Model-based Testing as a Service

Steffen Herbold, Andreas Hoffmann

International Journal on Software Tools for Technology Transfer, June 2017

SWVV Presentation 2017.11.30. – Pomázi Krisztián

Introduction

- Quality of Web Services – Number of failures
- Testing of service-oriented architectures is complex
- Model-based Testing (MBT) can deal with the complexity of the testing
- MBT yields a large amount of tests
- Can cloud computing be used to provide the required resources?

Table of contents

Motivation

- Failures caused by bad software quality:
  - HSBC bank payment delay for 275.000 payments
  - HSBC bank no online access for days
  - Smart termostat software failure in winter
  - And the ones mentioned in the class
- Service-oriented architectures (SOA): rise in numbers
- SOA: the software is decomposed into single Web Services that are coupled loosely with each other
- Web Services are described using their interfaces, independent of their implementation of location
- Example: flight search
- High quality is critical

Motivation

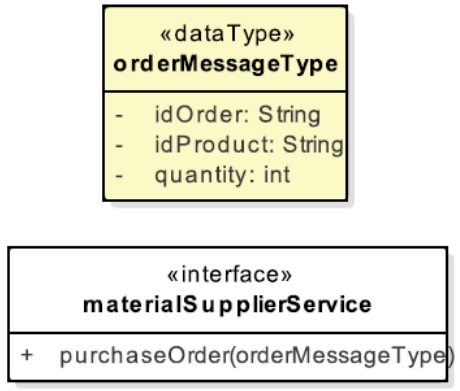
- Multiple Web Services: SOA orchestration
- Challenges in testing SOA within a full application:
  - Rapid change in the usage of services
  - Only interface of service available
  - Unanticipated evolution of services by other providers
  - Setting up a test environment is already a challenging task
- Models:
  - High level of abstraction
  - Defining complex behaviour in a compact way
  - Number of tests depend on the strategy: single/double/triple cov.
  - Resource limitations: too many resources needed for good coverage

Model-based testing

- „Testing based on or involving models"
- Different test artifacts:
  - Abstract test cases
  - Concrete test cases for manual testing
  - Concrete test cases for automatic testing
- Cloud execution: only possible without manual interaction
- Automated MBT -> it is important, that the test model is rich in terms of detailed informations about the SUT (examples shown in UML):
  - Behavioral model
  - Interface descriptions
  - Deployment information

Model-based testing – Structural description

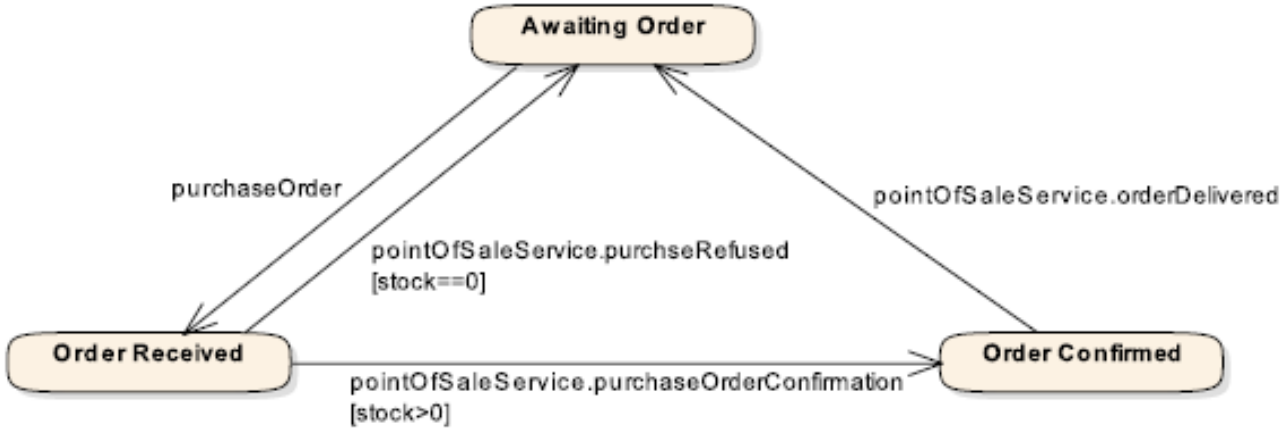
- Interfaces exposed by the SUT
  - Operations called
  - Data structures used for input and output
- Convenient solution: UML



