



Data, Information and Process Integration
with Semantic Web Services

DIP

Data, Information and Process Integration with Semantic Web Services

FP6 - 507483

Deliverable

WP 10: Case Study eBanking

D10.8

WSMO Descriptions of Application 2

Dariusz Kleczek

Silvestre Losada

Jose Luis Bas

Sergio Bellido

Jesús Contreras

Monica Martinez Montes

Dec 12th, 2005



EXECUTIVE SUMMARY

This deliverable contains the WSMO descriptions of Semantic Web Services which will be deployed in the context of a Stock Market application specified in DIP deliverable 10.6. WSMO descriptions include domain ontologies for Stock Market Domain as well as descriptions of WSMO goals and Web Services. These descriptions provide test cases for WSMX, the core component for this prototype. The operational application based on the descriptions and WSMX is supposed to be a proof of concept for the WSMO framework and DIP.

The deliverable contributes to the goals of DIP by specifying a use case, which will show the functionality and added value of architecture and methodologies developed in DIP. The Stock Market prototype will be developed to demonstrate the advantages and benefits of implementing this application using SWS. In the chosen scenario the advantages of a SWS based solution are clearly in the area of service composition and system integration.

Finally an exploitable tool will be developed. It will also provide test cases for the technical architecture and therefore contribute to the usefulness and quality of other technical deliverables of the project.

WSMO descriptions of Semantic Web Services, as specified in this deliverable, are relevant for all the DIP technical workpackages (1, 2, 3, 4a, 4b, 5 and 6), since they provide practical test cases for the developed components.

The target audience of this deliverable includes: the partners developing tools to be used for the description of SWS, the partners developing the DIP infrastructure and external readers interested in finding information about a use case for SWS.

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

IST Project Number	FP6 – 507483	Acronym	DIP
Full title	Data, Information, and Process Integration with Semantic Web Services		
Project URL	http://dip.semanticweb.org		
Document URL			
EU Project officer	Kai Tullius		

Deliverable	Number	10.8	Title	WSMO Descriptions of Application 2
Work package	Number	10	Title	Case study eBanking

Date of delivery	Contractual	M 24	Actual	12-Dec-05
Status	version 1.0		final <input checked="" type="checkbox"/>	
Nature	Prototype <input type="checkbox"/> Report <input type="checkbox"/> Dissemination <input type="checkbox"/> Ontology <input checked="" type="checkbox"/>			
Dissemination Level	Public <input checked="" type="checkbox"/> Consortium <input type="checkbox"/>			

Authors (Partner)	Dariusz Kleczek (iSOCO), Silvestre Losada (iSOCO), Jose Luis Bas, (Bankinter), Sergio Bellido (Bankinter), Jesús Contreras (iSOCO), Monica Martínez (Bankinter)			
Responsible Author	Silvestre Losada (iSOCO)		Email	slosada@isoco.com
	Partner	iSOCO	Phone	Partner





Abstract (for dissemination)	This deliverable contains the WSMO descriptions of Semantic Web Services which will be deployed in the context of a Stock Market application specified in DIP deliverable 10.6. WSMO descriptions include domain ontologies for Stock Market Domain as well as descriptions of WSMO goals, Web Services. These descriptions provide a proof of concept for WSMX, the core component for this prototype.
Keywords	WSMO, financial services, Ontology, Financial, Stock Market; SWS, SW, e-Banking


Version Log			
Issue Date	Rev No.	Author	Change
02-11-05	001	Dariusz Kleczek iSOCO	Creates initial version of the Document
15-11-05	002	Dariusz Kleczek iSOCO	WSMO Descriptions




18-11-05	003	Jesús Contreras	Changes in sections 2 & 3
21-11-05	004	Silvestre Losada	WSMO descriptions
9-12-05	005	JL Bas Sergio Bellido	Changes in introduction
12-12-05	006	Monica Martínez	Version for internal review.
17-01-06	009	Dariusz Kleczek	Final version

Reviewer			
	Laurentiu Vasiliu	Email	laurentiu.vasiliu@deri.org
	Partner DERI	Phone	
	Hung Le Vu	Email	lehung.vu@epfl.ch
	Partner EPFL	Phone	

Project Consortium Information

Partner	Acronym	Contact
National University of Ireland Galway	NUIG 	Dr. Sigurd Harand Digital Enterprise Research Institute (DERI) National University of Ireland, Galway Galway Ireland Email: sigurd.harand@deri.org Tel: +353 91 495112
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

Partner	Acronym	Contact
Institut für Informatik, Leopold-Franzens Universität Innsbruck	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
ILOG SA	ILOG  Changing the rules of business	Christian de Sainte Marie 9 Rue de Verdun, 94253 Gentilly, France Email: csma@ilog.fr Tel: +33 1 49082981
inubit AG	Inubit  the integration experts	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  The Open University	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

