Agile software development

Helena Holmström Olsson

Department of Computer Science and Engineering
Chalmers | University of Gothenburg

Agile software development: What?

- Development approach comprising methods that address turbulent business environments
  - Changing customer requirements
  - Shorter time-to-market
  - Rapidly advancing information technologies
- Development approach focusing on team-level software development
  - Small (cross-functional) development teams
  - Co-located development teams
  - Empowered development teams (end-to-end responsibility)
- Development approach trying to solve traditional SE problems
  - Long development life cycles
  - Difficulties to handle changing customer requirements
  - Customer value late in the development process

Agile software development: Definitions

- ”…readiness for motion, nimbleness, activity” (dictionary.oed.com)
- ”…moving quickly and lightly” (wordnet.princeton.edu)
- ”…being effective and manoeuvrable” (Cockburn and Highsmith, 2001)
- ”...the continual readiness of an entity to rapidly or inherently create change, proactively or reactively embrace change, and learn from change” (Conboy, 2009)

What makes a software development method 'agile'?

- **Adaptive,** i.e. it welcomes and responds to changes, evolutionary refinement of plans and goals
- **Iterative and incremental,** i.e. short/rapid iterations, small software releases, incremental growth of the system
- **People-oriented and co-operative,** i.e. relies on tacit knowledge within a team, actively involve users in the development process

Agile manifesto (http://agilemanifesto.org)

- **Individuals and interactions** over processes and tools
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan

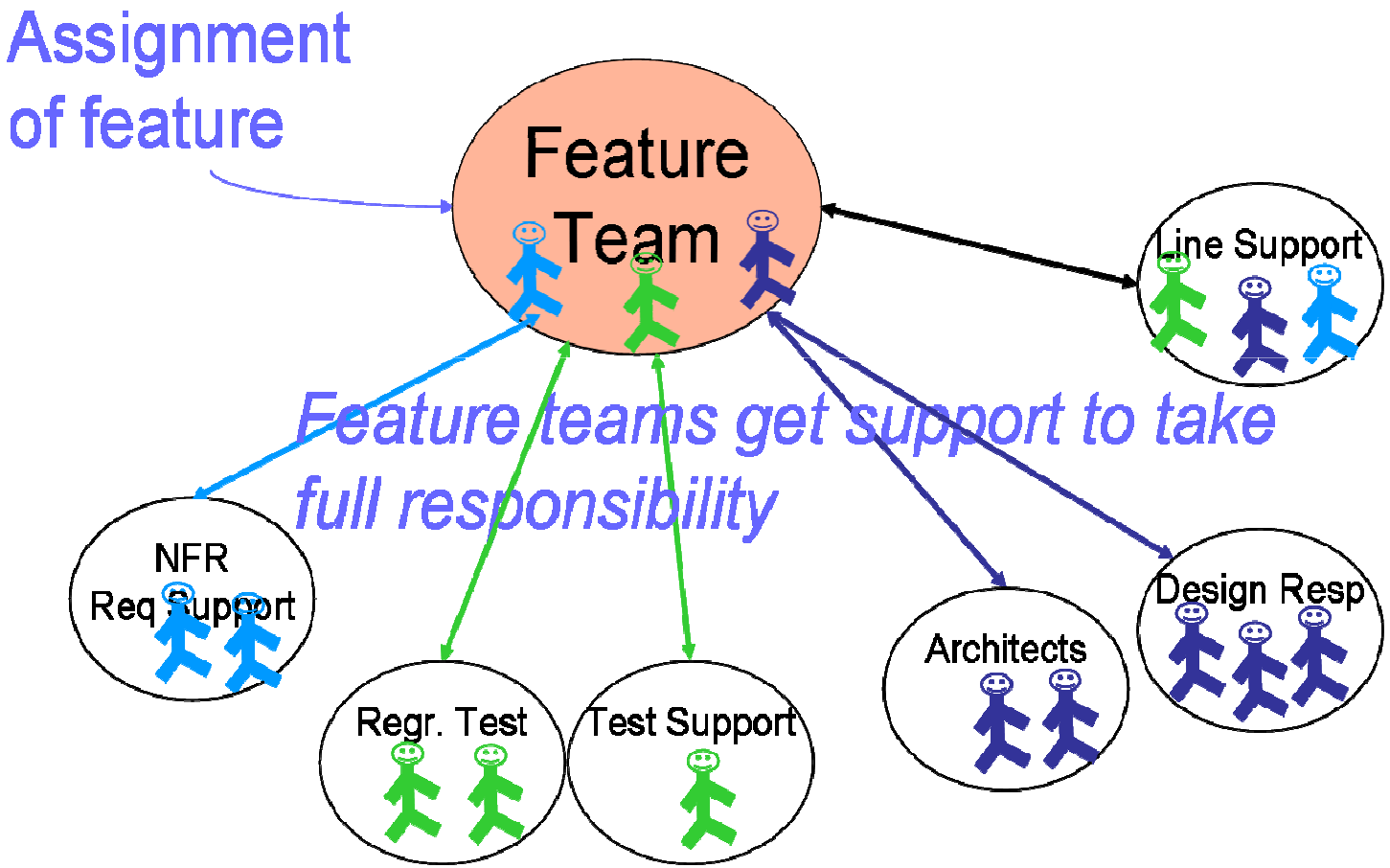
Agile software development: Why?

