DIP
Data, Information and Process Integration with Semantic Web Services
FP6 - 507483

Deliverable

WP 10: Case study eBanking

D 10.5
Mortgage Comparison Service

Silvestre Losada
Jordi Ribas
Jesús Contreras
José Luís Bas
Sergio Bellido
José Manuel Gómez
Richard Benjamins
Ignacio González

July 28th, 2005
SUMMARY

This deliverable contains the software description of the DIP mortgage comparison service implementation available on http://comparador.isoco.com/ and specified in deliverable D10.2. Deliverable 10.5 contains. This document describes each software component as well as installation instructions and licensing information.

The deliverable contributes to the goals of DIP by providing a concrete application based on Semantic Web Services in the financial domain.

This deliverable provides an insight in how the prototype uses the functionalities proposed by the DIP infrastructure: discovery and execution.

The target audience of this deliverable is as follows: the partners who are developing tools, and external readers who are interested in finding information about a real SWS implementation.

The application can be accessed at: http://comparador.isoco.com/

Disclaimer: The DIP Consortium is proprietary. There is no warranty for the accuracy or completeness of the information, text, graphics, links or other items contained within this material. This document represents the common view of the consortium and does not necessarily reflect the view of the individual partners.
Document Information

<table>
<thead>
<tr>
<th>IST Project Number</th>
<th>FP6 – 507483</th>
<th>Acronym</th>
<th>DIP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full title</td>
<td>Data, Information, and Process Integration with Semantic Web Services</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="http://dip.semanticweb.org">http://dip.semanticweb.org</a></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Document URL</td>
<td>EU Project officer</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Kai Tullius</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Deliverable Number</th>
<th>10.5</th>
<th>Title</th>
<th>Mortgage Comparison Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work package Number</td>
<td>10</td>
<td>Title</td>
<td>eBanking Use Case</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Date of delivery</th>
<th>Contractual</th>
<th>M18</th>
<th>Actual</th>
<th>27-July-05</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>version. 0.07</td>
<td>final</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nature</td>
<td>Prototype ☑ Report ☐ Dissemination ☑ Ontology ☐</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dissemination Level</td>
<td>Public ☑ Consortium ☐</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Authors (Partner)</th>
<th>Silvestre Losada (iSOCO), Jordi Ribas,(iSOCO), Jesus Contreras,(iSOCO), Jose Luis Bas (Bankinter), Sergio Bellido (Bankinter), Richard Benjamins (iSOCO), Ignacio Gonzalez (iSOCO)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Responsible Author</td>
<td>Silvestre Losada Email <a href="mailto:slosada@isoco.com">slosada@isoco.com</a></td>
</tr>
<tr>
<td>Partner</td>
<td>(iSOCO) Phone +34913349744</td>
</tr>
</tbody>
</table>

Abstract (for dissemination) The deliverable contributes to the goals of DIP by providing a concrete application based on Semantic Web Services in the financial domain. This deliverable provides an insight in how the prototype uses the functionalities proposed by the DIP infrastructure: discovery and execution. And provides a real SWS implementation.

Keywords SWS, Financial, Mortgage.

Version Log

<table>
<thead>
<tr>
<th>Issue Date</th>
<th>Rev No.</th>
<th>Author</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-june-05</td>
<td>001</td>
<td>Silvestre Losada</td>
<td>Initial Version</td>
</tr>
<tr>
<td>15-june-05</td>
<td>002</td>
<td>Jordi Ribas</td>
<td>Mortgage comparison service.</td>
</tr>
</tbody>
</table>

Deliverable 10.5 iii 28/07/2005
<table>
<thead>
<tr>
<th>Date</th>
<th>No.</th>
<th>Name</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-june-05</td>
<td>003</td>
<td>Jordi Ribas</td>
<td>Invocation module</td>
</tr>
<tr>
<td>22-june-05</td>
<td>004</td>
<td>Ignacion gonzalez</td>
<td>Licensing</td>
</tr>
<tr>
<td>23-june-05</td>
<td>005</td>
<td>JL Bas</td>
<td>Introduction</td>
</tr>
<tr>
<td>24 june-05</td>
<td>006</td>
<td>Richard Benjamins</td>
<td>Some explanations.</td>
</tr>
<tr>
<td>20 july-05</td>
<td>007</td>
<td>Silvestre Losada</td>
<td>Review updates.</td>
</tr>
</tbody>
</table>
# Project Consortium Information