Sirma AI Ltd.	 <p> Sirma Ontotext Knowledge and Language Engineering Lab of Sirma </p>	Atanas Kiryakov, Ontotext Lab, - Sirma AI EAD Office Express IT Centre, 3rd Floor 135 Tzarigradsko Chausse Sofia 1784, Bulgaria Email: atanas.kiryakov@sirma.bg Tel.: +359 2 9768 303
Unicorn Solution Ltd.	 <p> Unicorn </p>	Jeff Eisenberg Unicorn Solutions Ltd, Malcha Technology Park 1 Jerusalem 96951 Israel Email: Jeff.Eisenberg@unicorn.com Tel.: +972 2 6491111
Vrije Universiteit Brussel	 <p> VUB Vrije Universiteit Brussel </p>	Pieter De Leenheer Starlab- VUB Vrije Universiteit Brussel Pleinlaan 2, G-10 1050 Brussel ,Belgium Email: Pieter.De.Leenheer@vub.ac.be Tel.: +32 (0) 2 629 3749

LIST OF KEY WORDS/ABBREVIATIONS

SOAP	Simple Object Access Protocol
SWS	Semantic Web Services
UDDI	Universal Discovery, Description and Integration
WS	Web Services
WSDL	Web Service Description Language
WSML	Web Service Modeling Language
WSMO	Web Service Modeling Ontology
WSMX	Web Service Execution Environment
XML	eXtensible Markup Language

TABLE OF CONTENTS

EXECUTIVE SUMMARY	I
LIST OF KEY WORDS/ABBREVIATIONS	VII
TABLE OF CONTENTS	VIII
1 INTRODUCTION	1
2 USE CASE OVERVIEW	1
2.1 Description	1
2.2 Scope	2
2.3 Actors` Roles and Goals	2
2.4 System Architecture	3
3 WSMO USE CASE MODELS	5
3.1 Ontologies.....	15
3.2 Goals.....	28
3.3 Web Services	37
3.4 Interfaces	52
4 CONCLUSIONS	60
REFERENCES	60
ANNEX I	61

LIST OF FIGURES

Figure 1: Conceptual Architecture of eBanking Application.....	2
Figure 2: Overview of the system architecture.....	4

LIST OF TABLES

Table 1: Use Case Instances	5
Table 2: Stock Market Process Ontology	5
Table 3: Stock Market Ontology	6
Table 4: Goal UCI1	6
Table 5: Goal UCI2	7
Table 6: Goal UCI3	7
Table 7: Goal UCI4	7
Table 8: Goal UCI5	8

Table 9: Goal UCI6	8
Table 10: Goal UCI7	9
Table 11: Web Service UCI1.....	9
Table 12: Web Service UCI2 and UCI4.....	10
Table 13: Web Service UCI3.....	10
Table 14: Web Service UCI5.....	11
Table 15: Web Service UCI6.....	11
Table 16: Web Service UCI7.....	12
Table 17: Choreography Interface UCI1	12
Table 18: Choreography Interface UCI2 and UCI4	13
Table 19: Choreography Interface UCI3	13
Table 20: Choreography Interface UCI5	14
Table 21: Choreography Interface UCI6.....	14
Table 22: Choreography Interface UCI7	14

1 INTRODUCTION

This document presents the WSMO descriptions of application 2 in the Case Study eBanking (DIP WP10). WSMO descriptions are WSML documents which contain WSMO formalization. The objective of this document is to specify the WSMO layer of the prototype application which is going to be developed in WP10. The descriptions developed will be also used as a requirement set for the DIP technical architecture.

The evolution of information and communication technologies has enabled non-expert customers to participate in stock market operations. Such operations can be executed for example from the Bankinter web site. Its functions include consulting information about specific stocks, indices, markets, alerting service based on user's preferences and placing of buy or sell orders.

The above operations are currently implemented as Web Services, which can be accessed for example through the Bankinter Mobile Brokerage Application. Although a significant advance, Web Services on their own contain some deficiencies [7].

This barrier may be overcome by the development and adoption of Semantic Web Services. In the context of the present use case, their implementation brings twofold benefits. Firstly, the aggregation and discovery of existing services will lower the cost of developing and maintaining the application and shorten the time to market, which are normally high due to the heterogeneity of stock market operations and financial products.

This document is structured as follows: section 2 provides a general overview of the use case, section 3 gives a detailed view of the specific WSMO descriptions and section 4 presents some conclusions.

2 USE CASE OVERVIEW

This section provides a description of the setting of this use case. According to [1] we have distinguished the following elements: description, scope, actors' roles and goals, usage scenarios and system architecture.