Model-based testing – Behavioral models

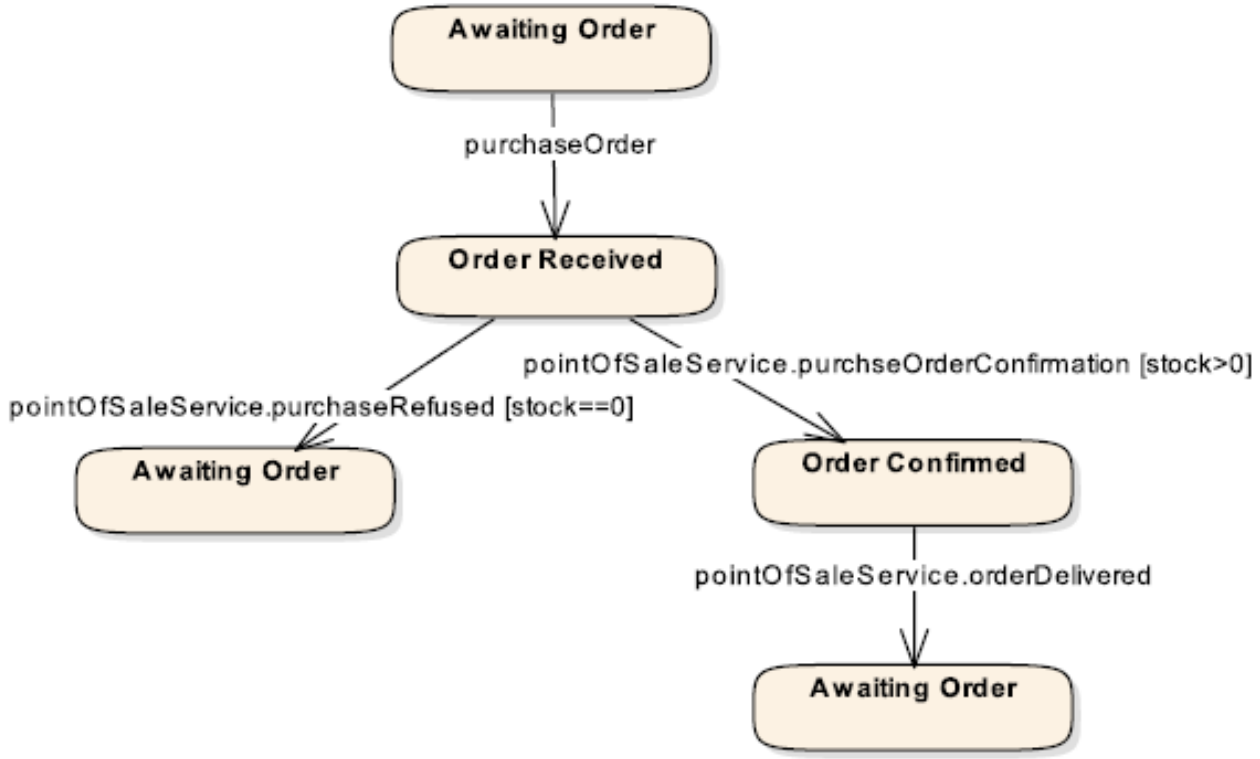
- How the SUT should behave when interacting with its environment
- Combination of behavioral model and test strategy decides which and how many test cases are derived from the model
- UML Sequence Diagram and State Machines