<table>
<thead>
<tr>
<th>Partner</th>
<th>Acronym</th>
<th>Contact</th>
</tr>
</thead>
</table>
| National University of Ireland Galway | NUIG | Prof. Dr. Christoph Bussler  
Digital Enterprise Research Institute (DERI)  
National University of Ireland, Galway  
Galway  
Ireland  
Email: chris.bussler@deri.org  
Tel: +353 91 512460 |
| Fundacion De La Innovacion.Bankinter | Bankinter | Monica Martinez Montes  
Fundacion de la Innovation. Bankinter  
Paseo Castellana, 29  
28046 Madrid,  
Spain  
Email: mmtnez@bankinter.es  
Tel: 916234238 |
| Berlecon Research GmbH | Berlecon | Dr. Thorsten Wichmann  
Berlecon Research GmbH  
Oranienburger Str. 32  
10117 Berlin,  
Germany  
Email: tw@berlecon.de  
Tel: +49 30 2852960 |
| British Telecommunications Plc. | BT | Dr John Davies  
BT Exact (Orion Floor 5 pp12)  
Adastral Park Martlesham  
Ipswich IP5 3RE,  
United Kingdom  
Email: john.nj.davies@bt.com  
Tel: +44 1473 609583 |
| Swiss Federal Institute of Technology, Lausanne | EPFL | Prof. Karl Aberer  
Distributed Information Systems Laboratory  
École Polytechnique Fédérale de Lausanne  
Bât. PSE-A  
1015 Lausanne, Switzerland  
Email: Karl.Aberer@epfl.ch  
Tel: +41 21 693 4679 |
| Essex County Council | Essex | Mary Rowlett,  
Essex County Council  
PO Box 11, County Hall, Duke Street  
Chelmsford, Essex, CM1 1LX  
United Kingdom.  
Email: maryr@essexcc.gov.uk  
Tel: +44 (0)1245 436524 |
| Forschungszentrum Informatik | FZI | Andreas Abecker  
Forschungszentrum Informatik  
Haid-und-Neu Strasse 10-14  
76131 Karlsruhe  
Germany  
Email: abecker@fzi.de  
Tel: +49 721 9654 0 |
<table>
<thead>
<tr>
<th>Partner</th>
<th>Acronym</th>
<th>Contact</th>
</tr>
</thead>
</table>
| Institut für Informatik, Leopold-Franzens | UIBK    | Prof. Dieter Fensel  
Institute of computer science  
University of Innsbruck  
Technikerstr. 25  
A-6020 Innsbruck, Austria  
Email: dieter.fensel@deri.org  
Tel: +43 512 5076485 |
| Universität Innsbruck                    |         |                                                                         |
| ILOG SA                                  | ILOG    | Christian de Sainte Marie  
9 Rue de Verdun, 94253  
Gentilly, France  
Email: csma@ilog.fr  
Tel: +33 1 49082981 |
| inubit AG                                | Inubit  | Torsten Schmale  
inubit AG  
Lützowstraße 105-106  
D-10785 Berlin  
Germany  
Email: ts@inubit.com  
Tel: +49 30726112 0 |
| Intelligent Software Components, S.A.    | iSOCO   | Dr. V. Richard Benjamins, Director R&D  
Intelligent Software Components, S.A.  
Pedro de Valdivia 10  
28006 Madrid, Spain  
Email: rbenjamins@isoco.com  
Tel. +34 913 349 797 |
| NIWA WEB Solutions                       | NIWA    | Alexander Wahler  
NIWA WEB Solutions  
Niederacher & Wahler OEG  
Kirchengasse 13/1a  
A-1070 Wien  
Email: wahler@niwa.at  
Tel:+43(0)1 3195843-11 |
| The Open University                      | OU      | Dr. John Domingue  
Knowledge Media Institute  
The Open University, Walton Hall  
Milton Keynes, MK7 6AA  
United Kingdom  
Email: j.b.domingue@open.ac.uk  
Tel.: +44 1908 655014 |
| SAP AG                                   | SAP     | Dr. Elmar Dörner  
SAP Research, CEC Karlsruhe  
SAP AG  
Vincenz-Priessnitz-Str. 1  
76131 Karlsruhe, Germany  
Email: elmar.dorner@sap.com  
Tel: +49 721 6902 31 |
<table>
<thead>
<tr>
<th>Company</th>
<th>Contact Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sirma AI Ltd.</td>
<td>Atanas Kiryakov, Ontotext Lab, - Sirma AI EAD</td>
</tr>
<tr>
<td></td>
<td>Office Express IT Centre, 3rd Floor</td>
</tr>
<tr>
<td></td>
<td>135 Tzarigradsko Chausse</td>
</tr>
<tr>
<td></td>
<td>Sofia 1784, Bulgaria</td>
</tr>
<tr>
<td></td>
<td>Email: <a href="mailto:atanas.kiryakov@sirma.bg">atanas.kiryakov@sirma.bg</a></td>
</tr>
<tr>
<td></td>
<td>Tel.: +359 2 9768 303</td>
</tr>
<tr>
<td>Unicorn Solution Ltd.</td>
<td>Jeff Eisenberg</td>
</tr>
<tr>
<td></td>
<td>Unicorn Solutions Ltd,</td>
</tr>
<tr>
<td></td>
<td>Malcha Technology Park 1</td>
</tr>
<tr>
<td></td>
<td>Jerusalem 96951</td>
</tr>
<tr>
<td></td>
<td>Israel</td>
</tr>
<tr>
<td></td>
<td>Email: <a href="mailto:Jeff.Eisenberg@unicorn.com">Jeff.Eisenberg@unicorn.com</a></td>
</tr>
<tr>
<td></td>
<td>Tel.: +972 2 6491111</td>
</tr>
<tr>
<td>Vrije Universiteit Brussel</td>
<td>Carlo Wouters</td>
</tr>
<tr>
<td></td>
<td>Starlab- VUB</td>
</tr>
<tr>
<td></td>
<td>Vrije Universiteit Brussel</td>
</tr>
<tr>
<td></td>
<td>Pleinlaan 2, G-10</td>
</tr>
<tr>
<td></td>
<td>1050 Brussel ,Belgium</td>
</tr>
<tr>
<td></td>
<td>Email: <a href="mailto:carlo.wouters@vub.ac.be">carlo.wouters@vub.ac.be</a></td>
</tr>
<tr>
<td></td>
<td>Tel.: +32 (0) 2 629 3719</td>
</tr>
</tbody>
</table>