2.1 Description

The area of financial operations can be characterized as heterogeneous and dynamic. It comprises of thousands of products, such as stocks, bonds, commodities and derivatives, which can be combined together. The parameters which influence decision making come from different sources and their value changes over time. These characteristics make it suitable as a testbed for Semantic Web Services, which aim at overcoming the difficulties of software engineering in such an environment. For this use case we have chosen to concentrate on stock market operations.

The application should enable Bankinter's customers to execute complex operations on the stock market. These might involve testing a condition before executing an action, such as a buy or sell order, or selecting the right stock to buy. Based on the user input, the system will find and fill a SWS Goal Template, which will be forwarded to the DIP architecture for the discovery and execution of appropriate Web Services. The long term goal is to free customers from restrictions posed by custom built applications. That means, that the customer is able to formulate whichever goal he wishes to achieve. If

currently no web service combination is able to satisfy such a goal, it can be stored as an incentive for businesses to offer the desired functionality. The way to achieve this long term goal is to build interfaces restricted to a well defined domain in the short term, such as is the case of the Broker application.

The deployment of WSMX environment in the broker application presents a significant added value, which lies in its capability of composing existing web services to achieve more complex functionality. This composition will be realized with an orchestration component developed inside WSMX¹.

Figure 1 gives an overview of the use case.

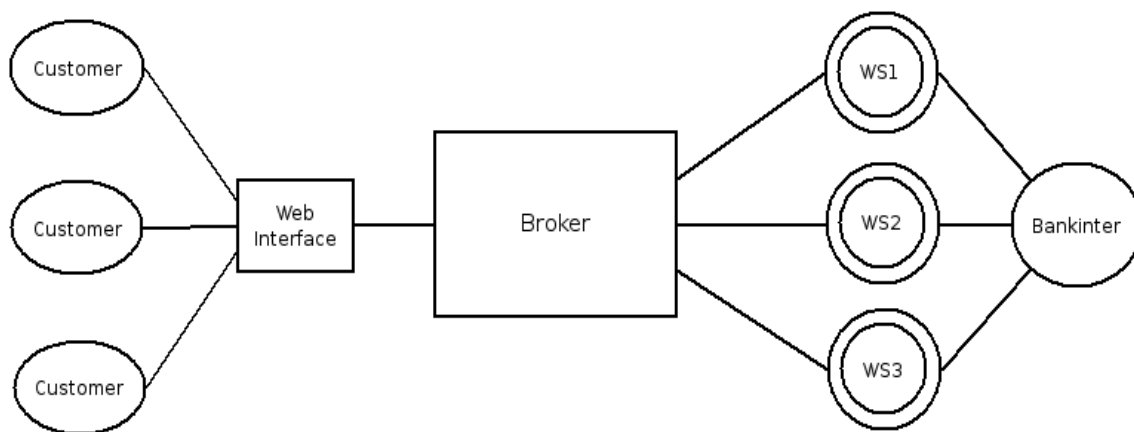


Figure 1: Conceptual Architecture of eBanking Application

2.2 Scope

The application is being developed for Bankinter and adds a semantic layer to the existing Web Services of Bankinter. However, it can easily be extended to incorporate other Semantic Web Services and to be deployed in another financial institution.

2.3 Actors` Roles and Goals

We have identified 3 actors in the use case. The goals represent the reason for their participation in the use case and the roles define the particular interactions they are involved in.

1. Customer: the end user that requests a service provided by the broker
 - *Goal*: execution of an operation on a stock market based on some kind of decision criteria
 - *Role*: end-user, interacts with the broker for service usage.

¹ The orchestration is however still under development and its specification is outside of the scope of this deliverable.

2. Broker: the intermediate between the Customer and the Service Providers. It provides brokerage services to customers by aggregating the separate services provided by the single Service Providers. In our use case Broker is an application based on WSMX and adapter framework.
 - *Goal*: provide the customer with the complex product as requested by the customer, without showing the complexity of services combined together to achieve the desired product
 - *Role*: interacting with customer, aggregating and invoking Web Services offered by Service Providers, centrally holding all functionalities for handling Semantic Web Services (mechanisms for discovery, composition, execution etc.).
3. Service Providers: commercial companies (currently only Bankinter) that provide services to execute operations on the stock market and to access decision support information for these operations
 - *Goal*: sell service to end customers, maximize profit as a commercial company
 - *Role*: provide operation execution and decision support services as Web Services as well as their semantic descriptions.

