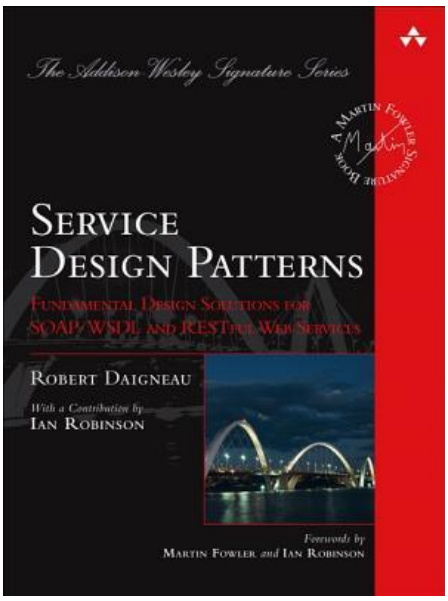

Service Design Patterns: Fundamental Design Solutions for SOAP/WSDL and RESTful Web Services

Robert Daigneau

Service Design Patterns: Fundamental Design Solutions for SOAP/WSDL and RESTful Web Services

Robert Daigneau

Service Design Patterns: Fundamental Design Solutions for SOAP/WSDL and RESTful Web Services
Robert Daigneau

Web services have been used for many years. In this time, developers and architects have encountered a number of recurring design challenges related to their usage, and have learned that certain service design approaches work better than others to solve certain problems.

In ***Service Design Patterns,*** Rob Daigneau codifies proven design solutions for web services that follow the REST architectural style or leverage the SOAP/WSDL specifications. This catalogue identifies the fundamental topics in web service design and lists the common design patterns for each topic. All patterns identify the context in which they may be used, explain the constituent design elements, and explore the relative strengths and trade-offs. Code examples are provided to help you better understand how the patterns work but are kept general so that you can see how the solutions may be applied to disparate technologies that will inevitably change in the years to come.

This book will help readers answer the following questions:

How do you create a web service API, what are the common API styles, and when should a particular style be used?
How can clients and web services communicate, and what are the foundations for creating complex conversations in which multiple parties exchange data over extended periods of time?
What are the options for implementing web service logic, and when should a particular approach be used?
How can clients become less coupled to the underlying systems used by a service?
How can information about a web service be discovered?
How can generic functions like authentication, validation, caching, and logging be supported on the client or service?
What changes to a service cause clients to break?
What are the common ways to version a service? How can web services be designed to support the continuing evolution of business logic without forcing clients to constantly upgrade?

This book is an invaluable resource for enterprise architects, solution architects, and developers who use web services to create enterprise IT applications, commercial or open source products, and Software as a Service (SaaS) products that leverage emerging Cloud platforms.

Service Design Patterns: Fundamental Design Solutions for SOAP/WSDL and RESTful Web Services Details

Date : Published October 28th 2011 by Addison-Wesley Professional (first published July 20th 2011)

ISBN : 9780321544209
Author : Robert Daigneau
Format : Hardcover 321 pages
Genre : Computer Science, Software, Science, Technology, Programming, Architecture, Internet, Web, Technical

From Reader Review Service Design Patterns: Fundamental Design Solutions for SOAP/WSDL and RESTful Web Services for online ebook

Robert Lara III says

For anyone who is building web services for the first time, this is a great book to learn about all the pitfalls of different architecture decisions. While a 5 to 10 year software engineer may know some of this due to experience, this book still has a lesson or two to teach any veteran.

Christophe Addinquy says

This book should have been a nice complement over Enterprise Integration Patterns, but in the end, it's not. I have a mixed feeling about this book. It misses the Pattern Language philosophy or even the Design Patterns concept at all. The author is looking for covering the field as well as possible.
In the end there is some useful stuff out there, such as clarification of what can be used in what situation. It makes this book not that useless.
ma note de lecture en français ici

Finlay says

The usual high standard from the Martin Fowler Series

Joshua says

A lot of good ideas here, but the language/environment choices (Java/C#) are very different from my native tongues, so was hard to find the wheat in the chaff.

J Lavoie says

As is true with all patterns books, your reaction might be, “I already know all of this”. This book doesn’t try to identify new ideas. Instead, it gives a name to the approaches we’ve all been using for some time, and lists their pros and cons. It’s cool that the author identified names that were, in many cases, useful for both RESTful and SOAP style services.

The code examples are helpful, but aren’t detailed or prescriptive “how-to” recipes. The author hints in the forward that you should probably have an understanding of or an acquaintance with the technologies and frameworks that are used. Fowler also suggests in the forward that there are many other books out there for that type of thing.

GoodReads ask you to identify the date you finished it. To be honest, I haven't read each and every page because the book clearly isn't meant to be read cover to cover (it's pretty heavy reading), it's more of a reference that you might look to when you need a nudge on the decision factors to consider. Still, I've read enough to evaluate it.

I would highly recommend it.

John says

This book is pretty good place to start if you're still new to web services and design patterns. But I stress the word *start*. If you've done any significant work in this area and aren't confused by the acronym GoF then you should probably skip this. You won't learn anything new and you'll probably be annoyed that some of his guidance is given no real context or in-depth discussion or even decent justification.

Rod Hilton says

Most patterns books contain very little new information, usually they just provide terminology for things an experienced developer has seen or done countless times. As such, I admit it is somewhat unfair for me to feel the way I do about Service Design Patterns: that contains staggeringly little new information for an experienced developer. I knew not to expect to learn a great deal, but I still managed to find even less information than I expected.

I think about 70% of the book will be immediately recognizable to anyone who has done any web service work before, and it won't particularly be new terminology, it's generally the standard terminology present in virtually any documentation. This is not to say it's a bad book; it's not bad, it's actually very well-written, well-organized, and very clear. The problem is that there simply doesn't seem to be enough material for an entire book here, and it often feels like the text is padded with common knowledge.

There were two things that really irked me about the book. One was that it contained mention of Resource API as an approach to designing a web service API, but contained very little about REST beyond the absolute basics. I realize entire books have been written about properly designing RESTful APIs, but as I mentioned this book feels padded, so it could have easily stood a more thorough treatment.

My biggest complaint with this book was its treatment of the issue of web service versioning. Versioning web services is mentioned, and the reasons to do so are enumerated, but very little guidance is provided for how to do so effectively. In my experience, properly versioning a web service API is one of the most challenging aspects of designing such an API. Obviously it's not difficult from the perspective of the external API itself, you just throw a version number in the URL or something, but it's very challenging in implementation. If you're at version 10 and a request comes in for version 3, what do you do? Do you have a request mapper for 3->10, process, and then a response mapper for 10->3? If so, then introducing 11 means writing 20 converters. Do you just write a 3->4 and a 4->3 when you introduce 4, then chain them together? It's all very complex and difficult to do properly, and I was hoping for some practical guidance. But instead, it has one of the weakest treatments in the book, without even the level of source code present for even trivial patterns elsewhere in the book.

The book is good overall, but it's not great. It doesn't offer much in the way of new information or even insight into existing information. It does the basic job of a patterns book, in that it provides shared terminology, but the bulk of that terminology is standard in web service documentation anyway, so even that is unlikely to add much. I'd say it's worth reading if you're really new to web services, and maybe worth skimming quickly if you're more experienced.

David Ruiz Ramirez says

ok
