IS212: Software Project Management

Week 1: Software Project Management Fundamentals

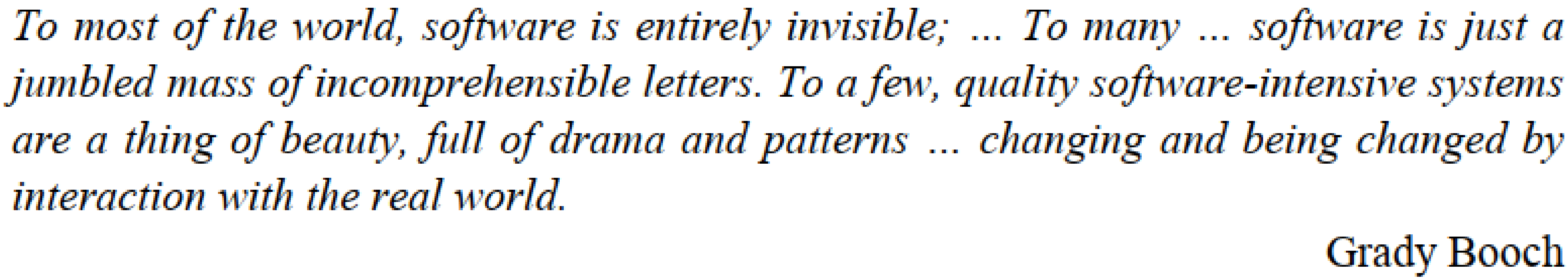
Agenda

- Software and software engineering
  - Characteristics of software
  - Limits of software
  - What is software engineering?
  - Problems confronted by software engineering
- Industrial software development
  - Software development: In the ideal and in the real
  - Software Life Cycle (SLC) vs. Software Development Life Cycle (SDLC)
  - Evolutionary trends in software development
  - Different development philosophies: Waterfall, rapid prototyping, iterative and incremental
  - Algorithm, process, methodology
  - Processes: Individual, team, organization
  - A quick comparison of methodologies
  - Different stakeholder views
  - Software development: Workflows and Phases

Agenda contd.

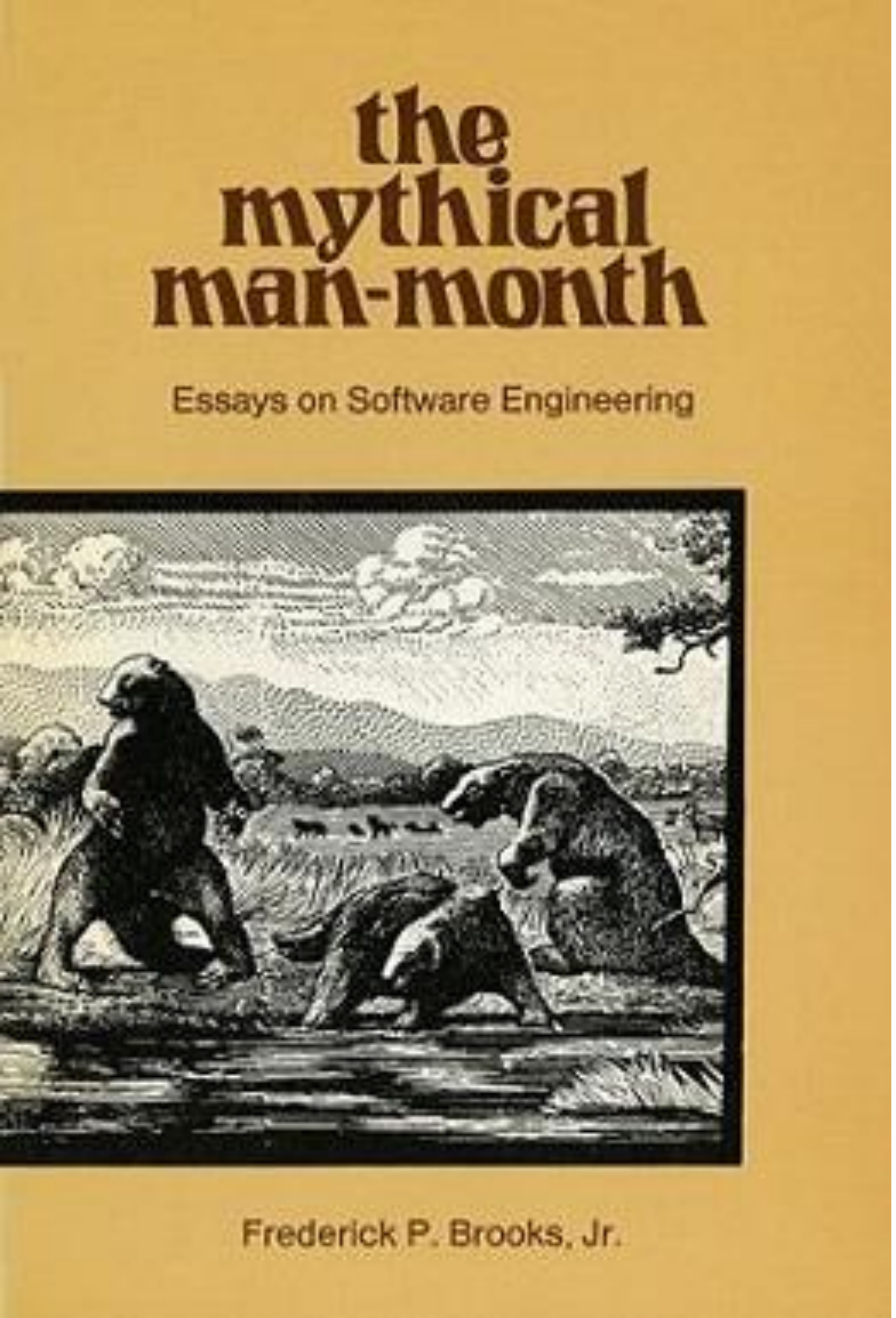
- Key themes in software project management
  - That elusive something ☺
  - Four P’s of software development
  - Software project life cycle
  - Variation of the cost of change and stakeholder influence with time
  - Co-evolution of the problem and solution domains
  - Role of automation in software development
  - Principles of software project management
  - Project management: Process Groups
  - Project management: Knowledge areas
  - Why, what, and how of a given project
  - Project management plan
  - Team dynamics
  - Important project management activities
  - Managing versus leading
- Software project management: A practitioner’s view

Software and Software Engineering



To most of the world, software is entirely invisible; … To many … software is just a jumbled mass of incomprehensible letters. To a few, quality software-intensive systems are a thing of beauty, full of drama and patterns … changing and being changed by interaction with the real world.

Grady Booch

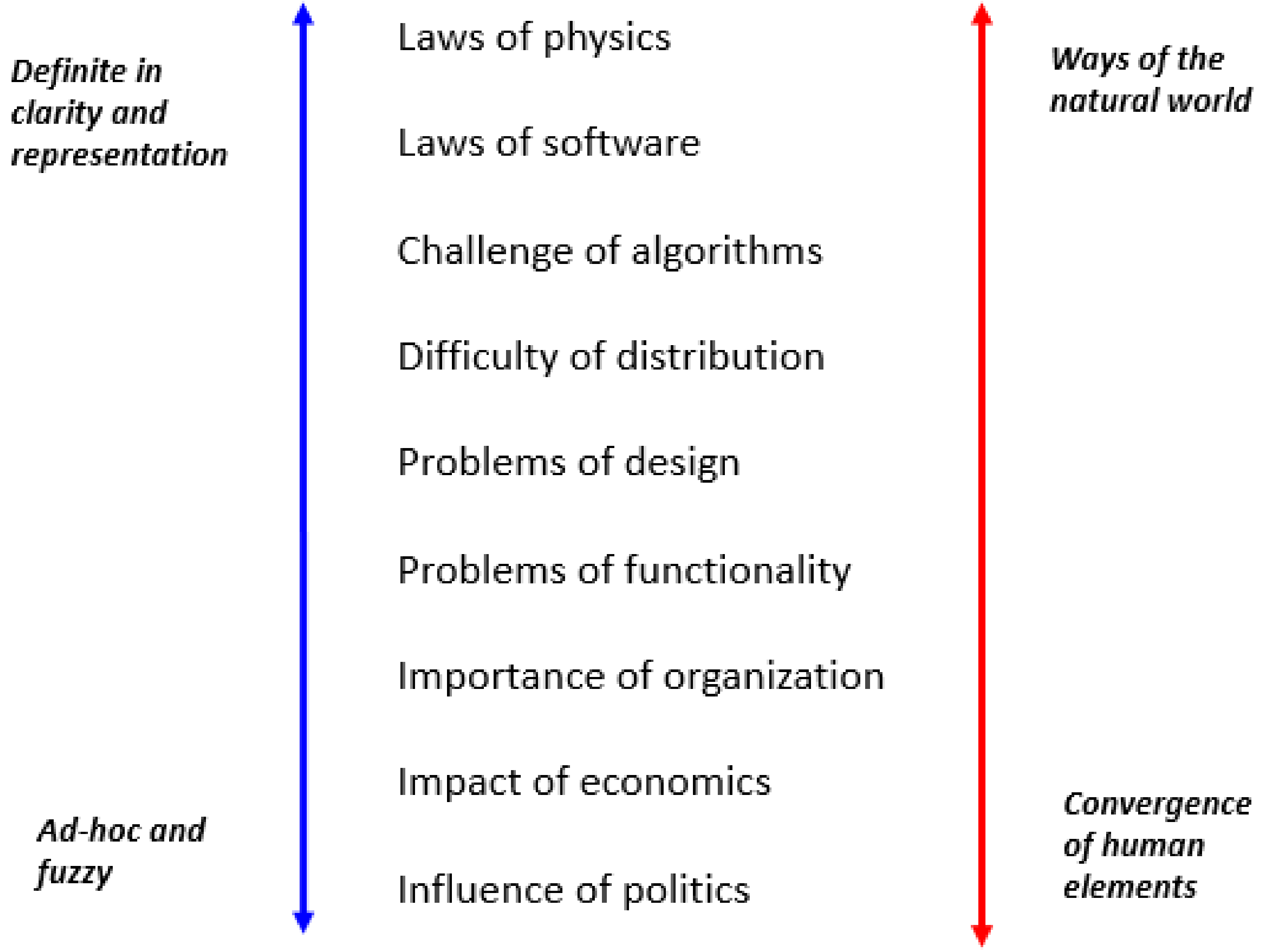


Characteristics of software

- Software is inherently complex
- Software must be made to conform to existing interfaces
- Software is constantly subject to change
- Software is invisible and unvisualizable

https://upload.wikimedia.org/wikipedia/en/f/fd/Mythical_man-month_%28book_cover%29.jpg

Limits of software



Definite in clarity and representation

Ad-hoc and fuzzy

Laws of physics

Laws of software

Challenge of algorithms

Difficulty of distribution

Problems of design

Problems of functionality

Importance of organization

Impact of economics

Influence of politics

Ways of the natural world

Convergence of human elements

Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

What is software engineering?

- According to Boehm, software engineering involves the application of science and mathematics through which the facilities of computer equipment are made useful to human beings via computer programs, procedures, and associated documentation.
- Pfleeger identifies software engineering with the utilization of tools, techniques, procedures, and paradigms toward quality improvement of the software product.
- Naur and Randall see software engineering in terms of establishing and using sound engineering principles to obtain economically effective and reliable software that can work efficiently on real machines

What is software engineering?

- According to Freeman and Von Staa, software engineering involves the organized application of methods, tools, and knowledge towards fulfilling stated technical, economic, and human goals for a software-intensive system.
- Kacmar says simply applying engineering principles to designing and constructing computer software can be termed software engineering.
- To Schach, software engineering is the discipline that aims at producing fault-free software, to be delivered on time and within budget, which satisfies the user’s needs.

What is software engineering?

- Whitmire describes software engineering as a ‘slippery’ term, and says for some it is something that can only be applied to a large project, while to others it is just a ‘figment of collective imaginations’. He gives a working definition as, ‘Software engineering is the science and art of designing and building, with economy and elegance, software systems and applications so they can fill the uses to which they may be subjected’

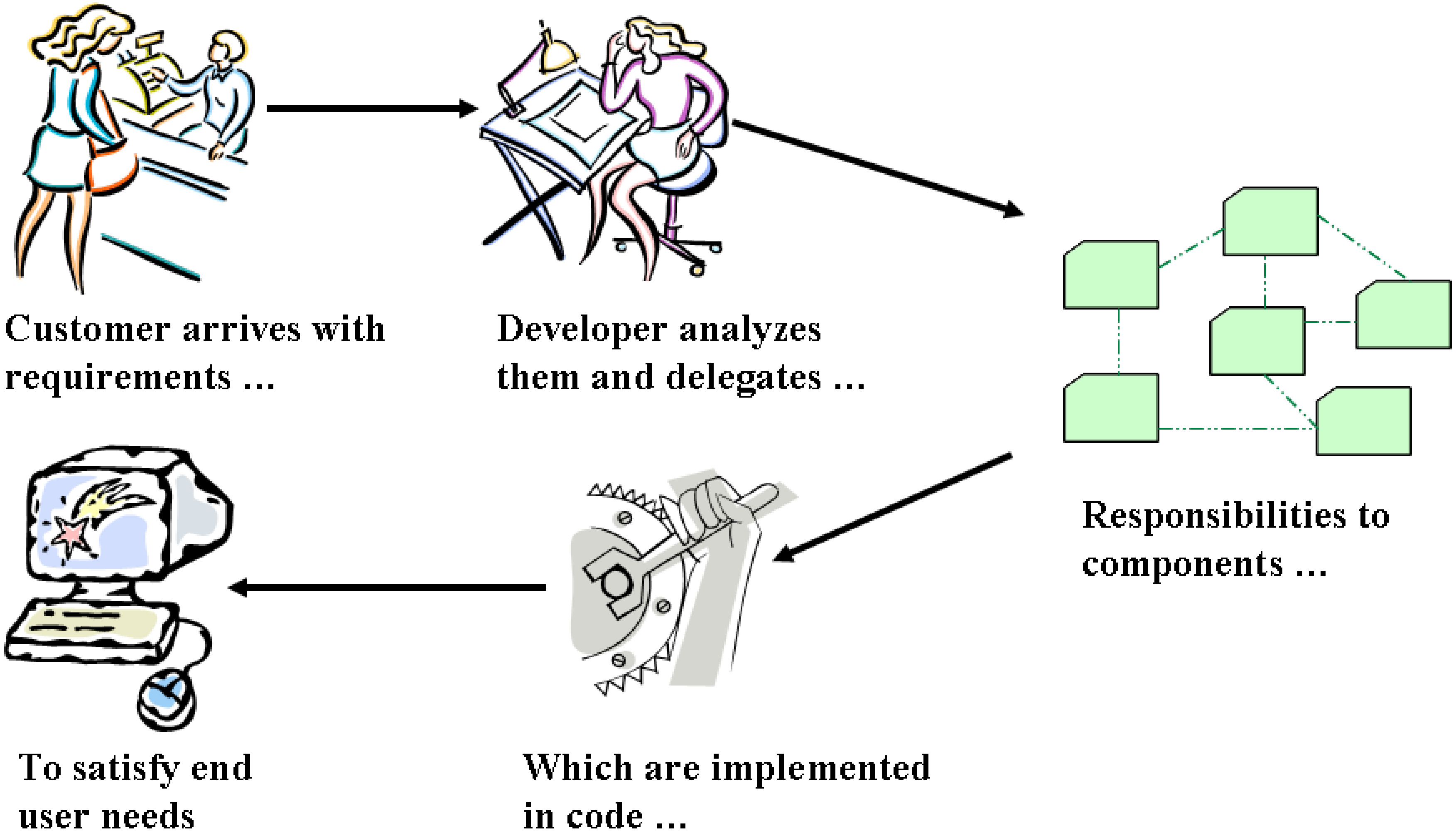
Problems confronted by software engineering