Mortgage Comparison Service
LIST OF KEY WORDS/ABBREVIATIONS

SOAP  Simple Object Access Protocol
SWS   Semantic Web Services
UDDI  Universal Discovery, Description and Integration
WSDL  Web Service Description Language
XML   eXtensible Markup Language
API   Application Programming Interface
TABLE OF CONTENTS

SUMMARY ......................................................................................................................... II
LIST OF KEY WORDS/ABBREVIATIONS ....................................................................... VIII
TABLE OF CONTENTS ...................................................................................................... IX
1 INTRODUCTION .............................................................................................................. 1

2 OVERVIEW MORTGAGE COMPARISON SERVICE .................................................... 1

2.1 Story board .................................................................................................................. 2

3 COMPONENTS AND ARCHITECTURE ............................................................................ 10

3.1 Comparator a Mortgage Comparison service ............................................................ 11

3.1.1 Current version & status ...................................................................................... 11

3.1.2 Requirements ........................................................................................................ 12

3.1.3 External systems: ................................................................................................... 12

3.1.4 External Libraries; ................................................................................................. 12

3.2 Discovery .................................................................................................................... 13

3.2.1 Current version and status ................................................................................... 13

3.2.2 Requirements ........................................................................................................ 15

3.2.3 External systems: XSB/Flora-2 ............................................................................. 15

3.2.4 External Libraries .................................................................................................. 15