- **Traditional problems**
  - Software projects exceeding budgets
  - Software projects exceeding schedules
  - Software projects having poor level of quality when/if completed
- **Questioning of traditional development methods**
  - Managed/controlled process vs. Random/opportunisitc process
  - Linear/sequential process vs. Simultaneous/overlapping process
  - Universal process vs. Unique process
  - Rational/determined process vs. Negotiated/compromised process
- **Agile software development methods proposed as a solution**
  - Promoting communication, flexibility, innovation, teamwork
  - Promoting productivity rather than process

Research at Ericsson 2008-2010

- Long development life cycles
- Difficulties to manage changing requirements
- Productivity bottlenecks (late testing and re-design)

- ‘Streamline development’ was introduced as an agile approach to increase flexibility, reduce lead time, narrow project scope (feature prioritizing) and empower development teams
- One central part in ‘streamline development’ is cross-functional (feature) teams consisting of core competences, i.e. test, design, system management
- Cross-functional teams take an end-to-end responsibility


Assignment of feature
Feature Team
Line Support
Feature teams get support to take full responsibility
NFR Req Support
Regr. Test
Test Support
Architects
Design Resp

Cross-functional teams – 'prerequisites'

- **Communication of high-level vision**
  - Clear communication of framework and high-level goals
  - Evolutionary transition to agile process
- **Continuous management commitment**
  - Close contact during process implementation
- **Coherent resource allocation**
  - Organize teams based on core competencies
  - Availability of associated competencies
- **Team stability**
  - 6 – 10 persons/team, stable at least 6-12 months
  - Change only parts of the team at the same time
  - Avoid having roles distributed between too many teams ("bumble-bees")
- **Team authority**
  - End-to-end responsibility
  - Operate freely within the agile framework
- **Team co-location**
  - Co-locate core competences ,i.e. test, design, system management
  - Encourage teams to use physical environment for team activities
  - Prioritize competence broadening within each discipline

Cross-functional teams – 'challenges'

- Competence
  - Parallel activities compete for similar resources (single/expert competences)
  - Stability enforces efficiency but sometimes certain teams experience a lack of expertise
  - How to find a balance between broad competence and domain-specific expertise?
- Company culture
  - Structure of organization and habitual behaviors (reward system)
  - How to encourage knowledge sharing, communication and coordination?
- Communication
  - Delayed or lost communication between teams and roles
- Feature vs. Overall view
  - Risk to focus only on the feature – limited understanding of overall system
  - Need an understanding for overall view/design to be efficient
- Team workload
  - System managers/designers/test workload