이영준

Agile 개발 방법과 모델기반 설계

AUTOMOTIVE CONFERENCE 2019
Agenda

- Agile Values and Typical Workflow
- Model-Based Design (MBD)
- Agile Development with MBD
- Scrum with MBD
### Agile Values

<table>
<thead>
<tr>
<th>Agile Values</th>
<th>Over</th>
<th>Other Agile Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Individuals &amp; Interactions</td>
<td>over</td>
<td>Process and Tools</td>
</tr>
<tr>
<td>Customer Collaboration</td>
<td>over</td>
<td>Contract Negotiation</td>
</tr>
<tr>
<td>Working Software</td>
<td>over</td>
<td>Comprehensive Documentation</td>
</tr>
<tr>
<td>Responding to Change</td>
<td>over</td>
<td>Following a Plan</td>
</tr>
</tbody>
</table>

"While there is value in the items on the right, we value the items on the left more."

- The Agile Manifesto, 2001
Agile is a mindset defined by values, guided by principles and manifested through many different practices. Agile practitioners select practices based on their needs.

~ Agile Practice Guide (PMI® and Agile Alliance®)
Typical agile development workflow

1-4 Weeks
Model-Based Design
Models == Understanding
Simulation is key to Level 4-5 autonomy

*Source: Center for Artificial Intelligence, Saarland University
Model-Based Design

Systematic use of models throughout the development process

Modeling

Simulation

Fast repeatable tests
Model-Based Design

**Systematic use of models throughout the development process**

- **Modeling**
  - Simulation
- **Automation**

Fast repeatable tests
Model-Based Design

Systematic use of models throughout the development process

- Modeling
  - Simulation
- Automation
  - Coding
  - Verification

Fast repeatable tests
Model-Based Design

**Systematic use of models throughout the development process**

- **Modeling**
  - Simulation
  - Fast repeatable tests

- **Automation**
  - Coding
  - Verification
  - Fast agile development loops
Model-Based Design

RESEARCH ACTIVITIES

REQUIREMENT DOCUMENTS

ANALYSIS • SPECIFICATION • DESIGN

Model

- Architecture
- Physical Components
- Constraints
- Algorithms
- Environment

INTEGRATION

IMPLEMENTATION

- C, C++
- VHDL, Verilog
- Structured Text
- MCU
- DSP
- FPGA
- ASIC
- PLC
- PAC

TEST & VERIFICATION

- Test Cases

Executable Specification
- Unambiguous – easy to understand
- Systems engineering - modeling whole system including environment
- Sharing of models to improve communication and collaboration
- Early validation and test development

Multi-domain Design
- Model algorithms and environment
- Perform integration testing at model level before implementation

Automatic Code Generation
- Eliminate errors from hand-coding
- Regenerate easily for different targets

Continuous Test and Verification
- Detect errors early in development
- Reduce use of physical prototypes
- Reuse tests throughout development process
Agile Development with MBD
Agile Development – Key Principles

Customer satisfaction by rapid delivery of useful software

Picture Source: http://www.softwareplant.com/scrum/
Principle: Customer satisfaction by rapid delivery of useful software

- **Simulation** allows customer evaluation of functional behavior **early and often**.

- **Useful software** can be delivered throughout the project via **code generation**.
Agile Development – Key Principles

Welcome changing requirements, even late in development.

Customer satisfaction by rapid delivery of useful software.

Picture Source: http://www.softwareplant.com/scrum/
Principle: Welcome changing requirements, even late in development

- **Requirements traceability** supports impact analysis of affected model components.

- Dynamic/executable models allow for **rapid evaluation of requirements** changes.

- **Regression testing** of simulation-based tests can be automated to confirm **new design meets existing requirements**.
Agile Development – Key Principles

Inputs from Executives, Team, Stakeholders, Customers, Users

- **Customer satisfaction by rapid delivery of useful software**

- **Welcome changing requirements, even late in development**

- **Working software is the principal measure of progress**

Picture Source: http://www.softwareplant.com/scrum/
Principle: Working software is the principal measure of progress

- Functional designs can be evaluated continuously via **executable models**.

- Code generation supports rapid software deliveries, **rapid prototyping** and HIL test.
Agile Development – Key Principles

Customer satisfaction by rapid delivery of useful software

Welcome changing requirements, even late in development

Continuous attention to technical excellence and good design

Working software is the principal measure of progress

Picture Source: http://www.softwareplant.com/scrum/
Principle: Continuous attention to technical excellence and good design

- Continuous development and testing in an executable modeling environment allows for – “build a little / test a little’’ workflows.

Where are you! ...still coding?

Work on your functionality, not on your code
Agile Development – Key Principles

Customer satisfaction by rapid delivery of useful software

Continuous attention to technical excellence and good design

Welcome changing requirements, even late in development

Working software is the principal measure of progress
Principle: Simplicity—the art of maximizing the amount of work not done—is essential

- Model-Based Design supports identification of bugs where they are introduced and “cheaper” to fix.

