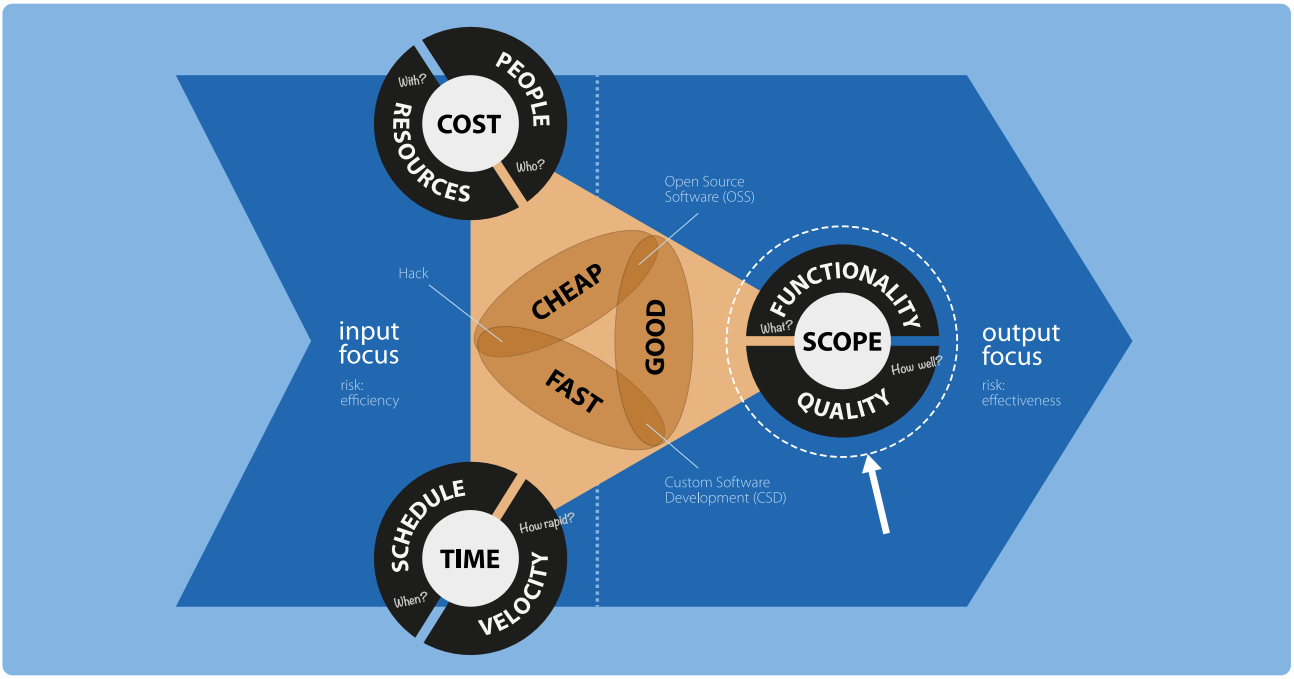


Software Engineering in Industrial Practice (SEIP)

Dr. Ralf S. Engelschall

Project Management Triangle



Definition of a *Project*:
"Temporary endeavor undertaken to create a unique product, service or result."
Temporary in that it has a defined beginning and end in time, and a defined scope and cost.
Unique in that it is not a routine operation, but a one-time, single-goal, and risk-containing operation.

Project Management *Iron Triangle*:
A project is constrained by **time**, **cost** and **scope**. No constraint in this triangle can be changed without affecting the others. Time splits into **schedule** and **velocity**. Cost splits into **people** and **resources**. Scope splits into **functionality** and result **quality**.

Project Management *Trilemma*:
"Fast. Cheap. Good. Pick two!"
Each project optimization effort has the choice among **three** favourable options — only **two** of them are possible at the same time.

