A Dynamic Configuration Architecture

Finn Arve Aagesen
Dept. of Telematics, Norwegian University of Science and Technology, Norway
aagesen@item.ntnu.no

Chutiporn Anutariya
Computer Science Program, Shinawatra University, Thailand
chutiporn@shinawatra.ac.th

Mazen Malek Shiaa, Bjarne E. Helvik and Paramai Supadulchai
Dept. of Telematics, Norwegian University of Science and Technology, Norway
{malek, helvik, paramai}@item.ntnu.no

Network-based services have during more than one decade been is an important research topic with a focus on service architecture solutions that give flexibility and efficiency in the definition, deployment and execution of the services. This focus is now slightly changing into focus on adaptability and evolution of such services. Due to the increase in both heterogeneity and complexity in today's networking, wide-area distributed computing and service provision systems, there arises a demand for a platform with functionalities beyond existing solutions.

The paper develops a formal framework for dynamic configuration and reconfiguration of services in *TAPAS*: *Telematics Architecture for Plug-and-Play Systems* (http://tapas.item.ntnu.no). It provides representation, computation and reasoning mechanisms for semantic description and matching of required and offered capabilities and status which are required by a particular service system. It employs *CIM* and recently developed languages for the Semantic Web (*RDF(S)* and DAML languages) in order to provide a mechanism for human-readable and machine-comprehensible descriptions of status, capabilities, system (re)configuration plans as well as the exchanging messages. In addition, it exploits *XML Declarative Description* (*XDD*) theory—an expressive XML rule-based, knowledge representation—to seamlessly unify such various languages into a single uniform formalism, hence allowing the integration, extraction, computation of and reasoning with instances/objects of those different languages. It permits formal definitions of application-specific configuration requirements and constraints as well as reconfiguration policies for handling particular events in terms of *XML clauses*. Reasoning about these definitions and the available capabilities/status of nodes in the system yields appropriate (re)configuration plans for composition of new services to be installed and for adaptation of the currently executing services.

The proposed architectural framework for dynamic configuration as depicted by Figure 1 comprises the following main entities:

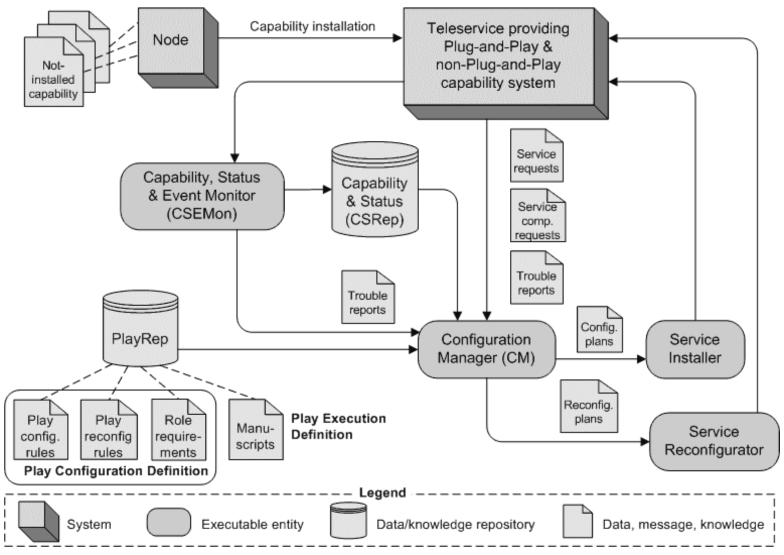

Figure 1: Dynamic Configuration Architecture

1. *Capability & Status Repository* (*CSRep*) stores specifications of capabilities offered by components in a system and maintains information reflecting the situation and status of the system at a particular time.
2. *Play Repository* (*PlayRep*) is a collection of:
   (i) *Role requirements* identify, for each role, capability and status requirements.
   (ii) *Play configuration rules* describe system configuration rules and constraints which must always be maintained.
   (iii) *Play reconfiguration rules* define application-specific reconfiguration policies for handling significant reconfiguration-related events.
   (iv) *Play execution definitions* define the entire functional behaviour of each role which includes also the interaction and cooperation with other roles.
3. *Capability, Status & Event Monitor* (*CSEMon*) monitors system capabilities/ status and maintains the CSRep.
4. *Configuration Manager* (*CM*) is the primary entity which reasons about the current system's capabilities & status, services' requirements and reconfiguration policies in order to dynamically generate appropriate service (re)configuration. In particular, it is responsible for: (i) Generation of appropriate configurations; (ii) Determination of a location for executing a particular role; and (iii) Computation of dynamic reconfiguration schemes.
5. *Service Installer* installs a service into the system by creating corresponding actors for execution of certain roles according to an obtained play configuration.
6. *Service Reconfigurator* initiates and performs service reconfiguration.

To verify the framework's feasibility and potential in real applications, a prototype system (http://tapas.item.ntnu.no/ipm) has been implemented using XET reasoning engine. Its integration into the *TAPAS platform* in order to provide a basis for experiments with dynamic configuration management is in progress.