- Reduces rework (design and test).
Agile Development – Key Principles

- Continuous attention to technical excellence and good design
- Self-organizing teams
- Welcome changing requirements, even late in development
- Working software is the principal measure of progress
- At regular intervals, the team reflects on how to become more effective
- Customer satisfaction by rapid delivery of useful software

**Inputs from Executives, Team, Stakeholders, Customers, Users**

- Product Owner
- The Team

**Scrum Master**

**Daily Scrum Meeting**

**Burndown/up Charts**

**Sprint Backlog**

**Sprint Planning Meeting**

**Task Breakout**

**Sprint 1-4 Week**

**Sprint Review**

**Finished Work**

**Sprint Retrospective**

*Picture Source: [http://www.softwareplant.com/scrum/]*
Principle: At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly

- Collect metrics from models to measure and improve process e.g. test coverage, model metrics
- Flexible tool suite supports wide variety of workflows and development processes.
- Open APIs.
Agile Development – Key Principles

- **Customer satisfaction by rapid delivery of useful software**
- **Welcome changing requirements, even late in development**
- **Continuous attention to technical excellence and good design**
- **Close, daily cooperation between business people and developers**
- **Self-organizing teams**
- **Working software is the principal measure of progress**
- **Simplicity**
- **At regular intervals, the team reflects on how to become more effective**

**Inputs from Executives, Team, Stakeholders, Customers, Users**

**Product Owner**

**Product Backlog**

**Sprint Planning Meeting**

**Sprint Backlog**

**Team selects starting at top as much as it can commit to deliver by end of Sprint**

**Task Breakout**

**Scrum Master**

**Burndown/up Charts**

**Daily Scrum Meeting**

**Every 24 Hours**

**Sprint Review**

**Finished Work**

**Sprint Retrospective**

**1-4 Week Sprint**

**Continuous attention to technical excellence and good design**
Scrum with MBD
Model-Based Design

- Executable Specifications from Models
- Design with Simulation
- Implementation with Automatic Code Generation
- Continuous Test and Verification

Integrate

Develop

Generate code

Simulate

Test
**Evolved System Requirements**

- Simulate
- Generate code
- Model-Based Design
  - Executable Specifications from Models
  - Design with Simulation
  - Implementation with Automatic Code Generation
  - Continuous Test and Verification

**MIL – Modeling only**

- **Integrate** subsystems
- **Verify** design choices
- **Integrate** external code
- **Develop/Design** plant and algorithm models

- **Test**
- **Simulate**
- **Generate code**
- **Evaluate/Decide**
- **Validation**
HIL Verification

Evolved System Requirements

Generate code of plant model

Integrate candidate with HIL system

Verify candidate implementation against simulated plant

Develop/Design plant model

Model-Based Design
- Executable Specifications from Models
- Design with Simulation
- Implementation with Automatic Code Generation
- Continuous Test and Verification

Integrate external code

Simulate Test

BACKLOG
DEFINITION OF READY
DEFINITION OF DONE
DELIVERY

• Automatic code generation
  Real-Virtual simulation

Integrate candidate with HIL system

Generate code of plant model

Verify candidate implementation against simulated plant

Evolved System Requirements
Rapid prototyping

Integrate with real system

Verify prototype against real plant

Evolved System Requirements

Develop/Design algorithm

Integrate external code

Generate code of algorithm to prototype board

Automatic code generation Real-Virtual simulation

Model-Based Design
- Executable Specifications from Models
- Design with Simulation
- Implementation with Automatic Code Generation
- Continuous Test and Verification

Simulate Test

BACKLOG DEFINITION OF READY DEFINITION OF DONE DELIVERY
Integrated Agile System Development through Rapid Development Iterations

Deploy

Design

Integrate

Implement

Evaluate

Real-Virtual Simulation

Controller | Plant
---|---
MIL, SIL, PIL | Virtual Virtual
RCP | Virtual RT Real
OTRP | Real Real
HIL | Real Virtual RT

External Code

Models (Components, Subsystems, System)

Algorithm

Plant Model

System Architecture

Automatic Code Generation

Algorithm | Plant
---|---

Desktop CPU: MIL, SIL
Target CPU: PIL
Prototyping RT CPU: RCP, HIL
Production EC: HIL, OTRP

Controller Plant

MIL, SIL, PIL Virtual Virtual
RCP Virtual RT Real
OTRP Real Real
HIL Real Virtual RT

Integrated Agile System Development through Rapid Development Iterations
Who will be successful in the future?

Mechanical-centric

Model-centric

Software-centric

Comprehensive models
Simulation based testing
Generate code and automate verification
Questions?