Model-based testing – Behavioral models



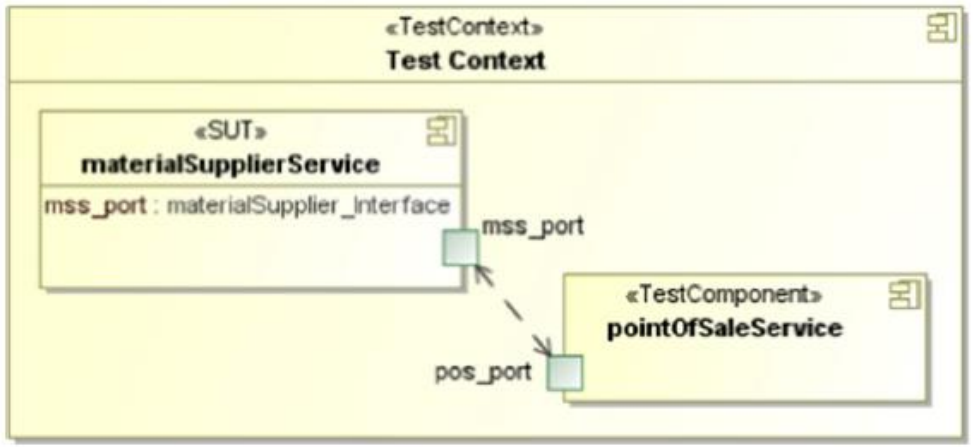
Model-based testing – Behavioral models

- Coverage of round-trip paths
- Round-trip paths are determined using the transition tree



Model-based testing – Deployment information

- Two possible locations:
  - Part of the test harness where the test cases are executed
  - In the model and part of the generated test cases
- Hybrid example below

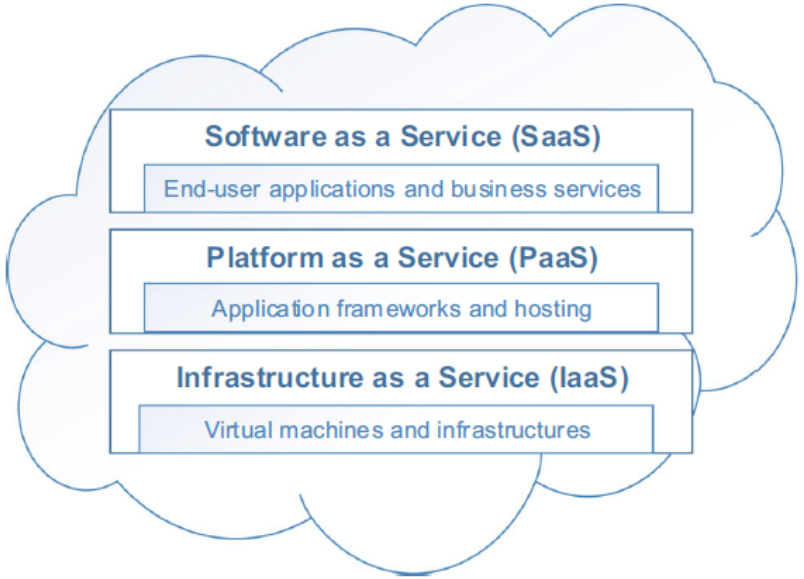


Cloud computing – Definition and Characteristics

- Definition by National Institute of Standards and Technology (NIST): A model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction
- 5 characteristics
- 2 Most important:
  - Rapid elasticity
  - On-demand services

Cloud computing – Service Models

- 3 service models
- IaaS: most influence
- PaaS: consumer still has control
- SaaS: consumer can acces fully running applications



Testing the service with MIDAS

- Back to inital vision: massively scaling up test campaigns using MBT and cloud computing
- Serveral conceptual and technical problems, that must be resolved
- MIDAS European project: development of such a platform for testing of Web services based on the SOA, specifically the orchestration of multiple services
- Great complexity (exponencially growing with the number of services)

