# Adaptive monitoring for continuous performance model integration

# An Approach for adaptive monitoring for continuous performance model integration

# In this chapter, we will introduce our approach for adaptive monitoring for continuous performance model integration.

# Context of our approach

# As mentioned before, our approach is part of the CIPM vision (section 8) which extends the agile and DevOps process and provides them with iterative and incremental Performance Model. Moreover, the Performance Models in CIPM are enriched with Performance Model Parameters. In the following, we will briefly depict and describe the process and the context in which our approach takes place.

# Figure 1 shows the process in which our approach takes place. this process is based on Vitruvius (section 2) which means its elements are either models or transformations. The Java code in Figure 1 is represented by a JaMoPP model. When the developer commits changes on this model, two transformations will be triggered. The first one is the Coevolution process of Langhammer (section 5) which keeps the Source Code and the models in the Palladio Component Model consistent, mainly the repository and the SEFF Model. The second Transformation is our Transformation which is specified for keeping the Source Code and the Instrumentation Model consistent. We proposed the Instrumentation Model in order to persist the Probes that will be required for instrumenting the Source Code.

# When the system under development is deployed, our Instrumentation Process will be triggered. This process receives as inputs the Probes from the Instrumentation Model and the Source Code of the System. It delivers afterwards the instrumented Source Code as a JaMoPP Model. After the instrumentation process has been finished, the instrumented Source code will be executed and monitored. The information provided by monitoring are encapsulated in a Measurement Model which describes the needed monitoring records. After the monitoring has been finished the Parameters Estimation Process (section 9) will be triggered. it uses the information in the Measurement Model to estimate the Performance Model Parameters and updates accordingly the SEFF Model in the Palladio Component Model. Afterwards, the user can use the updated Performance Model to simulate and evaluate the performance of the system.

Change Analysis / Transformation

Instrument Source Code

Run Instrumented Application

Parameter Estimation

Simulation

Feedback

Developer

Change

Java Code

Instrumentation Model

Instrumented Code

Measurement Model

Palladio Component Model

Legend:

Application

Model

Changes / Triggers

Operator

Operate

Figure 1: Context of our approach

# Approach

# In this section we will introduce the main activities of our approach. These activities will be described in details in the next chapters.

# Terminologies

# DevOps

# DevOps is a development process that aims to integrate

# Instrumentation Model

# 

**Figure 2: Instrumentation Model**

# Iterative Instrumentation

**Development**

Java Code

Generate Probes

Instrumentation Model

**Continuous Deployment**

Instrument the System

Instrumented System

**Monitoring**

Run the Instrumented System

Measurement Model

Developer

Commit

**Figure 3: Overview of our Approach Activities**

References

[1] Andreas Brunnert et al. Performance-oriented DevOps: A Research Agenda. Tech. rep. SPEC-RG-2015-01. SPEC Research Group - DevOps Performance Working Group, Standard Performance Evaluation Corporation (SPEC), Aug. 2015