A Model-based Framework for System Configuration Management

Azadeh Jahanbanifar  
PhD Candidate  

Jan 31st, 2015  
Concordia/Ericsson
The MAGIC (Modeling and Automatic Generation of Information for Configuration and Upgrade of HA Systems) was formed in 2007 as a collaborative project between Concordia University and Ericsson Canada.

The objective of the project summarizes as:
- Development of methods, frameworks, architectures for the (re-)configuration, deployment, upgrade, and monitoring of complex software systems
- Handle the performance, dependability and availability of such systems and their services.

The research and experiments were initiated in the context of the Service Availability Forum middleware (e.g. OpenSAF) and it is being extended to the context of cloud computing.

The methodology used in the project is based on the principles of Model Driven Engineering (MDE).
Outline

- Introduction
- Goals
- Challenges
- An application domain: SA Forum middleware
- System configuration generation
  - Mapping between the domain models
  - Weaving the configuration models
- Consistency preservation of configuration at runtime
- Summary
Introduction

- A system consists of various inter-related aspects and/or subsystems (e.g. software, platform, security, …)

- Each aspect is managed separately by a specific management service/subsystem.

- A service configuration is a logical representation of the system entities under the management of that service.

- An integration of all service configurations form a system configuration.
Goals

- Different services of a system require configurations each of which specifies the organization and the characteristics of the entities under their control.
- How to generate (consolidate) a system configuration automatically based on configuration fragments?
Challenge? Overlapping Elements

- A physical entity may have multiple representations in different configuration fragments.
- The relation between these overlapping elements is critical to avoid inconsistencies.
In this work the Service Availability Forum middleware is used as the context for explaining the proposed framework for system configuration management.

The middleware defines a set of services collaborating to facilitate the provisioning of high availability for the applications under its management.

Middleware services considered in our work are:

- Platform Management (PLM)
- Availability Management Framework (AMF)
Mapping Between Configurations:

- How to map the AMF Nodes to the PLM EEs to reduce the impact of hardware failure for software entities?
Mapping between the PLM and AMF Domain Models
System Configuration Generation

- Model weaving technique can help us to establish the links between the input profiles and the target profile.
- The mappings between the models can be captured in a weaving model that conforms to a Weaving metamodel which describes the link types.
Extending the Weaving Metamodell
An Example of Using the Link Type

DisjointGrouping:

- **<Source>**
  - **<LeaderElement>** (e.g. PlmEEVM)
  - **<FollowerElement>** (e.g. AmfNode)

- **<Target>**
  - **<TargetElement>** (e.g. AmfSU)
Extracting System Constraints

- The consistency rules of the models can be defined with constraints.
- The constraints (pertaining to the model integration) can be retrieved from the transformation rules as these rules are implicitly considering these constraints.
We can define different roles (Leaders, Followers and Peers) for the constrained elements of the constraints.

This means that changes in the leader elements can impact the followers but not the other way. Or peer elements can impact other peers in a constraint.

For the other constraints related to the configuration domains, the leadership info should be defined by the configuration designer.
Constraint Extraction

LeadedLink A:
-SahSource-
-<LeaderElement>-<FollowerElement>-<Target>-<TargetException>

TransformationRule A:
-From: LeaderElements, FollowerElements
-To: TargetElement

Constraint A:
-OCL Expression
-LeadershipInfo: Leader/Follower
Consistency Preservation of the Configuration

- Administrator/management applications may request to modify the configuration at runtime.

- Changing the configuration elements related to one service may affect the other services using the same or related elements.

- Can we control the side-effects and the propagation of the changes in the system configuration?
  - We keep consistency by validating the change requests against the system constraints.
Validation and Auto-adjustment

Administrator/Management Apps

- Modification Request
- Illegal Modification
  Reject Request

Constraint model

Configuration Generation Rules

Reject Modifications
Modifications are not Adjustable

Validator

- Legitimacy Check
- Integrity Check

System Configuration

- Passed the Consistency Checks
- Need for Adjustments
- Violating the Consistency Checks

Auto Adjustment Engine

- Adjusted Modifications
- Apply the Modifications with Adjustments
- Apply the Modification

Administrator/Management Apps

Constraint model uses

Configuration Generation Rules uses

Administrator/Management Apps uses
Research Summary

Configuration
Fragment A

Configuration
Fragment B

Configuration
Fragment C

Configuration Integration
- Mapping definition
- Model Integration (Weaving)

System Configuration

Consistency Constraints

Configuration Auto-adjustment

Modification Request

The System Configuration Generation

Maintaining the Consistency of the System Configuration

Done
In Progress
Not Started
Thank You!