Problem of change

Problem of complexity

Industrial software development

Software development: In the ideal



Customer arrives with requirements …

Developer analyzes them and delegates …

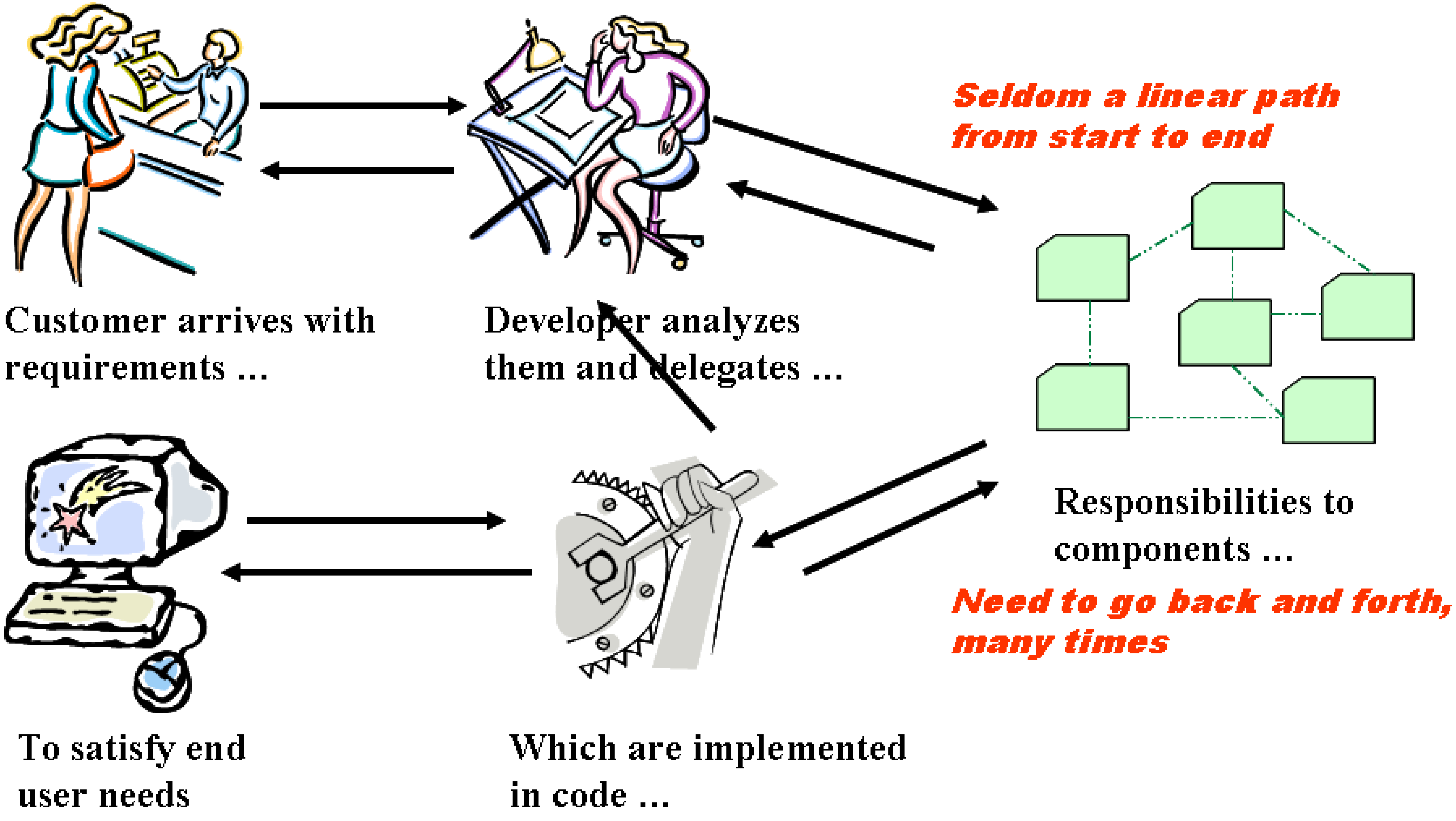
Responsibilities to components …

Which are implemented in code …

To satisfy end user needs

Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

Software development: In the real



Customer arrives with requirements …

Developer analyzes them and delegates …

Seldom a linear path from start to end

Responsibilities to components …

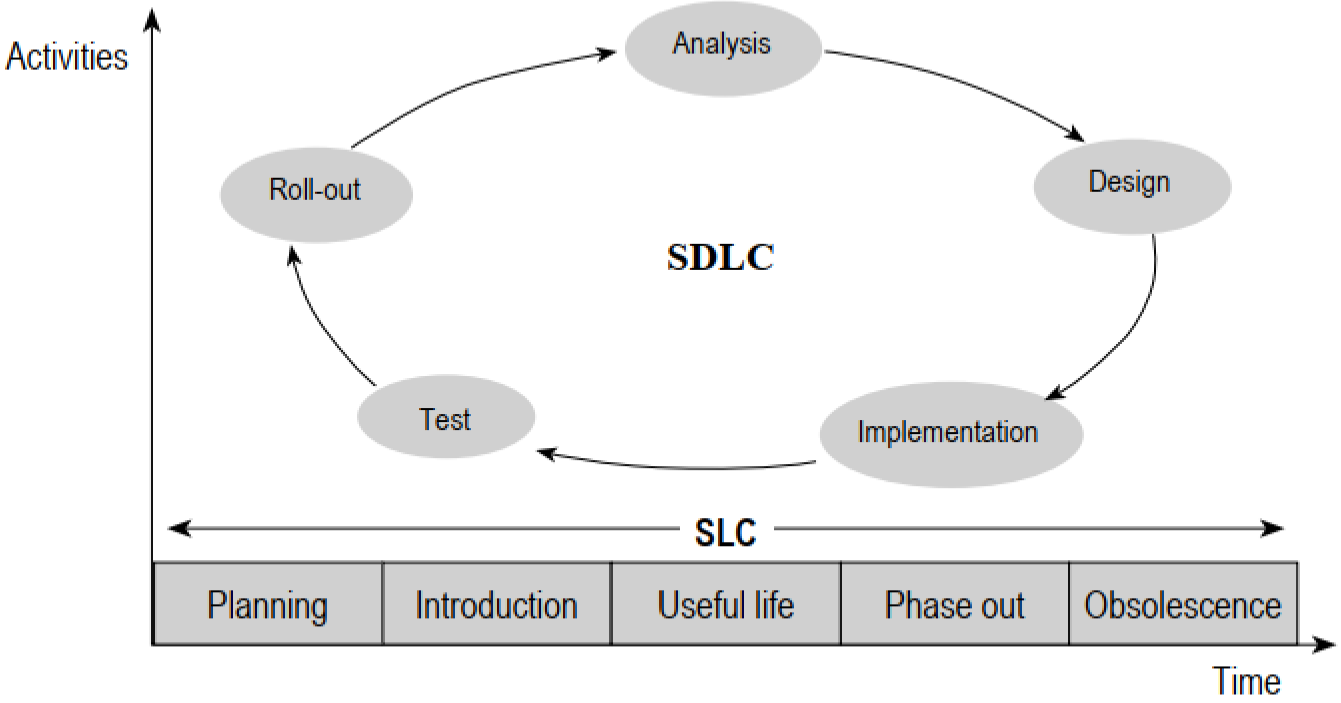
Need to go back and forth, many times

Which are implemented in code …

To satisfy end user needs

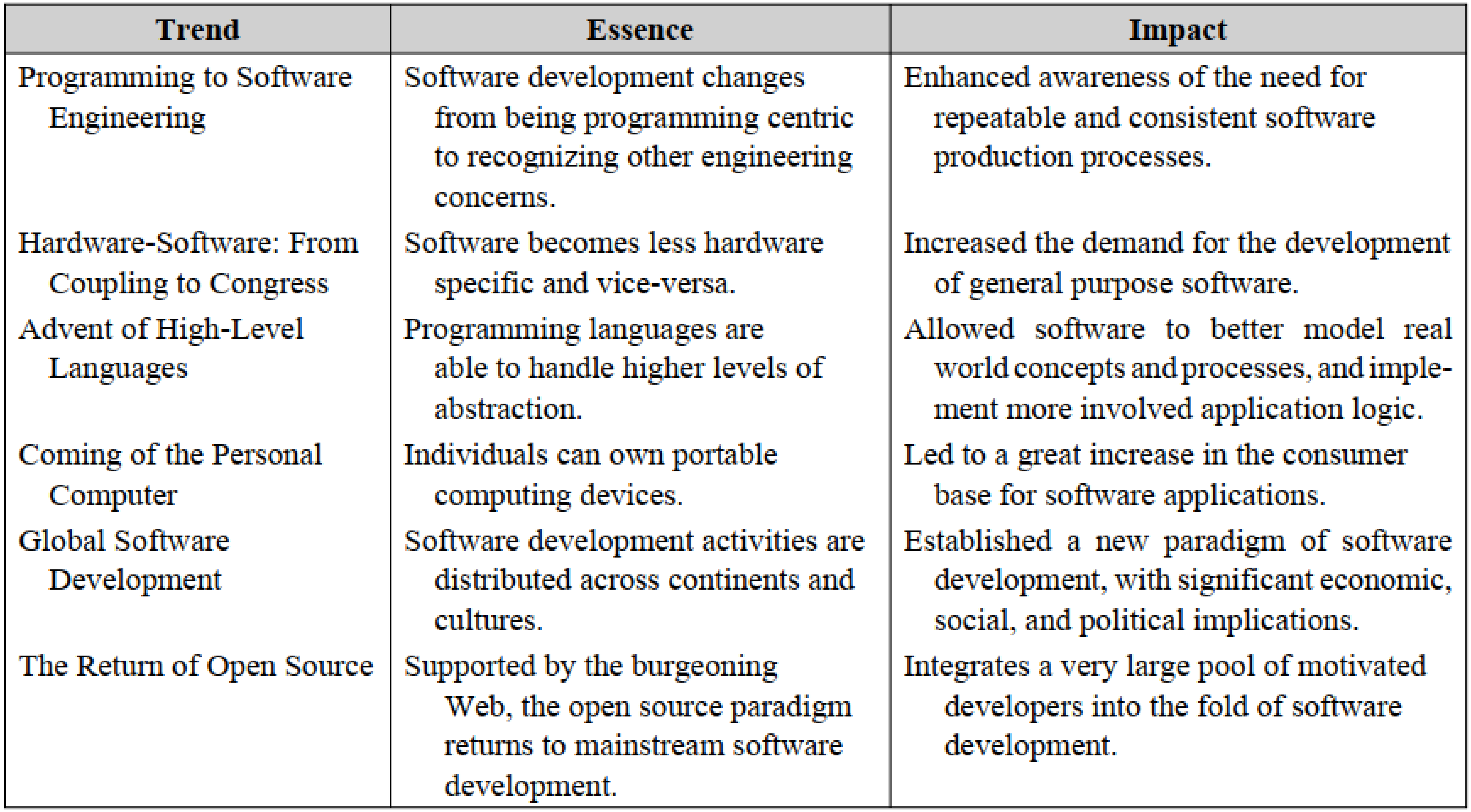
Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

Software Life Cycle (SLC) vs. Software Development Life Cycle (SDLC)



Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

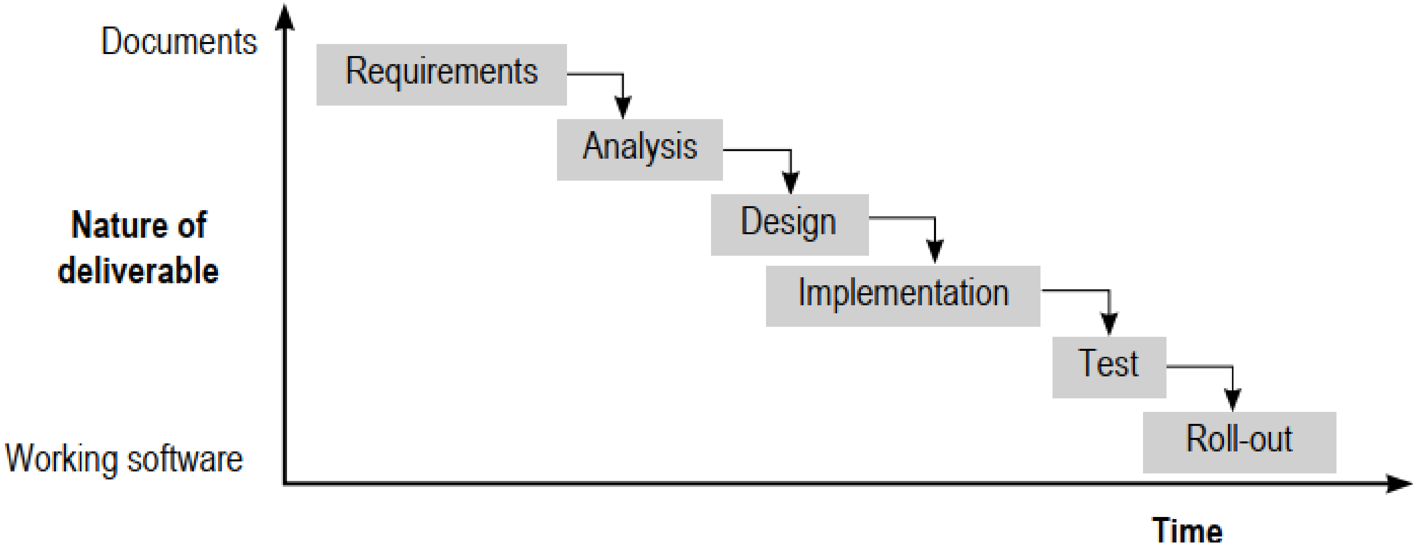
Evolutionary trends in software development



Trend	Essence	Impact
Programming to Software Engineering	Software development changes from being programming centric to recognizing other engineering concerns.	Enhanced awareness of the need for repeatable and consistent software production processes.
Hardware-Software: From Coupling to Congress	Software becomes less hardware specific and vice-versa.	Increased the demand for the development of general purpose software.
Advent of High-Level Languages	Programming languages are able to handle higher levels of abstraction.	Allowed software to better model real world concepts and processes, and implement more involved application logic.
Coming of the Personal Computer	Individuals can own portable computing devices.	Led to a great increase in the consumer base for software applications.
Global Software Development	Software development activities are distributed across continents and cultures.	Established a new paradigm of software development, with significant economic, social, and political implications.
The Return of Open Source	Supported by the burgeoning Web, the open source paradigm returns to mainstream software development.	Integrates a very large pool of motivated developers into the fold of software development.

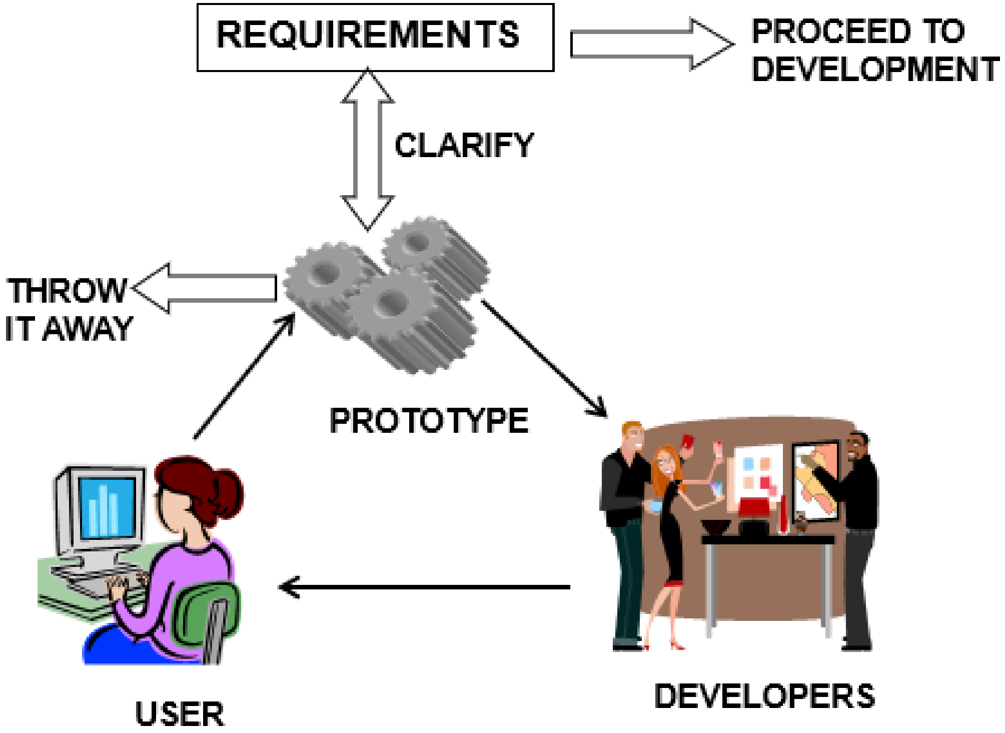
Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

Different development philosophies: Waterfall



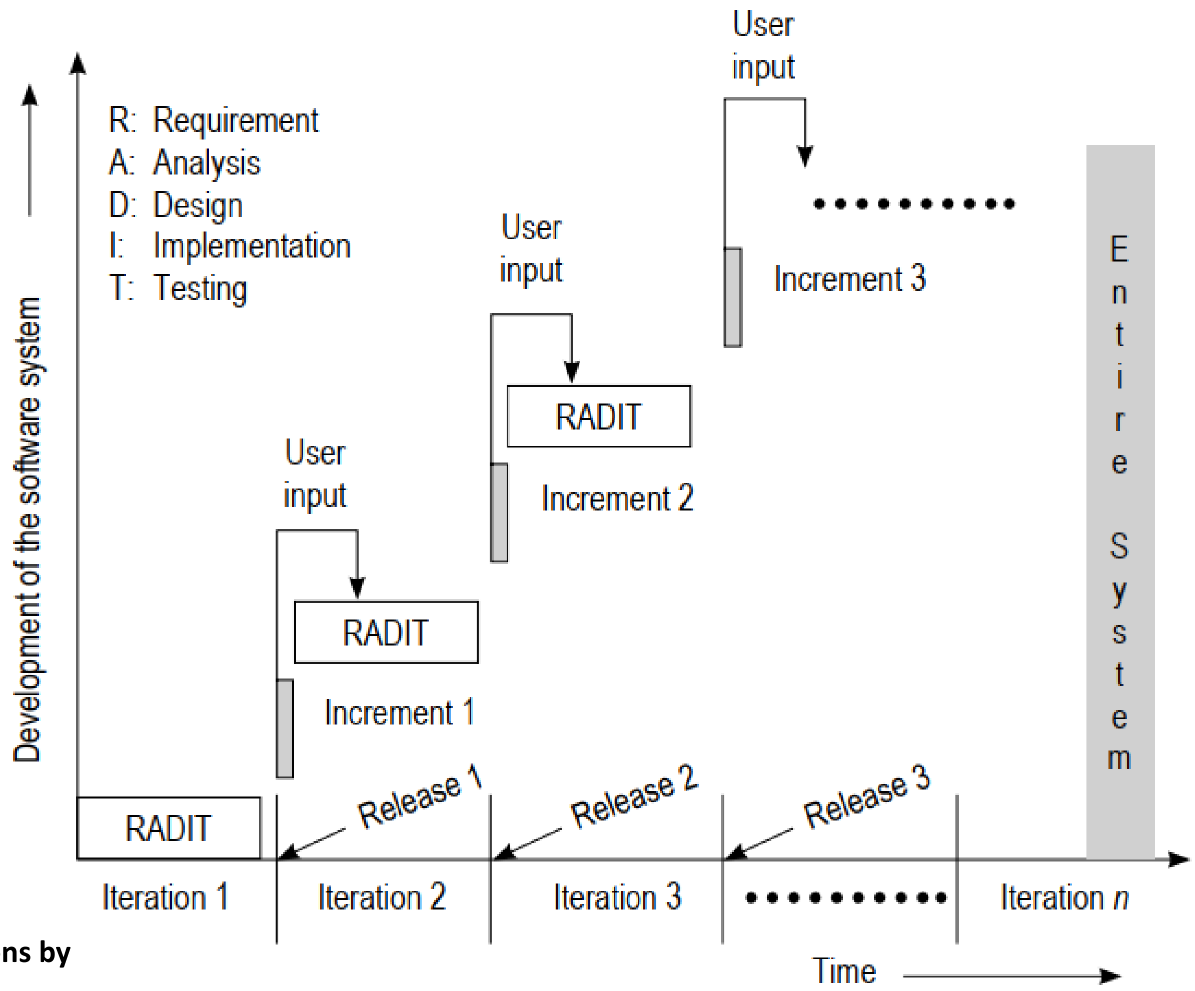
Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

