



Evaluating Software Architectures: Methods and Case Studies

By Paul Clements, Rick Kazman, Mark Klein

Download now

Read Online ➔

Evaluating Software Architectures: Methods and Case Studies By Paul Clements, Rick Kazman, Mark Klein

The foundation of any software system is its architecture, this text is intended to help architects determine what aspects of their architectures need improvement. The techniques put forward are based on actual practice in both government and industry development organizations.

↓ [Download Evaluating Software Architectures: Methods and Cas ...pdf](#)

📄 [Read Online Evaluating Software Architectures: Methods and C ...pdf](#)

Evaluating Software Architectures: Methods and Case Studies

By Paul Clements, Rick Kazman, Mark Klein

Evaluating Software Architectures: Methods and Case Studies By Paul Clements, Rick Kazman, Mark Klein

The foundation of any software system is its architecture, this text is intended to help architects determine what aspects of their architectures need improvement. The techniques put forward are based on actual practice in both government and industry development organizations.

Evaluating Software Architectures: Methods and Case Studies By Paul Clements, Rick Kazman, Mark Klein **Bibliography**

- Sales Rank: #229047 in Books
- Published on: 2001-11-01
- Original language: English
- Number of items: 1
- Dimensions: 9.40" h x 1.10" w x 6.50" l, 1.43 pounds
- Binding: Hardcover
- 368 pages

 [Download Evaluating Software Architectures: Methods and Cas ...pdf](#)

 [Read Online Evaluating Software Architectures: Methods and C ...pdf](#)

Editorial Review

From the Back Cover

Praise for *Evaluating Software Architectures*

“The architecture of complex software or systems is a collection of hard decisions that are very expensive to change. Successful product development and evolution depend on making the right architectural choices. Can you afford not to identify and not to evaluate these choices? The authors of this book are experts in software architecture and its evaluation. They collected a wealth of ideas and experience in a well-organized and accessible form. If you are involved in the development of complex systems or software, you will find this book an invaluable guide for establishing and improving architecture evaluation practice in your organization.”

—**Alexander Ran**, Principal Scientist of Software Architecture, Nokia

“Software engineers must own this book. It is a well-written guide to the steps for evaluating software architecture. It argues for the inclusion of architecture evaluation and review as a standard part of the software development lifecycle. It introduces some new and innovative methods for analyzing important architecture characteristics, like extensibility, portability, and reliability. I believe these methods will become new engineering cornerstones for creating good software systems.”

—**Joe Maranzano**, AT&T Bell Labs Fellow in Software Architecture (1990), and former head of the Bell Labs Software Technology Center

“Experience and teamwork are the only approaches I know of to deliver products faster, cheaper, and yet to delight your customers. In their first book, *Software Architecture in Practice*, Paul and Rick (and Len Bass) helped me match my experience with theory. Their invaluable approaches and case studies changed my practice and the way I proceed to design systems and software architectures. This second book, with Mark, covers what I will look at before I feel good about an architecture. It is about how I can tap other people's experience to produce an improved outcome, using other people's feedback. I have used many of the concepts explained in this book for my customers' benefit. Using this book, you—architects, developers, and managers—will develop a common language and practice to team up and deliver more successful products.”

—**Bertrand Salle**, lead architect with a major telecommunications company

“If architecture is the foundation of system construction, architectural evaluation is part of the foundation of getting to a ‘good’ architecture. In this book, the authors put their considerable expertise to one of the most pressing issues in systems development today: how to evaluate an architecture prior to system construction to ascertain its feasibility and suitability to the system of interest. The book provides a practical guide to architecture evaluation using three contemporary evaluation methods. It should prove valuable to practitioners and as a basis for the evolution of architectural evaluation as an engineering practice.”

—**Rich Hilliard**, Chief Technical Officer, ConsentCache, Inc., and technical editor, *IEEE Recommended Practice for Architectural Description of Software-Intensive Systems*

“Too many systems have performance and other problems caused by an inappropriate architecture. Thus problems are introduced early, but are usually detected too late—when the deadline is near or, even worse, after the problem makes the headlines. Remedies lead to missed schedules, cost overruns, missed market windows, damaged customer relations, and many other difficulties. It is easy to prevent these problems by evaluating the architecture choices early, and selecting an appropriate one.”

