README

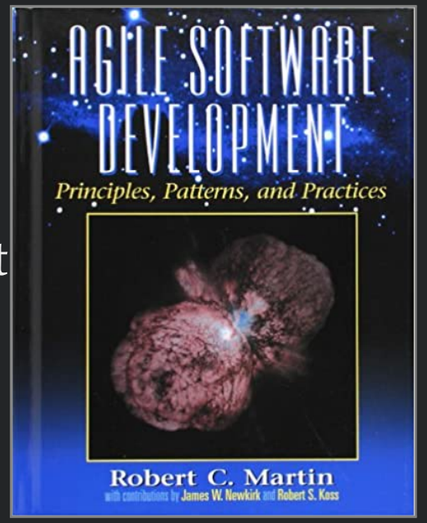
vs. IEEE, RUP, SWEBOK, CMMI

Yegor Bugayenko

Lecture #2 out of 16
90 minutes

All videos are in this YouTube playlist.

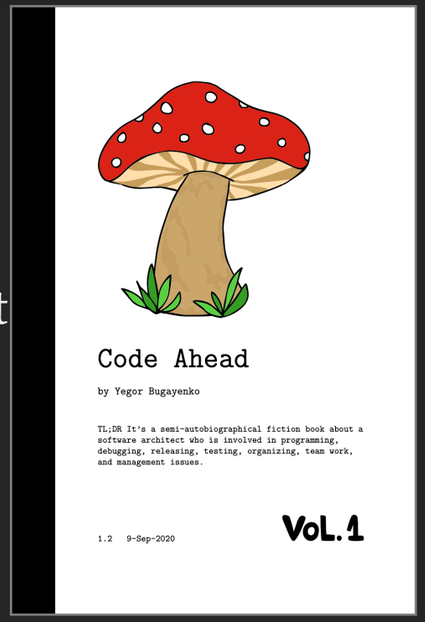
+2pt



“If you are lucky, you start a project with a clear picture of what you want the system to be. The design of the system is a vital image in your mind. If you are luckier still, the clarity of that design makes it to the first release.”

— *Agile Software Development. Principles, Patterns, and Practices*, Robert Martin

+2pt



"It's the responsibility of a programmer to make sure the tasks he is working with have explicit borders."

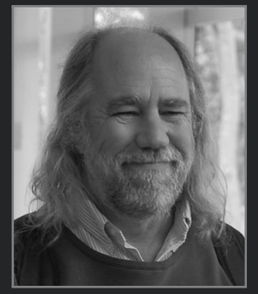
— *Code Ahead*, Yegor Bugayenko

- Use Cases (user stories)
- FPA, IFPUG, COSMIC
- Traceability Matrix
- Verification and Validation
- Non-Functional Requirements (NFRs)
- Estimates and COCOMO II
- Books, Venues, Call-to-Action

Chapter #1:

Use Cases (user stories)