Different development philosophies: Rapid prototyping



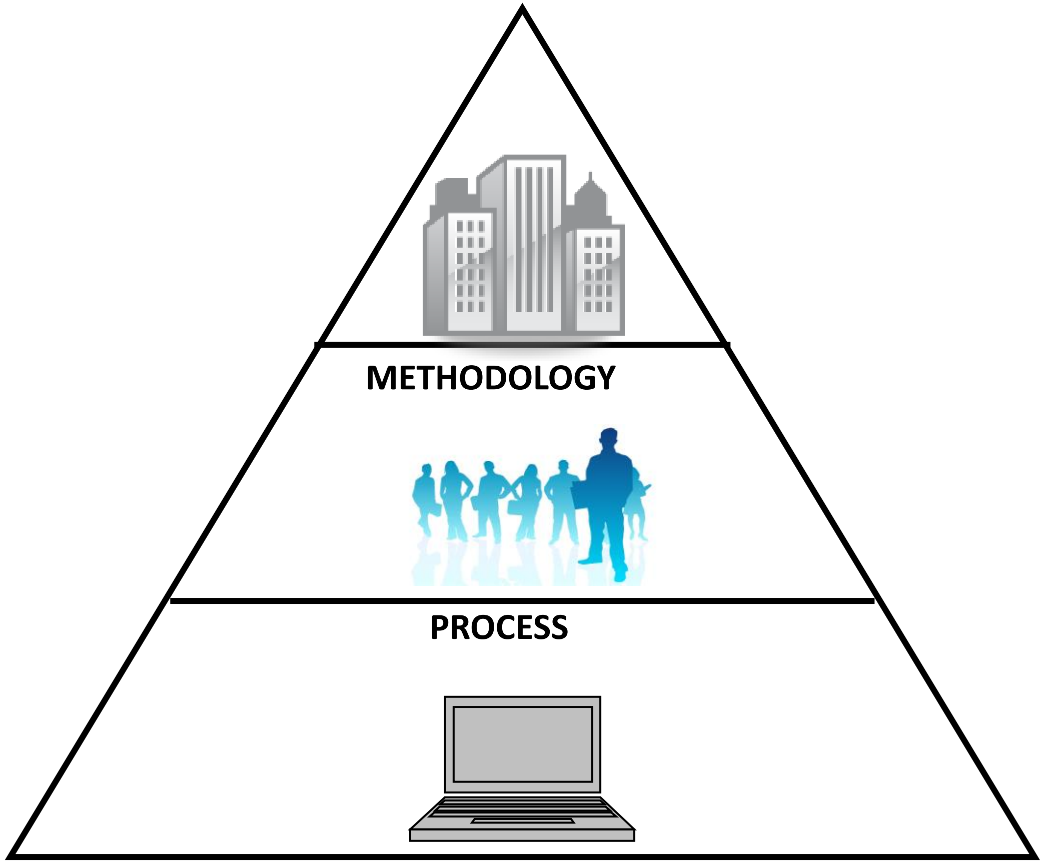
Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

Different development philosophies: Iterative and incremental



Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

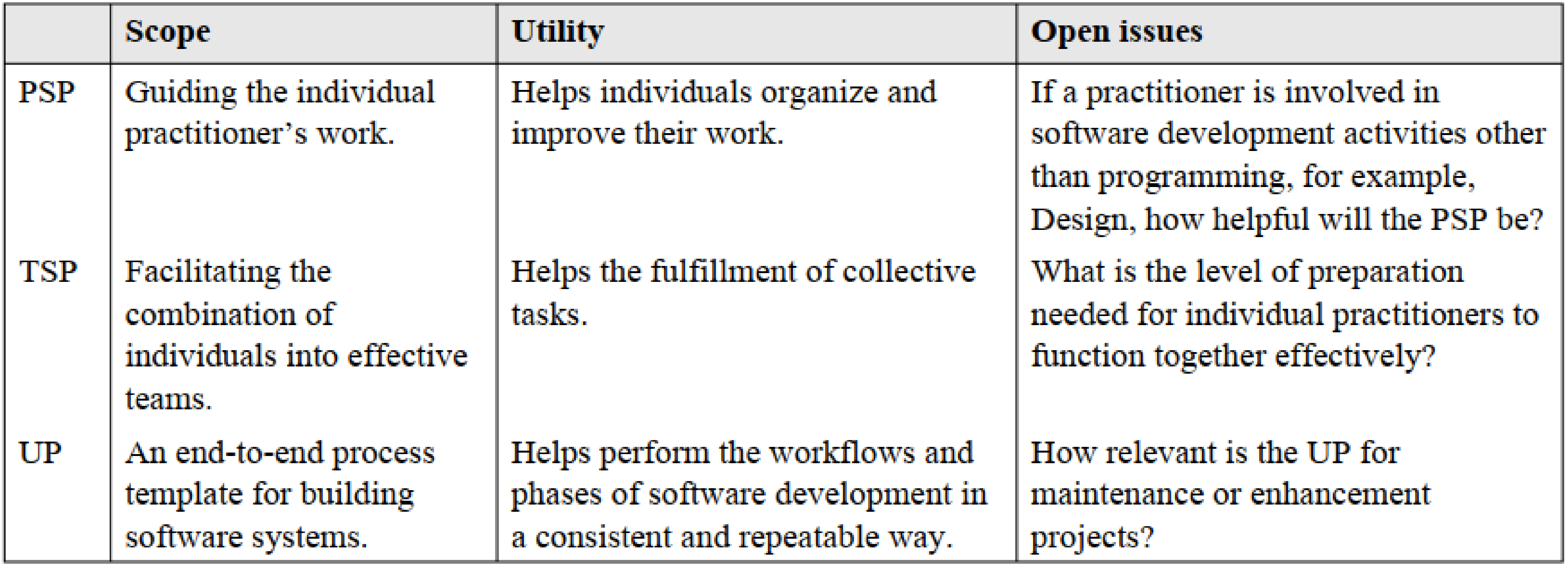
Algorithm, process, methodology



ALGORITHM

Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

Processes: Individual, team, organization

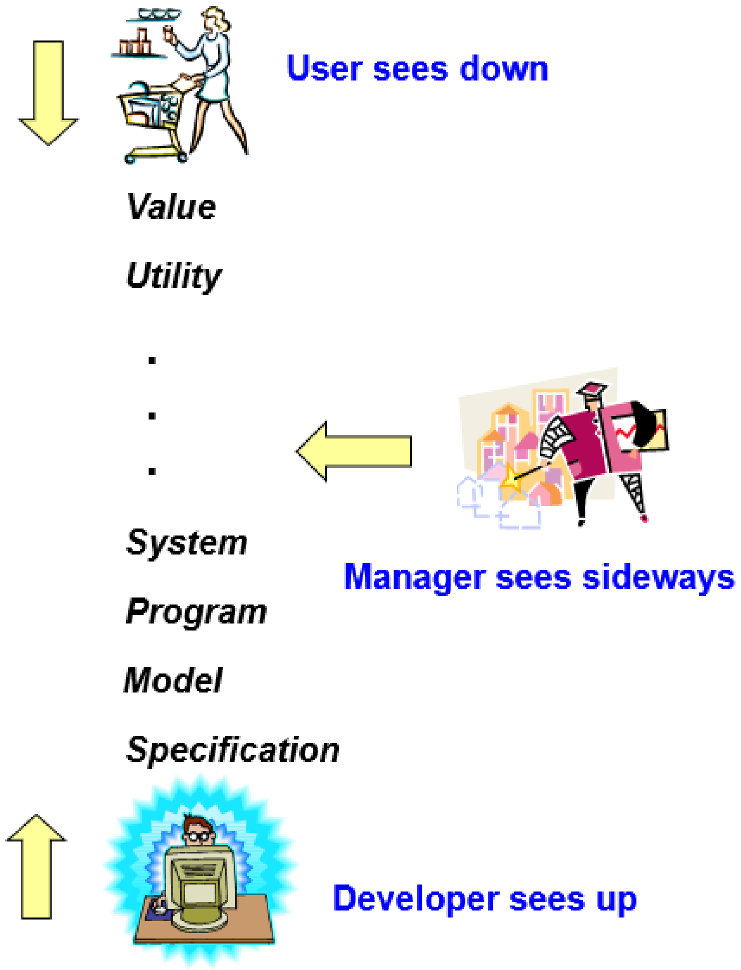


	Scope	Utility	Open issues
PSP	Guiding the individual practitioner's work.	Helps individuals organize and improve their work.	If a practitioner is involved in software development activities other than programming, for example, Design, how helpful will the PSP be?
TSP	Facilitating the combination of individuals into effective teams.	Helps the fulfillment of collective tasks.	What is the level of preparation needed for individual practitioners to function together effectively?
UP	An end-to-end process template for building software systems.	Helps perform the workflows and phases of software development in a consistent and repeatable way.	How relevant is the UP for maintenance or enhancement projects?

PSP = Personal Software Process. TSP = Team Software Process. UP = Unified Process

Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

Different stakeholder views



User sees down

Value

Utility

.

.

.

Manager sees sideways

System

Program

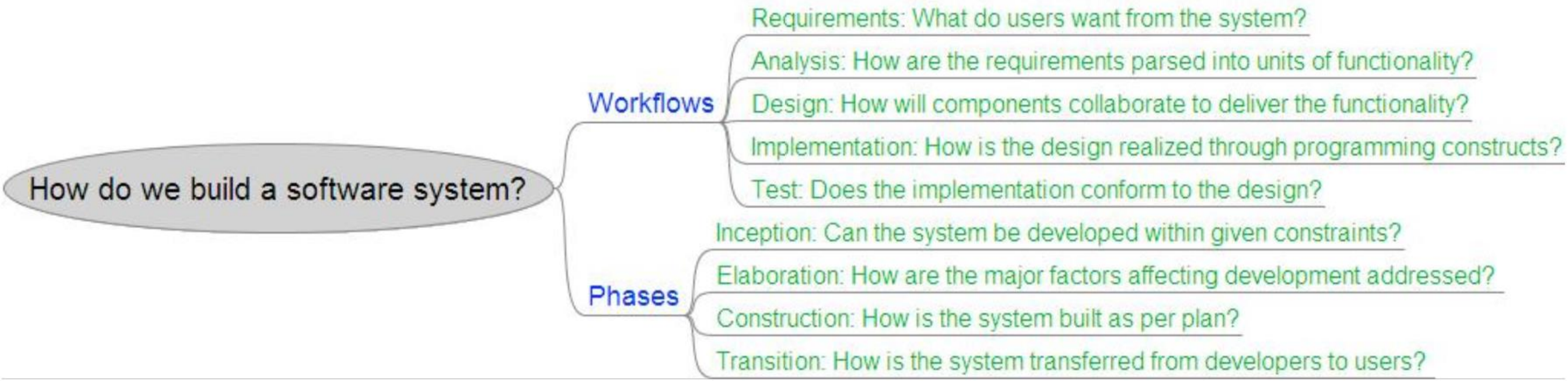
Model

Specification

Developer sees up

Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

Software development: Workflows and Phases

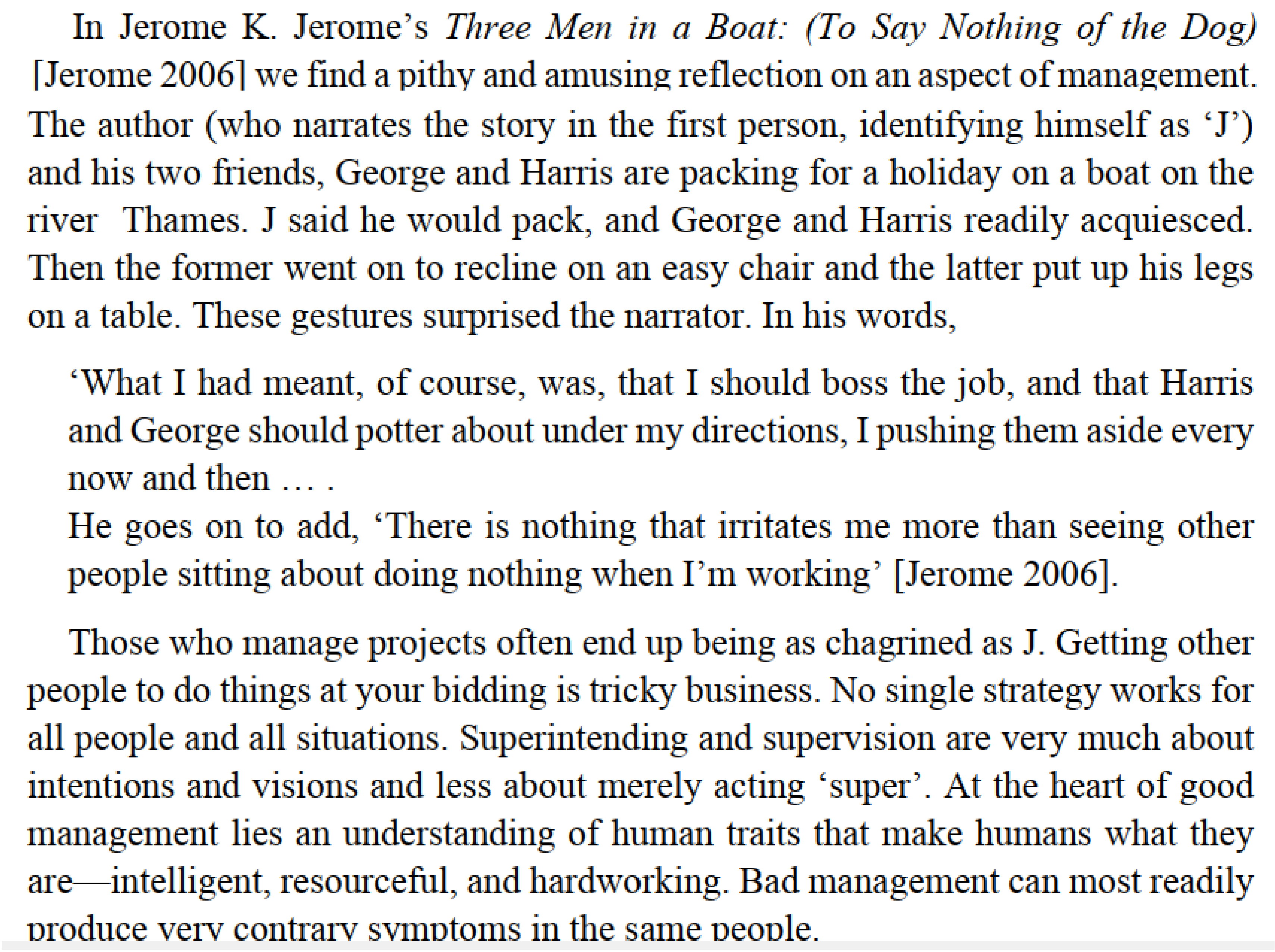


How do we build a software system?

- Workflows
  - Requirements: What do users want from the system?
  - Analysis: How are the requirements parsed into units of functionality?
  - Design: How will components collaborate to deliver the functionality?
  - Implementation: How is the design realized through programming constructs?
  - Test: Does the implementation conform to the design?
- Phases
  - Inception: Can the system be developed within given constraints?
  - Elaboration: How are the major factors affecting development addressed?
  - Construction: How is the system built as per plan?
  - Transition: How is the system transferred from developers to users?

Key themes in software project management

That elusive something ☺



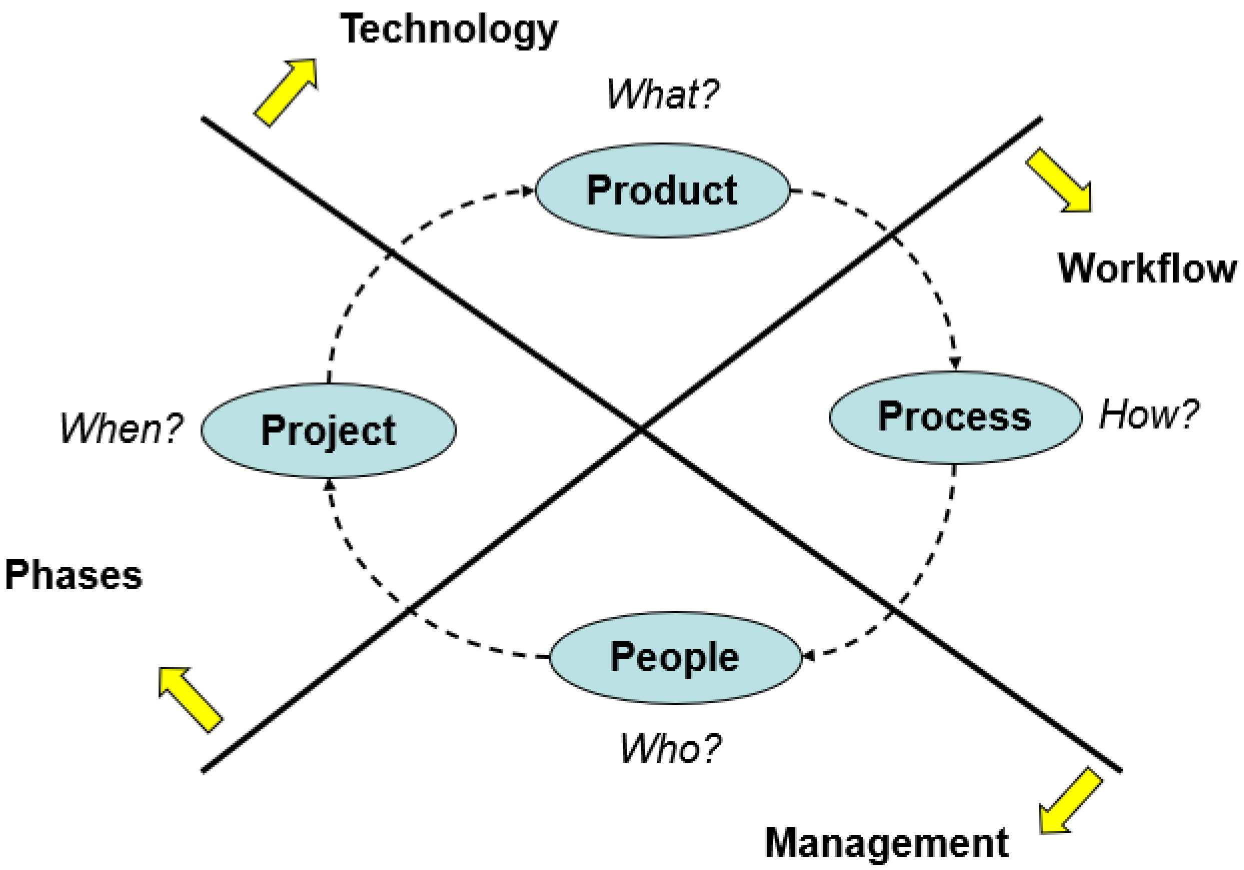
In Jerome K. Jerome's *Three Men in a Boat: (To Say Nothing of the Dog)* [Jerome 2006] we find a pithy and amusing reflection on an aspect of management. The author (who narrates the story in the first person, identifying himself as 'J') and his two friends, George and Harris are packing for a holiday on a boat on the river Thames. J said he would pack, and George and Harris readily acquiesced. Then the former went on to recline on an easy chair and the latter put up his legs on a table. These gestures surprised the narrator. In his words,

> 'What I had meant, of course, was, that I should boss the job, and that Harris and George should potter about under my directions, I pushing them aside every now and then … .
>
> He goes on to add, 'There is nothing that irritates me more than seeing other people sitting about doing nothing when I'm working' [Jerome 2006].

Those who manage projects often end up being as chagrined as J. Getting other people to do things at your bidding is tricky business. No single strategy works for all people and all situations. Superintending and supervision are very much about intentions and visions and less about merely acting 'super'. At the heart of good management lies an understanding of human traits that make humans what they are—intelligent, resourceful, and hardworking. Bad management can most readily produce very contrary symptoms in the same people.