3.2.5 Licensing ............................................................................................................... 16

3.2.6 Installation of discovery component .................................................................... 16

3.3 Invocation module ...................................................................................................... 16

3.3.1 Current version and status ................................................................................... 16

3.3.2 Requirements ........................................................................................................ 17

3.3.3 External Libraries ................................................................................................. 17

3.3.4 Licensing ............................................................................................................... 17

3.3.5 Installation & usage ............................................................................................... 17

4 CONCLUSIONS .............................................................................................................. 17

REFERENCES .................................................................................................................. 18

LIST OF FIGURES

Figure 1: Mortgage Comparison service overview. ............................................................ 1
Figure 2: Screenshot of mortgage comparison service. ..................................................... 3
Figure 3: Goal example generated by mortgage comparison service ............................... 5
Figure 4: List of web services that match with a given goal. ............................................ 6
Mortgage Comparison Service

Figure 5: Screenshot with the service returned by discovery................................. 7
Figure 6: Screenshot of mortgage comparison service results.............................. 8
Figure 7: System architecture.................................................................................. 10
Figure 8: Screenshot with results screen................................................................. 12

LIST OF TABLES
Table 1: Translation to English of terms from the application interface..................... 3
Table 2: Inputs outputs mortgage comparison service:............................................. 9
Table 3: Discovery API............................................................................................... 13
Table 4: Discovery WSDL.......................................................................................... 13
Table 5: Invocation Module Method Summary......................................................... 16
1 INTRODUCTION

The objective of this document is to describe the prototype of a financial application (the mortgage comparator/simulator) based on SWS (Semantic Web Services). As described in D10.1 [1], and D10.2 [2], SWS technology can optimise several processes in the financial domain. These processes are mainly related with human interactions and, consequently, with the costs associated to them. Hence the main benefit of applying SWS technology is that it could permit to develop and maintain financial services with lower costs.

The application automates the process of collecting mortgage data from several banks, taking into account that the data can be accessed by executing Semantic Web Services from different banks. Then it provides this aggregated information to users, according to the data that they have filled-in in appropriate query forms.

2 OVERVIEW MORTGAGE COMPARISON SERVICE

The main interaction of the Mortgage Offers Simulator/Comparator is very simple: each time a client wants to know the mortgage market proposals, the application will give him/her actual simulations made on-line in each bank WS-based Simulator, and the results are presented in a web interface in order to compare them. Some further filtering on the presented data could enhance the user experience.

![Diagram of Mortgage Comparison service overview.](image)
2.1 Story board

In this section we walk through a typical scenario of a user that uses the application.

The mandatory parameters for calculating the mortgage formula are:

- **Monthly payment**: The amount of money it is paid every month.
- **Number of payments**: Number of payments during the whole life of the mortgage.
- **Total mortgage amount**: Mortgage amount that user is looking for.
- **Maximum interest rate**: Interest rate limit. All mortgages which interest would be bigger than maximum interest rate will be discarded.
- **Type of interest**: Fixed type, variable type or mixed.
- **Home Insurance**: This parameter indicates if a home insurance must be contract with mortgage.
- **Life Insurance**: This parameter indicates if a home insurance must be contract with mortgage.

The user interaction is described in the following steps:(more information about these steps in [1]).

1) There are three kind of parameters in each mortgage Web Service. The **obligatory parameters**, the user has to provide are: type of interest and **maximum interest rate value**. If the user selects type of interest as variable or mixed, she/he also needs to provide the type of **reference index** (euribor, ceca, irph). The **optional parameters** for the mortgage calculation are: **home insurance** and **life insurance**. The specific **interest rate** for a given mortgage, **home insurance** and **life insurance** are **predefined** in each bank Web Service, so they do not take part in the input parameters provided by the user.

The user need to provide only two of the three obligatory parameters. The third, the missing one, will be provided by the web service execution.
Figure 2: Screenshot of mortgage comparison service.

