



Faculty of Sciences,
Technologie
and Communication

Application Layer Service's Description in RESIST Architecture

Cecilia Manzino
January 23, 2008

About the Speaker

- I'm doing a master in Computer Sciences at the University of the Republic, Uruguay.
- Since August, I'm doing an internship in the RESIST project advised by Marcos Da Silveira and Nicolas Guelfi.

Outline

- 1 Introduction
 - The Soa Paradigm
 - Web Services
 - The architecture of RESIST
- 2 A design method for describing utility services
 - Service Profiles
 - Service Factoring
 - Service's Descriptions
- 3 Conclusions

Outline

1

Introduction

- The Soa Paradigm
 - Web Services
- The architecture of RESIST

2

A design method for describing utility services

- Service Profiles
- Service Factoring
- Service's Descriptions

3

Conclusions

The Soa Paradigm

The need of an evolution of software architecturing

- With object oriented programming languages dynamicity was introduced at the process level, but changes were still dealt statically at the product level.
- Another big leap was the need for decentralizing the responsibility to provide some needed functionality to pre-existing components, and the ability of reusing previously developed subsystems into new systems.

The first ideas

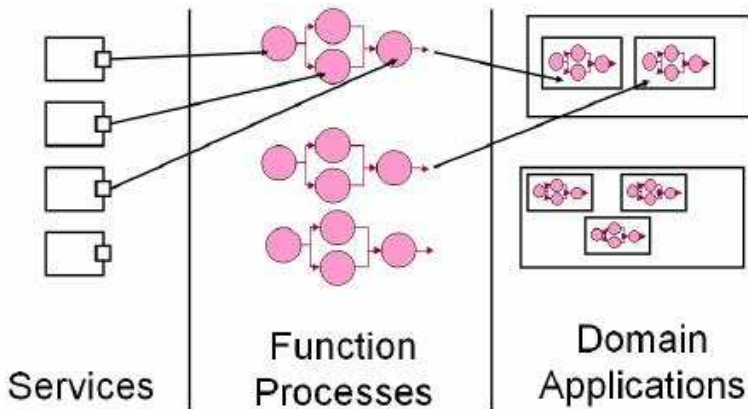
- The evolution of software development has been driven to increase degrees of dynamicity, decentralization, and decoupling.
- “We are moving from a situation where application development is mostly done from scratch, to a situation where components are integrated, maybe dynamically, to form new applications.”[C. Ghezzy and P. Picco]

SoC and SOA

Service-Oriented Computing (SoC) is a new computing paradigm, the visionary promise of it is a world of cooperating services where application components are assembled with little effort into a network of services that can be loosely coupled to create flexible dynamic business processes and agile applications. [Leymann 2005]

Service Oriented Architecture (SOA): is an architectural style that represents a model in which functionality is decomposed into small, distinct units (services), which can be distributed over a network and can be combined together and reused to create business applications.

SOA basic concepts



Web Services

Web Services are the current most promising technology based on the concept of SoC. This technology uses:

- 1 The internet or a variety of other networks as the communication medium.
- 2 Open Internet-based standards protocols as transmission medium, such as (SOAP).
- 3 A Web Service Description Language for interface's service definitions, such as (WSDL).
- 4 The Business Process Execution Language (BPEL) for orchestrating the services.

Web Services

Web Services are the current most promising technology based on the concept of SoC. This technology uses:

- 1 The internet or a variety of other networks as the communication medium.
- 2 Open Internet-based standards protocols as transmission medium, such as (SOAP).
- 3 A Web Service Description Language for interface's service definitions, such as (WSDL).
- 4 The Business Process Execution Language (BPEL) for orchestrating the services.

Web Services

Web Services are the current most promising technology based on the concept of SoC. This technology uses:

- 1 The internet or a variety of other networks as the communication medium.
- 2 Open Internet-based standards protocols as transmission medium, such as (SOAP).
- 3 A Web Service Description Language for interface's service definitions, such as (WSDL).
- 4 The Business Process Execution Language (BPEL) for orchestrating the services.