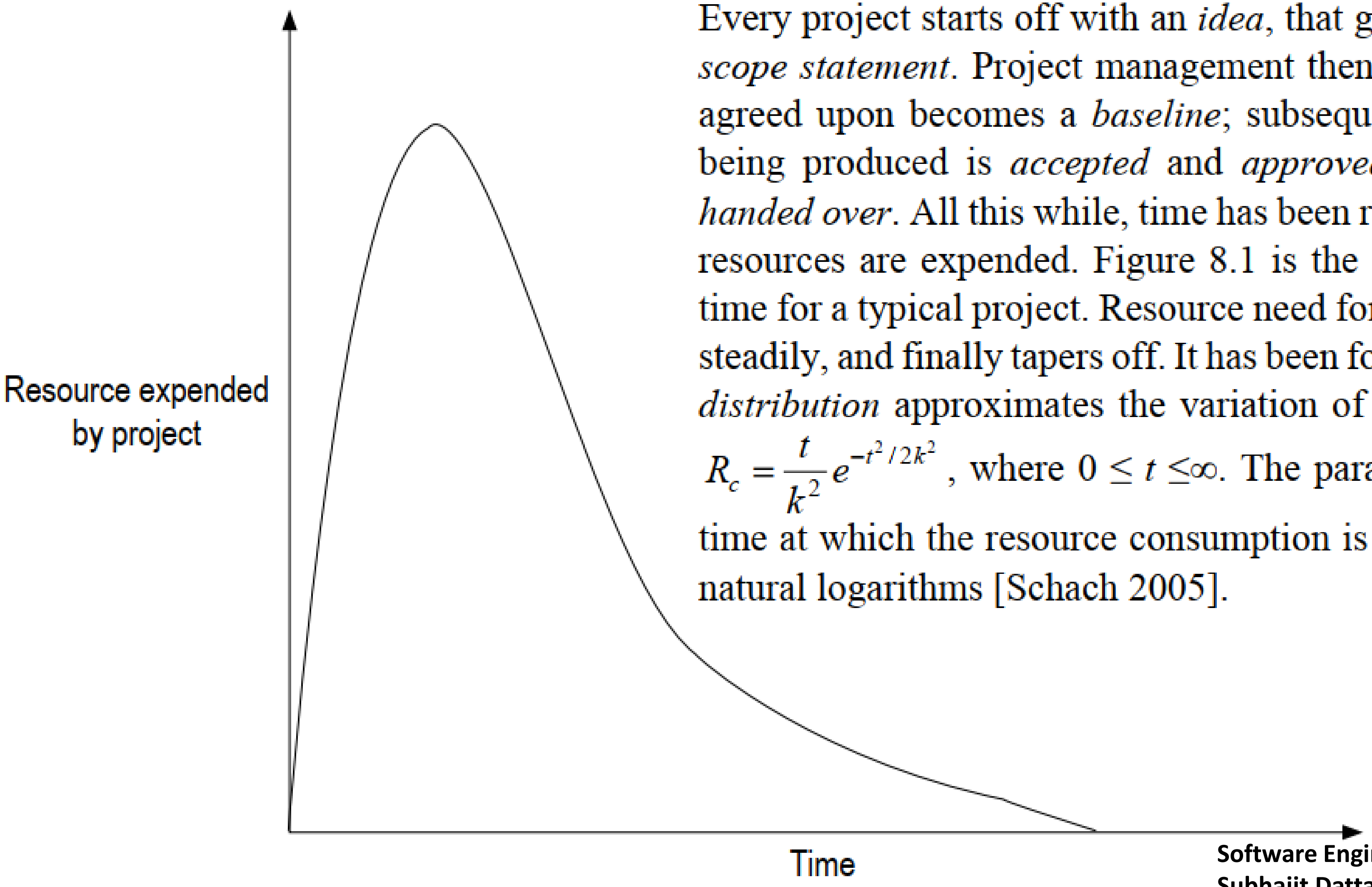
Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

The four P’s of software development



Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

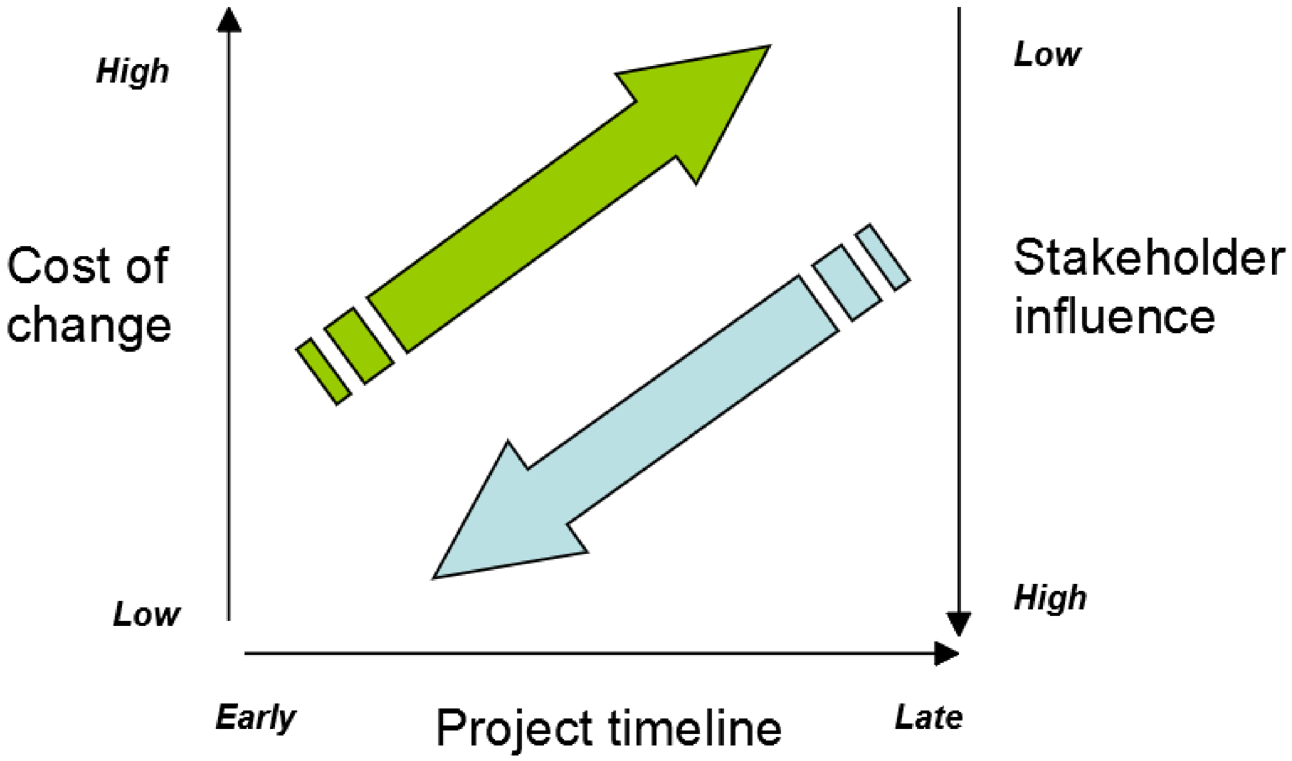
Software project life cycle



Every project starts off with an *idea*, that gets clarified in a *charter*, and then in a *scope statement*. Project management then needs to produce a *plan* which when agreed upon becomes a *baseline*; subsequently *progress* is tracked, the product being produced is *accepted* and *approved* by the customers, and it is finally *handed over*. All this while, time has been running and it is interesting to note how resources are expended. Figure 8.1 is the curve for resource expenditure versus time for a typical project. Resource need for a project starts off low, then increases steadily, and finally tapers off. It has been found that for large projects the *Rayleigh distribution* approximates the variation of resource consumption R_c with time t: $R_c = \frac{t}{k^2} e^{-t^2/2k^2}$, where $0 \leq t \leq\infty$. The parameter k is a constant and denotes the time at which the resource consumption is at its peak, and $e \approx 2.718$, the base of natural logarithms [Schach 2005].

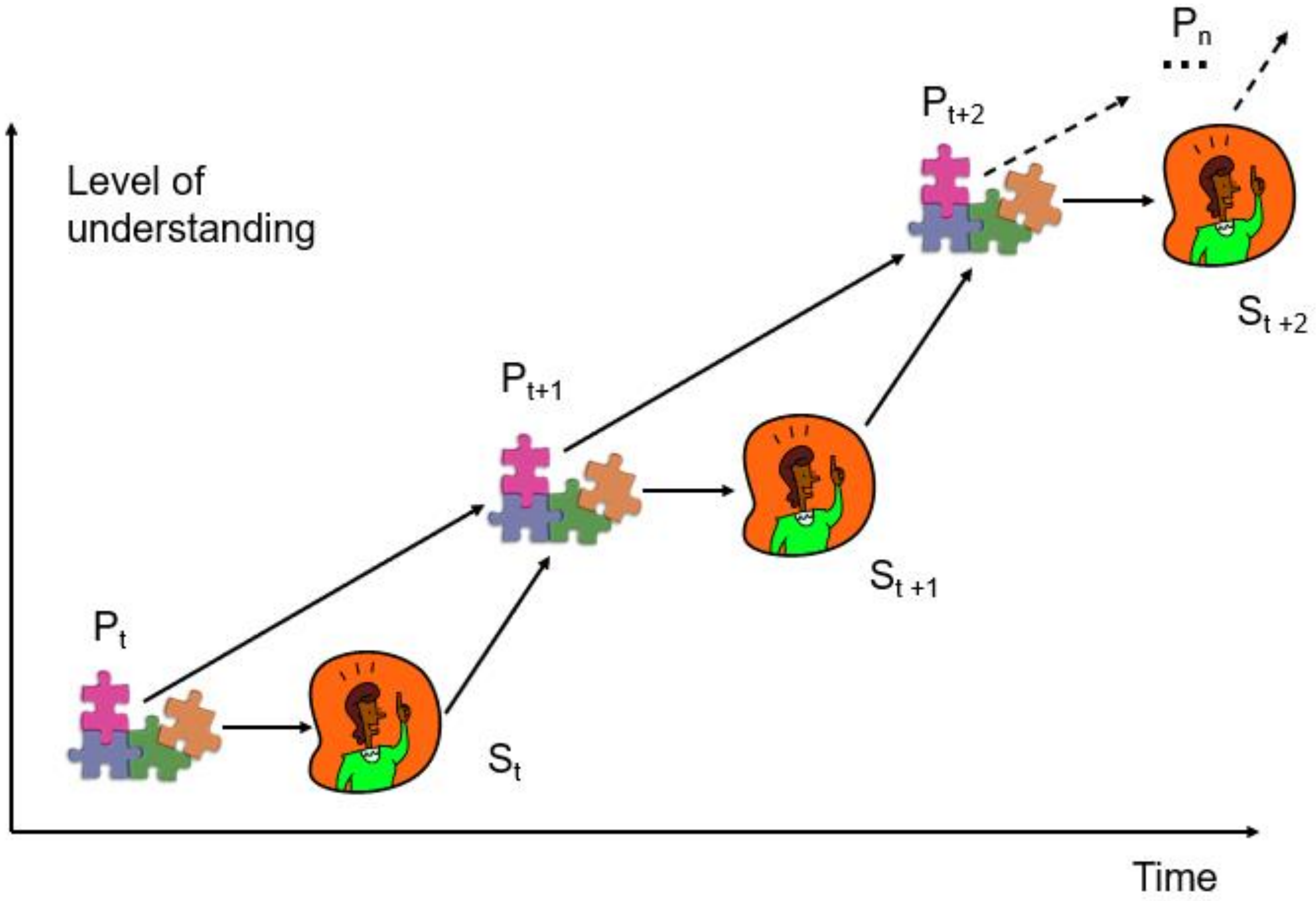
Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

Variation of the cost of change and stakeholder influence with time



Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

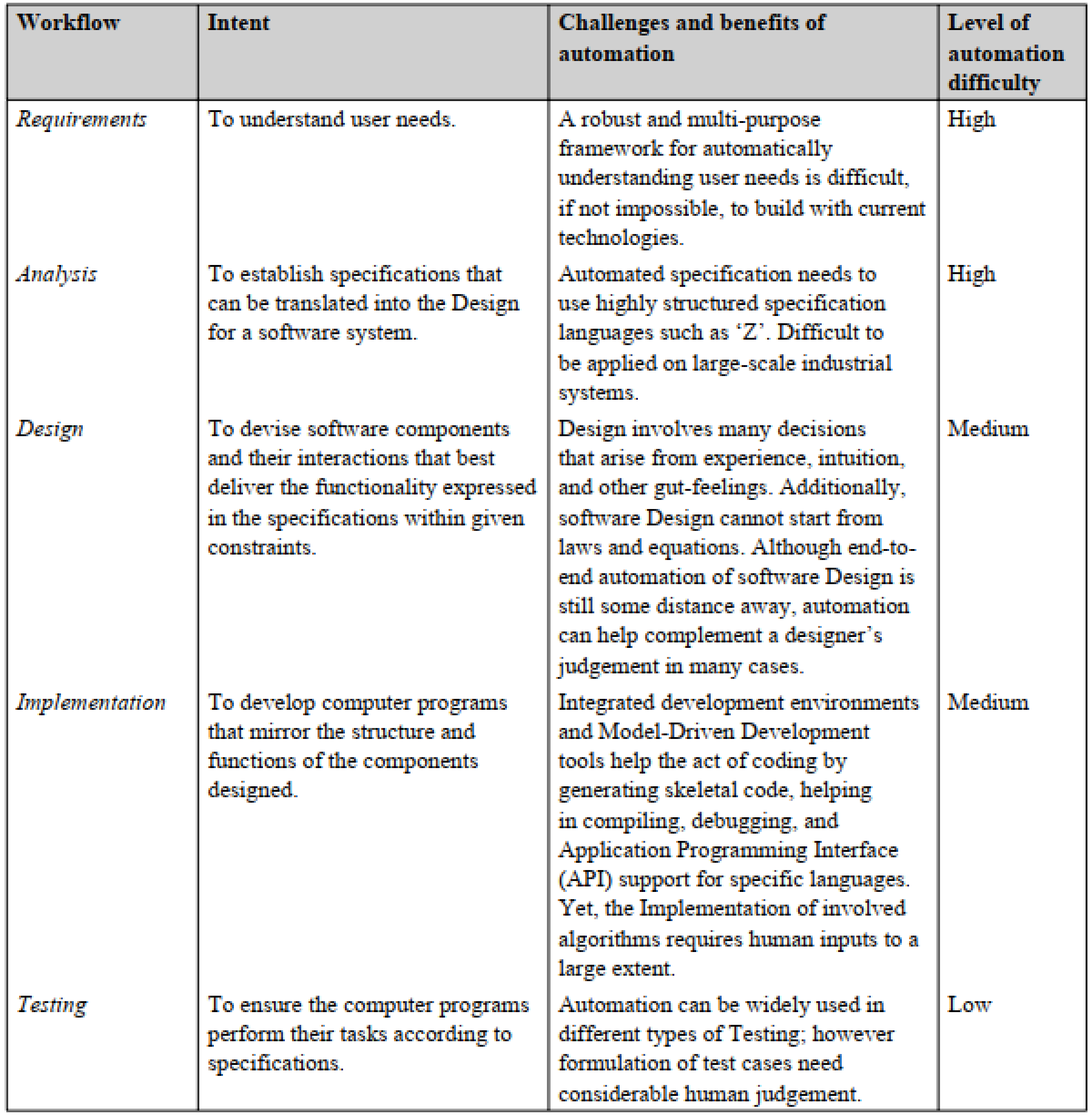
Co-evolution of the problem and solution domains



(P_t and S_t represent states of the problem and solution at time t)

Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

Role of automation in software development



Workflow	Intent	Challenges and benefits of automation	Level of automation difficulty
Requirements	To understand user needs.	A robust and multi-purpose framework for automatically understanding user needs is difficult, if not impossible, to build with current technologies.	High
Analysis	To establish specifications that can be translated into the Design for a software system.	Automated specification needs to use highly structured specification languages such as 'Z'. Difficult to be applied on large-scale industrial systems.	High
Design	To devise software components and their interactions that best deliver the functionality expressed in the specifications within given constraints.	Design involves many decisions that arise from experience, intuition, and other gut-feelings. Additionally, software Design cannot start from laws and equations. Although end-to-end automation of software Design is still some distance away, automation can help complement a designer's judgement in many cases.	Medium
Implementation	To develop computer programs that mirror the structure and functions of the components designed.	Integrated development environments and Model-Driven Development tools help the act of coding by generating skeletal code, helping in compiling, debugging, and Application Programming Interface (API) support for specific languages. Yet, the Implementation of involved algorithms requires human inputs to a large extent.	Medium
Testing	To ensure the computer programs perform their tasks according to specifications.	Automation can be widely used in different types of Testing; however formulation of test cases need considerable human judgement.	Low

Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

Principles of software project management

- Software project management needs to be based on an architecture-first approach
- Iterative development has to be adopted so that risks are met and resolved early
- Component-based development needs to be encouraged
- A change management environment has to be established
- Tools supporting round trip engineering need to be used
- Design has to be documented in formal model-based notation
- The quality of the product as well as progress of the project has to be objectively assessed
- Intermediate artefacts need to be assessed through a demonstration-based approach
- Intermediate releases should be planned with evolving levels of detail
- A configurable process that is economically scalable needs to be established

Project management: Process groups

- Initiating Process Group
  - Aims at defining and authorizing the project or a project phase
- Planning Process Group
  - Aims at defining and refining objectives, plans, and course of action towards fulfilling the scope and target of the project
- Executing Process Group
  - Aims at integrating people and other resources to ensure that the project management plan is carried out properly
- Monitoring and Controlling Process Group
  - Aims at regularly measuring and monitoring progress to detect variances from the project management plan such that remedial action may be taken as necessary.