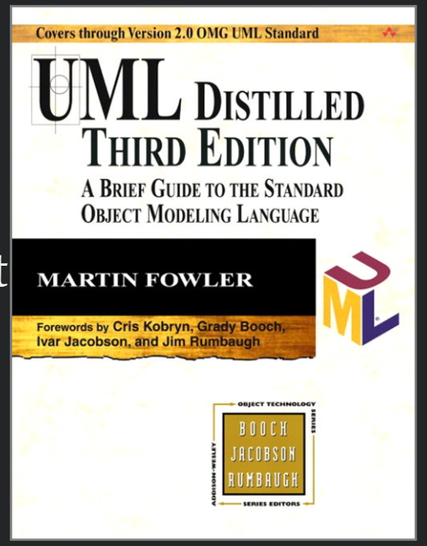
Ivar Jacobson
in 1987



Grady Booch

James Rumbaugh

+2pt

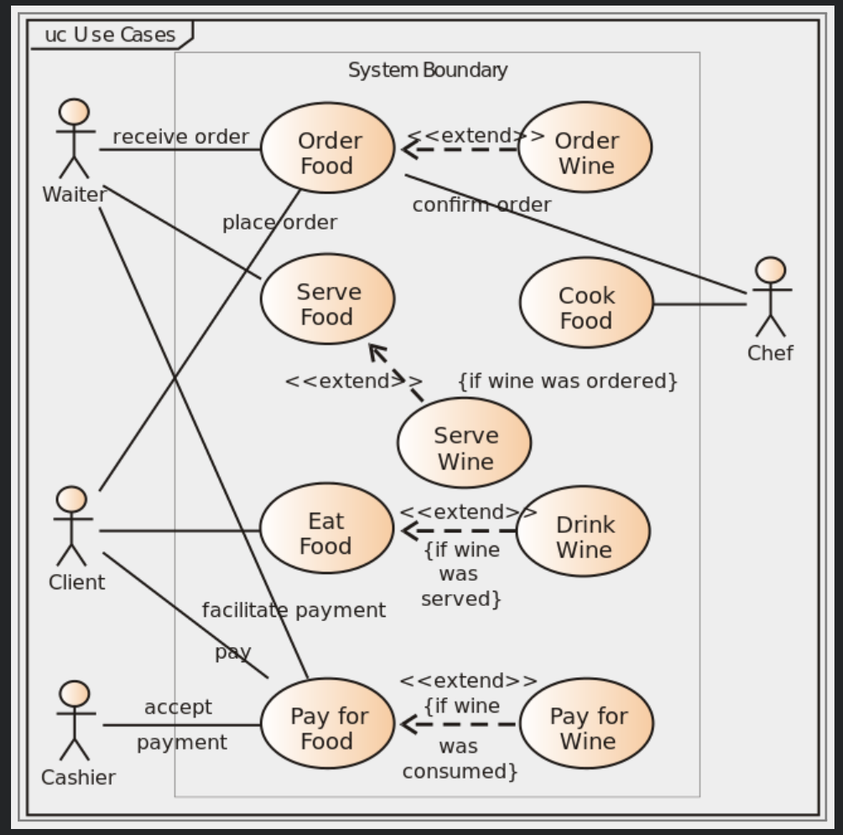


“There is no standard way to write the content of a use case, and different formats work well in different cases”

— UML Distilled
Martin Fowler

```
Use Case: Make a QR code
Primary Actor: User
Basic flow:
  1. The User enters the URL into the HTML box.
  2. The System creates a PNG image of a QR code.
  3. The User downloads the image through HTTP.
Extensions:
  1a. The User enters broken URL.
    1. The System shows an error modal dialog box.
    2. The User confirms.
  3a. The User cancels downloading.
    1. The System stops sending data over HTTP.
```

Use Case Diagram



Chapter #2:

FPA, IFPUG, COSMIC

Function Point Analysis (FPA)

FPA was originally developed by Allan Albrecht in the late 1970s at IBM

International Function Point Users Group (IFPUG) is in charge

Regulated by ISO 20296

COSMIC is the modern version: ISO/IEC 14143

The method breaks down the requirements into combinations of the four data movements types: Entry (E), Exit (X), Read (R), Write (W)

