

Preface

Welcome to the second edition of Essential Software Architecture. It is 5 years since the first edition was published, and in the software architecture world, 5 years is a long time. Hence this updated version, with refreshed chapters to capture new developments in methods and technologies, and to relate relevant experiences from practise. There's new material covering enterprise architecture, agile development, enterprise service bus technologies and RESTful Web services. All chapters have an updated and more extensive list of recommended reading, capturing many of the best new books, papers, web sites and blogs that I know of.

Most notably, the completely new Chap. 10 provides a case study on the design of the MeDICi technology, which extends an open source enterprise service bus with a component-based programming model. The MeDICi technology is open source and freely downloadable (<http://www.medic.pnl.gov>), making it a highly suitable tool for teaching the advanced concepts of middleware and architecture described in this text.

At its heart however, this remains a book that aims to succinctly impart a broad sweep of software architecture knowledge relating to systems built from mainstream middleware technologies. This includes a large, diverse spectrum of systems, ranging from Web-based ecommerce sites to scientific data management and high performance financial data analysis systems.

Motivation

What hasn't changed in the last 5 years is that many projects I work with or review lack an explicit notion of an architectural design. Functional requirements are usually captured using traditional or agile techniques, agreed with stakeholders, and addressed through highly iterative or traditional waterfall methods. But the architectural issues, the "how" the application achieves its purpose, the "what" happens when things change and evolve or fail, are frequently implicit (this means they are in somebody's head, maybe) at best. At worst, they are simply not addressed in any way that can be described in terms other than accidental. Frequently, when I ask for an overview of the application architecture and the driving nonfunctional

requirements at the first technical meeting, people start drawing on a whiteboard. Or they show me code and dive into the internals of the implementation based around their favorite, trendy technology. Either of these is rarely a good sign.

The problems and risks of poor architectural practices are well known and documented within the software engineering profession. A large body of excellent architectural knowledge is captured in broadly accessible books, journals and reports from members of the Software Engineering Institute (SEI), Siemens and various other renowned industrial and academic institutions.

While the focus of much of this literature is highly technical systems such as avionics, flight simulation, and telecommunications switching, this book leans more to the mainstream world of software applications. In a sense, it bridges the gap between the needs of the vast majority of software professionals and the current body of knowledge in software architecture. Specifically:

- It provides clear and concise discussions about the issues, techniques and methods that are at the heart of sound architectural practices.
- It describes and analyzes the general purpose component and middleware technologies that support many of the fundamental architectural patterns used in applications.
- It looks forward to how changes in technologies and practices may affect the next generation of business information systems.
- It uses familiar information systems as examples, taken from the author's experiences in banking, e-commerce and government information systems.
- It provides many pointers and references to existing work on software architecture.

If you work as an architect or senior designer, or you want to 1 day, this book should be of value to you. And if you're a student who is studying software engineering and need an overview of the field of software architecture, this book should be an approachable and useful first source of information. It certainly won't tell you everything you need to know – that will take a lot more than can be included in a book of such modest length. But it aims to convey the essence of architectural thinking, practices and supporting technologies, and to position the reader to delve more deeply into areas that are pertinent to their professional life and interests.

Outline

The book is structured into three basic sections. The first is introductory in nature, and approachable by a relatively nontechnical reader wanting an overview of software architecture.

The second section is the most technical in nature. It describes the essential skills and technical knowledge that an IT architect needs.

The third is forward looking. Six chapters each introduce an emerging area of software practice or technology. These are suitable for existing architects and

designers, as well as people who've read the first two sections, and who wish to gain insights into the future influences on their profession.

More specifically:

- *Chapters 1–3*: These chapters provide the introductory material for the rest of the book, and the area of software architecture itself. Chapter 1 discusses the key elements of software architecture, and describes the roles of a software architect. Chapter 2 introduces the requirements for a case study problem, a design for which is presented in Chap. 9. This demonstrates the type of problem and associated description that a software architect typically works on. Chapter 3 analyzes the elements of some key quality attributes like scalability, performance and availability. Architects spend a lot of time addressing the quality attribute requirements for applications. It's therefore essential that these quality attributes are well understood, as they are fundamental elements of the knowledge of an architect.
- *Chapters 4–10*: These chapters are the technical backbone of the book. Chapter 4 introduces a range of fundamental middleware technologies that architects commonly leverage in application solutions. Chapter 5 is devoted to describing Web services, including both SOAP and REST-based approaches. Chapter 6 builds on the previous chapters to explain advanced middleware platforms such as enterprise service bus technologies. Chapter 7 presents a three stage iterative software architecture process that can be tailored to be as agile as a project requires. It describes the essential tasks and documents that involve an architect. Chapter 8 discusses architecture documentation, and focuses on the new notations available in the UML version 2.0. Chapter 9 brings together the information in the first 6 chapters, showing how middleware technologies can be used to address the quality attribute requirements for the case study. It also demonstrates the use of the documentation template described in Chap. 8 for describing an application architecture. Chapter 10 provides another practical case study describing the design of the open source MeDICi Integration Framework, which is a specialized API for building applications structured as pipelines of components.
- *Chapters 11–15*: These chapters each focus on an emerging technique or technology that will likely influence the futures of software architects. These include software product lines, model-driven architecture, aspect-oriented architecture and the Semantic Web. Each chapter introduces the essential elements of the method or technology, describes the state-of-the-art and speculates about how increasing adoption is likely to affect the required skills and practices of a software architect. Each chapter also relates its approach to an extension of the ICDE case study in Chap. 9.



<http://www.springer.com/978-3-642-19175-6>

Essential Software Architecture

Gorton, I.

2011, XVI, 242 p., Hardcover

ISBN: 978-3-642-19175-6