2.4 System Architecture

The system architecture of the application has been specified in [3]. The application functions as an intermediary between the customer and service providers. Therefore, its features must consist of an interface for the customer and of mechanisms for discovery, composition and execution of Web Services. The high level functionality required is:

1. Semantic representation of services using WSMO.
2. Automatic service discovery that match with a given goal. It is necessary to discover the best WS that fulfils the defined requirements, which are expressed by the user. The discovery component is concerned with finding Web Service descriptions that match the goal specified by the service requester. The discovery component returns a (possibly empty) list of Web Service descriptions.
3. Service composition: It is probable that no WS will completely fulfil the selected goals, and it will therefore be necessary to compose several WS that will achieve that goal. This functionality will be achieved through WSMX orchestration component. With orchestration, multiple WS can be composed to provide complex functionality, e.g.: in order to execute a BUY operation we need to check the account balance. It is probable that no WS will completely fulfil the selected goals, and it will therefore be necessary to compose several WS that will achieve that goal partially.
4. Invocation of services. We need to invoke WS returned by the discovery component and fulfill a given Goal. The invocation component is responsible for this function.

Some components of DIP/WSMX offer these features, therefore the architecture of Stock Market application is based on DIP/WSMX architecture. Figure 2 shows the high level architecture for the prototype including the necessary DIP/WSMX Architecture components. This includes a set of design-time tools to create the semantic descriptions of the services, the WSMX core, communication manager to manage interaction, discovery component to discover services, choreography/orchestration component to

make composition and resources manager to manage storage, the adapter framework to convert between between WSMML and XML.

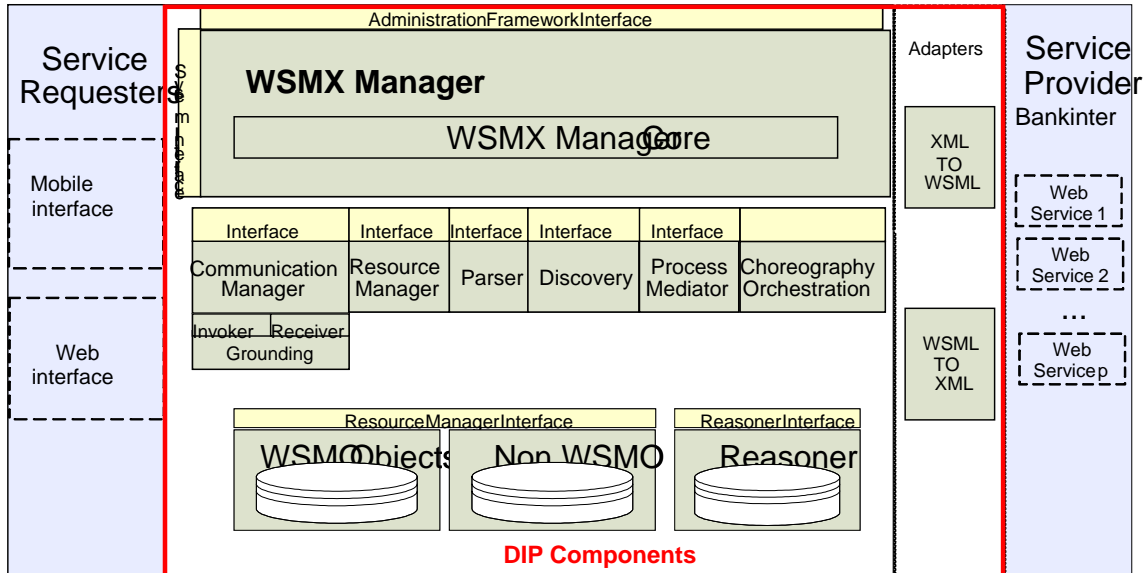


Figure 2: Overview of the system architecture

A very important issue in the context of this deliverable is the composition of Semantic Web Services. In the WSMO framework, orchestration provides a technique that allows service providers to realize the functionality of a Web Service by composition of other Web Services. This functionality is especially important in the finance industry, which is highly heterogeneous. We can easily imagine a financial contract which describes the right to choose between buying a certain stock at a future date and paying a certain amount of money at a different date.

The core contribution of Semantic Web Services lies in the fact, that these kinds of products may be offered in a dynamic and distributed manner, instead of hardcoding them inside an application.

In the long term, our vision is that user will be able to specify goals, which describe his requirement for the products.

In the short term, to prove these concepts, we restrict the functionality of prototype application to the domain of stock market operations. The orchestration of basic operations, such as obtaining information on products and executing actions, will be defined at design time based on the users' needs, as identified by Bankinter.

The current approach to orchestration that we are planning to implement in our case study is based on the formalism of abstract state machines. As this language is not user friendly, a graphical interface based on UML2 Activity Diagrams has been developed, which supports application developers in specifying the orchestration rules. We expect this interface together with the ASM formalism to provide a user friendly and efficient method of composing Semantic Web Services, which should solicit their acceptance in business.

3 WSMO USE CASE MODELS

This section exemplifies the specification of this use case in the “Web Service Modeling Ontology” (WSMO). These specifications comply with WSMO final version 1.2 and they have been validated with the WSML Online Validation Service. The resources are based on seven use case instances, which are listed in Table 1. Use case instances are defined as sets of composed operations, which will be performed by the prototype version of the system.