Project Management, alongside **Software Architecture**, is the second important Discipline in the field of **Software Engineering**. Therefore everyone should have at least a basic understanding of the essential task of Project Management: continuously finding the balance from the "Iron Triangle" of **Time**, **Cost** and **Scope**.

The adjusting screw **Time** is divided into the two aspects **Schedule** (When?) and **Velocity** (How rapid?). The adjusting screw **Cost** is divided into the two aspects **People** (Who?) and **Resources** (With?). The adjusting screw **Scope** is divided into the two aspects **Functionality** (What?) and **Quality** (How well?).

If a change is made to one of the three adjusting screws or one of the six aspects, the "Iron Triangle" will be unbalanced, and one must inevitably change one or more of the other screws or aspects to restore the balance.

Also worthy of mention is the **Trilemma**, which says that one can usually have only two out of three things at a time: either cheap and good (Open Source Software), but not fast; or good and fast (Custom Software Development), but then not cheap; or fast and cheap (the "Quick Hack"), but then not good.

In practice, the non-Project-Managers are co-responsible, especially in the area **Scope**, since here a change in the project usually requires a deeper technical understanding of the Application.

Questions

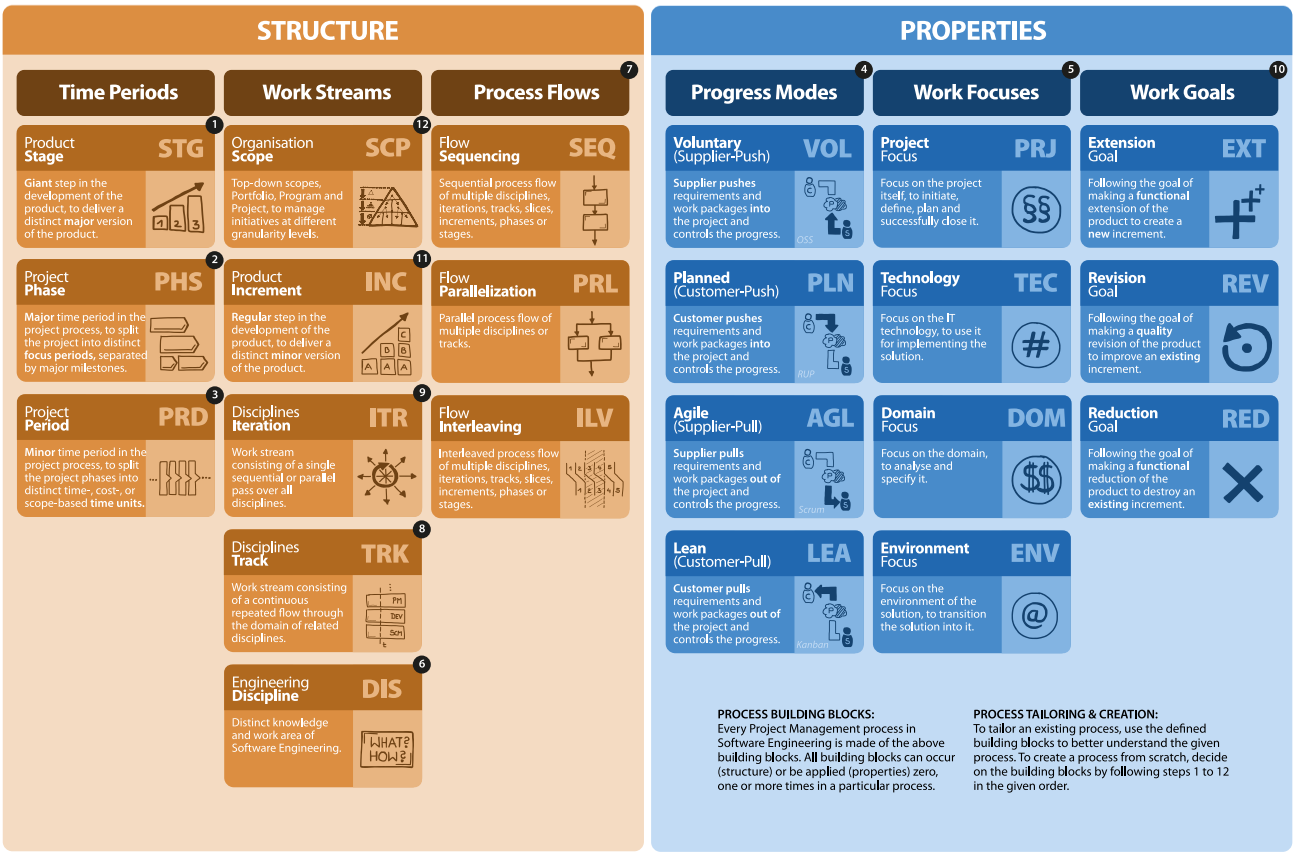
- At which adjusting screw of **Project Management** in practice are the non-Project-Managers co-responsible?

