Component-based Simulation Framework for Component Testing

Jaroslav Šnajberk, Tomáš Potužák, Richard Lipka
Department of Computer Science and Engineering
Faculty of Applied Sciences
University of West Bohemia, Pilsen, Czech Republic
Agenda

> Introduction
> Component frameworks
> Simulation framework description
> Simulation example
> Future work
System constructed from individual components
Components are blackbox entities that are part of third party composition
Components are deployed and executed in component framework
Component framework is implementation of component model
There are many component models and much more component frameworks
Component frameworks

- Manage communication and lifecycle of components
- Don’t support testing, statistics, scenarios
- Must be extended to test components

- OSGi component framework
  - Widely used in many fields of industry
- Spring DM extension
  - Simplifies the development process
Simulation framework for component testing

- A framework designed for testing of real components
  - No need for creation of models of the tested components
- Framework also constructed from components
  - Modularity of the framework
  - Easy to change the features of the simulation
- Utilization of SpringDM
Framework – Component types

- Components of the framework (Type 1)
  - Components necessary for functionality of the framework
- Real tested components (Type 2)
  - Components to be tested in the simulation framework
  - Components of a component based application
- Simulated components (Type 3)
  - Components for providing of the services required by the real tested components (type 2)
Framework – Framework components (type 1)

> Components, from which the framework is constructed
  > Functionality of the framework
  > Services for the simulation (e.g. logging)

> Calendar – most important component of the
  > Incorporates all events occurring during the simulation
  > Controls the progress of the simulation based on loaded scenario
Framework – Real components (type 2)

- Components to be tested in the framework for extrafunctional properties
- More than one real tested component in the simulation possible
  - Interaction with other real tested components
  - Interaction with their simulated environment (type 3 component, see next slide)
Framework – Simulated components (type 3)

> Components providing services required by the real tested components (type 2)
> Together with type 1 forming the environment for the real tested components to run
> No real functionality
  > The services provides predefined return values along with predefined delay
  > No real calculation of values
Framework – Real component interaction

> Two different requirements
  > The real components must not be aware they are in simulated environment
  > The simulation must be completely under control – all actions must be performed using calendar

> Services invocation:
  > By the real tested components (type 2) on the simulated components (type 3) performed using calendar
  > Between two real tested components performed using an intermediate components
Framework – Real component interaction II

A real component (type 2)  

A method invocation (foo())  

Returning of specified value  

A simulated component (type 3)  

suspend()  
add an event into the calendar  
wait for a specified time period  
resume()
Framework – Real component interaction III

Real component 1 (type 2) \(\xrightarrow{\text{A method invocation (foo())}}\) Intermediate \(\xrightarrow{\text{suspend()}}\) Real component 2 (type 2)

- A method invocation (foo())
- add an event into the calendar
- calculation of the return value
- resume()
Event based simulation vs. real time real tested components (type 2)

- Events necessary for the simulation → the real time for the real tested components must be simulated
- All Java API functions using the time must not return the time of the computer, but rather the “simulation” time

Similar problem with socket connections
- Must be also only simulated
Possible solutions to this problems:

- Override Java Core API methods directly in JRE
  - Problematic – Simulation framework also runs using the same JVM

- Create packages with overridden methods, replace imports in real tested components
  - More convenient
  - Manipulation with the bytecode of the component or using the Aspect Oriented Programming (AOP)
Framework – Simulation scenarios

- Description of the simulation
  - General description of the simulation – length of the simulation
  - Description of the environment (components and their connections)
  - Description of the activities (events)
  - Initial events of the calendar
  - (Return values of the simulated components (type 3))
- Loaded from a XML file(s)
General description of the simulation

- Structure of simulated application
  - Simulated and real components
  - Connections, their properties
  - Distribution of components
- Conversion of real and simulated time
  - Initial time
  - Length of simulation
Description of the environment

> Users
  > Expected activities in time
  > Generators or only prepared activities

> Required services from outside
  > Services provided by web applications of other sides
  > Services provided by hardware sensors etc.
Description of activities

- Activities of components within simulation
  - Source of activity / message
  - Target of message
  - Passed parameters
  - Time mark for event
Description of simulated components

- Generator-based components
  - Settings of generator – probability distributions, produced events

- Oracle-based components
  - Setting of input and output data
  - Setting of time for processing
Example of the simulation configuration
Future work

- Implementation of simulation framework
  - Three students are working on it
- Creating of scenarios
Thank you