The use case instances described in this deliverable are based on Bankinter’s experience and expertise in providing the customers’ with access to stock market operations. The deployment of Semantic Web Services makes the application very scalable with respect to widening the functionality. We expect that with the development and discovery of new Web Services the customers will be provided with a growing number of operations.

In further sections of this deliverable, the use case instances will be addressed and appropriate WSMO Goals, Web Services and interfaces will be shown. The following tables provide an overview of the resources specified in this use case. Additionally, in Annex I we provide WSML descriptions of Web Services which are composed to achieve functionality specified in the use case instances.

Table 1: Use Case Instances

Use Case Instance	Description
UCI1	<i>The user wants to execute some action (action can be a buy order, a sell order, or a request for an alert) on a specific stock based on the condition, that this stock belongs to top five variations on a specific stock market</i>
UCI2	<i>The user wants to execute some action on a stock on the condition that its value rises</i>
UCI3	<i>The user wants to execute some action on a stock on the condition that some or all of relevant entities advice this kind of action</i>
UCI4	<i>The user wants to execute some action on a stock on the condition that its value changes and the change is in a specified percentage range</i>
UCI5	<i>The user wants to execute some action on a stock if its rating changes to a specified value</i>
UCI6	<i>The user wants to execute some action on a stock provided that any of its numerical attributes satisfies certain statistical constraint</i>
UCI7	<i>The user wants to execute some action on the stock which has the highest variation on a certain market.</i>

Table 2: Stock Market Process Ontology

Attribute	Value
-----------	-------

WSMO component type	Ontology
Name	Stock Market Process Ontology
Description	Defines ontology constructs for the domain of Brokering services
Imported ontologies, used mediators	Stock Market Ontology
Main constructs	Action, CheckStock, ExecutelfRatingChanges, CheckIndexes, ExecutelfRecommendations, CheckStatistics, ListNews, ExecutelfInTopFiveVariations, CheckStocksOfIndex, ExecutelfValueChanges, CheckQuotations, Result, NewsDetail
WSML model	Listing 1

Table 3: Stock Market Ontology

Attribute	Value
WSMO component type	Ontology
Name	Stock Market Ontology
Description	Defines ontology constructs for the stock market domain
Imported ontologies, used mediators	OWL currency mediator Financial Ontology
Main constructs	Stock, Index, LastPrice, BuySellOrder, StockMarket, Dividend, StockPortfolio, IndexSession, Session, Quotation, StockWeight, Depository, Portfolio, StockMaxValue, StockVariation, Broker
WSML model	Listing 2

Table 4: Goal UCI1

Attribute	Value
WSMO component type	Goal
Name	Goal for variation dependent actions
Description	If the value of a specified stock belongs to the top five variations on a specified market, then the system execute some action
Imported ontologies, used mediators	Stock Market Ontology
Main constructs	Precondition: The stock specified by the user has to belong to the top five variations on a specified market

	Postcondition: Result of the web service is execution of the desired action
WSML model	Listing 3

Table 5: Goal UCI2

Attribute	Value
WSMO component type	Goal
Name	Goal for value dependent actions 2
Description	A Goal to execute an action on a stock if its value rises
Imported ontologies, used mediators	Stock Market Ontology
Main constructs	Precondition: The value of the stock specified by the user has to rise Postcondition: Result of the web service is execution of the desired action
WSML model	Listing 4

Table 6: Goal UCI3

Attribute	Value
WSMO component type	Goal
Name	Goal for recommendation dependent actions
Description	A Goal to execute an action on a stock based on its recommendations
Imported ontologies, used mediators	Stock Market Ontology
Main constructs	Precondition: All recommendations regarding stock specified by the user have to advise some action Postcondition: Result of the web service is execution of the desired action
WSML model	Listing 5

Table 7: Goal UCI4

Attribute	Value
WSMO component	Goal

type	
Name	Goal for value dependent action 1
Description	A Goal to execute an action on a stock based on its value
Imported ontologies, used mediators	Stock Market Ontology
Main constructs	Precondition: The value of the stock specified by the user has to fall more than specified user's value Postcondition: Result of the web service is execution of the desired action
WSML model	Listing 6

Table 8: Goal UCI5

Attribute	Value
WSMO component type	Goal
Name	Goal for rating dependent actions
Description	A Goal to execute an action on a stock based on its rating
Imported ontologies, used mediators	Stock Market Ontology
Main constructs	Precondition: The rating of the stock reaches a value specified by the user Postcondition: Result of the web service is execution of the desired action
WSML model	Listing 7

Table 9: Goal UCI6

Attribute	Value
WSMO component type	Goal
Name	Goal for statistics dependent actions
Description	A Goal to execute an action on a stock based on its statistics
Imported ontologies, used mediators	Stock Market Ontology
Main constructs	Precondition: Value of some stock statistics is bigger than the value specified by the user