Intellectual Content: Version 1.1.3 (2021-05-26), Authored 2019-2021 by Dr. Ralf D. Engel. Based on project management body of knowledge of PMI and PMA. Graphics: Illustration: Version 1.1.8 (2021-05-11). Copyright © 2021 Dr. Ralf D. Engel. http://engel.nu/cc/by-nc-sa/4.0/. All rights Reserved. Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science for Future contexts only.

ENGINEERING FUNDAMENTALS

Project Management Building Blocks

TECHNISCHE UNIVERSITÄT MÜNCHEN



Every Project Management process in Software Engineering is made out of the same set of building blocks. All building blocks can occur (structure) or be applied (properties) zero, one or more times in a particular process.

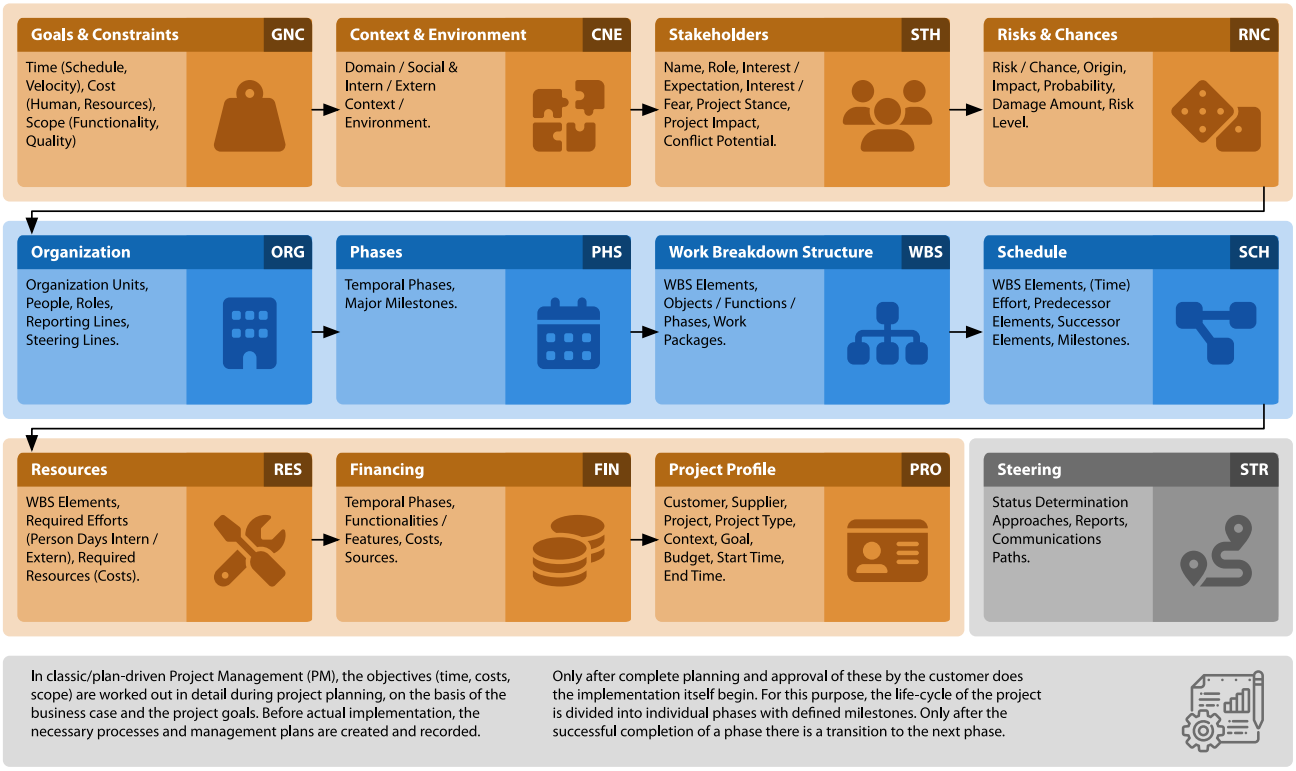
To tailor an existing process, use the defined building blocks to understand the given process better. To create a process from scratch, decide on the building blocks by following steps 1 to 12 in the given order.