Project management: Knowledge areas

- Project Integration Management
  - Includes the process and activities that are required to ‘identify, define, combine, unify, and coordinate’ the various processes and activities with the process groups
- Project Scope Management
  - Includes the processes that are needed to ensure all the work that the project needs to meet its goals successfully are addressed
- Project Time Management
  - Includes the processes needed to complete the project on time
- Project Cost Management
  - Includes the processes involved in planning, estimating, budgeting, and controlling costs for the project to be completed within budget

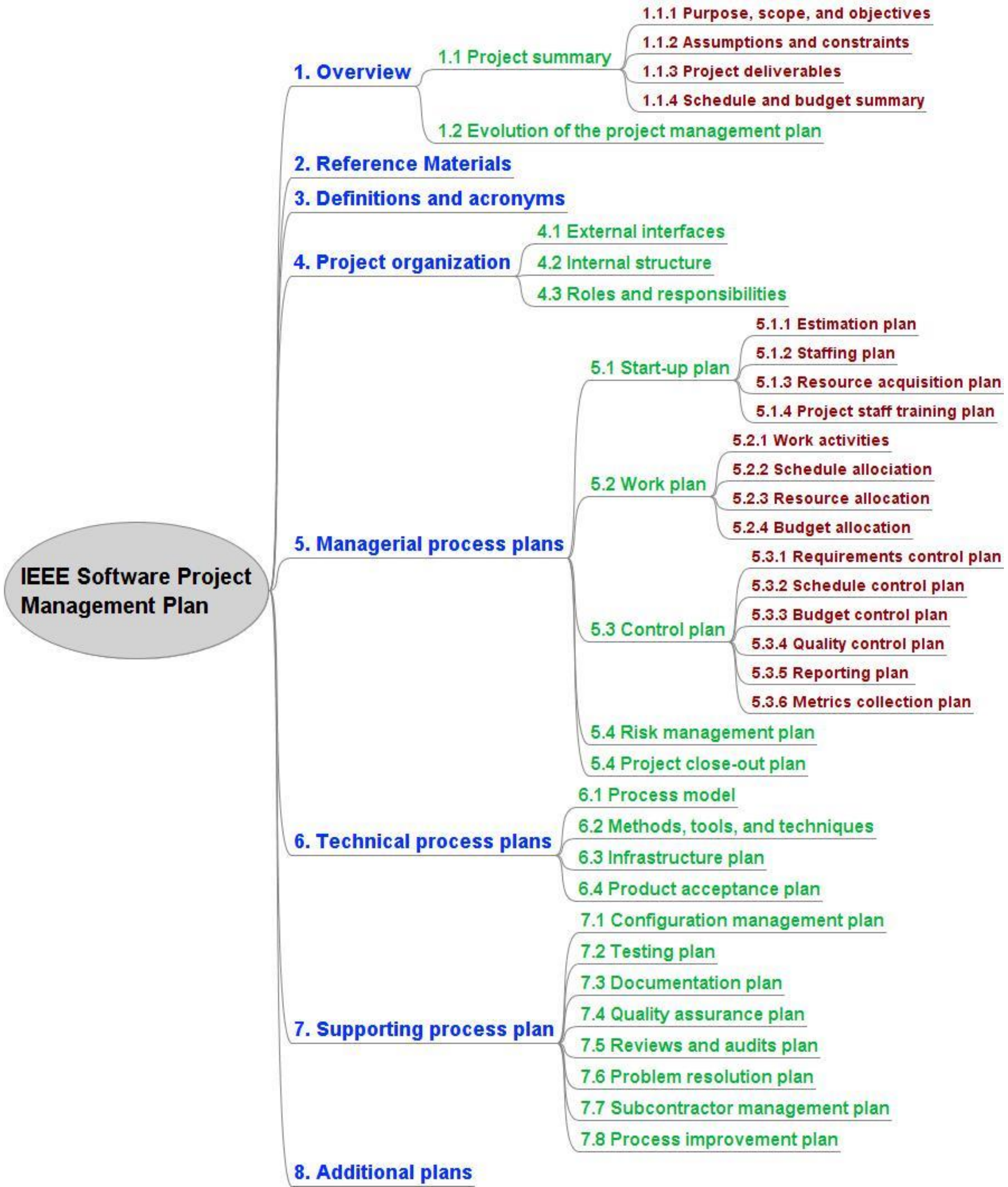
Project management: Knowledge areas (contd.)

- Project Quality Management
  - Includes all the activities that determine quality policies, objectives, and responsibilities towards satisfying the needs the project seeks to address
- Project Human Resource Management
  - Includes processes that organize and manage the project team
- Project Communications Management
  - Includes the processes needed to ensure ‘timely and appropriate generation, collection, distribution, storage, and retrieval of project information
- Project Risk Management
  - Includes processes to foresee, detect, and address risks associated with a project
- Project Procurement Management
  - Includes the processes to acquire goods or services from outside the project team that are essential for the completion of the project.

Why, what, and how of a given project

- Project Charter
  - Gives a formal authorization for the project, often with justification as to why the project was commissioned in the first place
- Project Scope Statement
  - Specifies what the project needs to accomplish and what deliverables need to be produced
- Project Management Plan
  - Describes how the project's work will be performed

Software project management plan



IEEE Software Project Management Plan

- 1. Overview
  - 1.1 Project summary
    - 1.1.1 Purpose, scope, and objectives
    - 1.1.2 Assumptions and constraints
    - 1.1.3 Project deliverables
    - 1.1.4 Schedule and budget summary
  - 1.2 Evolution of the project management plan
- 2. Reference Materials
- 3. Definitions and acronyms
- 4. Project organization
  - 4.1 External interfaces
  - 4.2 Internal structure
  - 4.3 Roles and responsibilities
- 5. Managerial process plans
  - 5.1 Start-up plan
    - 5.1.1 Estimation plan
    - 5.1.2 Staffing plan
    - 5.1.3 Resource acquisition plan
    - 5.1.4 Project staff training plan
  - 5.2 Work plan
    - 5.2.1 Work activities
    - 5.2.2 Schedule allocation
    - 5.2.3 Resource allocation
    - 5.2.4 Budget allocation
  - 5.3 Control plan
    - 5.3.1 Requirements control plan
    - 5.3.2 Schedule control plan
    - 5.3.3 Budget control plan
    - 5.3.4 Quality control plan
    - 5.3.5 Reporting plan
    - 5.3.6 Metrics collection plan
  - 5.4 Risk management plan
  - 5.4 Project close-out plan
- 6. Technical process plans
  - 6.1 Process model
  - 6.2 Methods, tools, and techniques
  - 6.3 Infrastructure plan
  - 6.4 Product acceptance plan
- 7. Supporting process plan
  - 7.1 Configuration management plan
  - 7.2 Testing plan
  - 7.3 Documentation plan
  - 7.4 Quality assurance plan
  - 7.5 Reviews and audits plan
  - 7.6 Problem resolution plan
  - 7.7 Subcontractor management plan
  - 7.8 Process improvement plan
- 8. Additional plans

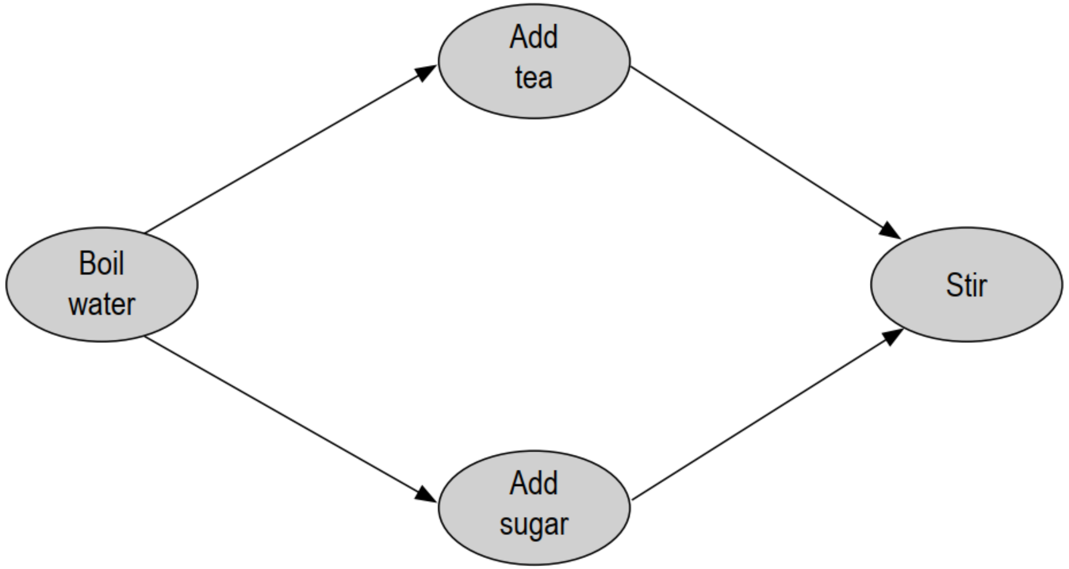
Team dynamics

- A team consists of at least two people
- The team members work towards a common goal
- Each member of the team is assigned a specific role
- Reaching the common goal needs some form of dependency among the team members

Important project management activities

- Defining a task network
  - Assuming that a “task” defines a unit of work in a project, a task network represents how tasks are dependent among one another.
  - It is a graph structure, where nodes (or vertices) denote tasks, and links (or edges) denote dependencies.
- Scheduling
  - The principal components of a project ecosystem are:
    - Tasks—what needs to be completed
    - Time—by when the tasks need to be completed
    - Resources—who will complete the tasks
  - For a project to succeed, the interaction of these components must be optimized.
    - This is the objective of scheduling.

Task network and corresponding timeline chart



Task	0 minutes		5 minutes		10 minutes
Boil water	▨	▨	▨	▨	
Add tea		▨			
Add sugar			▨		
Stir				▨	▨

Software Engineering: Concepts and Applications by Subhajit Datta (Oxford University Press, 2010)

Important project management activities (contd.)

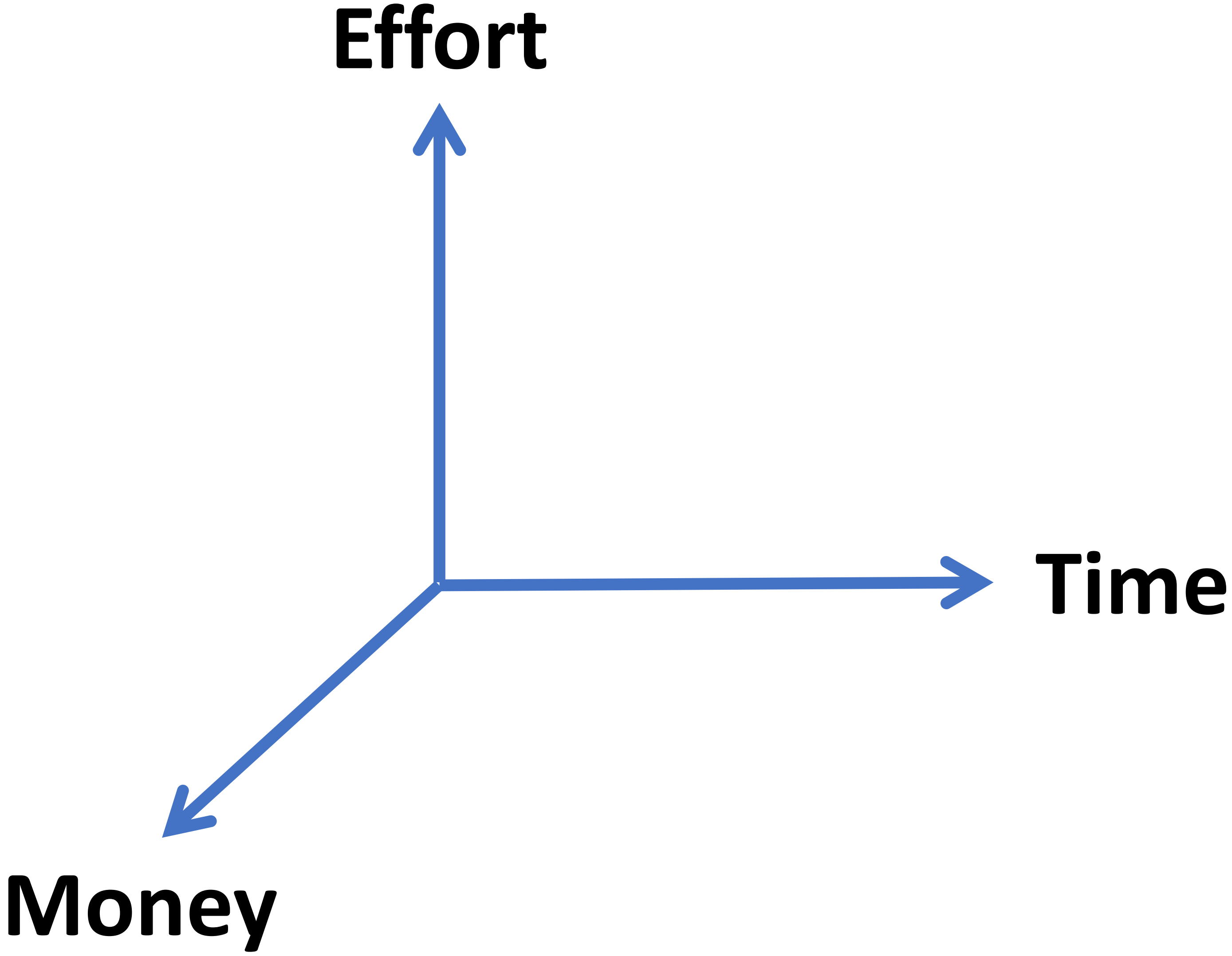
- Earned value analysis
  - A technique for measuring a software project's progress by providing a common value scale across the project's tasks
  - The quantitative insights available from earned value Analysis helps determine what percentage of the project has been completed
  - Essential for initiating feedback and taking corrective action
- Error tracking
  - The process of identifying and resolving errors in the project's execution path
  - Errors that are not identified during error tracking can later manifest as defects.
  - Defect removal efficiency—an important metric for the gauging the effectiveness of quality assurance processes—is closely related to error tracking

Managing versus leading

- “Management uses resources to accomplish results; leadership motivates people to achieve objectives” – Watts S. Humphrey
- Leadership is something every manager aspires for, and few achieve
- Motivation is vital if one expects a group of diverse individuals to be aligned to a common cause
- Management at its most sublime may meld into leadership
- Effective management of software projects can be brought about by systematic use of principles discussed in this lecture