Table 1: Translation to English of terms from the application interface

<table>
<thead>
<tr>
<th>Spanish</th>
<th>English</th>
</tr>
</thead>
<tbody>
<tr>
<td>Importe Cuota</td>
<td>Monthly payment</td>
</tr>
<tr>
<td>Interés más índice de referencia</td>
<td>Maximum interest rate of mortgage</td>
</tr>
<tr>
<td>Importe del préstamo</td>
<td>Total amount of mortgage</td>
</tr>
<tr>
<td>Número de cuotas mensuales</td>
<td>Number of payments</td>
</tr>
<tr>
<td>Seguro de vida</td>
<td>Life Insurance</td>
</tr>
<tr>
<td>Seguro de vivienda</td>
<td>House Insurance</td>
</tr>
<tr>
<td>Indice de referencia</td>
<td>Reference index</td>
</tr>
<tr>
<td>Interes fijo</td>
<td>Fixed interest</td>
</tr>
<tr>
<td>Interes mixto</td>
<td>Mixed interest ( rate variable + rate fixed)</td>
</tr>
<tr>
<td>Interes variable</td>
<td>Variable interest</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Banco</td>
<td>Bank</td>
</tr>
<tr>
<td>Comparar</td>
<td>Compare</td>
</tr>
</tbody>
</table>
2) Generates a goal to find a SWS that fulfill a goal. Then press the continue button to send the goal generated to the discovery component.

![comparador.com](image_url)

User Goal

```xml
myGoal-goal
nlp -> nIF [ 
title -> "Goal to find mortgage simulator with value restrictions",
creator -> "blank 2",
type -> "http://www.w3no.org/2004/d1#goals",
description -> "",
contributor -> ['Simulators', 'Mortgage', 'Financial', 'Product'],
date -> "2005-03-01",
format -> "text/html",
language -> "en-US",
rts -> "http://www.naco.com/privacy.html"
]
capability->nOPCapability
]

nOPCapability: capability
precondition -> nOPprecondition,
postcondition -> nOPpostcondition
]

nOPprecondition: 'MortgageLoan'
loanCapital -> 200000.0,
initialQuota -> 500.0,
interestRateType -> Interest

Interest: 'ProductRateApplicationVariable'
interestRateValue -> 3.5,
referenceType -> "FURIBOR"

nOPpostcondition: 'MortgageLoan'
term -> 
openingCommission -> 2,
lifeInsurance -> 'true',
homeInsurance -> 'true'
```

Figure 3: Goal example generated by mortgage comparison service.
3) The discovery component returns a list of available Web Services. Then press the continue button to send them to the invocation module.

![Web services returned by](comparador.com)

- bank 2
- bank 3
- bank 1

**Figure 4: List of web services that match with a given goal.**

User can press a bank name in the list of available Web Services to view their description.
Figure 5: Screenshot with the service returned by discovery.
4) Finally once Web Services are invoked, the application receives the output data to be presented to the user. View translation to English in Table 1: Translation to English of terms from the application interface.

![Screenshot of mortgage comparison service results](image)

**Figure 6: Screenshot of mortgage comparison service results**

As mentioned earlier in [3], the application completes the missing parameter from the user input. It can be used for the mortgage amount or number of payments or monthly payment calculation, depending on the user input, given the parameters, the application allows for twelve combinations of possible input data along with corresponding output data, the table below shows the different combinations
Table 2: Inputs outputs mortgage comparison service:

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monthly payment + Number of Payments + Type of Interest + Max. Interest Rate</td>
<td>Mortgage Amount</td>
</tr>
<tr>
<td>Monthly payment + Mortgage Amount + Type of Interest + Max. Interest Rate</td>
<td>Number of Payments</td>
</tr>
<tr>
<td>Number of Payments + Mortgage Amount + Type of Interest + Max. Interest Rate</td>
<td>Monthly payment</td>
</tr>
<tr>
<td>Monthly payment + Number of Payments + Type of Interest + Max. Interest Rate + Home Insurance</td>
<td>Mortgage Amount</td>
</tr>
<tr>
<td>Monthly payment + Mortgage Amount + Type of Interest + Max. Interest Rate + Home Insurance</td>
<td>Number of Payments</td>
</tr>
<tr>
<td>Number of Payments + Mortgage Amount + Type of Interest + Max. Interest Rate + Home Insurance</td>
<td>Monthly payment</td>
</tr>
<tr>
<td>Monthly payment + Number of Payments + Type of Interest + Max. Interest Rate + Life Insurance</td>
<td>Mortgage Amount</td>
</tr>
<tr>
<td>Monthly payment + Mortgage Amount + Type of Interest + Max. Interest Rate + Life Insurance</td>
<td>Number of Payments</td>
</tr>
<tr>
<td>Number of Payments + Mortgage Amount + Type of Interest + Max. Interest Rate + Life Insurance</td>
<td>Monthly payment</td>
</tr>
<tr>
<td>Monthly payment + Number of Payments + Type of Interest + Max. Interest Rate + Life Insurance + Home Insurance</td>
<td>Mortgage Amount</td>
</tr>
<tr>
<td>Monthly payment + Mortgage Amount + Type of Interest + Max. Interest Rate + Life Insurance + Home Insurance</td>
<td>Number of Payments</td>
</tr>
<tr>
<td>Number of Payments + Mortgage Amount + Type of Interest + Max. Interest Rate + Life Insurance + Home Insurance</td>
<td>Monthly payment</td>
</tr>
</tbody>
</table>

For the sake of simplicity the following assumptions are considered by each bank:

- All monthly payments are equal
- All monthly payments include amortization of and interest.
3 COMPONENTS AND ARCHITECTURE

The banking business model requires a flexible product creation processes in an extremely short time-to-market. Every bank needs to react to market needs and monitor competitors very closely. For that reason the technological environment must allow a flexible application creation on a low cost base. In this prototype we have succeeded on showing a working environment using SWS technology with a specific focus on automatic discovery of new (mortgage) services.

Comparator uses dip components to find mortgage simulation services, offered by banks and aggregates them. When a bank creates new SWS and registries this service in a SWS repository in order to discovery component can locate this service. Comparator service sends a goal to discovery component in order to locate suitable services that offers mortgage comparison services. Discovery component returns a set of services and each service is invoked using invocation module. As a result obtain different mortgage offers to compare them.

Components from dip architecture must provide:

1. It provides a user interface for customer interaction, web interface, Web Service Interface, mobile devices interface, etc
2. It provides appropriate Web Services for mortgage simulator.
3. It discovers suitable Web Services for a user request.
4. It invokes external Semantic Web Services
5. It registries providers Semantic Descriptions.

3.1 Comparator- A Mortgage Comparison service

The comparison service is the module that acts as a user interface and interacts with the DIP components and with financial entities that offers mortgage comparison services. Mortgage comparison service aggregates all services offers by financial entities as shows Figure 7: System architecture.

Present the user web interface and:
1) Receives the user data inputs
2) Generates a goal, user desires.
3) Sends the goal generated to the discovery component
4) Mortgage comparator service receives a list of Web Services
5) Invoke every service using the invocation module.
6) Finally Mortgage comparison service receives the Web Services data output, this information is presented in a web interface.

3.1.1 Current version & status

The current version of the mortgage comparison service is able to collect several mortgage offers and range this offers. The application is available at http://comparador.isoco.com/mortgagecomparator/comparador.jsp. Actual version generates a goal based on F-logic [5] using a goal template predefined. The component generates a goal based on F-logic because the discovery component uses this formalism for reasoning. This component is ready to order the mortgage offers by selecting the field to order.
3.1.2 Requirements

**Nature:** Web Application Component.

**Interfaces (API, Web Services):** non provided

**Platform:** Application Server (Tomcat, Resin, Weblogic).

3.1.3 External systems:

- Discovery component (see section 3.2).
- Invocation Module (see section 3.3)

3.1.4 External Libraries;

The following libraries are needed, and hence they are included in the war file that deploys the application:

- wsdl4j.jar Provides WSDL libraries.
- mail.jar Provides a platform-independent and protocol-independent framework to build mail and messaging applications.
- activation.jar The classes that make up JavaBeans Activation Framework.
- servlet.jar Allow servlets execution.