Questions

? Is a special Project Management Process in Software Engineering crucial?

Intellectual Content: Version 1.0.8 (2022-08-11), Authored 2016-2022 by Dr. Ralf S. Engelschall
Graphical Illustration: Version 1.0.9 (2022-09-11), Copyright © 2016-2022 Dr. Ralf S. Engelschall <http://engelschall.com>, All Rights Reserved.
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contents only.

Plan-Driven Project Management



In classic/plan-driven Project Management (PM), the objectives (time, costs, scope) are worked out in detail during project planning, on the basis of the business case and the project goals. Before actual implementation, the necessary processes and management plans are created and recorded.

Only after complete planning and approval of these by the customer does the implementation itself begin. For this purpose, the life-cycle of the project is divided into individual phases with defined milestones. Only after the successful completion of a phase there is a transition to the next phase.

Intellectual Content: Version 1.0.0 (2021-12-20), Authored 2022 by Dr. Ralf S. Engelschall based on concepts from Steve McConnell
Graphical Illustration: Version 1.0.0 (2022-04-10), Copyright © 2022 Dr. Ralf S. Engelschall <http://engelschall.com> - All Rights Reserved
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

In classic, plan-driven Project Management (PM), the objectives (time, costs, scope) are worked out in detail during project planning, on the basis of the business case and the project goals. Before actual implementation, the necessary processes and management plans are created and recorded.

Only after complete planning and approval of these by the customer does the implementation itself begin. For this purpose, the life-cycle of the project is divided into individual phases with defined milestones. Only after the successful completion of a phase is there a transition to the next phase.

Questions

- What is the major problem of plan-driven Project Management?

ENGINEERING FUNDAMENTALS

Agile Project Management

TECHNISCHE UNIVERSITÄT MÜNCHEN

EF 04.4

Agile Manifesto & Scrum Values

"We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and Interactions	over	Processes and Tools
Working Software	over	Comprehensive Documentation
Customer Collaboration	over	Contract Negotiation
Responding to Change	over	Following a Plan

That is, while there is value in the items on the right, we value the items on the left more."

— *Manifesto for Agile Software Development*
http://agilemanifesto.org

Scrum Theory:
Empiricism
Lean Thinking

Scrum Values:
Courage
Focus
Commitment
Respect
Openness

Essence of Agility:
Driving On Sight
Deliver Regularly

Sprint

The heartbeat of a Scrum process, where ideas are turned into value. They are fixed length events of 2–4 weeks. A new Sprint starts immediately after the conclusion of the previous Sprint.

Scrum Roles

Product Owner

Captures the needs of customers and stakeholders.

