Requirements

Engineering
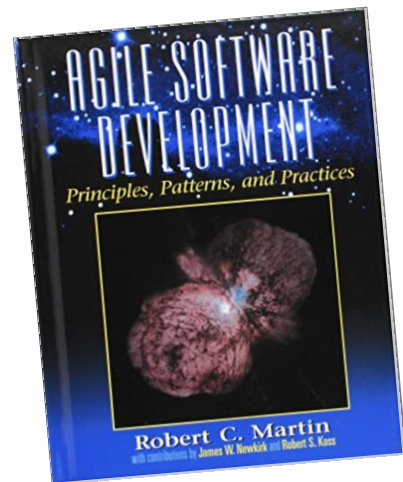
Yegor Bugayenko

Lecture #2 out of 16
80 minutes

The slidedeck was presented by the author in this YouTube Video

"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."

— Robert C. Martin. *Agile Software Development, Principles, Patterns, and Practices.* Prentice Hall, 2002. doi:10.5555/515230
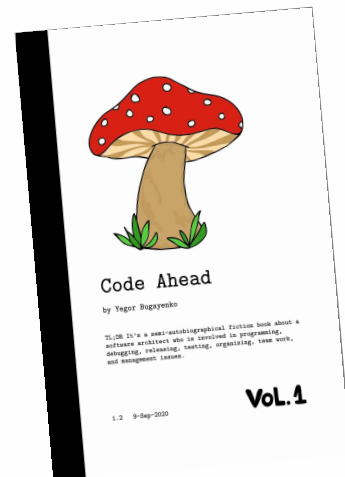
"It's the responsibility of a programmer to make sure the tasks he is working with have explicit borders."

— Yegor Bugayenko. *Code Ahead*. Amazon, 2018

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
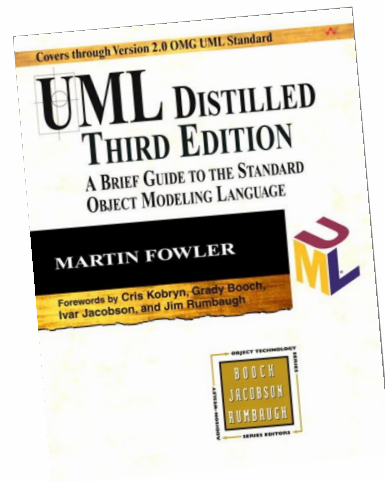
Chapter #1:

Use Cases (user stories)

Ivar Jacobson
in 1987

Grady Booch

James Rumbaugh

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

— Martin Fowler. *UML Distilled: A Brief Guide to the Standard Object Modeling Language.* Addison-Wesley, 3 edition, 2004

```
 1 Use Case: Make a QR code
 2 Primary Actor: User
 3 Basic flow:
 4   1. The User enters the URL into the HTML box.
 5   2. The System creates a PNG image of a QR code.
 6   3. The User downloads the image through HTTP.
 7 Extensions:
 8   1a. The User enters broken URL.
 9     1. The System shows an error modal dialog box.
10     2. The User confirms.
11   3a. The User cancels downloading.
12     1. The System stops sending data over HTTP.
```