Software project management: A practitioner’s view

Or some tips for the managing your course project ☺



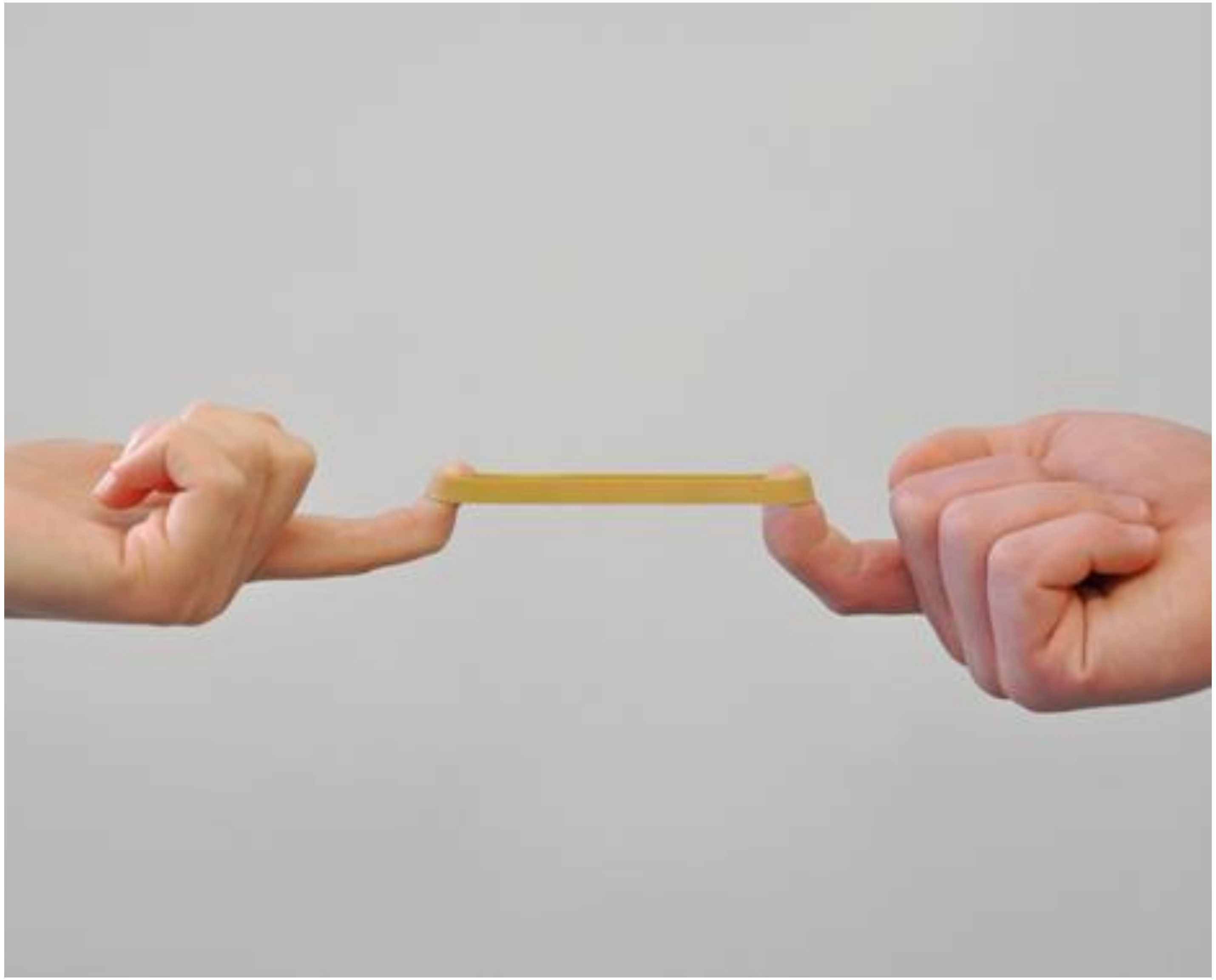
Effort

Time

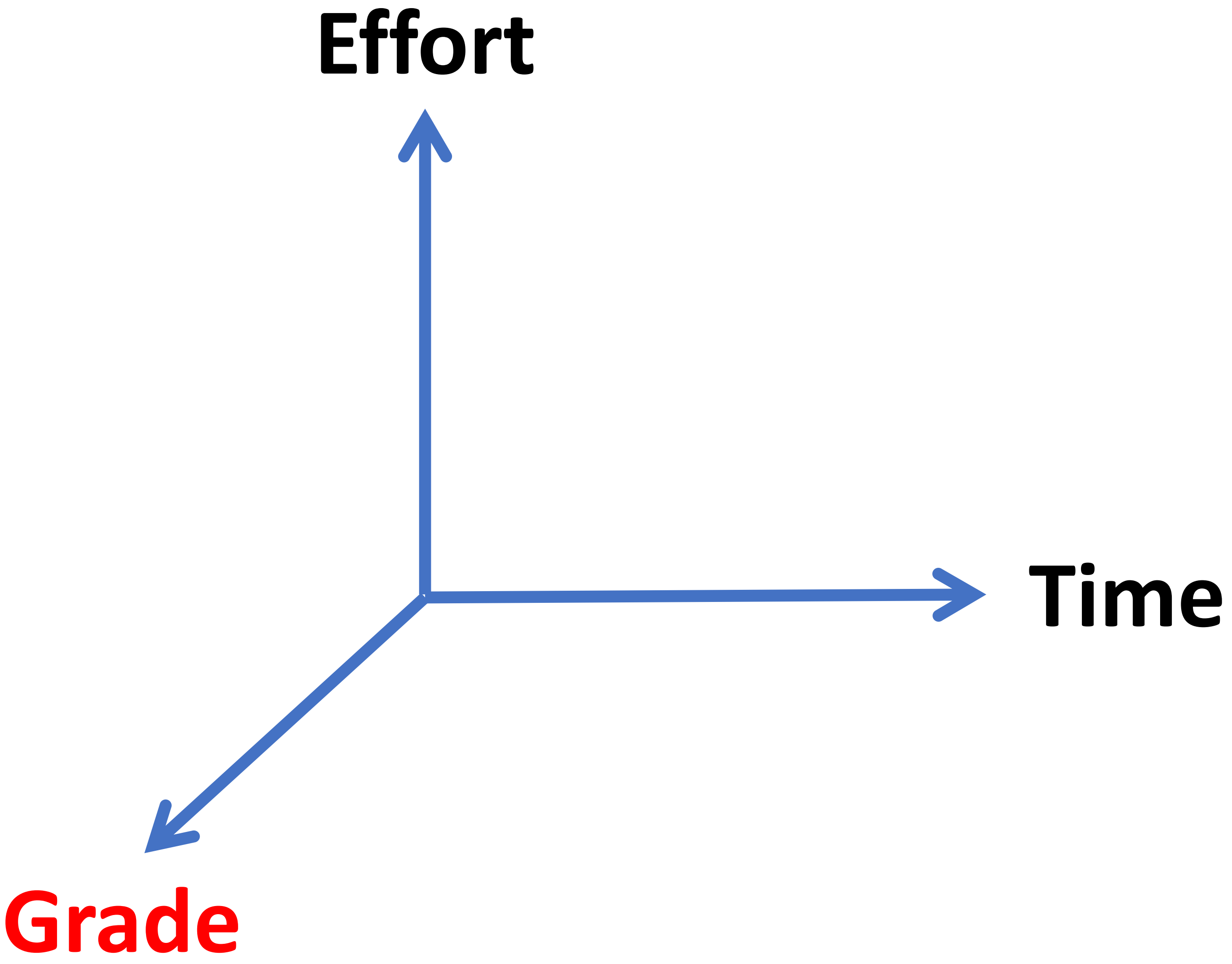
Money

http://thec0keman.blogspot.com/2008/12/three-sides-of-balance.html

Balancing the pulls of time, effort, and money …



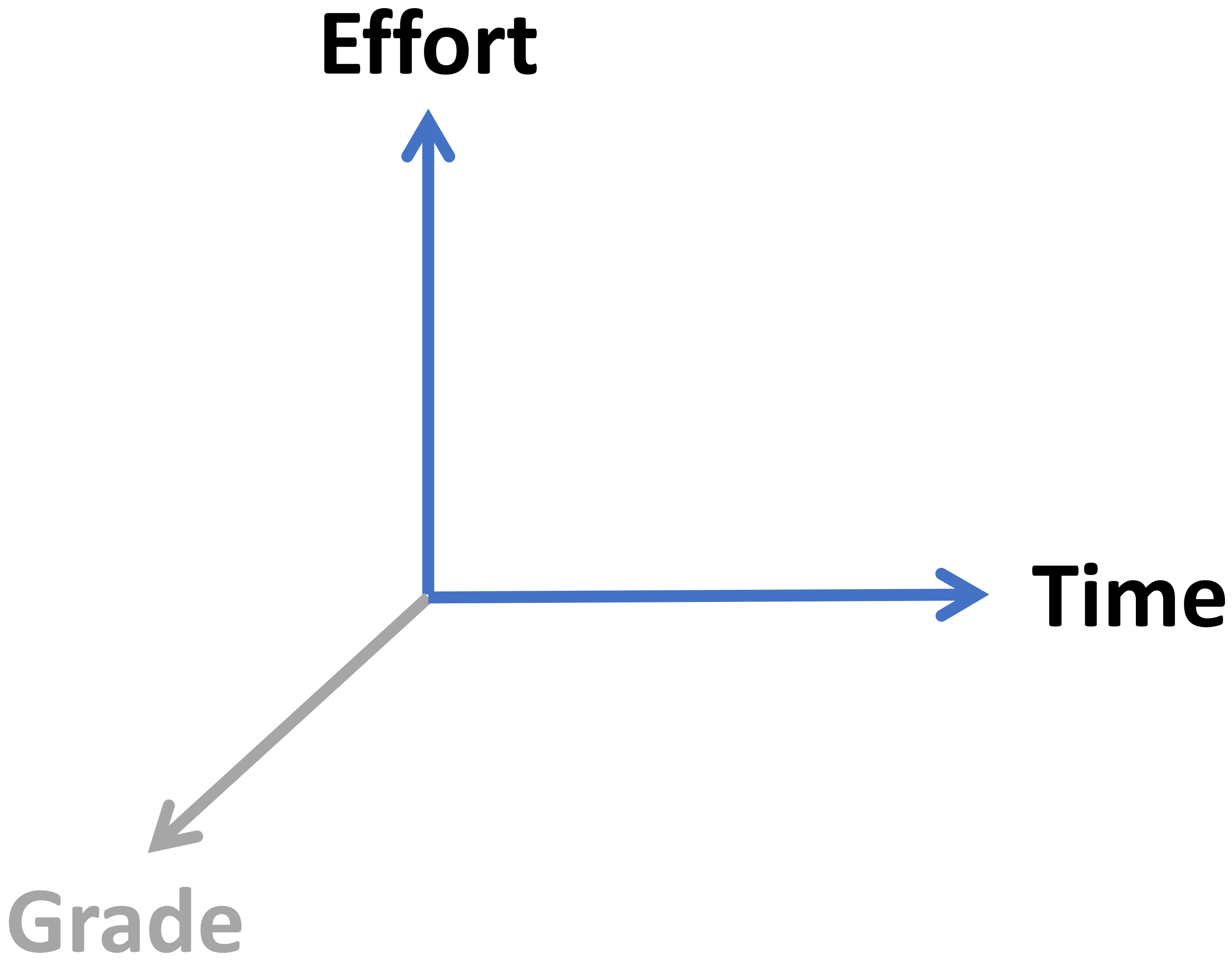
https://www.dezeen.com/2011/02/15/rubber-by-taf-arkitektkontor-for-zero/

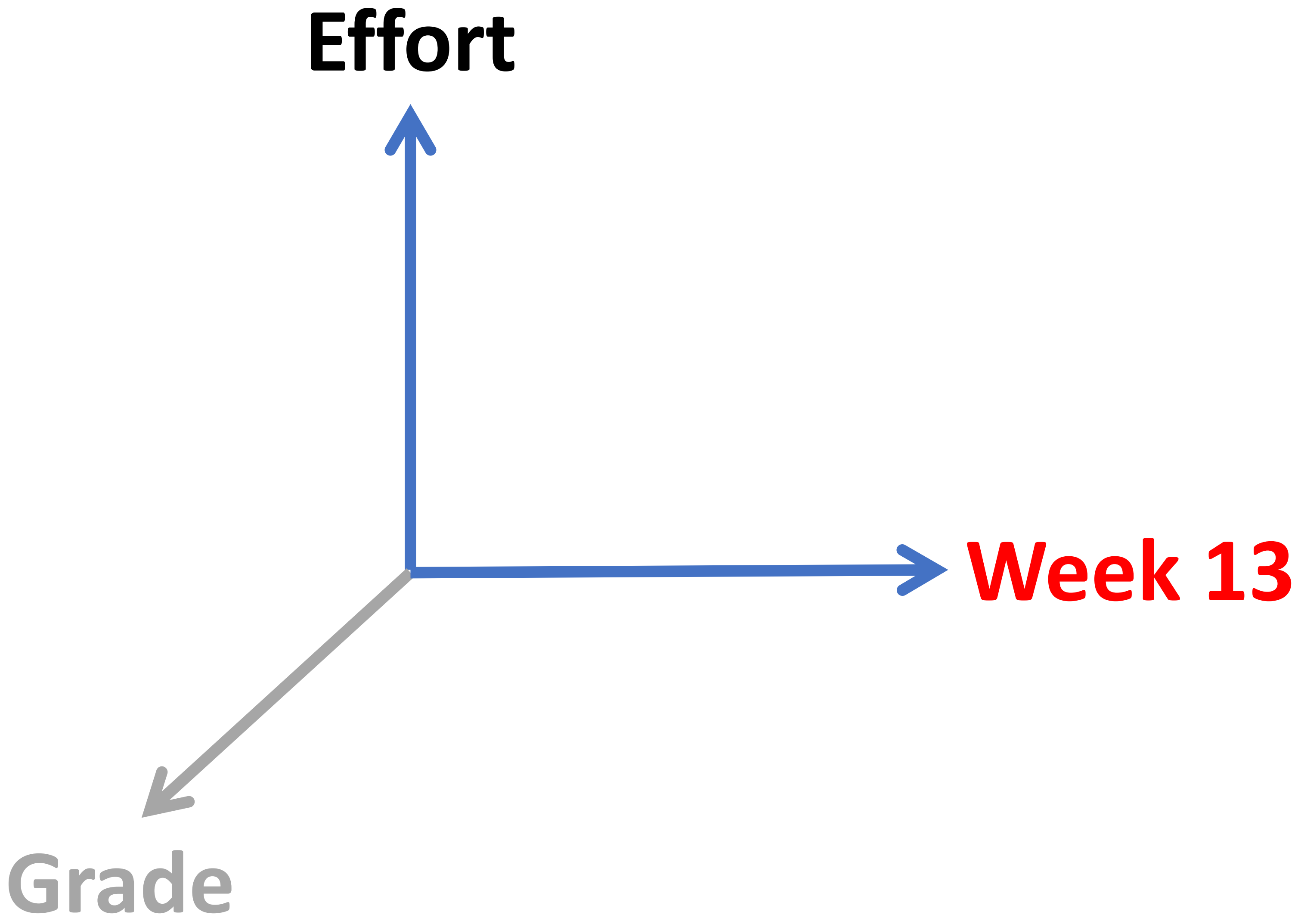


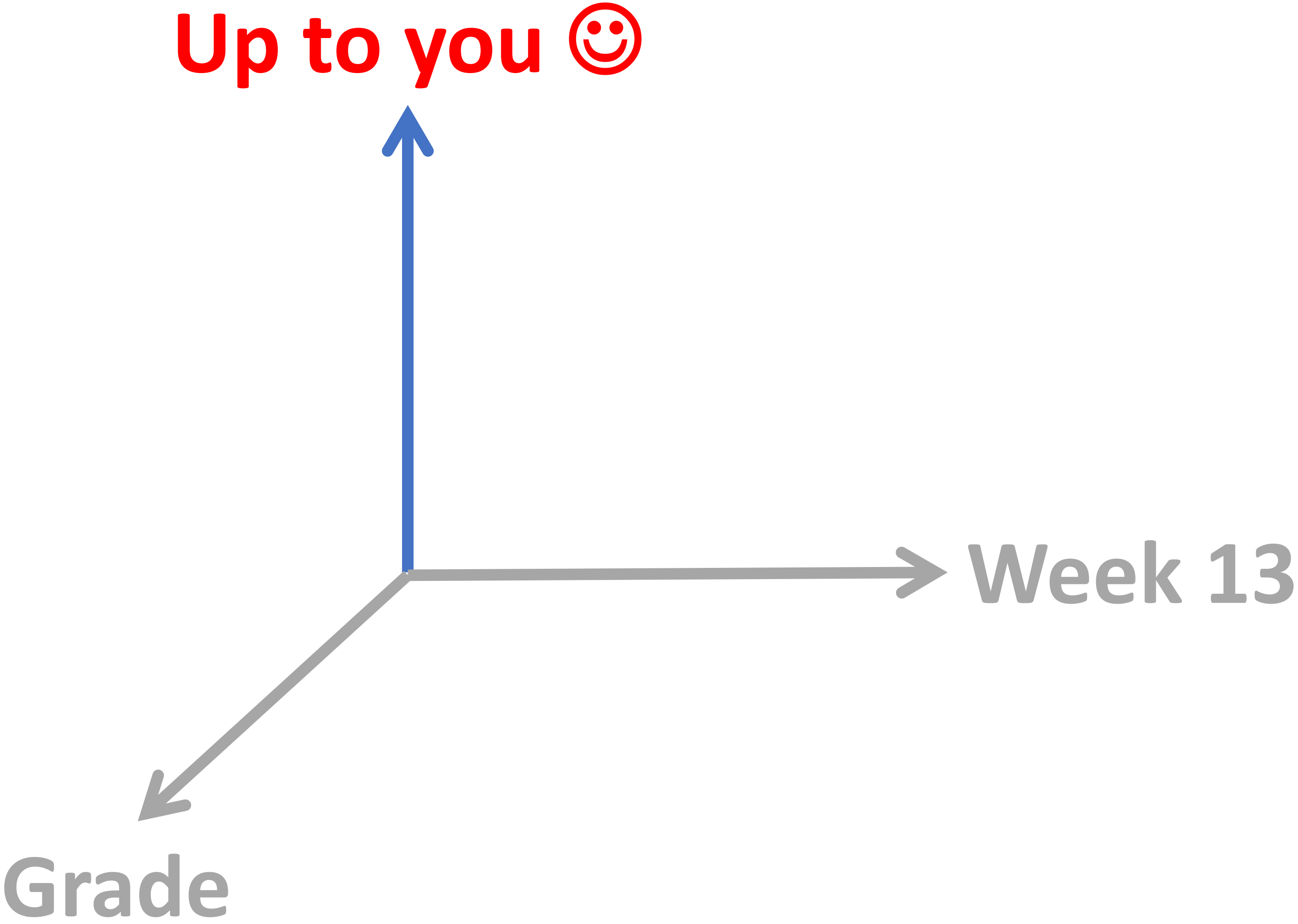
Effort

Time

Grade


Effort
Time
Grade

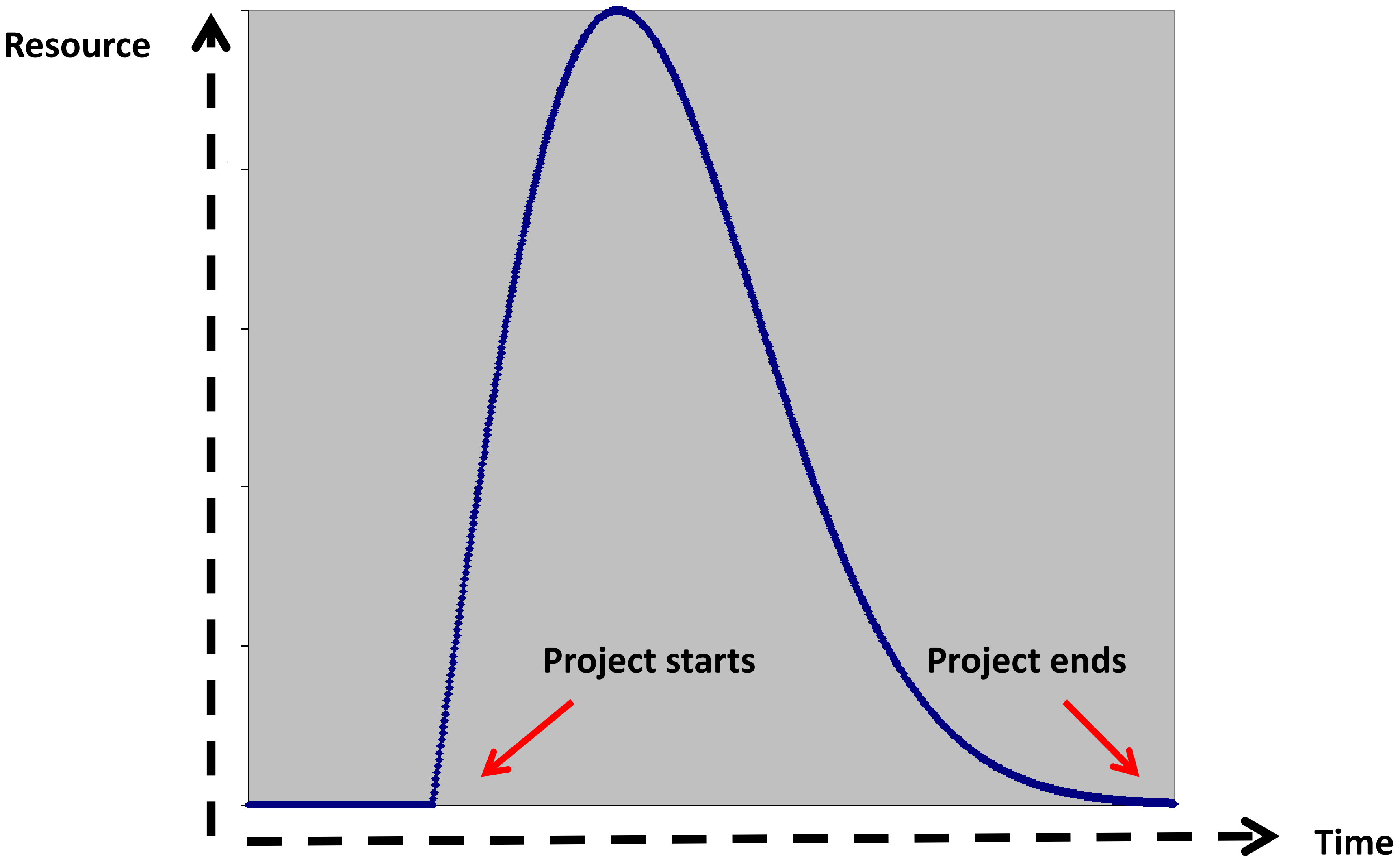

Effort
Week 13
Grade

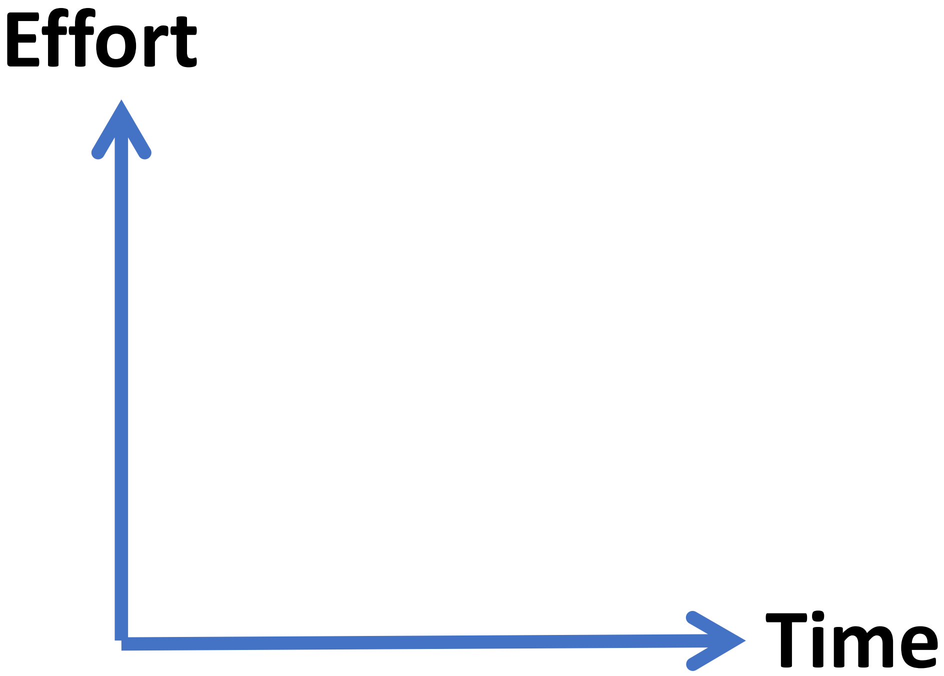


Up to you ☺

Week 13

Grade


Resource
Project starts
Project ends
Time


Effort
Time

$$\text{Effort} = \left[\frac{\text{Size}}{\text{Productivity} \cdot \text{Time}^{4/3}}\right]^3 \cdot B$$

https://en.wikipedia.org/wiki/Putnam_model

$$\text{Effort} = \left[\frac{\text{Size}}{\text{Productivity} \cdot \text{Time}^{4/3}}\right]^3 \cdot B$$

