MARKUS VOELTER

HOW TO UNDERSTAND ALMOST ANYTHING

A PRACTITIONER'S GUIDE TO DOMAIN ANALYSIS

VOELTER.DE/HTUAA

Preface

I have previously written several books related to software engineering, all covered technical content: software architecture, middleware, modelling, code generation, as well as domain-specific language (DSL) design and implementation. Why, you might ask, is the guy writing about "soft" topics now. Things like analysis, writing, communication, and processes. Has he become one of these lots-of-talk-no-walk consultants?

I hope not. However, the DSL[1] projects I have been involved in grew in scope, size and importance (for the client) and my role has shifted away from purely technical advise and language implementation. I spend more of my time with domain analysis, architecting systems in which DSLs and "executable subject matter" play a central role, and I also consult on how processes should change to take full advantage of this approach. So, while I've been writing code probably every day over the last couple of years, a significant portion of my days was spent with these "softer" topics.

There's an additional thing that changed for me. When I started out with modelling, DSLs and MPS around 15 years ago, I was totally fascinated by the technology. I didn't care too much what people used it for, as long as they paid me to "play" with it. mbeddr and KernelF are outgrowths of this phase. Now that I got used to the technologies and their potential benefits, I am more interested in actually reaping these benefits for the customers I am working with. In other words, I am much more interested in solving *real* problems using DSLs and tools focusing on capturing subject matter. And this means that all these "soft" topics I mentioned above have to be taken into account. Hence my personal evolution in this direction.

So, several of my buddies, colleagues and customers asked me how I go about this domain analysis thing. How do you find out how the language should look like? How do you extract all that domain knowledge from people's heads and capture it in a set of consistent abstractions? Is there a methodology (hint: no!)? So I decided to write down my experience in this space and sell it as a little eBook. Note the word "experience". This is not a scientific book. I am not

---

[1] This is not a DSL book. Please continue reading :-)

citing 250 other people who analyse domains. It's my own approach. And experience can be positive and negative. So you'll read suggestions on what to do and what to avoid. And sometimes my frustration about how (badly) some particular project went oozes from the text. So what I write in this book is not the one and only way to approach a domain analysis, but it is one that has worked in several projects of various sizes and in domains ranging from science to tax calculation to healthcare. That has to count for something.

This is also not a DSL book. Domain analysis is important for non-DSL projects as well. Whenever you need a consistent set of abstractions — a language — that allows users to express subject matter concisely and correctly, in a way that can be analysed and executed by a machine, domain analysis is relevant.

There's an even bigger picture perspective for this book (and sorry for zooming all the way out). With all the changes going on in the world, the (western) industrial countries have to become more competitive to retain the standard of living for our societies. One way of achieving this is with breakthrough ideas: if we can get fusion to work, lots of problems will be solved (only half-joking here). Another way of increasing competitiveness is perform existing activities more efficiently, making the existing workforce more effective: build better software faster, optimise how we capture subject matter, de-crappify processes. This seems especially relevant because of the skills shortage due to an ageing population. To do things more efficiently, organisations have to clearly understand what they are actually doing. They have to precisely understand their domain and optimise details. And the analysis process that finds out these details has to run efficiently itself. Anybody who has ever tried to change "how things are done" in an organisation with more than a few dozen people knows how hard this is for all kinds of reasons, from domain complexity to political games to (some) peoples' unwillingness to change. But in addition to these non-technical reasons, understanding how a domain works, what the organisation does now and how things should work going forward is genuinely non-trivial. With this book, I hope to help a little bit in running this knowledge creation process more effectively.

The book you are reading is relatively short. You should be able to read it in less about 4 hours. It could have been longer, with more examples, maybe a complete case study. But after writing several "large" books in my career, I didn't have the energy to do another one, which is why this one is shorter. A nice side effect is that it is less of a commitment for you to read. I chose a very colloquial style, I really wanted my voice to be discernible from the text[2]. I hope this makes reading the book more fun.

Now, before I let you move on to the core of the book, please join me in thanking my reviewers. It is hard to overstate how important their feedback was for this book. I wrote a lot of this stuff almost as a therapy for some of the frustrating experiences in recent projects. In the original draft, some things sounded (even more) frustrated, several things were too black and white, too unsystematic. And in Chapter Six, things were really waaay to brief. Without the honest, constructive and plentiful feedback — I counted 288 comments, excluding typos and other trivial stuff — this book would not be what it is now. So thank you very much to (in alphabetical order): Boris Holzer, Federico Tomassetti, Kresimir Vukorepa, Lisa Hartl, Mike Vogel, Sascha Lißon, and Sergej Koscejev.

I'm also looking forward to your feedback! Let me know whether you like what you're reading, or whether you disagree or have additional input. Contact me via http://voelter.de/ping or voelter@acm.org . You can use this same email adress to "register the book" — I will notify you when there's a new version.[3]

Enjoy the ride!

Markus Voelter, March 2023

---

[2] I am a big fan of audio books. Maybe this plays into the decision.

[3] Needless to say I won't spam you or try to sell you my services through this channel.

Table of Contents