Agile Process Models

Agile Methodologies

- Traditional IS development Methodologies "are treated primarily as a necessary fiction to present an image of control or to provide a symbolic status"
- Lightweight alternative to traditional heavyweight processes.
- Began with introduction of extreme programming (XP) by Beck in 1999 is starting point for Agile Methodologies.
- Agile Manifesto in 2001
- A number of other methods have been invented or rediscovered since.
- However, no clear agreement on how to distinguish Agile from Traditional Methodologies.

Agile Values

- Individuals and interactions over tools and processes.
- Working software over comprehensive documentation – continuously turn out working tested software.
- Customer collaboration over contract negotiation.
- Responding to change rather than following a plan – adaptive rather than prescriptive.

Agile Characteristics & Principles

- Satisfy the customer through early and continuous delivery of valuable software.
- Incremental and timebound with short iteration cycles from 1 to 6 weeks.
- Development group consists of both software developers and customer representatives. Regular feedback to developers. Authorised to consider adjustments during development.
- Working software is the primary measure of progress
- What is new is the recognition of people as the primary drivers of project success along with intense focus and manoeuvrability.

Agile Characteristics & Principles

- Fully automated regression testing.
- Experienced developers – speeds up project by a factor anywhere from 2 to 10 times.
- Minimal process overheads, remove unnecessary process activities.
- Modelling tools not as useful as generally thought.
- Continuous attention to technical excellence, e.g. refactoring in XP.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.

Agile Methodologies

- Extreme Programming - XP
- Scrum
- Crystal
- Feature Driven Development
- UP?

XP

- Evolved from problems caused by long development cycles.
- Has 5 phases
  - Exploration phase
  - Planning phase
  - Iterations to release phase
  - Productionising
  - Maintenance and Death

Exploration Phase

- Customers write story cards for first release.
- Each story card describes feature of system (like a brief use-case).
- Project team familiarise themselves with tools, technology and practices they use in the project.
- Try out tools and explore possibilities for architecture via prototyping.
- Phase can take from a few weeks to a few months.

Planning Phase

- Set priority for stories.
- Agree contents of first small release.
- Estimate effort for each story and agree schedule.
- First release normally not more than 2 months.
- Planning phase itself takes a couple of days.

Iterations to Release Phase

- Several iterations from 1 to 4 weeks.
- First iteration creates a system with architecture for whole system.
- Select story that will enforce building of structure for whole system.
- Customer selects stories for each iteration.
- Functional tests created by customer run at end of each iteration.

Productionising Phase

- Requires extra testing and performance checking before release to customer.
- New changes may still be found.
- Shorter iterations.

Maintenance and Death Phase

- Keep system in production while producing new increments.
- Requires effort for customer support tasks.
- Development speed may slow.
- Maybe incorporate new people into team.
- Death phase – no more stories to be implemented, or if system not delivering desired outcomes or is proving too expensive for more development.

XP Roles

- Programmer
- Customer – writes stories and functional tests
- Tester – helps customer write functional tests
- Tracker – gives feedback on schedule to team
- Coach – has sound understanding of XP
- Consultant – specific technical knowledge
- Manager

XP Practices

- Planning game
- Small short releases
- Simple design – simplest possible design that is implementable at the moment.
- Testing – development is test driven. Unit tests are implemented before code and run continuously. Customers write functional tests.
- Refactoring
- Pair programming

XP Practices

- Collective ownership – anyone can change any part anytime
- Continuous integration – new piece of code integrated into code-base as soon as it is ready. System integrated and built many times daily. All tests run and have to be passed for change to be accepted.
- 40-hour week
- On site customer
- Open workspace

XP – scope of use

- Small to medium sized teams, 3 to 20 project members.
- Communication between members at all times, even scattering across two floors is not acceptable.
- Any resistance by team members, management or customers may fail the process.
- Growing research on XP

Scrum

- Developed for managing the systems development process.
- No specific software development technique, can be adopted to manage an organisations own engineering practices.
- Focuses on flexibility of team members in producing software in a constantly changing environment.
- Frequent management activities to identify and rectify any impediments in the development process.

Scrum

- Suitable for small teams of less than 10
- Development relies on several environmental and technical variables which are likely to change:
  - Requirements
  - Time frame
  - Resources
  - Technology
  - Development Methods.

Scrum – 3 phases

- Pregame phase, which consists of 2 sub-phases
  - Planning
  - Architecture and high level design
- Development or Game phase
- Postgame phase

Pregame Planning sub-phase

- Product Backlog List
  - All requirements that are currently known
  - Prioritised
  - Development effort estimated
- Includes definition of team, tools and other resources.
- Risk assessment
- Training needs

Pregame Architecture sub-phase

- High level design of system including architecture planned based on current Product Backlog.
- For enhancement to an existing system, changes needed for implementing backlog are identified along with any problems they may cause.
- Design review meeting held to go over proposals and decisions are made on the basis of this review.

Development or Game Phase

- Agile part of Scrum, unpredictable expected.
- System developed in Sprints
  - Iterative cycles where functionality is developed or enhanced to produce new increments.
  - Included traditional development activities of analysis, design, implementation, delivery
  - A Sprint is planned to last from 1 week to 1 month
  - Could be 3 to 8 sprints in a system development process before ready for distribution.
- Environment and technical variables are observed and controlled during sprints, not just at beginning.

Postgame Phase

- Entered when agreement reached that environmental variables such as requirements are completed
- System ready for release
- Integration
- System Testing
- Documentation

Scrum Roles & Responsibilities

- Six roles:
  - Scrum Master
  - Product Owner
  - Scrum Team
  - Customer
  - User
  - Management

Scrum Roles

- Scrum Master
  - Responsible for ensuring that project is run according to practices, values and rules of Scrum.
  - Interacts with team, customer and management.
  - Responds to impediments

Scrum Roles

- Product Owner
  - Responsible for managing and controlling backlog list
  - Selected by Scrum Master, Customer and Management
  - Makes final decisions on backlog, decides on features to be developed, estimates development effort.
- Scrum Team
  - Self organising in order to achieve goals

Scrum Roles

- Customer – participates in creating product Backlog
- Management
  - in charge of final decision making
  - Also involved in selecting Product Owner
  - Gauging progress with Scrum Master