Use Case Diagram
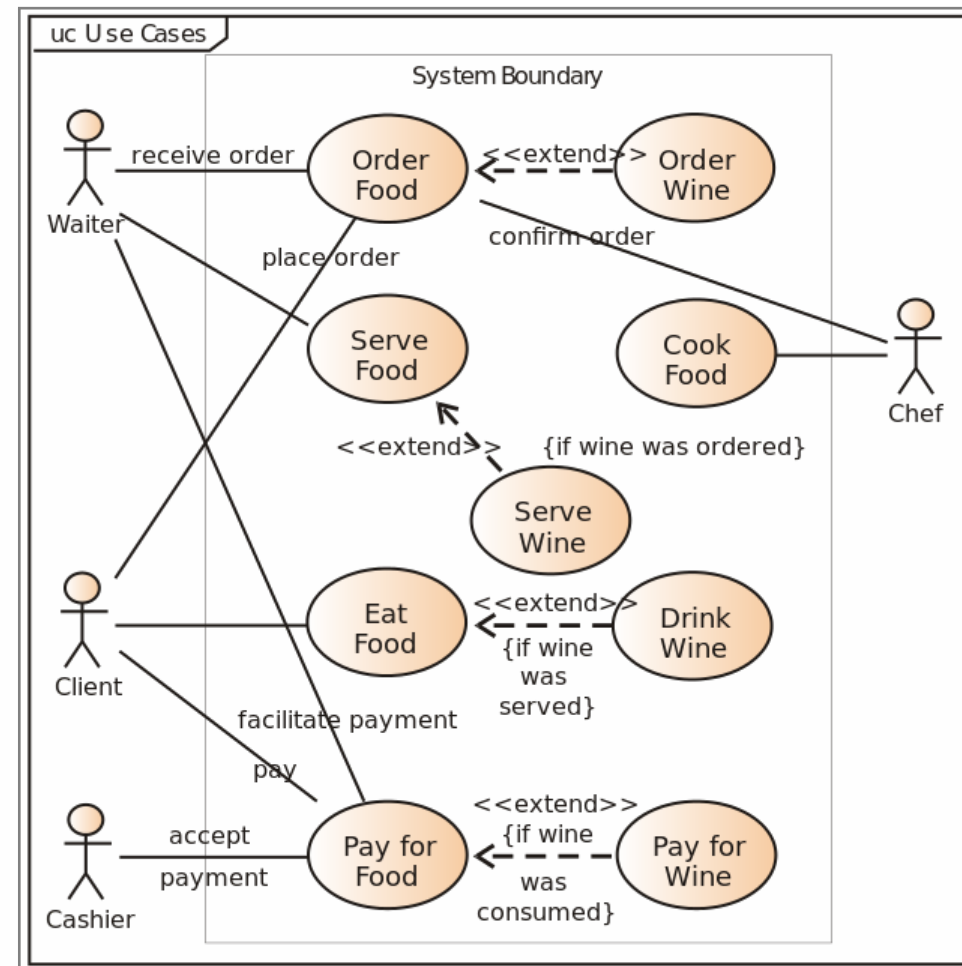
Chapter #2:
FPA, IFPUG, COSMIC

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)

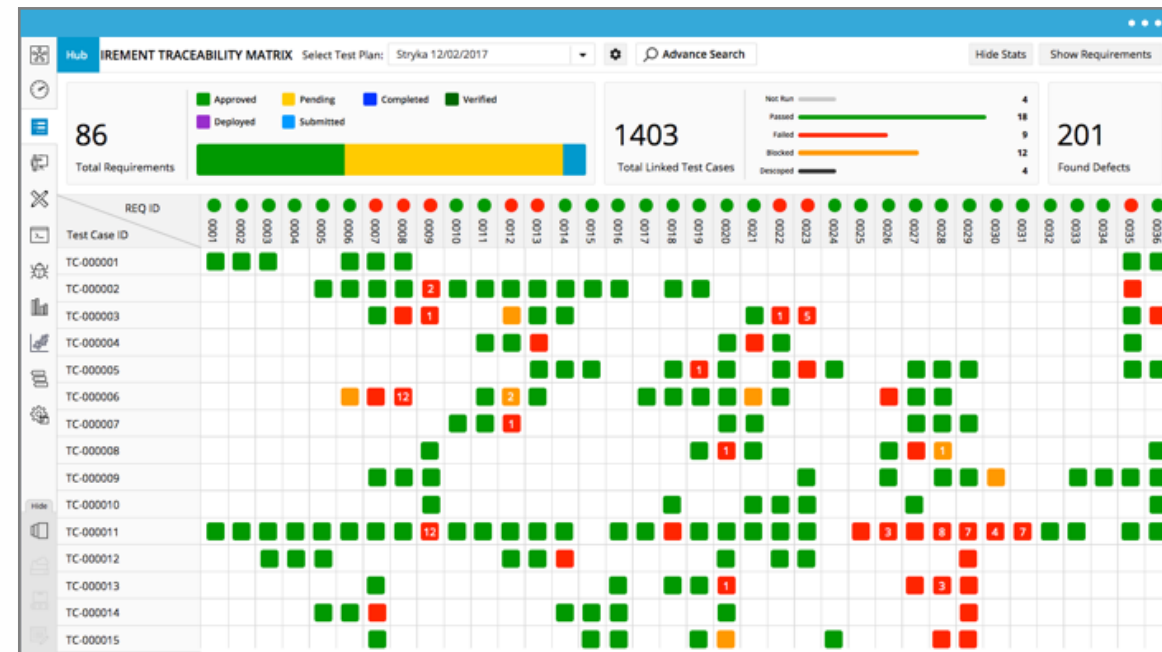Function Points (FPs) are used for estimates

