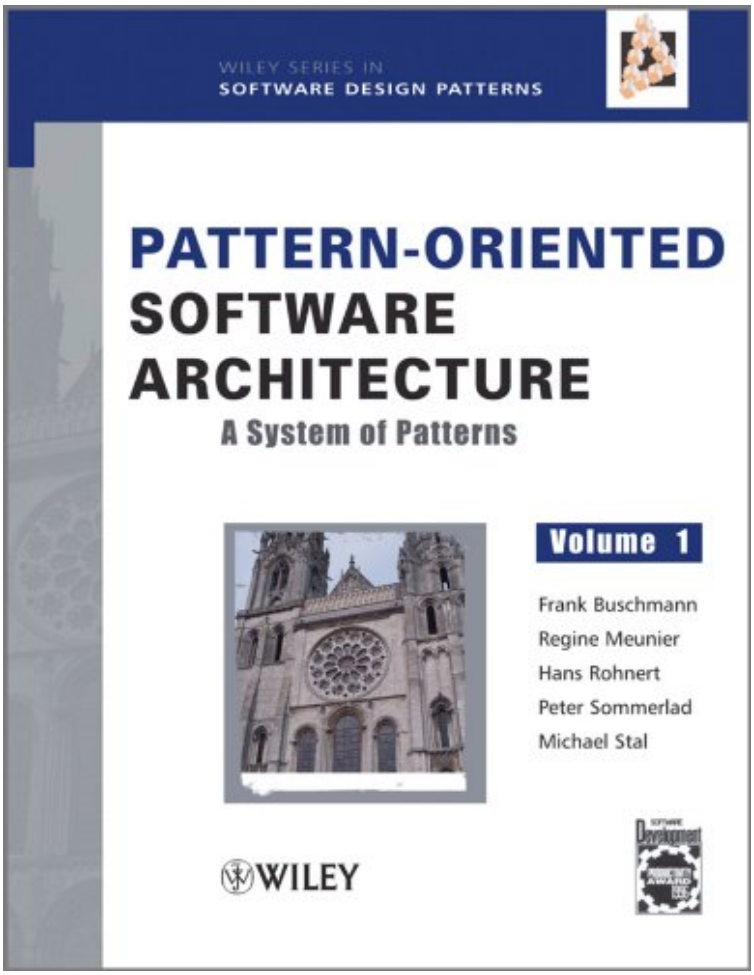
Pattern-Oriented Software Architecture, A System of Patterns: Volume 1 (Wiley Software Patterns Series)

Von Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal
*ebooks | Download PDF | *ePub | DOC | audiobook*



Produktinformation -Verkaufsrang: #92220 in eBooksVerffentlicht am: 2013-04-18Erscheinungsdatum: 2013-04-18File Name: B00CGH2KXA | File size: 39.Mb

Von Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal : Pattern-Oriented Software Architecture, A System of Patterns: Volume 1 (Wiley Software Patterns Series) before purchasing it in order to gage whether or not it would be worth my time, and all praised Pattern-Oriented Software Architecture, A System of Patterns: Volume 1 (Wiley Software Patterns Series):

KundenrezensionenHilfreichste Kundenrezensionen12 von 13 Kunden fanden die folgende Rezension hilfreich. Simply outstandingVon Robert E. NewbyIt is a shame that this excellent book does not receive the same level of attention (at least in the U.S.) as the Gang of Four's Design Patterns: Elements of Resusable Object-Oriented Software. Pattern-Oriented Software Architecture is more comprehensive in scope and, to my mind, better written. In particular,

it provides a solid, highly pragmatic framework for understanding and utilizing patterns at 3 levels of scope/abstraction -- i.e., architecture, design, and implementation -- not only design. Moreover, the authors pay significant attention to variations of patterns, trade-offs between alternative patterns, and the relationships between patterns that in practice may need to be combined in order to accomplish the objectives at hand. The book's overall organization, presentation of material, and referencing of related pattern sources (including the GoF's work) is superb. This is my top patterns reference, with others as serving as adjuncts.9 von 10 Kunden fanden die folgende Rezension hilfreich. This is *the* Patterns BookVon R. WilliamsWhile I have argued since it came out that the G of 4 book is the most important programming book of the decade, I have to agree with the other, lone reviewer here, that this is a deeper, more mature work. I rediscovered this book when Alan Holub's series of recent articles began to appear in JavaWorld about implemnting UIs and I realized that he was taking a lot of his ideas from Buschman. One of the reasons I bring this up is that it made me realize that this is the great thing about this book: it dares to wrestle some of the complex issues and tradeoffs to the ground, presenting the reader with a more useable guide to the practice of implementing patterns. You may have read John Vlissides' (Go4 author) comments about how for years after the publication of his book he'd ask when he spoke who had read the book and nearly everyone would raise their hands, then he'd ask who wanted to come up and explain the momento pattern or the bridge and only a couple of people's hands would be raised. This is in part due to the fact that the Go4 book encourages the concept of simple ICs that can just be retrieved and plugged in. In reality, as anyone who has read Vlissides' other book which spends its whole duration talking just about Visitor, the opposite is true. Buschman's book is the best in this regard at spanning the range of design issues but still dealing with the complexities of implementation, and helping the reader through the process of assessing trade-offs and still matching requirements.7 von 8 Kunden fanden die folgende Rezension hilfreich. The second book on patternsVon Christophe AddinquyDefinitively, after reading the GoF book, get this one ! The Design Patterns receive more attention in my own opinion, because patterns newcomers (I was one of them, few years ago) found at first (nices) solutions that may be applied directly. It take a little time to understand that the great idea is elsewhere. POSA book (this one) tackle directly to this great idea, and exposes several patterns categories (3, in fact). Architectural patterns are really well explained, much better than in the GoF. The POSA book receive less attention probably because authors forget the idea to "attract" newcomers, at least in my mind. Anyway, the Design Patterns is my "book of the Decade". I consider this one as the second to purchase in the patterns movement, but it's not a second hand material, it's a complementary materials.