Depends on you and your team

$$\text{Effort} = \left[\frac{\text{Size}}{\text{Productivity} \cdot \text{Time}^{4/3}}\right]^3 \cdot B$$

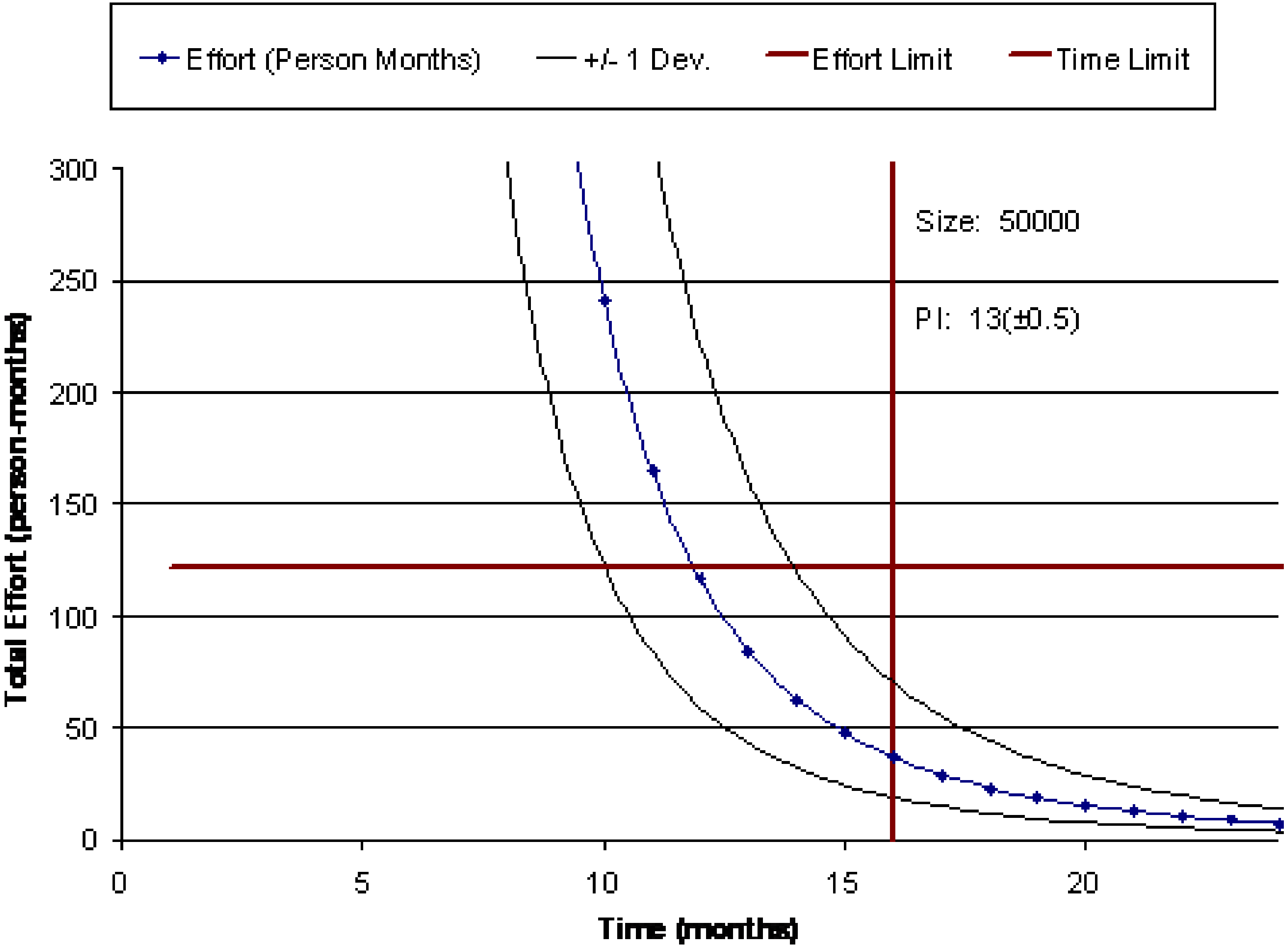
Depends on the scale of your project

https://en.wikipedia.org/wiki/Putnam_model

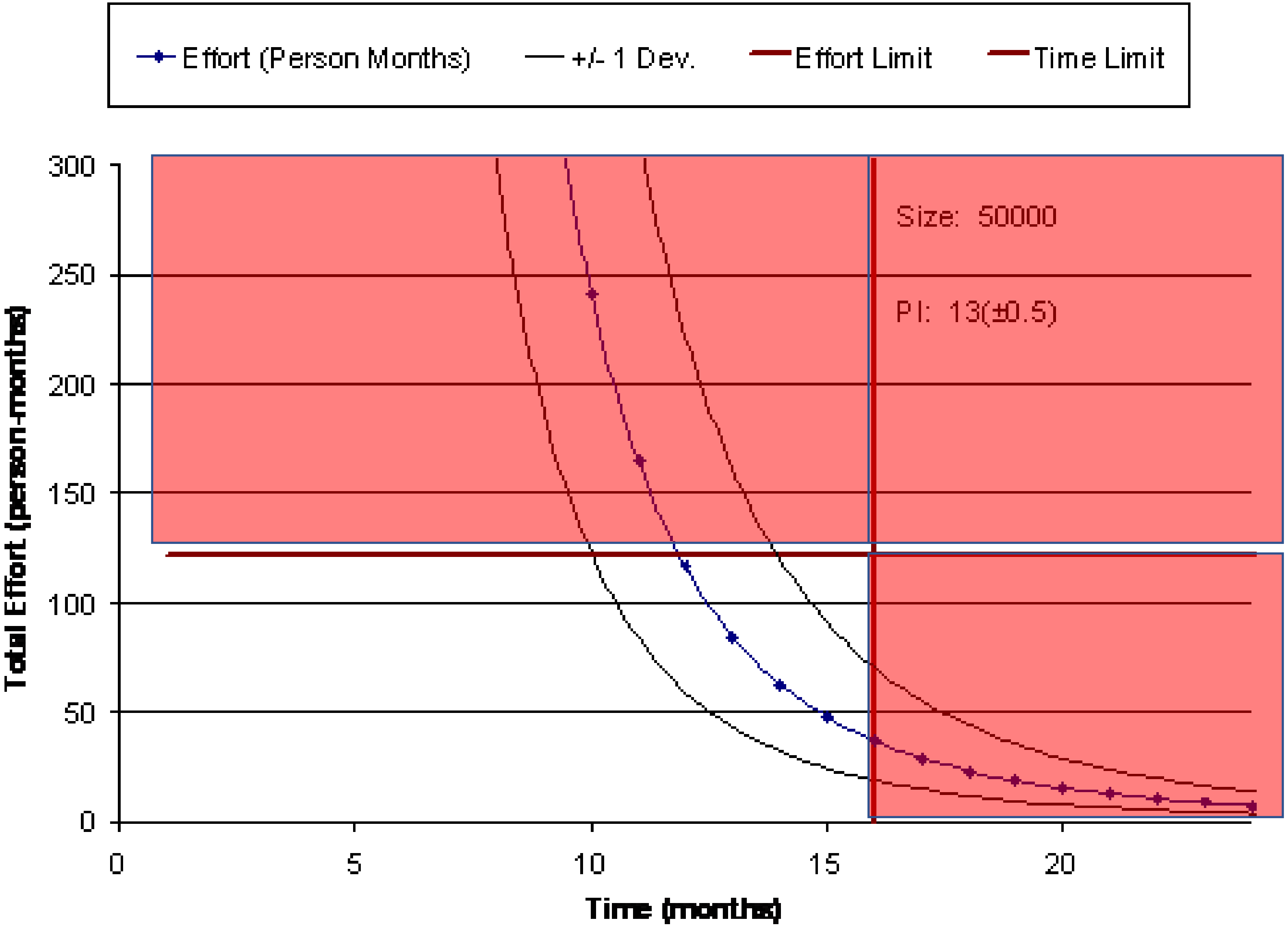
$$\text{Effort} = \left[\frac{\text{Size}}{\text{Productivity} \cdot \text{Time}^{4/3}}\right]^3 \cdot B$$

Inverse relationship

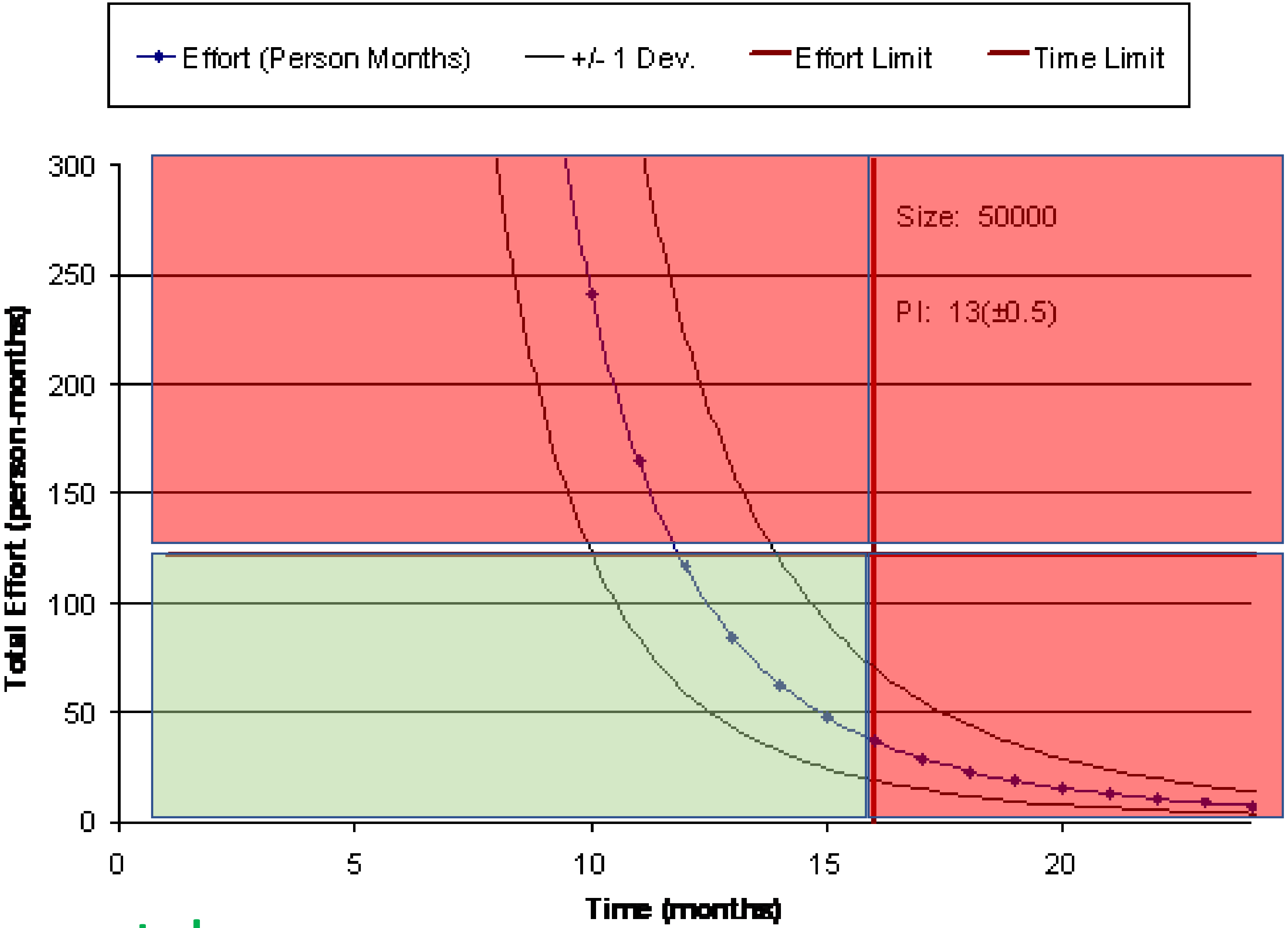
https://en.wikipedia.org/wiki/Putnam_model



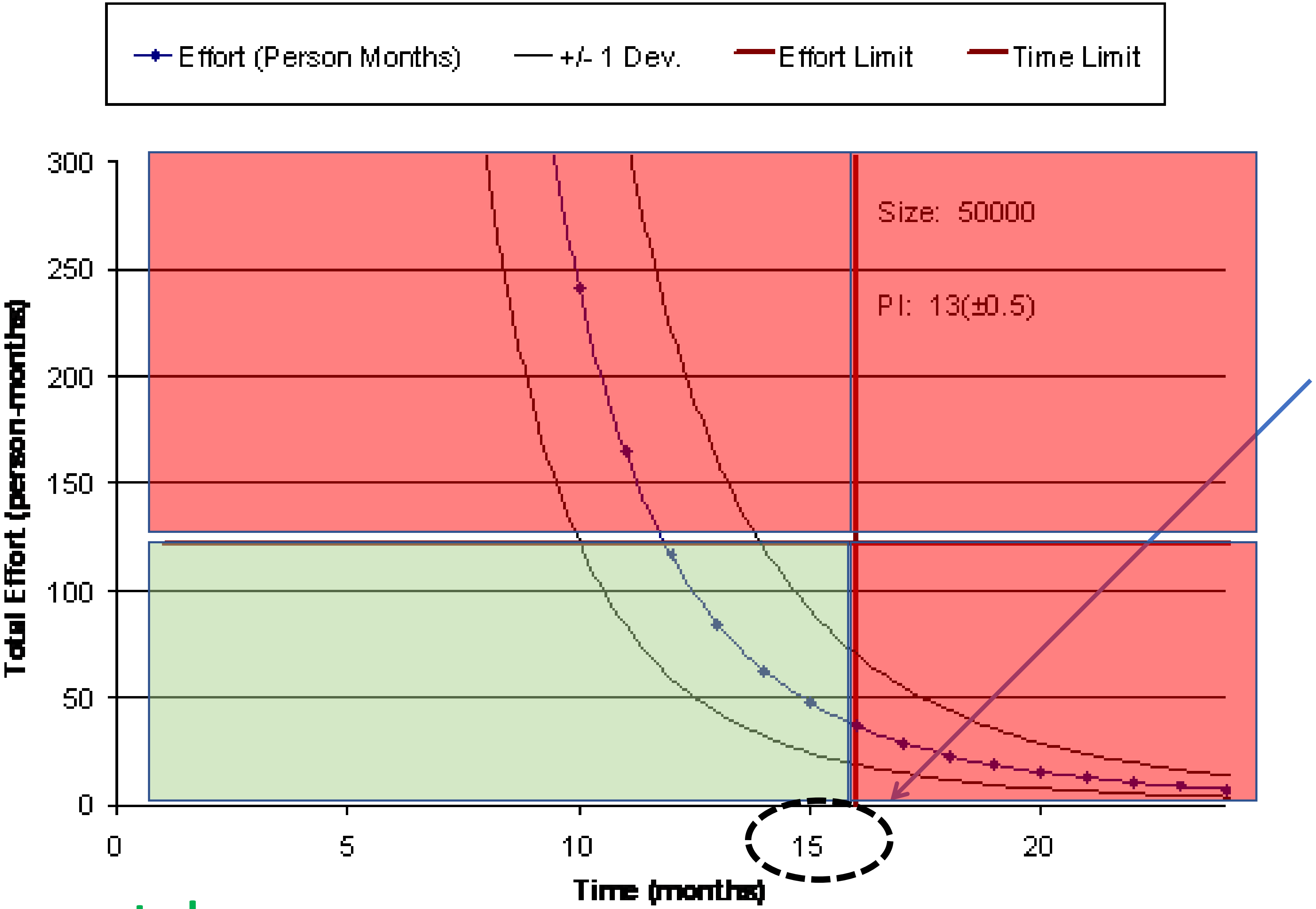
https://en.wikipedia.org/wiki/Putnam_model