	Postcondition: Result of the web service is execution of the desired action
WSML model	Listing 8

Table 10: Goal UCI7

Attribute	Value
WSMO component type	Goal
Name	Goal for select and execute orders
Description	A Goal for selecting stocks and executing on them Buy or Sell orders
Imported ontologies, used mediators	Stock Market Ontology
Main constructs	Postcondition: Result of the web service is execution of the desired action on the stock which has the highest variation
WSML model	Listing 9

Table 11: Web Service UC11

Attribute	Value
WSMO component type	Web Service
Name	Web Service for variation dependent actions
Description	If the stock value belongs to the top five variations, then the system execute some action
Imported ontologies, used mediators	Stock Market Ontology Financial Ontology
Main constructs	<p>Precondition: The system has to know the portfolio ID, from which to execute action, and the user, to check if the portfolio belongs to the user. The stock name, number of them and market have to be known to the system, if it is going to check whether it belongs to the top five variations and then do something with it. The action has to be specified as well. The stock value has to belong to top five variations. If not, no action is performed. In case of SELL action user has to possess at least the specified number of stocks in his portfolio. In case of BUY action, the value of stocks the user intends to buy must not be bigger than the balance on user's account</p> <p>Postcondition: In case of SELL, the user deposes of the number of stocks from the portfolio. The value is added to his savings account. In case of BUY, the number of stocks is added to</p>

	portfolio. The value is subtracted from the savings account
WSML model	Listing 10

Table 12: Web Service UCI2 and UCI4

Attribute	Value
WSMO component type	Web Service
Name	Web for value dependent actions 2
Description	If the value of some stock changes then execute BUY SELL ALERT
Imported ontologies, used mediators	Stock Market Ontology Financial Ontology
Main constructs	<p>Precondition: The necessary information to be provided to the system includes: stock, market, kind of action, customer, and portfolio. The stock has to belong to the specified market and the portfolio to the customer. The relative variation of the stock has to satisfy the condition provided by the customer. In case of SELL action user has to possess at least the specified number of stocks in his portfolio. In case of BUY action, the value of stocks the user intends to buy must not be bigger than the balance on user's account</p> <p>Postcondition: In case of SELL, the user deposes of the number of stocks from the portfolio. The value is added to his savings account. In case of BUY, the number of stocks is added to portfolio. The value is subtracted from the savings account</p>
WSML model	Listing 11

Table 13: Web Service UCI3

Attribute	Value
WSMO component type	Web Service
Name	Web Service for recommendation dependent actions
Description	Execute BUY (SELL) if recommendation of some or all entities is BUY (SELL)
Imported ontologies, used mediators	Stock Market Ontology Financial Ontology
Main constructs	Precondition: The necessary information to be provided to the system includes: stock, market, kind of action, customer, portfolio, quantifier - some or all regarding the recommendations. The stock has to belong to the specified market and the

	<p>portfolio to the customer. In case of SELL action user has to possess at least the specified number of stocks in his portfolio. In case of BUY action, the value of stocks the user intends to buy must not be bigger than the balance on user's account</p> <p>Postcondition: In case of SELL, the user deposes of the number of stocks from the portfolio. The value is added to his savings account. In case of BUY, the number of stocks is added to portfolio. The value is subtracted from the savings account</p>
WSML model	Listing 12

Table 14: Web Service UCI5

Attribute	Value
WSMO component type	Web Service
Name	Web Service for rating dependent actions
Description	WS to execute some action if the rating of some stock changes
Imported ontologies, used mediators	Stock Market Ontology Financial Ontology
Main constructs	<p>Precondition: The system has to know the portfolio ID, from which to execute action, and the user, to check if the portfolio belongs to the user. The stock name, number of them and market have to be known to the system, if it is going to check whether its rating has changed and then do something with it. The action has to be specified as well. The stock value has to belong to top five variations. If not, no action is performed. In case of SELL action user has to possess at least the specified number of stocks in his portfolio. In case of BUY action, the value of stocks the user intends to buy must not be bigger than the balance on user's account</p> <p>Postcondition: In case of SELL, the user deposes of the number of stocks from the portfolio. The value is added to his savings account. In case of BUY, the number of stocks is added to portfolio. The value is subtracted from the savings account</p>
WSML model	Listing 13

Table 15: Web Service UCI6

Attribute	Value
WSMO component type	Web Service
Name	Web Service for statistics dependent actions
Description	WS to execute some action on the stock if some statistics

	concerning this stock reach a specified value
Imported ontologies, used mediators	Stock Market Ontology Financial Ontology
Main constructs	<p>Precondition: The necessary information to be provided to the system includes: stock, market, kind of action, customer, portfolio. The stock has to belong to the specified market and the portfolio to the customer. The statistics of the stock has to satisfy the condition provided by the customer. In case of SELL action user has to possess at least the specified number of stocks in his portfolio. In case of BUY action, the value of stocks the user intends to buy must not be bigger than the balance on user's account</p> <p>Postcondition: In case of SELL, the user deposes of the number of stocks from the portfolio. The value is added to his savings account. In case of BUY, the number of stocks is added to portfolio. The value is subtracted from the savings account</p>
WSML model	Listing 14