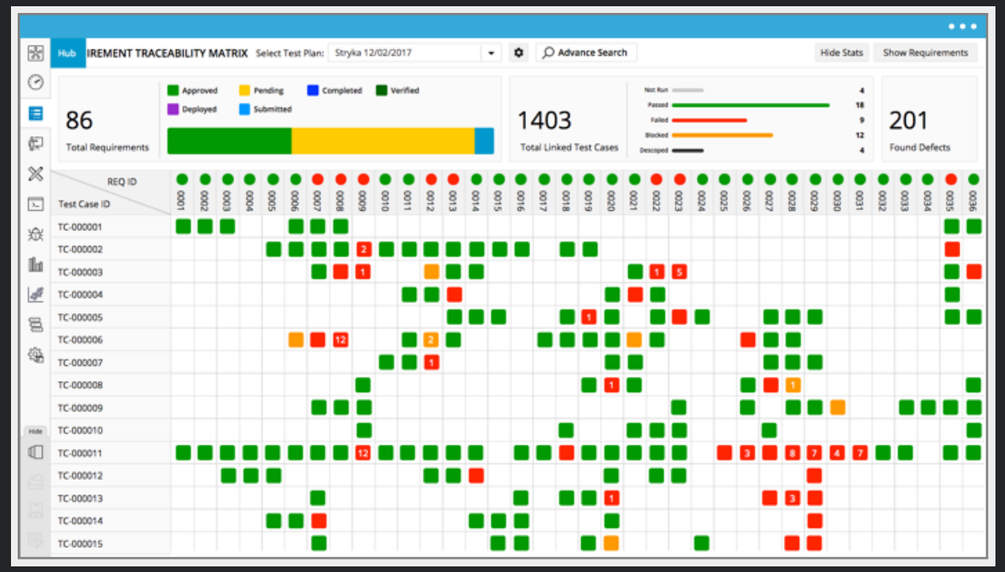
Function Points (FPs) are used for estimates

Also, read about Use Case Points (UCP)

Chapter #3:

Traceability Matrix

Traceability Matrix

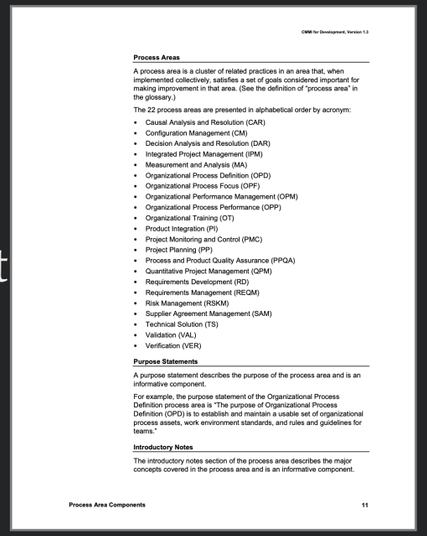


Use Cases, Non-Functional Requirements, Test Cases, Classes, Packages, Servers, Containers, Components, Nodes, etc.

Chapter #4:

Verification and Validation

+2pt



CMMI for Development, Version 1.3

Process Areas

A process area is a cluster of related practices in an area that, when implemented collectively, satisfies a set of goals considered important for making improvement in that area. (See the definition of "process area" in the glossary.)

The 22 process areas are presented in alphabetical order by acronym:

- Causal Analysis and Resolution (CAR)
- Configuration Management (CM)
- Decision Analysis and Resolution (DAR)
- Integrated Project Management (IPM)
- Measurement and Analysis (MA)
- Organizational Process Definition (OPD)
- Organizational Process Focus (OPF)
- Organizational Performance Management (OPM)
- Organizational Process Performance (OPP)
- Organizational Training (OT)
- Product Integration (PI)
- Project Monitoring and Control (PMC)
- Project Planning (PP)
- Process and Product Quality Assurance (PPQA)
- Quantitative Project Management (QPM)
- Requirements Development (RD)
- Requirements Management (REQM)
- Risk Management (RSKM)
- Supplier Agreement Management (SAM)
- Technical Solution (TS)
- Validation (VAL)
- Verification (VER)

Purpose Statements

A purpose statement describes the purpose of the process area and is an informative component.

For example, the purpose statement of the Organizational Process Definition process area is "The purpose of Organizational Process Definition (OPD) is to establish and maintain a usable set of organizational process assets, work environment standards, and rules and guidelines for teams."

Introductory Notes

The introductory notes section of the process area describes the major concepts covered in the process area and is an informative component.