Determines delivery date and delivery content.

Is responsible for product and project success on ROI and TCO basis.

Defines and priorizes the Product Backlog, depending on business value.

Accepts or rejects deliverables.

Developers

Typically a maximum of 8 people.

Is cross-functional, by coping, as a team, all necessary Software Engineering disciplines.

Members are usually assigned full-time, and membership can only change between Sprints.

Scrum Master

Is responsible for adherence of the process to the Scrum Theory and the Scrum Values, and acts as a coach and mediator.

Helps to remove obstacles, and protects Developers from external disturbances.

Is responsible to drive the learning process and self-management of Developers.

Has no authority to give disciplinary instructions to the Developers.

Should *not* be a Developer in parallel.

Scrum Events

Sprint Planning

First meeting (max. 8h) in the Sprint. Developers, with Product Owner, select the highest priority features of Product Backlog. Developers break down features to tasks and estimate efforts. Developers and Product Owner agree on Sprint Goal. Developers also make commitment to Sprint Goal.

Daily Scrum (Stand-Up)

Short (0,25h) daily meeting during the Sprint, always in same place and at same time. Everyone answers 4 questions: *What have I achieved since the last Daily? What would I like to achieve by the next Daily? What are the obstacles for me to do this? Are there any obstacles to reaching our Sprint Goal?* Discussions only afterwards.

Sprint Review

Second to last meeting (max. 4h) of the Sprint, before the Spring Retrospective. The Developers present the Increment they have achieved during the Sprint. Stakeholders give feedback. Only real artifacts or "live demonstrations" are allowed. The whole world is invited to participate!

Sprint Retrospective

Last meeting (max. 3h) in the Sprint. The "Process, Skill and Individual" dimensions are reviewed as well as "Lessons Learned" are discussed. Improvements are addressed immediately. Supports the mantra of Continuous Improvement (KAIZEN).

Scrum Artifacts & Commitments

Product Backlog

Emergent, ordered list of what is needed to improve the product, formulated as User Stories and managed by the Product Owner. User Story granularity is from small to large.

Sprint Backlog

Is managed by the Developers. Contains the User Stories that are planned for implementation in the current Sprint. These are split into tasks by the Developers.

Increment

Usable and thoroughly verified state of the product toward the Product Goal. Additive to all prior Increments.

Product Goal

Long-term goal for the product, formulated as a future state of the product, and provided by the Product Owner. Only one Product Goal is worked towards at any time.

Sprint Goal

Short-term goal for the current Sprint. What is the priority to be achieved in the Sprint? Why should we start the Sprint (Motivation)?

Definition of Done (DoD)

Formal description of the state of the Increment, when it meets the quality measures required for the product. When a Product Backlog item meets the Definition of Done, an Increment is born.

Intellectual content: Version 1.0.0 (2021-12-20). Authored 2022 by Dr. Ralf S. Engelschall based on concepts from Steve McConnell.
Graphical illustration: Version 1.0.0 (2022-04-10). Copyright © 2022 Dr. Ralf S. Engelschall <http://engelschall.com>. All rights reserved.
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

flexibility & change

Agility is a mindset for developing software where one values (according to the "Manifesto for Agile Software Development"): Individuals and Interactions over Processes and Tools, Working Software over Comprehensive Documentation, Customer Collaboration over Contract Negotiation, and Responding to Change over Following a Plan.

Scrum is a simple, lightweight, methodological framework that (according to the "Scrum Guide") "helps people, teams and organizations to generate value through adaptive solutions for complex problems", and which follows and supports the mindset of Agility.

Scrum requires an environment where: a Product Owner orders the work for a complex problem into a Product Backlog; a Developer Team turns a selection of the work into a usable Increment of value during a Sprint; the Product Owner, the Developer Team, and its stakeholders inspect the results and adjust the process for the next Sprint; finally this process just repeats.

Questions

- What is the main problem that Agile software development addresses?