Web Services

Web Services are the current most promising technology based on the concept of SoC. This technology uses:

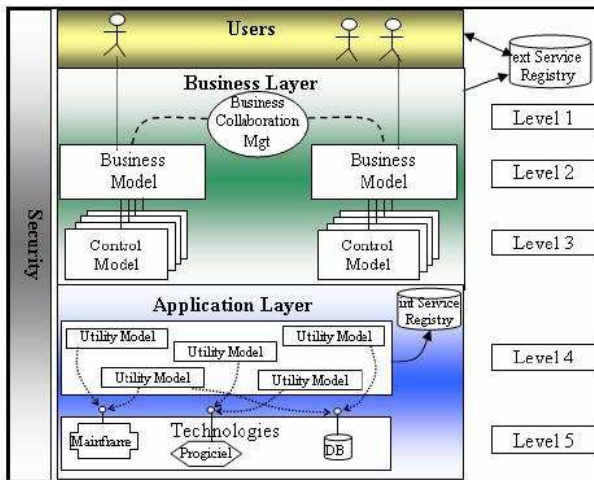
- 1 The internet or a variety of other networks as the communication medium.
- 2 Open Internet-based standards protocols as transmission medium, such as (SOAP).
- 3 A Web Service Description Language for interface's service definitions, such as (WSDL).
- 4 The Business Process Execution Language (BPEL) for orchestrating the services.

Web Services

Web Services are the current most promising technology based on the concept of SoC. This technology uses:

- 1 The internet or a variety of other networks as the communication medium.
- 2 Open Internet-based standards protocols as transmission medium, such as (SOAP).
- 3 A Web Service Description Language for interface's service definitions, such as (WSDL).
- 4 The Business Process Execution Language (BPEL) for orchestrating the services.

The architecture of Resist



Outline

- 1 Introduction
 - The Soa Paradigm
 - Web Services
 - The architecture of RESIST
- 2 A design method for describing utility services
 - Service Profiles
 - Service Factoring
 - Service's Descriptions
- 3 Conclusions

A design method for describing services

The following steps guide the definition of an utility service. A service will be described [E-Framework]:

- 1 As a functional definition using natural language and predicate calculus for modeling constrains and the environment.
- 2 As an abstract model of data and behaviour, using UML
- 3 As a web service specification, using WSDL-S.

Service Profiles

Functional Definition

- Identification
 - Name
 - Id
- Functionality
 - Tasks
 - Name
 - Description
 - Input
 - Output
 - Precondition
 - Postcondition
 - Provider
 - Data
 - Security Policies
- Description

Abstract model of data and behaviour

- A **Class Diagram** will give an structural view of the service, which illustrates the attributes and methods of each class that represents one service.
- **Activity Diagrams** will describe the controlflow of the most complex tasks.
- **State Machine Diagrams** will illustrate the state evolution of part of the system.

Web service specification

This part contains the description of the Web service interface, also called the abstract description of the service, as a WSDL-S document.

An interface's definition of a service must contains all the information necessary to invoke this, and it must be independent of implementation details.

Service Factoring

An scenario is presented in a narrative description and use-case diagrams can be used to illustrate it.

Scenario 1: *“It concerns a patient who can start a measurement remotely. The application asked for the login and password of the patient. When the login data was introduced it checks the identity and then verify if the sensors are connected and working. If the sensors are not working a procedure to activate them are indicated in the screen. When the sensors are connected the software interface shows two buttons for start and cancel the measurement. After a succesfully measurement the patient can validate it and re-start the measurement. If the journey of the measurement wasn't respected by a patient the HMC can start some kind of alert.”*

Service Factoring

We distinguish the following application services:

- Identification management service
- Measurements management service
- Sensors data read service
- Journey service
- Alarm service

Service's Descriptions

Functional Definition

- **Identification:**
 - **Name:** Identification management
 - **Id:** UM01
- **Description:** The purpose of this service is to provide authentication and authorization to users. Authentication is the process of uniquely identifying an individual based on the credentials provided by him. This service use as credentials a username and password. Authorization is the process of establishing what a principal is permitted to do. This service is used for managing the authorizations for executing service's task.