—**Connie U. Smith**, Ph.D., principal consultant, Performance Engineering Services Division, L&S Computer Technology, Inc., and coauthor of the new book, *Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software*

“The ATAM an evaluation method described in this book is the natural quality-gate through which a high-level design should pass before a detail design project is initiated. Why use the ATAM to evaluate an architecture? Mitigation of design risk is a major reason, but more importantly, the ATAM provides an interactive vehicle that can give key development and user stakeholders architectural visibility—visibility that can lead to an important ‘early buy-in.’”

—**Rich Zebrowski**, Software Technology Manager, Motorola, Inc.

“Caterpillar's experience with architecture reviews includes SAAM, ATAM, ARID, and ADR evaluation methods described in this book, the first three in detail. These reviews ensured that the needs of the user community were being met, and they exposed the architecture to others in the organization helping with understanding and organizational buy-in. The SAAM- and ATAM-based evaluations worked well to expose the architecture early in the development cycle to a broad range of people. The ARID- and ADR-based evaluations facilitated the exposure of technical details of the architecture later in the development cycle. As the architect of the pilot project for ARID, I observed that this review even served as an architecture training session before the architecture was fully documented.”

—**Lee R. DenBraber**, former Lead Software Architect, Caterpillar, Inc.

“We’ve heard all the management hype about harnessing the innovative creativity of our teams, establishing integrated customer-developer-product teams, and better targeting our systems to meet end user needs. The ATAM techniques described in this book give technical managers, system architects, and engineers proven tools for breaking down the communications barriers that impede our ability to realize these goals. We have successfully integrated the ATAM techniques throughout our lifecycle, including development and maintenance, and have found that they provide the strong technical basis we need to evaluate the many difficult trades required by a system as large as EOSDIS.”

—**Mike Moore**, Deputy Manager, Science Systems Development Office, Earth Observing System Data Information System (EOSDIS) Project, NASA Goddard Space Flight Center

“If you know how difficult architecture reviews are, you will be amazed how effective ATAM evaluations can be. For example, an ATAM evaluation we conducted on an important software product line identified a major architectural risk, which we subsequently were able to avoid—a benefit we expect to continue seeing. Moreover, ATAM techniques have enabled us to explain such risks to stakeholders far more clearly than by any other review method.”

—**Stefan Ferber**, Corporate Research, Robert Bosch GmbH

Drawing on clearly identified connections between architecture design decisions and resulting software properties, this book describes systematic methods for evaluating software architectures and applies them to real-life cases. It shows you how such evaluation can substantially reduce risk while adding remarkably little

expense and time to the development effort (in most cases, no more than a few days). *Evaluating Software Architectures* introduces the conceptual background for architecture evaluation and provides a step-by-step guide to the process based on numerous evaluations performed in government and industry.

In particular, the book presents three important evaluation methods:

- Architecture Tradeoff Analysis Method (ATAM)
- Software Architecture Analysis Method (SAAM)
- Active Reviews for Intermediate Designs (ARID)

Detailed case studies demonstrate the value and practical application of these methods to real-world systems, and sidebars throughout the book provide interesting background and hands-on tips from the trenches.

All software engineers should know how to carry out software architecture evaluations. *Evaluating Software Architectures* is the chance to get up to speed quickly by learning from the experience of others.

About the Author

Paul Clements is a senior member of the technical staff at the SEI, where he works on software architecture and product line engineering. He is the author of five books and more than three dozen papers on these and other topics.

Rick Kazman is a senior member of the technical staff at the SEI. He is also an Associate Professor at the University of Hawaii. He is the author of two books, editor of two more, and has written more than seventy papers on software engineering and related topics.

Mark Klein is a senior member of the technical staff at the SEI. He is an adjunct professor in the Masters of Software Engineering program at Carnegie Mellon and a coauthor of *A Practitioner's Handbook for Real-time Analysis: Guide to Rate Monotonic Analysis for Real-Time Systems* (Kluwer Academic Publishers, 1993).

020170482XAB01162003

Excerpt. © Reprinted by permission. All rights reserved.

The foundation of any software system is its architecture, that is, the way the software is constructed from separately developed components and the ways in which those components interact and relate to each other. If the system is going to be built by more than one person—and these days, what system isn't?—it is the architecture that lets them communicate and negotiate work assignments. If the requirements include goals for performance, security, reliability, or maintainability, then architecture is the design artifact that first expresses how the system will be built to achieve those goals. The architecture determines the structure of the development project. It is the basis for organizing the documentation. It is the first document given to new project members, and the first place a maintenance organization begins its work. Schedules, budgets, and workplans all revolve around it. And the senior, most talented designers are paid to create it.