Also, read about Use Case Points (UCP)
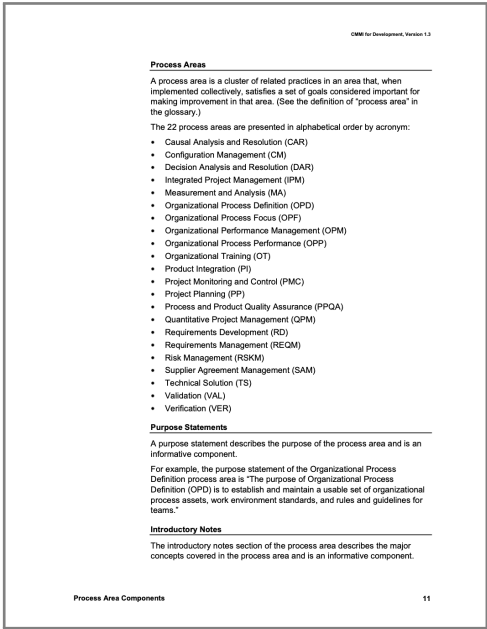
Chapter #3:
Traceability Matrix

Traceability Matrix

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

Chapter #4:
Verification and Validation

"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: $A = \dfrac{E_{\text{up}}}{E_{\text{down}} + E_{\text{up}}}$

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."

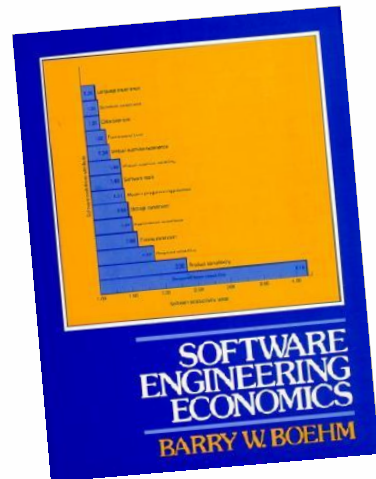"At least 80% of beta users must anonymously confirm that the UI is attractive enough."

"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

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

— Barry W. Boehm. Software Engineering Economics. *IEEE Transactions on Software Engineering*, 10(1):4–21, 1984. doi:10.1109/tse.1984.5010193

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

Then, we find effort adjustment factor (F).

