Guided Tour of *Domain-Driven Design*

These are references to specific selections of the book *Domain-Driven Design*, by Eric Evans (Addison-Wesley, 2004). They are meant to guide a manager or other project leader quickly to the main points of interest and use to a decision-maker on a software project.

Managers have particular interests, less technical, more organizational. The selections below do not require much technical background, but they do assume some experience with software development teams.

*Just find an interesting question and follow it to the indicated pages in the book.*

**I. What Is Domain-Driven Design and Why Care? (~1 hr)**

*These excerpts from Part I of the book walk you through the fundamental principles of domain-driven design and some of the benefits of applying it.*

Contrasting three projects concretely illustrates how design style can be a factor in success and failure.

- Why design style is an inextricable factor in the development process. xxii-xxiv
- The benefits of committing the whole team to domain-driven design. xxvii
- What is a model? 2-4
- The heart of software and why it gets neglected. 4-6
- Knowledge Crunching: How a team can accumulate, distill and apply domain knowledge to software development 12-15 (from "Ingredients of...")
- Ubiquitous Language: How to bring about a clearer and more dynamic flow of domain knowledge throughout the project 24-27
  - Example 27-30
  - Consequences 32-34
- How documents & diagrams can work for a project instead of just being work. 35-40
- How to make modeling relevant to the the goals of a software project: Model-Driven Design. 47-50
- Why models matter to users. 57-59
- The necessity of eliminating the distinction between modelers and programmers. 60-62
II. Building Blocks

*Part II lays a foundation of detailed modeling techniques that underpin effective model-driven design. This tour skips to Part III.*

III. What Constitutes a Useful Model-Driven Design and How to Go About Finding Such a Design (~ 1 hr)

True story: How model-driven design rescued a project and created unexpected opportunities. 193-203
How software experts can work with domain experts to explore and refine models. 207-210
Supple Design: How a system can become *easier* to extend and adapt rather than ossify into legacy 243-245
Overview of the rhythm of domain-driven design and how it allows for upside surprise opportunities to emerge. 321-326

IV. Strategic Design: Team Decisions That Affect the Trajectory of the Entire Project (~ 2 hr)

Introduction to three principles for applying domain-driven design to large projects and enterprises. 328-329
Bounded Context: Strategies for dealing with the inevitability of multiple viewpoints and conflicting needs. 331-338
- How much integration do you need? How can you structure relationships between teams to get it? 341-371, (headings and bold)
- Whimsical, non-technical example 378-381
- Broad tradeoffs between context strategies Figure 14.13 (on p. 388)
Distillation: How do you focus on your central problem and keep from drowning in a sea of side issues? 400-405
- A Tale of Two Time Zones: A right way and a wrong way to deploy your people to tackle essential supporting components 410-412
- Reducing project risk by tackling the core domain early 413-414
- Crafting a domain vision statement 415-416

Large-scale structure: How to make a sprawling system comprehensible and encourage consistency across subsystems. 439-442
- How to have structure without stifling development 444-446

*Several specific techniques for large-scale structure are discussed, but are*
skipped in this tour.
Non-technical example of how a large-scale structure allowed thousands of
people to contribute to the AIDS Memorial Quilt 478(bottom)-479
Putting the pieces together to develop a design strategy 490-497

V. Conclusion (~1/4 hour)

Tracking five real domain-driven design projects and their long-term outcomes. 500-505
The future of domain-driven design 505-506

This work is licensed under a Creative Commons License.