Table 16: Web Service UCI7

Attribute	Value
WSMO component type	Web Service
Name	Web Service for select and execute orders
Description	Web Service for selecting stocks and executing on them Buy or Sell orders
Imported ontologies, used mediators	Stock Market Ontology Financial Ontology
Main constructs	<p>Precondition: The stock has the highest variation on the market. The user has more money than the value of the stocks he wants to buy, or more stocks than the number he wants to sell</p> <p>Postcondition: In case of SELL, the user deposes of the number of stocks from the portfolio. The value is added to his savings account. In case of BUY, the number of stocks is added to portfolio. The value is subtracted from the savings account</p>
WSML model	Listing 15

Table 17: Choreography Interface UCI1

Attribute	Value
WSMO component type	Choreography Interface

Name	interface WSVariationDependentActionInterface
Description	Choreography for UCI1
Imported ontologies, used mediators	Stock Market Ontology Stock Market Process Ontology
Main constructs	State Signature Transition Rules
WSML model	Listing 16

Table 18: Choreography Interface UCI2 and UCI4

Attribute	Value
WSMO component type	Choreography Interface
Name	interface WSValueDependentActionInterface
Description	Choreography for UCI2 and UCI4
Imported ontologies, used mediators	Stock Market Ontology Stock Market Process Ontology
Main constructs	State Signature Transition Rules
WSML model	Listing 17

Table 19: Choreography Interface UCI3

Attribute	Value
WSMO component type	Choreography Interface
Name	interface WSRecommendationsDependentActionInterface
Description	Choreography for UCI3
Imported ontologies, used mediators	Stock Market Ontology Stock Market Process Ontology
Main constructs	State Signature Transition Rules
WSML model	Listing 18

Table 20: Choreography Interface UCI5

Attribute	Value
WSMO component type	Choreography Interface
Name	interface WSRatingDependentActionInterface
Description	Choreography for UCI5
Imported ontologies, used mediators	Stock Market Ontology Stock Market Process Ontology
Main constructs	State Signature Transition Rules
WSML model	Listing 19

Table 21: Choreography Interface UCI6

Attribute	Value
WSMO component type	Choreography Interface
Name	interface WSStatisticsDependentActionInterface
Description	Choreography for UCI6
Imported ontologies, used mediators	Stock Market Ontology Stock Market Process Ontology
Main constructs	State Signature Transition Rules
WSML model	Listing 20

Table 22: Choreography Interface UCI7

Attribute	Value
WSMO component type	Choreography Interface
Name	interface WSSelectAndExecuteInterface
Description	Choreography for UCI7
Imported ontologies, used mediators	Stock Market Ontology Stock Market Process Ontology
Main constructs	State Signature Transition Rules

WSML model

Listing 21

3.1 Ontologies

The WSMO Descriptions in this case study are based on two ontologies, which have been defined in WP10 [4,5] and a stock market process ontology, which has been constructed for the purpose of this application. In the following two listings we present the stock market process ontology, which depicts the domain of the broker application, and the stock market ontology, which concentrates on the area of supported stock market operations.

Listing 1 Stock Market Process Ontology

```

namespace {
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"
  ,
  xsd _"http://www.w3c.org/2001/XMLSchema#" ,
  dc _"http://purl.org/dc/elements/1.1#" ,
  foaf _"http://xmlns.com/foaf/01/" ,
  wsml _"http://www.wsmo.org/2004/wsml#" ,
  dt _"http://www.wsmo.org/ontologies/dateTime/#" ,
  cu
  _"http://www.wsmo.org/2004/d3/d3.2/v0.1/20040628/resources/owlCurrencyMediator.wsml#"
  ,
  financial
  _"http://users.isoco.net/~slosada/ontologies/bankinter/FinancialOntology.wsml#" ,
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#" }

ontology
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml"

  importsOntology
    _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml"

concept CheckStock
  nonFunctionalProperties
    dc#description hasValue "A request to get detailed view of a given stock, based
on its name and code of the relevant market"
  endNonFunctionalProperties
  stockName impliesType (1 1) _string
  marketCode impliesType (1 1) _integer

concept CheckStocksOfIndex
  nonFunctionalProperties
    dc#description hasValue "A request to list all stocks of a given index"
  endNonFunctionalProperties
  indexName impliesType (1 1) _string
  marketCode impliesType (1 1) _integer

concept ListNews
  nonFunctionalProperties
    dc#description hasValue "A request to list news for a particular stock"
  endNonFunctionalProperties
  stockISIN impliesType (1 1) _string
  marketCode impliesType (1 1) _integer