A system's longevity—how viable it remains in the face of evolutionary pressure—is determined primarily

by its architecture. Some architectures go on to become generic and adopted by the development community at large: three-tier client-server, layered, and pipe-and-filter architectures are well known beyond the scope of any single system. Today, organizations are recognizing the importance and value of architectures in helping them to meet corporate enterprise goals. An architecture can give an enterprise a competitive advantage and can be banked like any other capitalized asset.

The right architecture is the first step to success. The wrong architecture will lead to calamity. This leads to an important question: If your organization is betting its future—or at least a portion of it—on an architecture for a system or family of related systems, how can you be sure that you're building from the right architecture and not the wrong one?

The practice of creating an architecture is maturing. We can identify causal connections between design decisions made in the architecture and the qualities and properties that result downstream in the system or systems that follow from it. This means that it is possible to evaluate an architecture, to analyze architectural decisions, in the context of the goals and requirements that are levied on systems that will be built from it.

And yet even though architecture is regarded as an essential part of modern system development, architecture evaluation is almost never included as a standard part of any development process. We believe it should be, and this book is an attempt to help people fill that gap.

The time has come for architecture evaluation to become an accepted engineering practice for two reasons. First, architecture represents an enormous risk in a development project. As we've said, the wrong one leads to disaster. It makes good sense to perform an evaluation on such a pivotal artifact, just as you would plan risk-mitigation strategies for other sources of uncertainty. Second, architecture evaluation can be remarkably inexpensive. The methods described in this book add no more than a week to the project schedule, and some abridged forms require no more than a day or two. Architecture evaluation represents a very cheap insurance policy. Compared to the cost of a poor architecture, the modest expense of a software architecture evaluation makes all the sense in the world. What has been lacking up to this point is a practical method for carrying it out, which is where this book comes in.

This is a guidebook for practitioners (or those who wish to become practitioners) of architecture evaluation. We supply conceptual background where necessary, but the intent of the work is to provide step-by-step guidance in the practice of architecture evaluation and analysis. To help put the methods into practice, we have included sample artifacts that are put into play during an architecture evaluation: viewgraph presentation outlines, scenarios, after-action surveys, final report templates, and so forth. The goal is that after reading this book, you will feel confident enough to try out the methods on an architecture in your own organization. We have tried to help answer the question, during an evaluation, "What should I do now?"

While the book is written from the point of view of the evaluator, there are others involved in an evaluation—project managers, architects, other stakeholders—who will gain valuable insights by reading this book. They will come to understand how their products will be evaluated and thus can position themselves to make those products fare better with respect to the evaluation criteria. This is rather like scoring well on a test because you've seen an early copy of the test, but in this case it isn't cheating but rather sound management and engineering practice. But know that when we use the word *you* in the text, we are speaking to the evaluator.

The techniques in this book are based on actual practice in government and industry. Most of the methods were developed by ourselves and others at the Software Engineering Institute and applied by ourselves and others to our customers' and collaborators' systems. Other material was gleaned by holding industrial workshops whose participants were experts in the analysis and evaluation of architecture. In short, we have

learned by doing, and we have learned from others' doing.

This book will not teach you how to become a good architect, nor does it help you become fluent in the issues of architecture. We assume that you already have a good grasp of architectural concepts that comes from practical experience. This book will not help you assess the job performance of any individual architect nor a project's architecture (or development) process. What it will do is show you how to evaluate an architecture with respect to a broad spectrum of important quality attributes having to do with the architecture and the future system(s) that will be built from it.

Finally, we should say a word about *software* versus *system* architecture—that is, the architecture of software-intensive systems. This is a book about the evaluation of software architectures, but we often hear the question, "Well, what about the architecture of the system, not just the software? It's just as vital." We couldn't agree more. System architectures embody the same kinds of structuring and decomposition decisions that drive software architectures. Moreover, they include hardware/software tradeoffs as well as the selection of computing and communication equipment, all of which are completely beyond the realm of software architecture. System architectures hold the key to success or failure of a system every bit as much as the software architecture does for the software. Hence, they deserve to be evaluated every bit as much and for exactly the same reasons.