Testing the service with MIDAS – Setting up the infrastructure

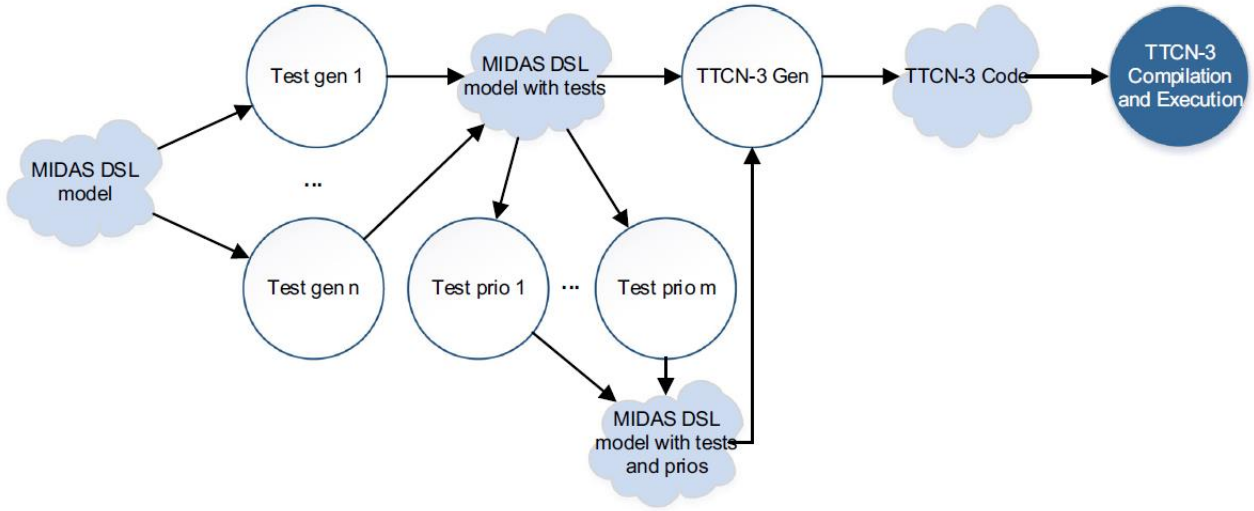
- PaaS solution based on an existing IaaS infrastrucutre
- SOAP interfaces based on Web Service Description Language (WSDL) provide:
  - User authentication
  - Object storage based file management
  - Accounting and billing
  - Definition of test campaigns
  - Test script compilation
  - Test execution

Testing the service with MIDAS – Setting up the infrastructure

- Major problem for many applications: many existing model-based tools as well as high-quality test execution engines are proprietary and require licensing
- These licenses are usually bound to a single machine
- Common problem for porting applications to the cloud, no general solution yet
- MIDAS project: TTCN-3 (Testing and Test Control Notation) compiler and execution engine
- Solution: agreement with the provider
- Could be a blueprint for the future

Testing the service with MIDAS – Model-based testing in the cloud

- Maximising flexibility and scalability -> each service defined as a closed unit
- Test services communicate via files (=serialized representations of the artifact that was generated)
- Figure: example for services deployed and their interactions

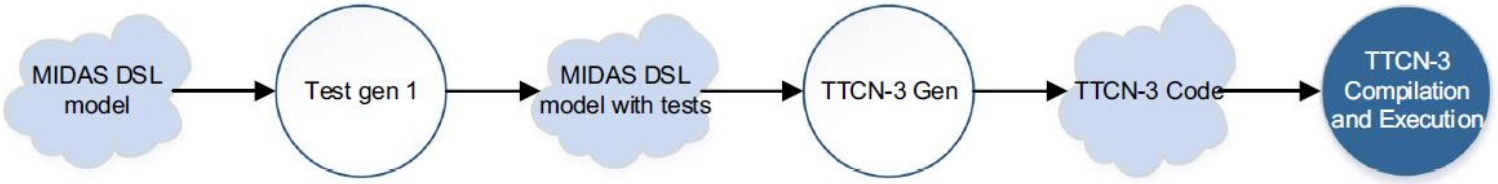


Testing the service with MIDAS – Model-based testing in the cloud

- Services defined as a single and isolated unit -> new services can be added without much effort
- Interchangeable parts
- Allows for the definition of a variety of test approaches

Testing the service with MIDAS – Definition of test campaigns

- Services are available -> looks like a SaaS or TaaS
- Testers only need to provide the required inputs:
  - a MIDAS DSL model (structural and deployment information about the SUT, behavioral model)
  - an orchestration for the services provided by MIDAS (which of the services provided by MIDAS are called with which files as input and in which order)
- One path through the data flow diagram
- Example of a concrete test campaign:



Conclusion

- How MBT as a potential solution to deal with the complexity of the Web service orchestration testing can be scaled up using cloud infrastructures
- MBT can generate a massive amount of tests to a large for execution on normal commodity test hardware
- Cloud computing's flexible elasticity and scaling mechanism together with its pay-per-use service model provides a solution that can be used to execute such a large amount of tests
- However, a testing solution requires a specialized cloud platform, as certain needs like licenses for test software are not within the portfolio of test providers. Within the MIDAS project a service solution for software testing was developed and explored.
- Other examples considering this idea: Test@Cloud Project, TravisCI, GitHub

Thank you for your attention!