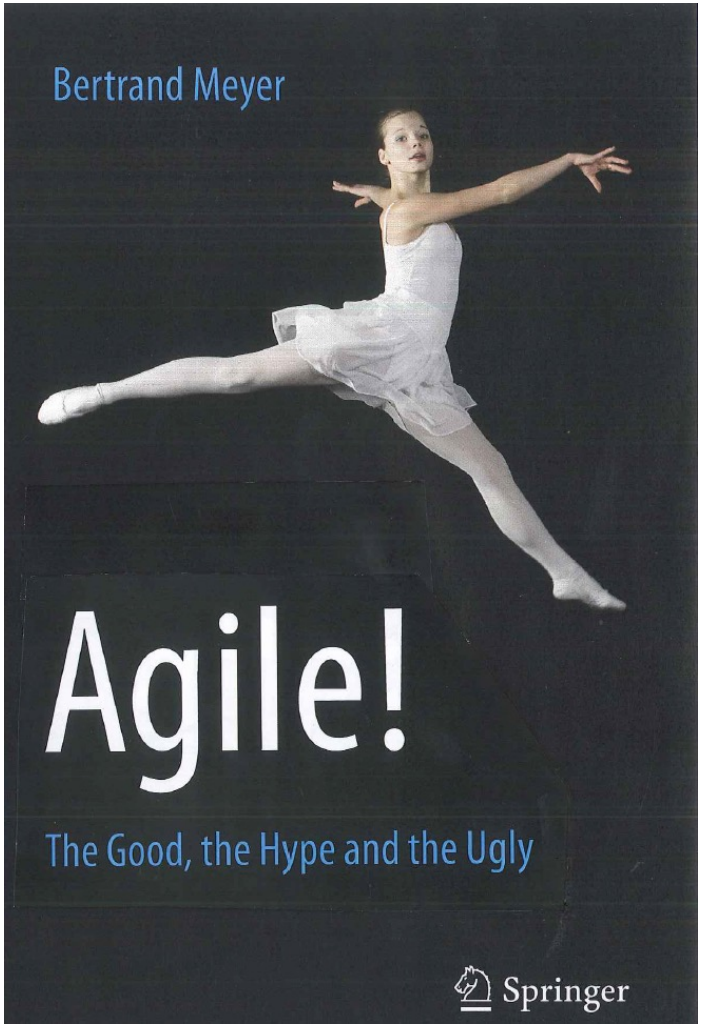
ETH zürich

Agile software development

Bertrand Meyer

Part F: assessment


Bertrand Meyer
Agile!
The Good, the Hype and the Ugly
Springer

The ugly

- Rejection of upfront tasks
  - Particularly: no upfront requirements
  - Dismissal of a priori architecture work
- User stories as a replacement for abstract requirements
- Tests as a replacement for specifications
- Feature-based development & ignorance of dependencies
- Method keeper (e.g. Scrum Master) as a separate role
- Test-driven development (but not the rest of agile's emphasis on tests)
- Dismissal of traditional manager tasks
- Dismissal of auxiliary products and non-shippable artifacts
- Dismissal of a priori concern for extendibility
- Dismissal of a priori concern for reusability

The indifferent

- Pair programming
- Open-space working arrangements
- Self-organizing teams
- Maintaining a sustainable pace
- Producing minimal functionality
- Planning poker
- Cross-functional teams
- Embedded customer

The good

- Acceptance of change
- Iterative development
- Emphasis on working code
- Tests as one of the key resources of the project
- Constant test regression analysis
- Notion of velocity
- No branching
- Product (but not user stories!) burndown chart
- Daily meeting

The brilliant

- Short iterations
- Closed-window rule
- Refactoring (but not as a substitute for design)
- Associating a test with every piece of functionality
- Continuous integration

Final observations

Software development is hard; quality is key

Lots of good ideas can help; there is no reason to reject those from any particular style of software engineering

- Particularly in the absence of credible empirical data

Agile will find its place in the history of productive software engineering ideas

Agile Software Development

Bertrand Meyer

Part F: Assessment

What we have seen:

Agile is a mix of good and bad ideas

(some very bad, and some very good)

and others not particularly important

It is a major step in the evolution of software engineering