The methods presented in this book will, we believe, apply equally well to system architectures as to software architectures. If modifiability is a concern, the methods can be used to gauge the expense of making changes over the system's lifetime; if performance is a concern, the methods can be used to spot bottlenecks and problem areas in the system as well as the software; and so forth.

Why, then, do we call it a book about *software* architecture evaluation? Because that is the realm in which the methods were invented, developed, tested, and matured. In the remainder of this book when we speak of *architecture*, you can always safely prefix it with *software*. You can prefix it with *system* depending on how applicable you feel the methods are to system architectures and how confident you are about our intuition in the matter.

As a final word, we invite you to share your experiences with us. We would be keenly interested in knowing what you discover works well and what doesn't work so well. Writing a book is an opportunity to share lessons, but more importantly to us, it is an opportunity to gather new ones.

—PCC, Austin, Texas

—RK, Pittsburgh, Pennsylvania

—MHK, Pittsburgh, Pennsylvania

020170482XP10082001

Users Review

From reader reviews:

Jennifer Meeks:

A lot of people always spent their particular free time to vacation or maybe go to the outside with them household or their friend. Were you aware? Many a lot of people spent many people free time just watching TV, as well as playing video games all day long. If you want to try to find a new activity honestly, that is look different you can read some sort of book. It is really fun to suit your needs. If you enjoy the book that

you just read you can spent the whole day to reading a book. The book Evaluating Software Architectures: Methods and Case Studies it is extremely good to read. There are a lot of folks that recommended this book. These people were enjoying reading this book. In case you did not have enough space to create this book you can buy the e-book. You can m0ore easily to read this book from a smart phone. The price is not too costly but this book offers high quality.

James Cummings:

Playing with family in a very park, coming to see the coastal world or hanging out with pals is thing that usually you have done when you have spare time, in that case why you don't try point that really opposite from that. One particular activity that make you not experiencing tired but still relaxing, trilling like on roller coaster you are ride on and with addition of information. Even you love Evaluating Software Architectures: Methods and Case Studies, you may enjoy both. It is good combination right, you still need to miss it? What kind of hang type is it? Oh can happen its mind hangout fellas. What? Still don't have it, oh come on its called reading friends.

Yolanda Harris:

Reading a book being new life style in this yr; every people loves to read a book. When you read a book you can get a lots of benefit. When you read books, you can improve your knowledge, mainly because book has a lot of information into it. The information that you will get depend on what kinds of book that you have read. If you need to get information about your study, you can read education books, but if you want to entertain yourself you can read a fiction books, this kind of us novel, comics, and also soon. The Evaluating Software Architectures: Methods and Case Studies will give you a new experience in examining a book.

Morgan Johnson:

In this age globalization it is important to someone to find information. The information will make someone to understand the condition of the world. The healthiness of the world makes the information much easier to share. You can find a lot of personal references to get information example: internet, classifieds, book, and soon. You can see that now, a lot of publisher which print many kinds of book. The actual book that recommended for your requirements is Evaluating Software Architectures: Methods and Case Studies this book consist a lot of the information from the condition of this world now. This particular book was represented so why is the world has grown up. The language styles that writer make usage of to explain it is easy to understand. Often the writer made some analysis when he makes this book. Honestly, that is why this book appropriate all of you.

Download and Read Online Evaluating Software Architectures: Methods and Case Studies By Paul Clements, Rick Kazman, Mark

Klein #25U4R6HLG1P

Read Evaluating Software Architectures: Methods and Case Studies By Paul Clements, Rick Kazman, Mark Klein for online ebook

Evaluating Software Architectures: Methods and Case Studies By Paul Clements, Rick Kazman, Mark Klein
Free PDF d0wnl0ad, audio books, books to read, good books to read, cheap books, good books, online
books, books online, book reviews epub, read books online, books to read online, online library, greatbooks
to read, PDF best books to read, top books to read Evaluating Software Architectures: Methods and Case
Studies By Paul Clements, Rick Kazman, Mark Klein books to read online.

Online Evaluating Software Architectures: Methods and Case Studies By Paul Clements, Rick Kazman, Mark Klein ebook PDF download

**Evaluating Software Architectures: Methods and Case Studies By Paul Clements, Rick Kazman,
Mark Klein Doc**

Evaluating Software Architectures: Methods and Case Studies By Paul Clements, Rick Kazman, Mark Klein Mobipocket

Evaluating Software Architectures: Methods and Case Studies By Paul Clements, Rick Kazman, Mark Klein EPub