Process Area Components 11

"Two different questions may be asked by a project team: 'We do it right?' vs. 'We do the right thing?' CMMI-Dev has two separate process areas: VER and VAL."

Chapter #5:

Non-Functional Requirements (NFRs)

Quality Attributes or just “-ilities”:

Availability: $A = \frac{E_{\text{up}}}{E_{\text{down}} + E_{\text{up}}}$

Capacity: Clicks Per Second (CPS)

Recovery: Recovery Time Objective (RTO)

Maintainability: Mean Time To Repair (MTTR)

Usability: focus group surveys?

Bad NFRs

“The software must be fast.”

“It must be easy to maintain.”

“The user interface must be attractive.”

Good NFRs

“Being installed on a test server (see Annex A), the software must respond is less than 20ms on any request from UC1–UC7.”

“The maximum time required to fix a bug much be less than two hours.”

“At least 80% of beta users must anonymously confirm that the UI is attractive enough.”

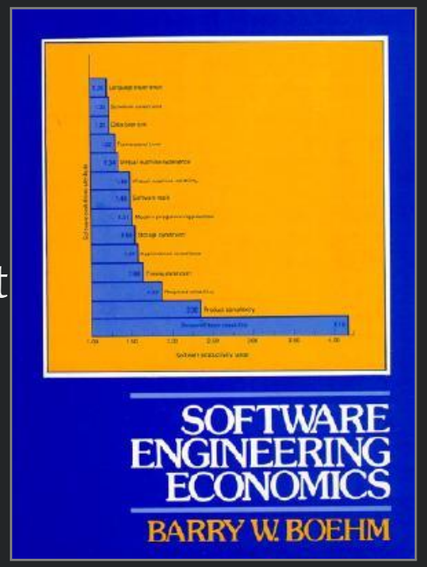
“Ten Mistakes in Specs” (2015)

- No Glossary or a Messy One
- Questions, Discussions, Opinions
- Mixing Functional and NFRs
- Mixing Requirements and Docs
- Un-measurable NFRs
- Implementation Instructions
- Lack of Actor Perspective
- Noise
- Will, Need, Must

Chapter #6:

Estimates and COCOMO II

+2pt



“For software decisions, the most critical and difficult of these inputs to provide are estimates of the cost of a proposed software project.”

— Software Engineering Economics
Barry W. Boehm

First, we predict the size in Kilo Lines of Code (K).

Then, we find effort adjustment factor (F).

Then, we find coefficients a, b, and c using the table.

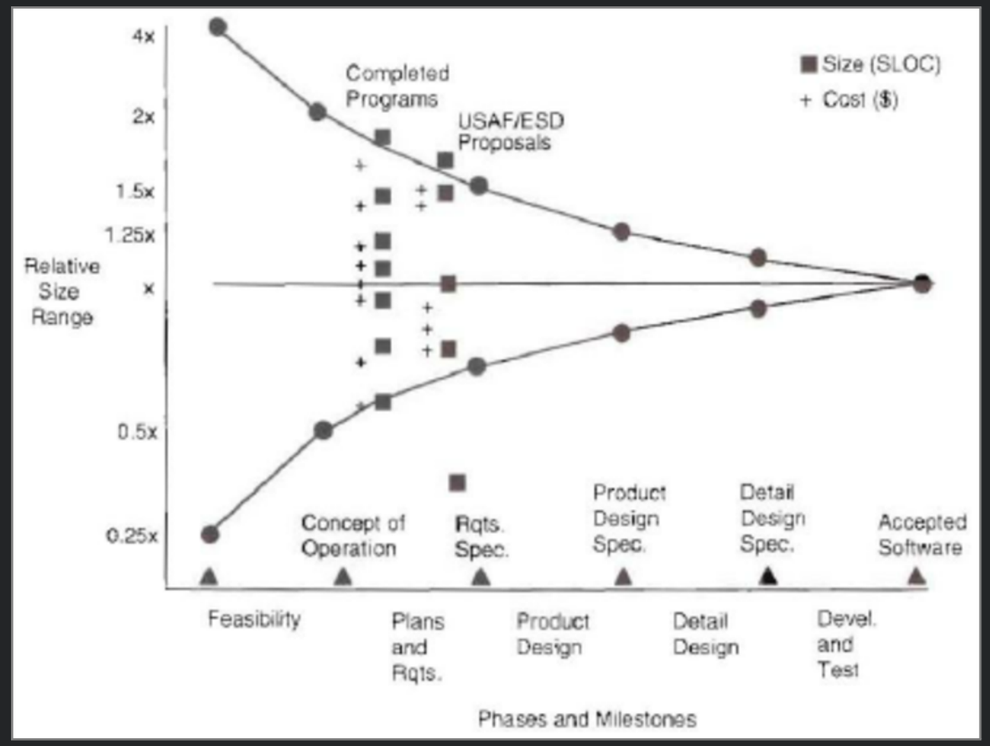
Then, we calculate the size in man-months:

$$E = a \times K^b \times F$$

Then, we calculate the duration in months (D):

$$D = 2.5 \times E^c$$

Cone of Uncertainty

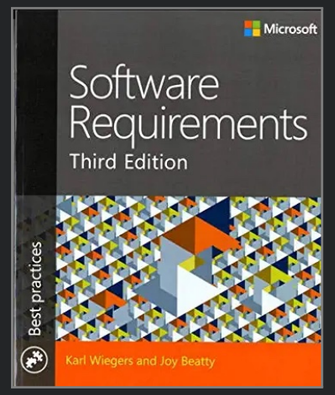


“How much for this software?” they ask.

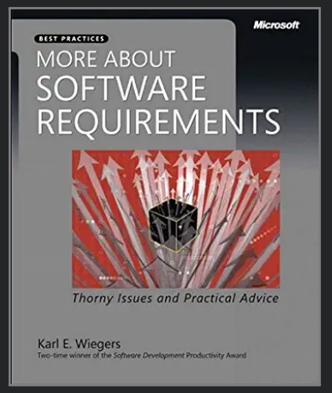
“The development will take forever and will consume all your money,” we answer.

Chapter #7:

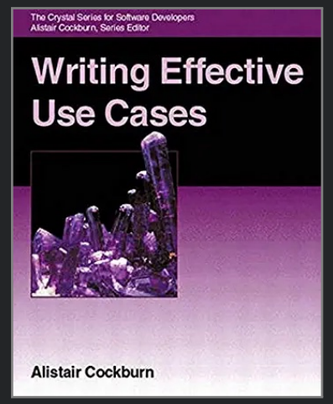
Books, Venues, Call-to-Action



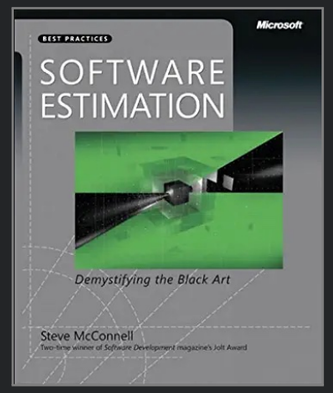
“Software Requirements” by Karl Wiegers et al.



“More About Software Requirements: Thorny Issues and Practical Advice” by Karl Wiegers



“Writing Effective Use Cases” by Alistair Cockburn



“Software Estimation: Demystifying the Black Art” by Steve McConnell

Where to publish:

IEEE International Requirements Engineering Conference (RE)

Call to Action:

Specify 4+ use cases and 10+ non-functional requirements for your app. Then, count all FPs, estimate the size in LoC, and then estimate the cost in man-months using COCOMO II.

Still unresolved issues:

- How to validate requirements automatically?
- How to trace them automatically?
- How to specify in Controlled Natural Language (CNL)?
- How to reduce them?