Embedded Systems Software Engineering

Jarosław Kuchta

Software lifecycle
“Waterfall” classic model

- Planning
- Analysis
- Design
- Implementation
- Testing
- Maintenance
“Waterfall” model - characteristics

**Pros**

- Well organized process
- Complete model
- Milestones
- Well documented

**Cons**

- Costly
- Long process
- Requirements hard to be completely specified
- Frequent turn-backs
- Uncertainty of success
Model with a prototype

1. Preliminary requirements
2. Prototype development
3. Prototype evaluation
   - Ready
   - Not ready
4. Is prototype in target technology?
   - T: Prototype refinement
   - N: Final software development
5. Testing
6. Product
Model with a prototype - characteristics

Pros
- Better requirements recognition
- Lower development risk

Cons
- Lack of business goals analysis
- Misunderstanding of prototype role (may be treated as final version)
- Disproportionately high costs of final development phase
- Prototype solution may be not optimal
“Spiral” iterative model
Iterative model - characteristics

**Pros**
- More complete requirements specification achieved in many cycles
- Evaluation of solution in each cycle - lower development risk

**Cons**
- Large costs of error location and elimination in the software being used.
- New versions with new errors
- Opinion of unreliability of the software provider (risk)
- Customers can feel lack of benefit from continuous improvement
RUP – Rational Unified Process

• Iterative software development
• Requirements management
• Graphical modeling and visual design
• Off-shelf components usage
• Software Quality Assurance
• Change management
Requirements management

- Business goals analysis
- Understanding of stakeholders needs
- Use cases identification
- Project scope determining
- Requirements specification
- Requirements management
Graphical modeling and visual design

• Unified Modeling Language (UML)
  – modeling with diagrams
  – well known and understandable
  – solution structure better understanding
  – no support for programming

• Visual software design
  – ease to use visual tools
  – visual tools ≠ UML diagrams
Off-shelf components usage

**Pros**
- Generally shorter development time
- Proven patterns to use
- Ease of division of labor at independent teams

**Cons**
- The necessity of matching ready-made components to suit the customer’s needs
- Problems with understanding of solution frameworks
- The risk to choose the wrong framework
- High risk of project delay
Software Quality Assurance

• Standards management
• Artifacts technical reviews
• Early error detection and elimination (costs and benefits)
Change management

- Tracking, recording and monitoring introduced changes
- Informing the persons concerned about these changes
- Maintaining consistency of the project artifacts
RUP phases

- Inception phase
- Elaboration phase
- Construction phase
- Transition phase
Initial phase

• Business goals analysis
• Success factors
• Use-cases model
• Project plan
• Preliminary risk analysis
• Preliminary requirements specification
Initial phase milestone

- Understanding of the main requirements
- The agreement on the estimated cost / time
- The credibility of the preliminary schedule
- Acceptance of the prototype
Elaboration phase

• System analysis and design

• Milestone:
  – Completeness of use case model at 80%
  – Designed system architecture
  – Ability to implement the main use cases in the designed architecture
  – Checking objectives reachability and possible risks
  – A detailed schedule for the entire project
Construction phase

- Implementation and testing
- Milestone:
  - The product is stable and ready for deployment
  - Customers are ready for product acceptance
  - Actual costs and time are within acceptable limits
Transition phase

• The transition from development to deployment
• User training
• Beta testing
• Milestone:
  – Satisfaction of the users / customers
  – Actual costs and time are acceptable
RUP disciplines (activities)

• Engineering disciplines:
  – Business modeling
  – Requirements
  – Analysis and design
  – Implementation
  – Test
  – Deployment

• Auxiliary disciplines:
  – Managing changes and configuration
  – Project management
  – Environment
Alternative approaches

• Open Unified Process (OpenUP)
• Eclipse Process Framework (EPF)
• Enterprise Unified Process
• Agile programming
Bibliography

• Roger S. Pressman: Software Engineering. A Practitioner's Approach (book, PDF)
• OMG.ORG: Unified Modeling Language (PDF)
• Eclipse.org: Introduction to OpenUP (PDF)
• Scott W. Ambler: Introduction to the Enterprise Unified Process (PDF)
• Agile Manifesto (HTML)
• Kent Beck: Extreme Programming Explained. Embrace Change (book with Cynthia Andres, PDF)
• scrum.org, scrumalliance.org (HTML)