● Functionality:

● Tasks:

- **Name:** Verify_user_identity
- **Description:** This task accepts an username and a password of a user and validates them. It stores the session id of the user in the databases and return success, if validation succeeded and failure and reason for failure, if validation failed.
- **Input:** username, password
- **Output:** validation, failure_login_reason
- **Precondition:** *True*
- **Postcondition:**

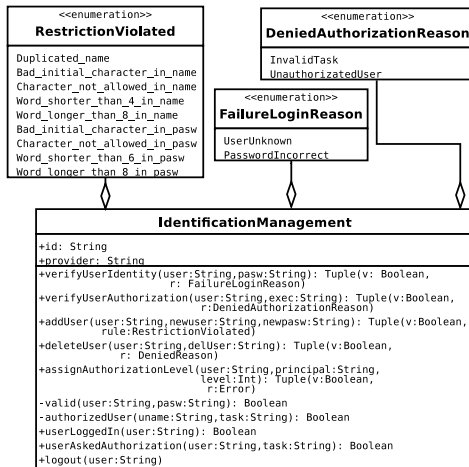
$$\begin{aligned} & (valid(username, password) \Rightarrow UserLoggedIn(username) \\ & \wedge validation) \wedge (\neg valid(username, password) \Rightarrow \\ & \neg UserLoggedIn(username) \wedge \neg validation) \end{aligned}$$

Data:

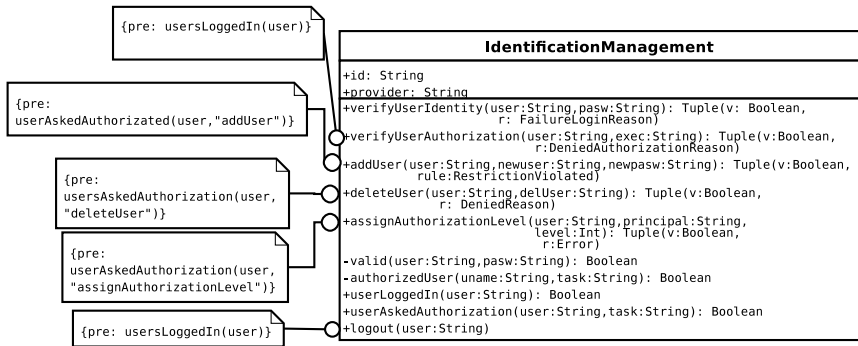
- **username:** is an authentication data of a user used for the purposes of identification.
- **password:** is a secret authentication data of a user used for the purposes of identification.
- **failure_login_reason:** this data can be any of the following values `user_unknown` or `incorrect_password`, and represents reasons why the identification of a user wasn't accepted.
- **principal:** this data represents an end user, a service, a server, a group or an organization that may be assigned a level of authorization.
- **identifier:** is a unique object used to represent a user once authenticated.
- **service_task:** is a tuple with the name of a task and the identifier of a service.
- **denied_authorization_reason:** this data can be any of the following values `invalid_task` or `unauthorized_user`, and represents reasons why authorization has been denied.

Abstract model of data and behaviour

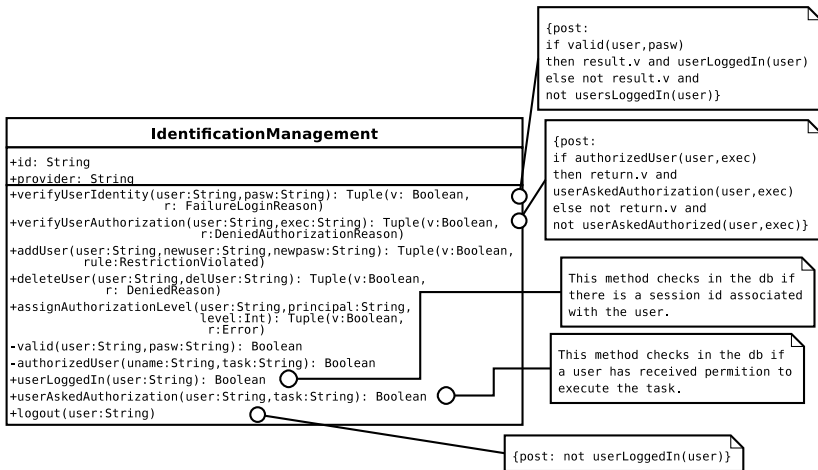
Class Diagram



Class Diagram (2)