**Figure 8: Screenshot with results screen.**

<table>
<thead>
<tr>
<th>Banco/Caja</th>
<th>Importe del préstamo</th>
<th>Tipo de interés</th>
<th>Número de cuotas mensuales</th>
<th>Importe cota</th>
</tr>
</thead>
<tbody>
<tr>
<td>BBVA</td>
<td>60000.0</td>
<td>2.23%</td>
<td>150</td>
<td>500.0</td>
</tr>
<tr>
<td>BBVA 2</td>
<td>60000.0</td>
<td>2.2%</td>
<td>150</td>
<td>500.0</td>
</tr>
<tr>
<td>BBVA 3</td>
<td>60000.0</td>
<td>2.23%</td>
<td>143</td>
<td>500.0</td>
</tr>
</tbody>
</table>

(See terms and conditions and reference indices. (Bank of Spain))
o soap-2.3.1.jar Provides Soap interfaces.

3.2 Discovery

3.2.1 Current version and status

The current version of discovery component can be used via web service or java library and the tool offers the Java interfaces presented in Table 3 and Web Service interface presented in Table 4: Discovery WSDL. The discovery component is distributed as a .jar to be used directly by your application or as a web service.

Table 3: Discovery API.

<table>
<thead>
<tr>
<th>Method Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>String[] Discover(String Goal)</td>
</tr>
<tr>
<td>Receive a goal and returns a list of web service descriptions that match with a given goal.</td>
</tr>
</tbody>
</table>

