RESUME

Of the Master Thesis

System for registration and management of JNDI service providers

Student: Elitsa Nikolova Pancheva, FN M-21360, MSc program: Computer science – Distributed systems and mobile technologies

Scientific coordinator: Boyan Bontchev, Assoc. Prof., PhD

Date: 24th October 2007, Sofia

Key words: Java, JNDI, SPI, classloading, naming service, registration, management, system

Master Thesis Annotation: Naming service broadly refers to a mechanism for associating an object with a specified name and later searching the object by this name. Java programming language supports the development and maintenance of naming services with the means of the JNDI: Java Naming and Directory Interface specification.

The SPI (Service Provider Interface) part of the JNDI specification facilitates the development of naming service providers and makes the services they offer accessible for the client applications through JNDI API (Application Programming Interface).

The master thesis paper studies the restrictions set by the JNDI specification presents the issues provoked by these restrictions and proposes a way to overcome them by using a system for registration and management of naming service providers. This system is designed as an extension to the JNDI SPI framework and embodies the following characteristics:

- Provides a way for locating and loading jndi services implementations which does not depend on the client's classloader, thus releasing the naming service providers from the restriction – their implementation classes to be mandatory available to the client's classloader. This solution makes possible the free usage of JNDI services in the classloader isolation enabled Java Enterprise Server Systems.

- Provides the possibility for every naming service provider to choose and implement its own mechanism for locating and loading the classes of the jndi implementations it offers. This way the restriction that only the first naming provider which succeeds to register in the JNDI framework gets this opportunity is suppressed.

- Provides the clients of the naming services the possibility to use these services with the new features enabled through the JNDI API, i.e. the system we offer is fully compatible and integrated with the JNDI SPI and can be accessed in a common way by using JNDI API.

The preface of the paper includes a brief overview of the technologies used in the development process - Java, JNDI, Java EE Server, naming service, semantic object.

The interactions and integration among them are described to build more global picture of the examined area.

A detailed investigation is performed to examine the means of integration of the naming providers in the standard JNDI framework. The semantics of the most common jndi operations (get initial context, lookup, and bind) is described along with the central role of the javax.naming.spi.NamingManager class in it. As a result of the performed functionality analysis the restrictions and omissions of the JNDI framework are identified and later used to form the functional requirements for the developed module.

The test-driven development methodology is naturally chosen to be used in the JNDI SPI extension module realization. According to its principles a set of automated JUnit tests are designed and implemented in order to validate the satisfaction of the functionality requirements, the efficiency and performance of the developed system and its compatibility against the standard functionality of the JNDI framework.

The architecture and the actual implementation of the system for registration and management of jndi services are described in details along with the optimizations and enhancements made as a result of the validation tests outcome.
The system is developed only with the means of standard technologies – JDK 5 and JNDI library version 1.2.1 included in the standard jdk distribution. It can be used standalone or integrated in jndi enabled enterprise server architecture.

The closing part of the paper contains a user manual describing the ways and prerequisites to use the newly developed system by the naming service providers and their clients. It also comments the compatibility of the developed system with the standard functionality of the JNDI framework and the migration of already existing applications towards the new system. The possible further enhancements are also discussed.