colegas ofrece al lector una visión panorámica de los patrones con "alcance arquitectónico", de algunos patrones de diseño que representan microarquitecturas de propósito general y brevemente se refiere a los ya conocidas frases idiomáticas (del inglés idioms) para el diseño de cápsulas recurrentes de código.

El libro está organizado en 8 capítulos, que pueden leerse de forma secuencial o atendiendo a la propia Guía para el Lector que se ofrece, y que puede satisfacer tanto al lector novato como al experto en patrones.

En el capítulo I se presenta una introducción del tema y se brinda una clasificación de los patrones atendiendo al grado de abstracción que desean modelar. Aunque no se presenta una descripción formal de los patrones, el lector tendrá un conocimiento suficiente de lo qué es y lo qué no es un patrón, después de concluir este capítulo.

Para ilustrar el concepto de patrones arquitectónicos, los autores utilizan el patrón Modelo-Vista-Controlador y logran el objetivo. Este patrón, descrito en 1988 por Krasner, para ilustrar el paradigma de interface de usuario del sistema Smalltalk-80, es usado por muchos autores puesto que resulta muy intuitivo para describir la estructura fundamental de un subsistema parte de un sistema interactivo.

El capítulo II de este libro, conjuntamente con los capítulos V, VI y VIII son los más interesantes para aquellos que ya han entrado a la comunidad de los patrones, a través de los patrones de diseño de Gamma. Aquí se presentan a los patrones arquitectónicos como esquemas de una organización estructural global, que brindan subsistemas con responsabilidades propias y reglas de uso para su conexión con otros subsistemas. Los autores presentan patrones representativos para diferentes dominios; los sistemas poco formalizados, los sistemas distribuidos, los sistemas interactivos y los sistemas adaptables. Para cada uno de estos dominios se presentan uno o más patrones que resuelven problemas recurrentes de diseño arquitectónico.

La descripción del patrón Pizarrón (del inglés Blackboard) presenta –con la misma descripción que el resto de los patrones- uno de los primeros métodos utilizados para la especificación de sistemas para los que no se encuentran estrategias determinísticas de solución. Para el lector familiarizado con las técnicas de inteligencia artificial que se usaron en sistemas de reconocimiento de voz en los años 70 (los sistemas HearSay y Harpy), la descripción del método del Pizarrón con un enfoque de patrones resulta muy atractiva.

El patrón Broker, descrito dentro del dominio de sistemas distribuidos brinda una disertación muy clara y concisa de la importancia de agentes de middleware en sistemas compuestos por componentes individuales que brindan y solicitan servicios mediante invocaciones remotas. Esta descripción puede acompañarse de la del patrón Proxy que aparece en el capítulo III y que describe el problema y la solución de un cliente conectándose con un representante de una componente remota, en lugar de conectarse directamente a ella misma. Muchos de los servicios que se especifican por CORBA siguen estas reglas.

El capítulo III describe 8 patrones de diseño aplicables a casi todos los dominios. Cada uno de esos patrones refleja por su propia estructura, el carácter de microarquitectura de diseño. De particular interés resultan los patrones que ofrecen variantes a los descritos por Gamma, como es el caso del patrón creacional Todo-Parte, comparado con el patrón Composite descrito por Gamma. En ambos casos se persigue el mismo objetivo; el de crear estructuras compuestas que permitan un tratamiento uniforme del todo y sus partes, pero los contextos en los que se aplican son diferentes.

Los lenguajes de patrones y el desarrollo basado en ellos se estudian en los capítulos V y VI. Los autores utilizan el término sistema de patrones para describir lo que en la literatura ya se conoce y discute como lenguaje de patrones. Esta distinción de términos se justifica atendiendo a la completitud computacional que un lenguaje formal debe poseer, que para el caso de los lenguajes de patrones no se cumple. Aunque puede aceptarse ese argumento, lo cierto es que modificar un término acuñado dentro de la comunidad científica no es prudente. A la larga, los propios patrones se describen por nombres que para algunos pueden resultar inapropiados, pero uno de los objetivos que se persiguen es que una vez que el término se hace familiar y se introduce en el vocabulario, debe usarse para

comunicar la abstracción correspondiente: la solución a un problema en un contexto recurrente.

En el capítulo VI se presenta una distinción clara entre las propiedades funcionales y no funcionales de un producto de software. Esta presentación puede complementarse por la presentada por Meyer [*Meyer, 1997*] para describir las entidades externas que gobiernan la construcción de software con objetos.

Todo libro que concluye con un panorama de las investigaciones actuales y futuras en el campo, es bien visto por los lectores y estimula la creatividad en el área. El libro que aquí presentamos, nos proporciona una visión global de lo que se está haciendo en el campo y lo mucho que queda por hacer.

Buchmann y sus colegas han logrado un libro que muy pronto se ha convertido en uno de los textos de referencia obligada en la comunidad de científicos y profesionales del campo. Aconsejamos al lector interesado en el mundo de los patrones y que ha tomado el software como una forma de vida a consultar este libro. No dude, que muy pronto todos los "metodólogos" conocidos introducirán en sus métodos lo concerniente al diseño orientado a patrones. Ya está sucediendo.

Referencias

[*Gamma, 1995*]. **Design Patterns. Elements of Reusable Object-Oriented Software**. Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides. Addison-Wesley Publishing Company, Reading, Massachusetts, 1995.

[*Meyer, 1997*]. **Object-Oriented Software Construction, Second Edition**. Bertrand Meyer. Prentice Hall PTR, New Jersey, 1997.