EXPERIENCES FROM APPLYING MBT IN AN AGILE SCRUM CONTEXT

AN MBT UC 2011 PRESENTATION

ATHANASIOS KARAPANTELEAKIS (athanasios.karapantelakis@ericsson.com)
ALLAN ANDERSSON (allan.andersson@ericsson.com)
JOHAN MALMBERG (johan.malmberg@ericsson.com)
BOGUMILA RUTKOWSKA (bogumila.rutkowska@ericsson.com)

ERICSSON AB, TELLUSBORGSVÄGEN 83-85, 126 25 STOCKHOLM, SWEDEN
OVERVIEW

› Introduction to test domain
› Approach
› Contributions and results
› Technical challenges
› A mental model for deployment of MBT in SCRUM projects
› Conclusion
Test Scope: modeling of O&M interfaces (man-machine communication) for a telecommunication system.

- User point of view when creating models
- Because of a strict, formalized structure of commands, there is no requirement for wrappers or APIs.
  - Success story with MBT: Generic command and printout handling in test execution harness.
  - Cost efficiency: wrapper class per interface versus wrapper class per command.

Test methodology: application of MBT for testing a system being developed using SCRUM.
APPROACH (1/3)

› Background and tool selection
  - Conformiq:
    › Provider of an MBT tool suite used to design models and generate test cases out of those models.
    › Model design is UML-based complemented by Java-like code.
    › Black-box testing approach: models describe sequences of incoming and outgoing messages to and from the system being modeled.
  - Glue logic
    › Code between the model domain and the test execution platform
    › *Translates* the sequence diagrams produced from the model to executable test cases.
    › Incoming messages to the system are O&M commands, and outgoing messages are command printouts.
      - Based on logic in the model, glue logic creates a set of executable test scripts, through a process within which incoming messages from the model are interpreted as O&M commands, and outgoing messages are interpreted as command printouts.
APPROACH (2/3)

Introduction → **Approach** → Contributions and results → Technical Challenges → MBT in SCRUM mental model → Conclusion
APPROACH (3/3)

Timeline

<table>
<thead>
<tr>
<th>MBT Evaluation</th>
<th>MBT Deployment</th>
</tr>
</thead>
</table>

Test specifications of previous system versions

Product backlog

Models

Glue Logic code and test execution harness

Sprint 1

Design Model and execute

Reuse
Contribute
Execute and verify

Sprint 2

... → Sprint X

Reuseable Model Assets

Test Automation Framework

Introduction → Approach → Contributions and results → Technical Challenges → MBT in SCRUM mental model → Conclusion
EVALUATION

› Duration (approximately two months)
› Time segmentation (man hours)
  – As a percentage of total time
    › Creation and refinement of glue logic (one time effort): 53%
    › Creation of models (including verification of models/execution of test cases): 47%

Efficiency of model based testing versus manual testing in evaluation phase

Test cases covered by MBT as a percentage of the total number of test cases selected for the evaluation phase

Completeness: We managed to cover 78% of the test specification

Introduction → Approach → Contributions and results → Technical Challenges → MBT in SCRUM mental model → Conclusion
TECHNICAL CHALLENGES

› Read data from printouts
  – Contracts between the test harness and model-level design.

› Non-deterministic situations
  – Ambiguous command printouts

› Large number of test cases (impacts test execution time)
  – Compacting test suite
MODELING A PROCESS

› Value of modeling the “MBT introduction” process
  - Simulations help correlate measurable parameters to varying values of preset parameters.
  - Facilitates project planning, assignment of resources, estimation of costs.

› Using System Dynamics (SD) mental models as a tool for planning for MBT deployment within a SCRUM project.
  - Define MBT introduction stages
    › Preparation
      - Automated test execution framework
      - MBT training
    › Deployment
      - Define model parameters
        › Measurable parameters
          - Cost of resources, time to deliver, quality
        › Preset parameters
          - Number of engineers allocated, project/training deadlines
MBT TRAINING

Mental model for capturing the process of MBT training within AXE I&V

Introduction → Approach → Contributions and results → Technical Challenges → MBT in SCRUM mental model → Conclusion
MBT TRAINING DYNAMICS

Introduction → Approach → Contributions and results → Technical Challenges → MBT in SCRUM mental model → Conclusion
DEVELOPMENT OF TEST EXECUTION AUTOMATION FRAMEWORK

Mental model for capturing the process of developing an MBT test automation framework

Introduction → Approach → Contributions and results → Technical Challenges → MBT in SCRUM mental model → Conclusion
DYNAMICS OF THE FRAMEWORK DEVELOPMENT PROCESS

Maturity of test execution framework and MBT adaptor

Introduction → Approach → Contributions and results → Technical Challenges → MBT in SCRUM mental model → Conclusion
MBT TEAM EFFICIENCY DYNAMICS

Introduction → Approach → Contributions and results → Technical Challenges → MBT in SCRUM mental model → Conclusion
Combining models

Introduction → Approach → Contributions and results → Technical Challenges → MBT in SCRUM mental model → Conclusion
LESSONS LEARNED

Experiences from evaluation
- Models can focus on the final solution
  › In every sprint, execute only the subset of test cases generated from the model that correspond to implemented functionality.
- Design teams may come up with temporary workarounds, not present in the final version
  › Model workarounds can be introduced and deactivated later
  › Save efforts for redesigning the model later

Experiences from simulation of SD models
- SD models capture the inter-relations of variables that determine project success.
  › Resource allocation, based on engineer experience, that leads to lower costs.
  › Resource allocation, based on engineer experience, that delivers results faster.
  - But also: Optimal allocation of engineers that leads to the best compromise of time and costs.
A **third** level of testing process automation

- Complete model creation versus model “stubs”.
- Generated test cases consistency, correctness.
- Reduced testing costs, lead-time.