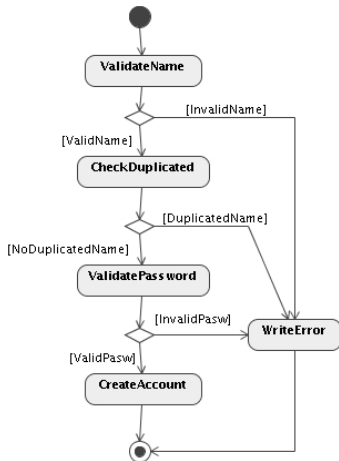


Class Diagram (3)



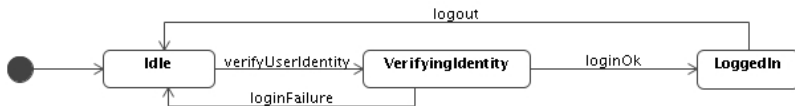
Activity Diagram

The task "Add User"



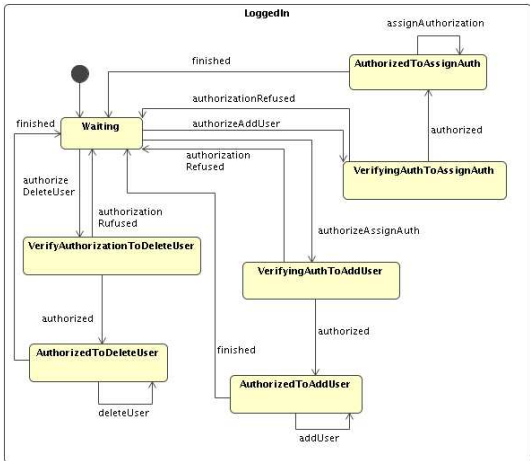
State Machine Diagrams

The top-level state machine diagram



State Machine Diagrams

The LoggedIn state



Web service specification

```
<definitions name="IdentificationManagementService"
targetNamespace="xs:anyURI"
xmlns:credential="http://www.csl.sri.com/users/denker/
owl-sec/credential.owl#>
```

```
<documentation> The purpose of this service is to
provide the operations needed for identification and
authorization of users </documentation>
```

```
<simpleType name='VerifyUserIdentityOutput'>
  <restriction base='string'>
    <enumeration value='UserUnknown' />
    <enumeration value='PasswordIncorrect' />
    <enumeration value='UserLoggedInSuccessfully' />
  </restriction>
</simpleType>
```

Web service specification (2)

```
<interface name="Authenticate">
  <operation name="VerifyUserIdentity">
    <input messageLabel="UserName&Password"
      element="credential:LoginWithPassphrase"/>
    <output messageLabel="Validation&FailureLoginReason"
      element="tns:VerifyUserIdentityOutput"/>
  </operation>
</interface>
...
</definitions>
```

Outline

- 1 Introduction
 - The Soa Paradigm
 - Web Services
 - The architecture of RESIST
- 2 A design method for describing utility services
 - Service Profiles
 - Service Factoring
 - Service's Descriptions
- 3 Conclusions

Conclusions

- The state of the art of SOA based designs was presented.
- The RESIST architecture proposition was introduced.
- A design method for describing web-services, based on the Soa paradigm, was addressed.
- This method was applied in a real case study named “Remote monitoring of system of cardiovascular disease”.

Future Work

- Implement the services.
- Detail the analysis model, according to MDE.
- Define how to publish the services.
- Define how to use the proposed model in the business layer.
- Improve the security in the architecture of RESIST.

Questions?

Cecilia Manzino