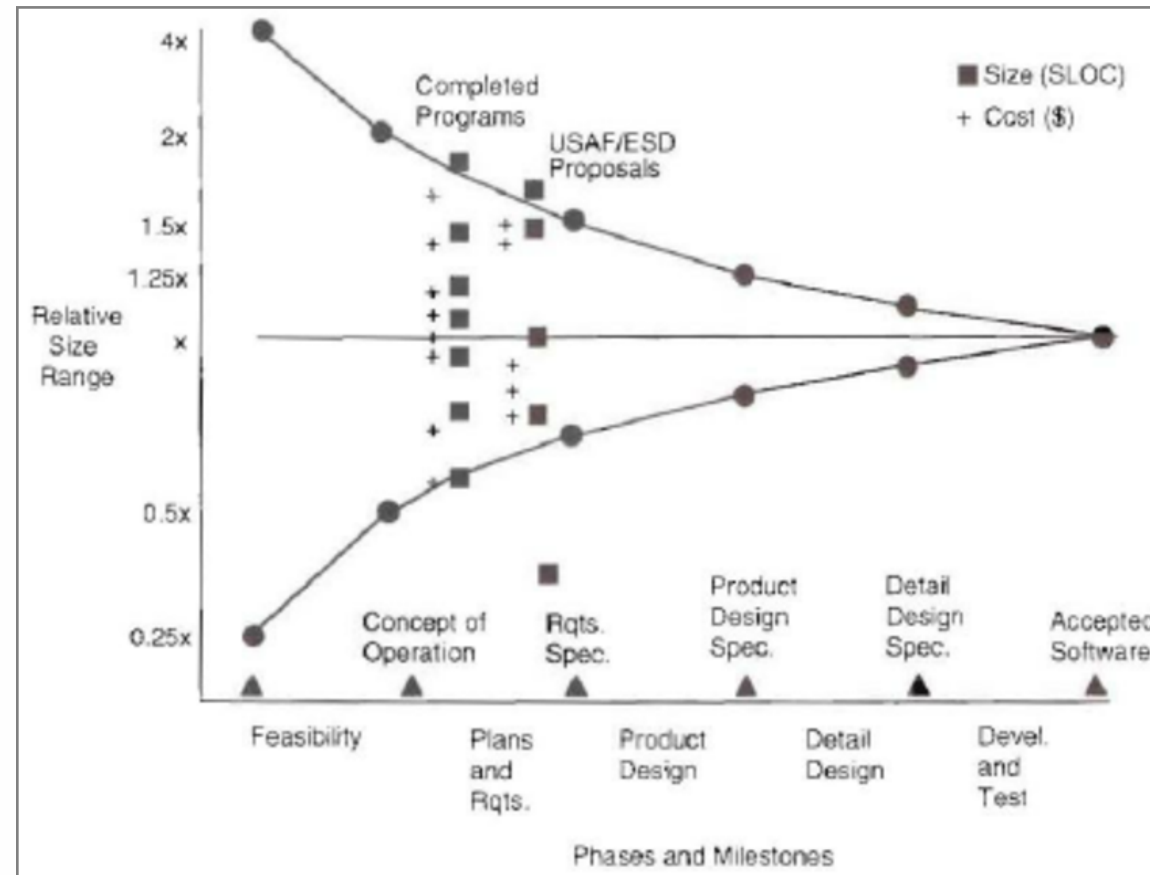Then, we find coefficients a, b, and c using the table.

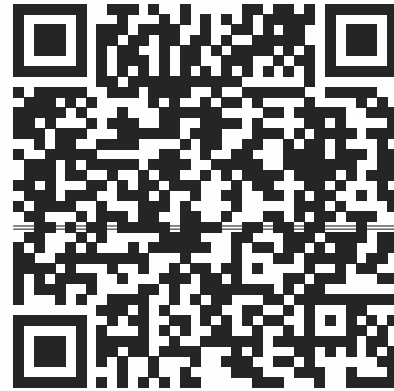Then, we calculate the size in man-months:

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

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

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

Cone of Uncertainty
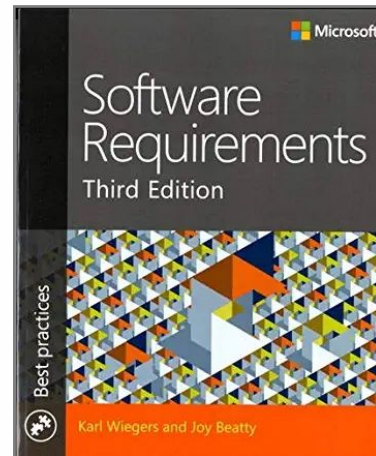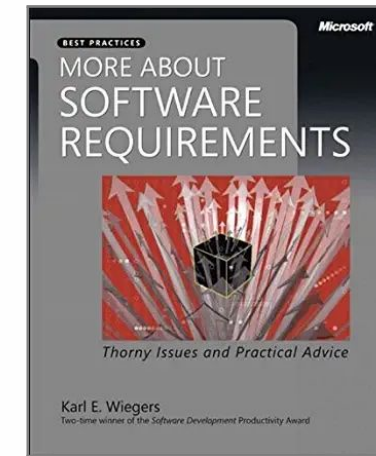
"How much for this software?" they ask.

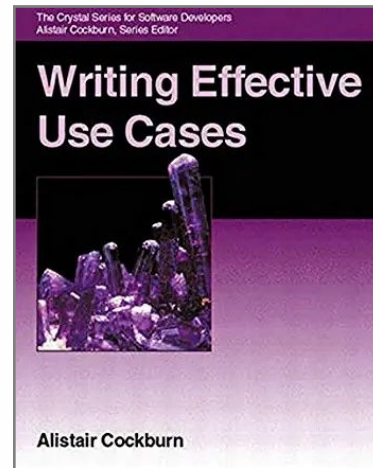"The development will take forever and will consume all your money," we answer.
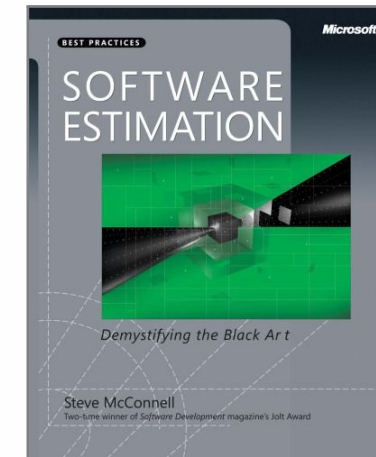
Chapter #7:
Books, Venues, Call-to-Action

Karl E. Wiegers and Joy Beatty. *Software Requirements.* Microsoft Press, 2 edition, 2003

Karl E. Wiegers. *More About Software Requirements: Thorny Issues and Practical Advice.* Microsoft Press, 2006

Alistair Cockburn and Lord Cockburn. *Writing Effective Use Cases.* Addison-Wesley, 2000. doi:10.5555/517669

Steve McConnell. *Software Estimation: Demystifying the Black Art.* Microsoft Press, 2006. doi:10.5555/1204642

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 <u>validate</u> requirements automatically?

- How to <u>trace</u> them automatically?

- How to <u>specify</u> in Controlled Natural Language (CNL)?

- How to <u>reduce</u> them?

Bibliography

Barry W. Boehm. Software Engineering Economics. *IEEE Transactions on Software Engineering*, 10(1):4–21, 1984. doi:10.1109/tse.1984.5010193.

Yegor Bugayenko. *Code Ahead.* Amazon, 2018.

Alistair Cockburn and Lord Cockburn. *Writing Effective Use Cases.* Addison-Wesley, 2000. doi:10.5555/517669.

Martin Fowler. *UML Distilled: A Brief Guide to the Standard Object Modeling Language.* Addison-Wesley, 3 edition, 2004.

Robert C. Martin. *Agile Software Development, Principles, Patterns, and Practices.* Prentice Hall, 2002.

doi:10.5555/515230.

Steve McConnell. *Software Estimation: Demystifying the Black Art.* Microsoft Press, 2006. doi:10.5555/1204642.

Karl E. Wiegers. *More About Software Requirements: Thorny Issues and Practical Advice.* Microsoft Press, 2006.

Karl E. Wiegers and Joy Beatty. *Software Requirements.* Microsoft Press, 2 edition, 2003.