KurzbeschreibungPattern - Oriented Software Architecture A System of Patterns Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal of Siemens AG, Germany Pattern-oriented software architecture is a new approach to software development. This book represents the progression and evolution of the pattern approach into a system of patterns capable of describing and documenting large-scale applications. A pattern system provides, on one level, a pool of proven solutions to many recurring design problems. On another it shows how to combine individual patterns into heterogeneous structures and as such it can be used to facilitate a constructive development of software systems. Uniquely, the patterns that are presented in this book span several levels of abstraction, from high-level architectural patterns and medium-level design patterns to low-level idioms. The intention of, and motivation for, this book is to support both novices and experts in software development. Novices will gain from the experience inherent in pattern descriptions and experts will hopefully make use of, add to, extend and modify patterns to tailor them to their own needs. None of the pattern descriptions are cast in stone and, just as they are borne from experience, it is expected that further use will feed in and refine individual patterns and produce an evolving system of patterns. Visit our Web Page http://www.wiley.com/compbooks/.dePattern-Oriented Software Architecture: A System of Patterns looks at how patterns occur on three different levels--in software architecture, in everyday design, and in idioms (which describe how a particular design pattern is implemented in a programming language like C++). This synthetic approach is a little theoretical at times, but the authors also present over a dozen patterns and provide real-world examples wherever possible. For architectural patterns, the authors look at the Layers pattern, used in operating systems such as Windows NT and virtual machines. They also consider Pipes and Filters, which process streams of data. (This pattern, the authors point out, is a lynchpin of Unix.) Their Blackboard pattern shows how a complex problem, such as image or speech recognition can be broken up into smaller, specialized subsystems that work together to solve a problem. (For recognizing words from a raw waveform input, a Blackboard approach might have separate processes to find phonemes, then words, then sentences.) This book also looks at today's distributed systems in considering the Broker pattern, which is used on the Internet and in Microsoft's OLE technology. This section also presents several powerful patterns for building effective graphical user interfaces, such as Model-View-Controller. The authors define several well-known design patterns, such as the Proxy and Command patterns, and also basic, far-reaching patterns, such as Whole-Part and Master-Slave, which are widely used throughout computing. Their survey ends with a discussion on the way objects can communicate (using such patterns as Forwarder-Receiver, Client-

Dispatcher-Server, and Publisher-Subscriber), which many developers will recognize as familiar patterns, but are codified here as "official" patterns. The book then discusses some idioms in C++ and a more far-reaching role for patterns in software design and architecture. By fitting patterns into traditional software engineering practices, the authors of Pattern-Oriented Software Architecture successfully argue that the role for patterns will only continue to diversify and enrich tomorrow's software engineering tools and methodologies. --Richard DraganPressestimmenThe GoF book concentrates on design-level patterns, whereas out patterns span several levels of abstraction... from high-level architectural patterns through design patterns to low-level idioms' writes Beushcmann and his four co-authors from Siemens of Germany about Pattern-oriented software architecture. An example of an architectural pattern is the mocrokernal which 'separates a minimal r-specific parts'. They (prophetically) illustrate with a fictitious operating system named - Hydra! For design patterns they cite (among others) the master-slave pattern that supports fault-tolerance, parallel computation and computational accuracy. Idioms are 'low-level patterns specific to a programming language'. Two alternative versions of a string copy function, written in C are compared: one by a devotee of Kernighan Ritchie's terse style takes three lines and includes masterpieces of compaction such as (*d++=*s++); while the other by a Pascal enthusiast takes eight lines but is more readable. There is plenty of discussion and explanation and two interesting closing chapters. One describes the leading figures in the 'Pattern Community', many of them the authors reviewed here, plus Ward Cunningham and Kent Beck who first translated Alexander's ideas into software. The other asks where patterns are heading, for example a new area is organisational structures to support software development, such as Architect Controls Product which ensures team output is elegant and cohesive, not shapeless and nondescript. Other shadows in the crystal ball are the use of repositories and indexing (this has, inevitably, given birth to a new creature; the pattlet) and the development of specific pattern languages. -- Unix NT News, January 99