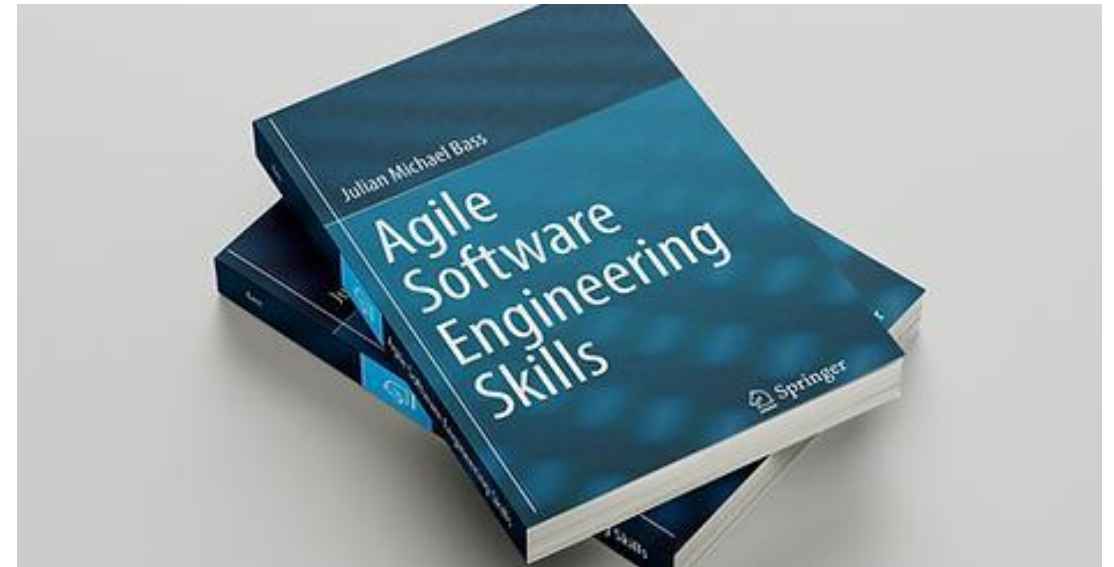


Agile Software Engineering Skills

Self-Organising Teams
Chapter 2
Julian M. Bass



Introduction

- Teams are groups of people that work together on shared tasks
- Teams are small
- Large projects use many teams

Contents

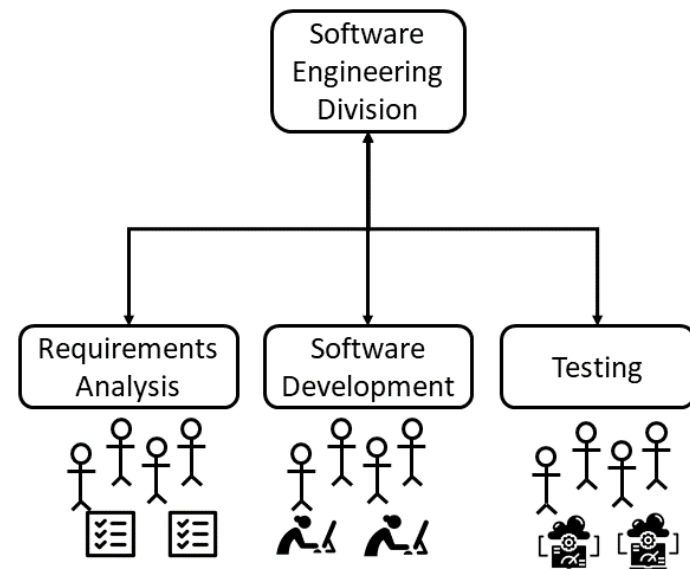
- Self-Organising Teams
- Groups and Teams
- Team Performance
- Agile Principles
- Collaboration Activities in Agile Teams
- Virtual Teams
- Communities of Practice

Self-Organising Teams

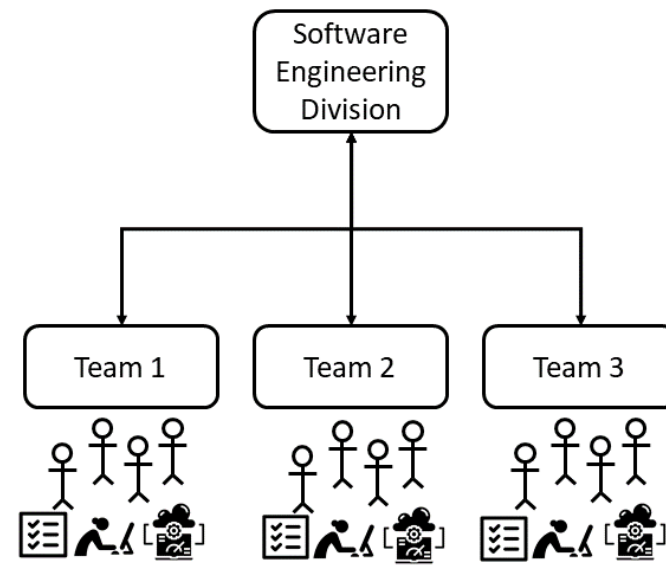
- Customers and clients set goals about ***what*** must be accomplished
- Team members collaborate to decide ***how*** to accomplish goals
- Functional teams are organised around a job function
- Multi-disciplinary teams pull together range of skills needed to fulfil goals