Table 4: Discovery WSDL.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="http://comparador.isoco.com/discovery/services/Matcher"
xmlns="http://schemas.xmlsoap.org/wsdl/
xmlns:tns1="http://discovery.dip.isoco.com"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:apachesoap="http://xml.apache.org/xml-soap"
xmlns:intf="http://comparador.isoco.com/discovery/services/Matcher"
xmlns:impl="http://comparador.isoco.com/discovery/services/Matcher" type="schema"
xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://discovery.dip.isoco.com"><import
namespace="http://schemas.xmlsoap.org/soap/encoding/"/><complexType
name="ArrayOfString"><complexType
base="soapenc:Array"><attribute
ref="soapenc:arrayType" wsdl:arrayType="xsd:string[]"/></complexType></complexType></schema></types>
<wsdl:message name="discoverRequest">
<wsdl:part name="goal" type="xsd:string" />
<wsdl:message>
<wsdl:message name="mainRequest">
<wsdl:part name="args" type="tns1:ArrayOfString" />
<wsdl:message>
<wsdl:message name="mainResponse">
<wsdl:message>
<wsdl:message name="discoverResponse">
<wsdl:part name="discoverReturn" type="tns1:ArrayOfString"/>
```
<wsdl:message
</wsdl:message>
<wsdl:portType name="Matcher">
<wsdl:operation name="main" parameterOrder="args">
<wsdl:input name="mainRequest" message="impl:mainRequest"/>
<wsdl:output name="mainResponse" message="impl:mainResponse"/>
</wsdl:operation>
<wsdl:operation name="discover" parameterOrder="goal">
<wsdl:input name="discoverRequest" message="impl:discoverRequest"/>
<wsdl:output name="discoverResponse" message="impl:discoverResponse"/>
</wsdl:operation>
</wsdl:portType>
<wsdl:binding name="MatcherSoapBinding" type="impl:Matcher">
<wsdlsoap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
<wsdl:operation name="main">
<wsdlsoap:operation soapAction=""/>
<wsdl:input name="mainRequest">
<wstx:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="http://discovery.dip.isoco.com"/>
</wsdl:input>
<wsdl:output name="mainResponse">
<wstx:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="http://comparador.isoco.com/discovery/services/Matcher"/>
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="discover">
<wsdlsoap:operation soapAction=""/>
<wsdl:input name="discoverRequest">
<wstx:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="http://discovery.dip.isoco.com"/>
</wsdl:input>
<wsdl:output name="discoverResponse">
<wstx:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="http://comparador.isoco.com/discovery/services/Matcher"/>
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="MatcherService"
3.2.2 Requirements
Nature: Web Service component.
Interfaces (API, Web Services): Web Service
Platform: JDK 1.4.2 or 1.5.

3.2.3 External systems: XSB\textsuperscript{1}/Flora-2\textsuperscript{2}
FLORA-2 is an advanced object-oriented knowledge base language and application development environment. The language of FLORA-2 is a dialect of F-logic with numerous extensions, including meta-programming in the style of HiLog and logical updates in the style of Transaction Logic.

XSB is a Logic Programming and Deductive Database system for UNIX and Windows. It is being developed at the Computer Science Department of the Stony Brook University, in collaboration with Katholieke Universiteit Leuven.

3.2.4 External Libraries
The following libraries are needed, and hence they are included in the war file that deploys the application:

- saaj.jar Apache Axis dependency.
- jaxrpc.jar Apache Axis dependency.
- axis.jar Framework\textsuperscript{3} for constructing SOAP processors such as clients, servers etc. It can be plugged into servlet engines such as Tomcat, Resin….
- log4j-1.2.8.jar. Apache Axis dependency.
- commons-logging.jar. Apache Axis dependency.
- commons-discovery.jar. Apache Axis dependency.
- xercesImpl.jar Axis dependency.
- xmlParserAPIs.jar Axis dependency.
- interprolog.jar\textsuperscript{4} A Java XSB prolog interface.

\textsuperscript{1} See \url{http://xsb.sourceforge.net/} for more information.
\textsuperscript{2} See \url{http://flora.sourceforge.net} for more information.
\textsuperscript{3} See \url{http://ws.apache.org/axis/} for more information.
○ servlet.jar. Allow servlets execution.
○ XSBFlora.jar Flora2Java allow a transparent usage of flora2 from Java. It uses interprolog to access the XSB from Java, and, via some additional piece of custom code added, it allows its users to evaluate FLogic expressions.

3.2.5 Licensing

Copyright (c) 2005, iSOCO

The discovery component is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version. This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details. You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA.

3.2.6 Installation of discovery component

This component is distributed as a war file. To install it copy, discovery.war inside the Webapps directory of the application server. The service descriptions are stored in (%APP-SERVER%/webapps/discovery). In file discovery.properties is necessary specify directory where service descriptions are storage. Finally sets to Java Virtual Machine java.library.path=%APP-SERVER%/webapps/discovery/XSB/config/x86-pc-windows/bin to run discovery properly.

This component is used via web service using a WSDL interface provided in Table 4: Discovery WSDL.or as JAVA component using API provided in Table 3: Discovery API.

3.3 Invocation module

The invocation module expects a web service description in F-logic and executes this.

3.3.1 Current version and status

The current version of invocation module is able to use as java library. The tool offers the Java interfaces presented in Table 5.

Table 5: Invocation Module Method Summary

<table>
<thead>
<tr>
<th>Method Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean invoke (string webServiceDescription)</td>
</tr>
</tbody>
</table>

4 See http://www.declarativa.com/interprolog for more information.
3.3.2 Requirements

Nature: Standalone component.

Interfaces (API, Web Services): Java API.

Platform: JDK 1.4.2 or 1.5.

3.3.3 External Libraries

- soap-2.3.1.jar. Provides Soap interfaces.
- jdom.jar. Parses, manipulates, and outputs XML using standard Java constructs.
- wsdl4j.jar. Provides WSDL libraries.

3.3.4 Licensing

Copyright (c) 2005, iSOCO

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version. This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details. You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA.

3.3.5 Installation & usage

This component is distributed as a java library to use this component add into CLASPATH or web application lib if component is used in web application.

4 CONCLUSIONS

The objective of the e-banking mortgage comparator prototype was to implement and test the DIP conceptual architecture [4] and to provide a working example for further project work and demonstrations. As the result of ongoing DIP activities, the conceptual architecture provided offers a common model, tool and vocabulary for facing most of integration projects using Semantic Web technology. Having decomposed the prototype in terms of DIP architecture we have defined valuable requirements for different components to DIP technological partners as well as we are still cross-fertilizing other case studies with our experience. In order to have a working application and a real proof
of concept the application has been implemented and deployed in the case study partner: Bankinter.

REFERENCES