Impossible zones

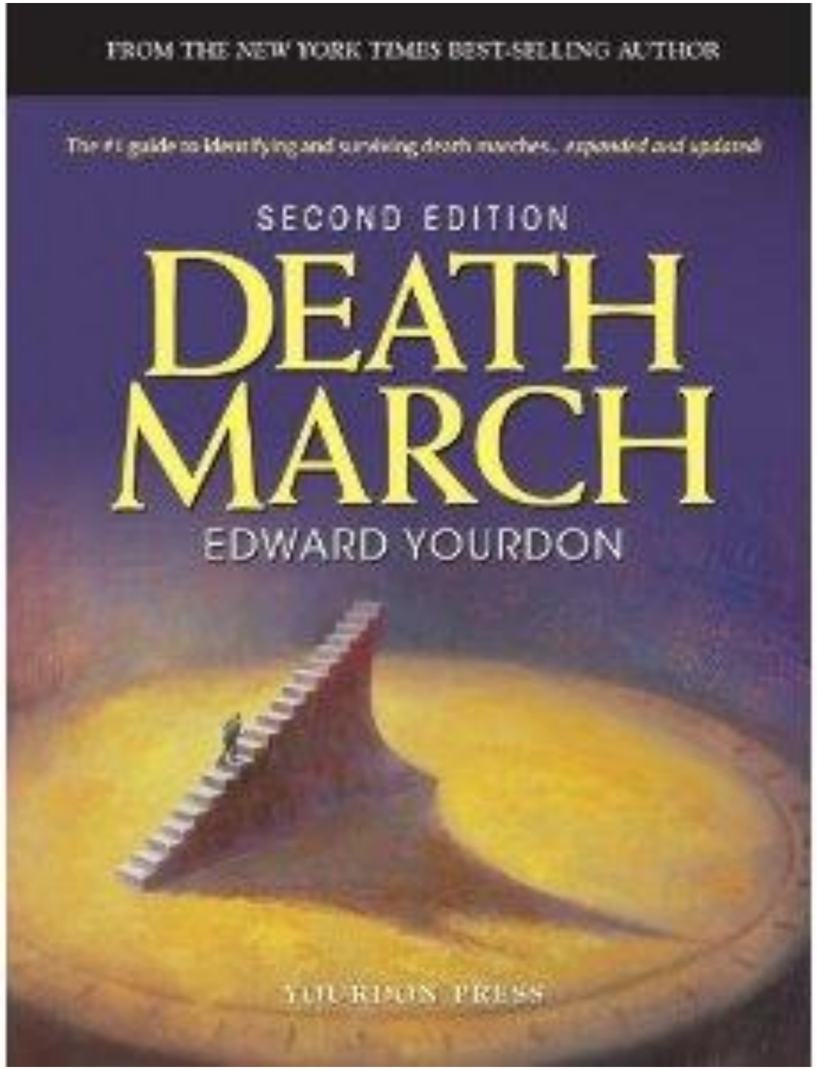


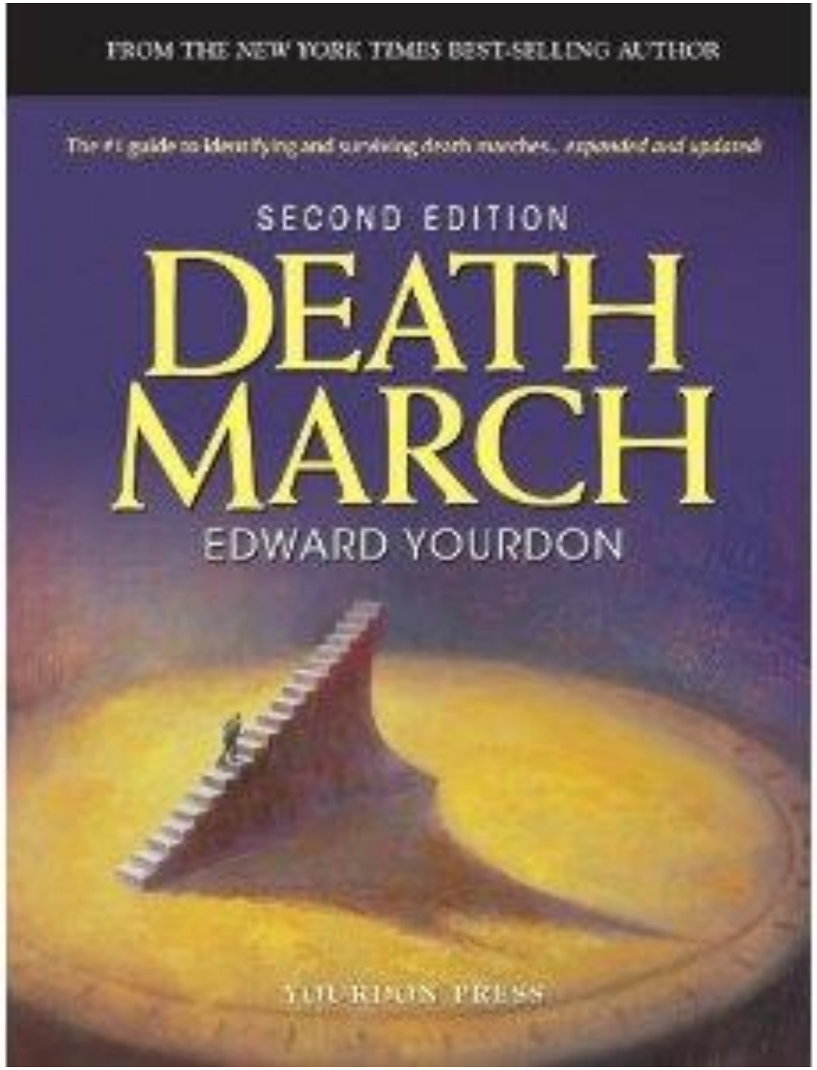
You can only operate here



In case you have not noticed, this is just an example. You do not have 15 months, just 13 weeks!

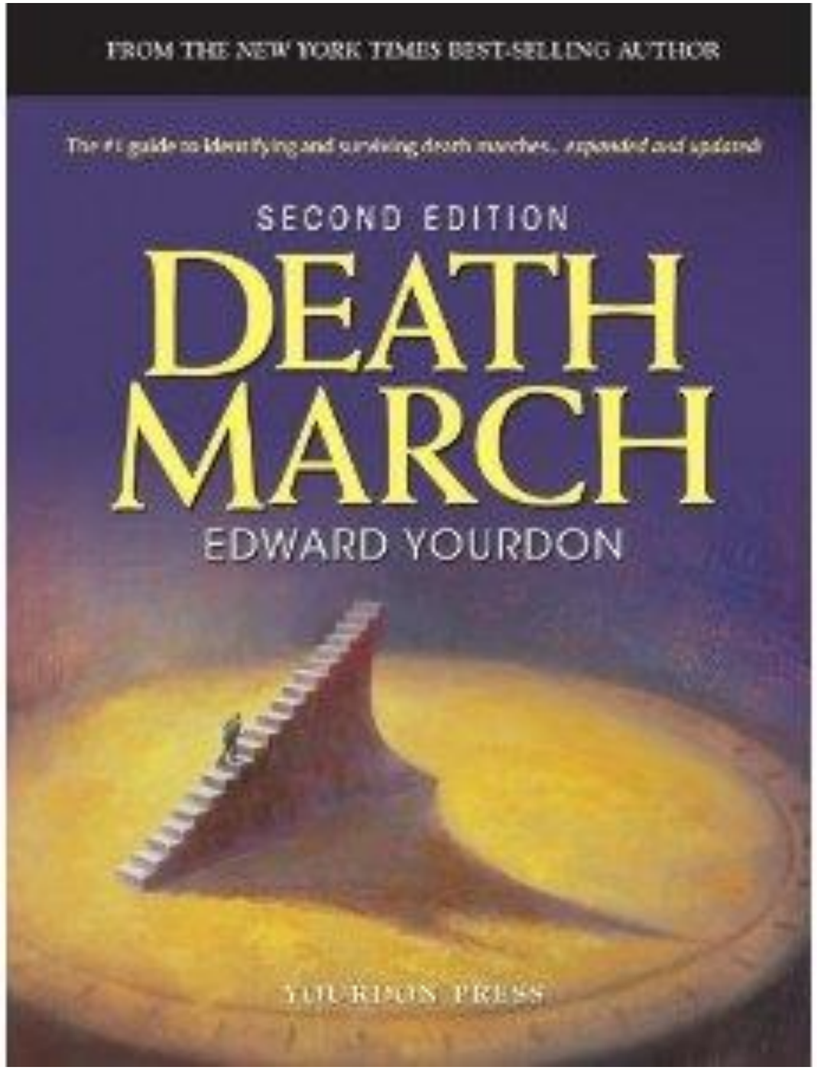
You can only operate here


FROM THE NEW YORK TIMES BEST-SELLING AUTHOR
The #1 guide to identifying and surviving death marches... expanded and updated
SECOND EDITION
DEATH MARCH
EDWARD YOURDON
YOURDON PRESS


FROM THE NEW YORK TIMES BEST-SELLING AUTHOR
The #1 guide to identifying and surviving death marches, expanded and updated
SECOND EDITION
DEATH MARCH
EDWARD YOURDON
YOURDON PRESS

“In project management, a death march is a project where the members feel it is destined to fail and/or requires a stretch of unsustainable overwork.”

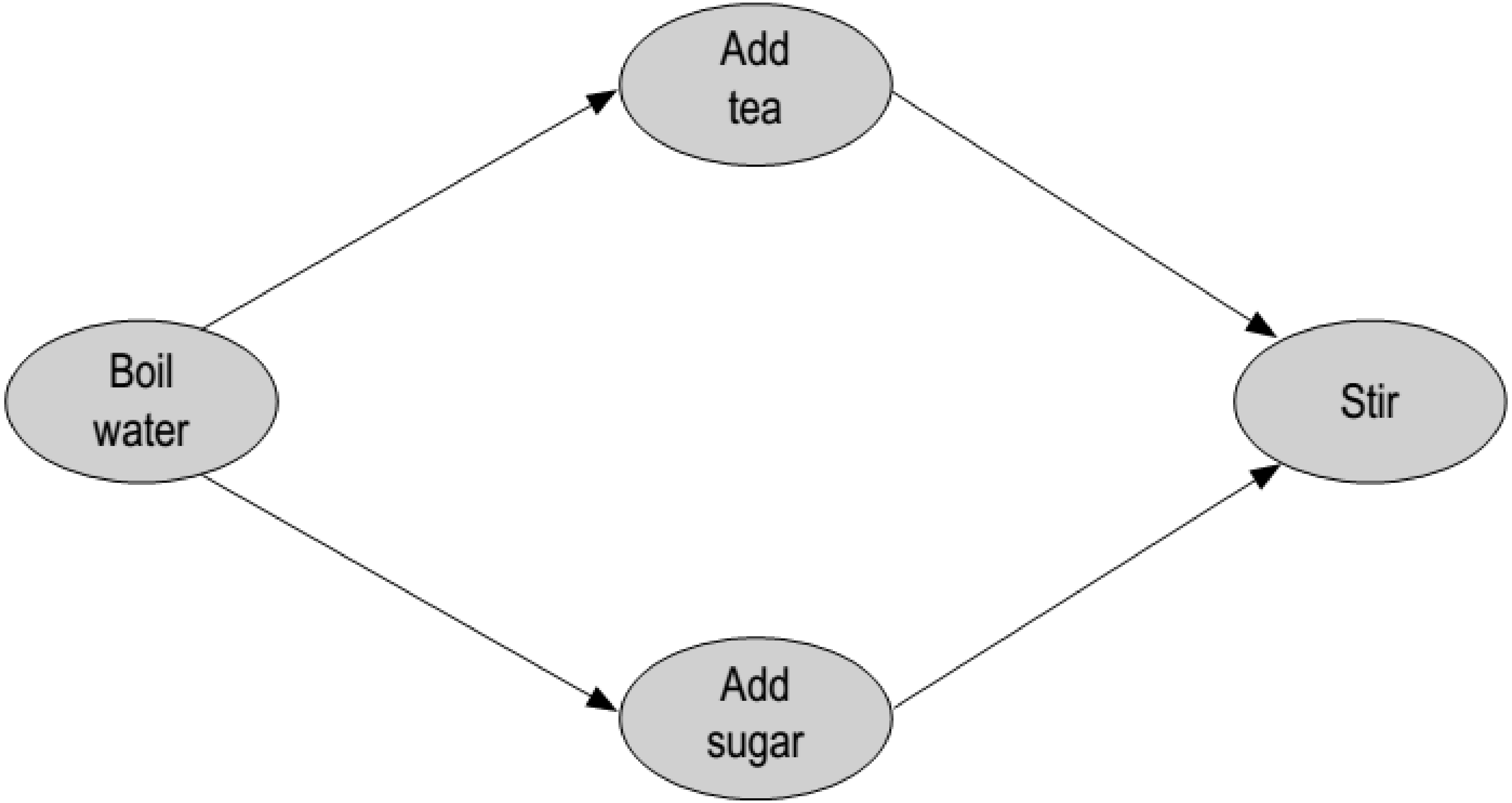
https://en.wikipedia.org/wiki/Death_march_%28project_management%29


FROM THE NEW YORK TIMES BEST-SELLING AUTHOR
The #1 guide to identifying and surviving death marches, expanded and updated
SECOND EDITION
DEATH MARCH
EDWARD YOURDON
YOURDON PRESS

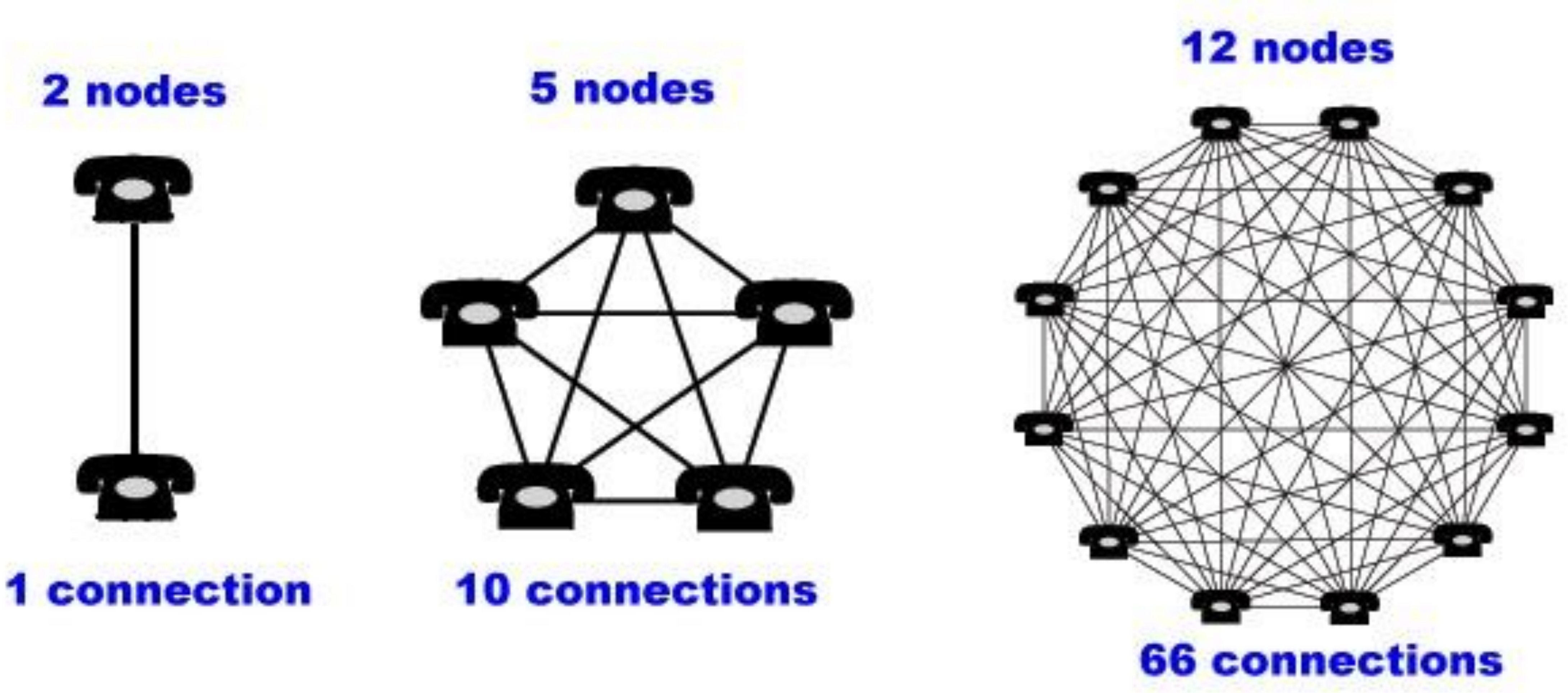
"In project management, a death march is a project where the members feel it is destined to fail and/or requires a stretch of unsustainable overwork."

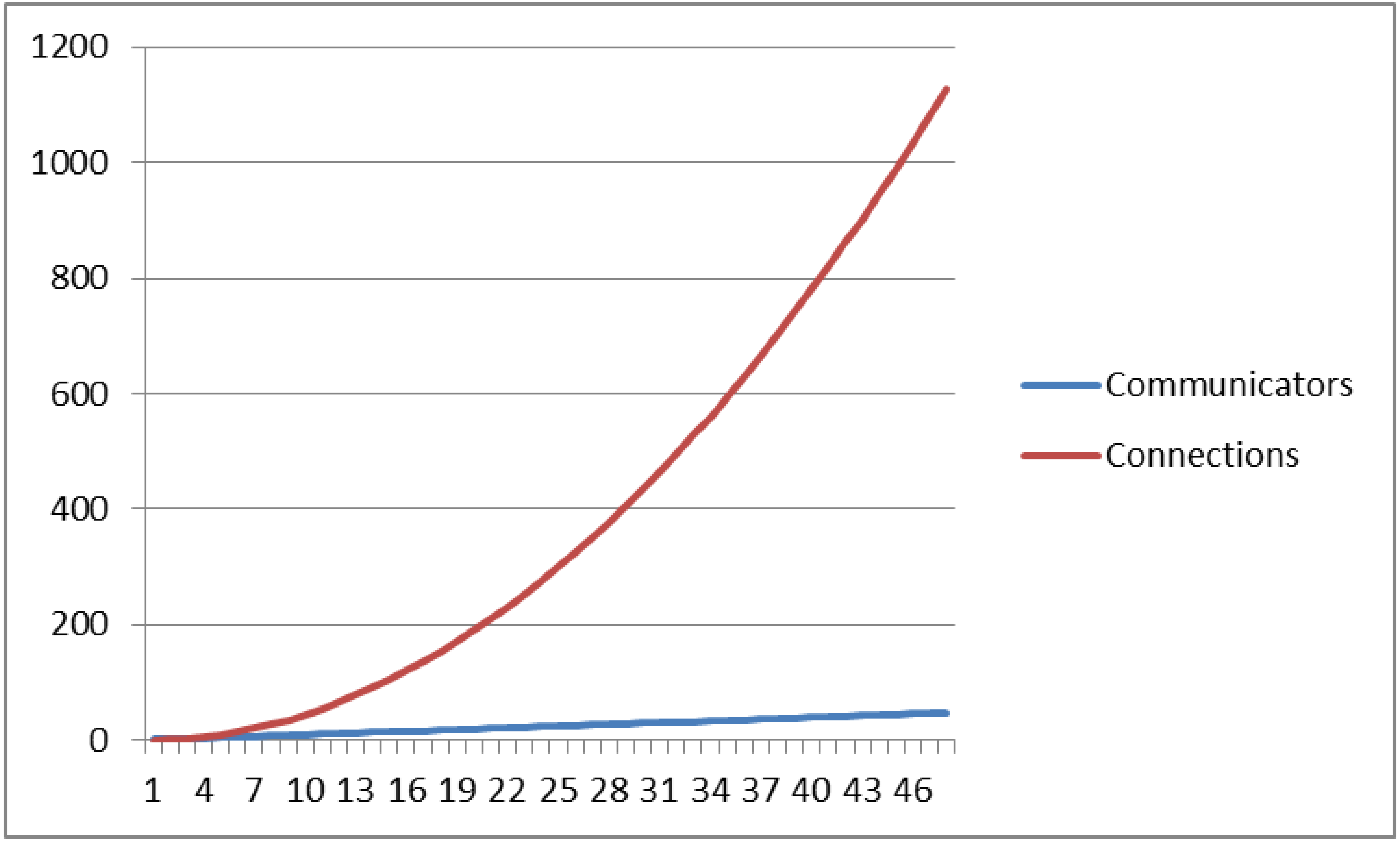
https://en.wikipedia.org/wiki/Death_march_%28project_management%29

You can pour more effort into less time, but there are limits.


Add
tea
Boil
water
Stir
Add
sugar

Understand the sequence of activities, and plan.


2 nodes
1 connection
5 nodes
10 connections
12 nodes
66 connections


1200
1000
800
600
400
200
0
1
4
7
10
13
16
19
22
25
28
31
34
37
40
43
46
Communicators
Connections

Talk is cheap, but communication has a cost, connect wisely.

But make sure each team member knows what is going on.

http://debsdustbunny.blogspot.com/2013/04/how-to-make-perfect-cup-of-tea.html