Team Composition

A) Functional Teams



B) Multi-disciplinary Teams



Self-Organising Teams

- Attributes include
 - Autonomy, to manage and assume responsibility
 - Requisite variety, employing a range of skills to meet goals
 - Learning to learn, to review and refine working methods
 - Cross-fertilisation, diversity helps consider different perspectives
 - Self-evaluation, to monitor and enhance productivity

Groups and Teams

- Teams are small, typically 7 plus or minus 2 members
- Draw on full portfolio of skills needed
- Share common purpose
- Achieve consensus on common approach
- Mutually accountable for goal achievement

Team Performance

- Teams and team members are different
- Teams work in different contexts
- No rule book to automatically create high performing teams
- There are some techniques that can help build performance...
 - Best if team members share sense of direction and urgency of purpose
 - Technical, problem-solving and interpersonal skills needed

Team Performance

- Establish urgency and direction
- Select membership based on skills and potential
- Pay attention to actions
- Set rules for acceptable behaviour
- Focus on few short-term performance tasks and goals
- Update facts regularly
- Spend time together, to build trust and empathy
- Exploit positive feedback

Agile Principles

- Software development teams use creativity and problem solving skills
- Working long, night time, hours has diminishing returns
 - Paid work is different from a personal project or Hackathon setting
 - Working long, night time, hours once or twice might be fine for your own learning or entertainment
- Working at a sustainable pace is best

Agile Principles

- In collective code ownership, developers are empowered to make enhancements across the code-base
- Source code is a resource of the entire team
- Individuals might create a feature but anyone in the team can make enhancements

Forming Teams

- New teams often go through a series of developmental stages
- One simplified view is
 - Forming, define structures, goals and direction
 - Storming, build-up of frustrations and friction within team
 - Norming, team members resolve differences and improve communications
 - Performing, significant progress made towards goals and
 - Adjourning, complete deliverables and wrap up project

Collaboration Activities within Software Teams

- Mentor
 - Mentoring is supporting and assisting other team members
- Co-Ordinator
 - Co-ordinates collaboration with customers
- Translator
 - Explains business language used by customers to team members

Hoda, R., Noble, J., & Marshall, S. (2013).
Self-Organizing Roles on Agile Software Development Teams.
IEEE Transactions on Software Engineering, 39(3), 422–444. <https://doi.org/10.1109/TSE.2012.30>

Collaboration Activities within Software Teams

- Champion
 - Proponent of agile methods with senior management
- Promoter
 - Proponent of agile methods with customers
- Terminator
 - Agitates to remove unhelpful team member

Hoda, R., Noble, J., & Marshall, S. (2013).
Self-Organizing Roles on Agile Software Development Teams.
IEEE Transactions on Software Engineering, 39(3), 422–444. <https://doi.org/10.1109/TSE.2012.30>

Virtual Teams

- Avoids need for work-related travel
- Absence of informal networking and trust building
- Collaboration processes needed to diffuse conflict
- Build awareness of strengths and weaknesses within team

Virtual Teams

- Cultural diversity
 - Sensitivity needed to build trust and empathy across cultures
- Remote pair programming
 - Video conferencing and screen sharing
 - Driver/navigator approach to source code development

Virtual Teams

- Launch
 - Get to know other team members
 - Clarify goals
 - Clarify team member roles and functions
 - Select communication technologies
 - Develop team work rules

Virtual Teams

- Performance improvement through
 - Creating opportunities for social communication to build cohesion, trust and motivation
 - Goal setting and feedback on task fulfilment
 - Feedback should be frequent, concrete and timely
 - Select appropriate communication tools, e.g.
 - Textual tools for recording source code structures and design patterns
 - Video conferencing tools for build trust and empathy

Community of Practice

- Do not contributed to common work tasks
- But share methods, skillsets and approaches
- Voluntary membership
- Goal to improve shared practices
- In Spotify (streaming service) software development, known as Guilds

Exercises

- Exercises 2.1 and 2.6 encourage creation of a learning journal
- Exercise 2.2 and 2.3 help you to learn about members of your team
- Exercise 2.4 agreeing group behaviours
- Exercise 2.5 establishing team goals

Summary

- Self-Organising Teams
 - Decide **HOW** to meet goals
- Groups and Teams
 - Teams have shared purpose
- Team Performance
 - Build urgency around shared purpose
- Agile Principles
 - Sustainable Pace, Collective code ownership

Summary

- Collaboration Activities in Agile Teams
 - Team members take action to help team performance
- Virtual Teams
 - Remote working convenient, but lacks informal networking
- Communities of Practice
 - Share methods and approaches to improve practice