```

```

concept NewsDetail
  nonFunctionalProperties
    dc#description hasValue "A request to get detailed view of a given news"
  endNonFunctionalProperties
  newsID impliesType (1 1) _string

concept CheckIndexes
  nonFunctionalProperties
    dc#description hasValue "A request to list available indexes"
  endNonFunctionalProperties

concept CheckQuotations
  nonFunctionalProperties
    dc#description hasValue "A request to list quotations for a given stock"
  endNonFunctionalProperties
  stockISIN impliesType (1 1) _string

concept ExecuteIfInTopFiveVariations
  nonFunctionalProperties
    dc#description hasValue "A request to execute an action on a stock if it
belongs to top five variations"
  endNonFunctionalProperties
  actionType impliesType (1 1) _string
  number impliesType (0 1) _integer
  customerID impliesType (1 1) _string
  portfolioID impliesType (0 1) _string
  stockISIN impliesType (1 1) _string
  market impliesType (1 1) _integer
  index impliesType (1 1) _string

axiom actionTypeConstraint
  nonFunctionalProperties
    dc#description hasValue "The type of actionType can only be 'BUY' or 'SELL' or
'ALERT' "
  endNonFunctionalProperties
  definedBy !-
    ?x memberOf ExecuteIfInTopFiveVariations or
    ?x memberOf ExecuteIfRecommendations or
    ?x memberOf ExecuteIfValueChanges or
    ?x memberOf ExecuteIfValueRises or
    ?x memberOf ExecuteIfStatistics or
    ?x memberOf ExecuteIfRatingChanges or
    ?x memberOf Action and
    ?x[actionType hasValue ?type] and
    (
      ?type = "BUY" or
      ?type = "SELL" or
      ?type = "ALERT"
    ).

concept ExecuteIfRecommendations
  nonFunctionalProperties
    dc#description hasValue "A request to execute an action on a stock if it
recommended by some or all of entities"
  endNonFunctionalProperties

```

```

actionType impliesType (1 1) _string
number impliesType (0 1) _integer
customerID impliesType (1 1) _string
portfolioID impliesType (0 1) _string
stockISIN impliesType (1 1) _string
quantifier impliesType (1 1) _string

axiom quantifierConstraint
  nonFunctionalProperties
    dc#description hasValue "The type of quantifier can only be 'SOME' or 'ALL'"
  endNonFunctionalProperties
  definedBy !-
    ?x memberOf ExecuteIfRecommendations and
    ?x[quantifier hasValue ?type] and
    (
      ?type = "SOME" or
      ?type = "ALL"
    ).

concept ExecuteIfValueChanges
  nonFunctionalProperties
    dc#description hasValue "A request to execute an action on a stock if its value
changes. Reference value is in the attribute valueChange"
  endNonFunctionalProperties
  actionType impliesType (1 1) _string
  number impliesType (0 1) _integer
  customerID impliesType (1 1) _string
  portfolioID impliesType (0 1) _string
  stockISIN impliesType (1 1) _string
  valueChange impliesType (1 1) _float
  changeDirection impliesType (1 1) _string

axiom changeDirectionConstraint
  nonFunctionalProperties
    dc#description hasValue "The type of changeDirection can only be '<' or '>' or
'=>' or '>=' or '=' "
  endNonFunctionalProperties
  definedBy !-
    ?x memberOf ExecuteIfInTopFiveVariations and
    ?x[changeDirection hasValue ?type] and
    (
      ?type = "<" or
      ?type = ">" or
      ?type = "<=" or
      ?type = ">=" or
      ?type = "="
    ).

concept ExecuteIfValueRises
  nonFunctionalProperties
    dc#description hasValue "A request to execute an action on a stock if its value
rises"
  endNonFunctionalProperties
  actionType impliesType (1 1) _string
  number impliesType (0 1) _integer

```

```

customerID impliesType (1 1) _string
portfolioID impliesType (0 1) _string
stockName impliesType (1 1) _string
marketCode impliesType (1 1) _string

concept SelectAndExecute
  nonFunctionalProperties
    dc#description hasValue "A request to select the stock with highest variation
and execute buy or sell action on it"
  endNonFunctionalProperties
  actionType impliesType (1 1) _string
  number impliesType (0 1) _integer
  customerID impliesType (1 1) _string
  portfolioID impliesType (0 1) _string
  marketCode impliesType (1 1) _string

concept ExecuteIfStatistics
  nonFunctionalProperties
    dc#description hasValue "A request to execute an action on a stock if its
statistics satisfy some constraint. Reference value is in the attribute valueChange"
  endNonFunctionalProperties
  actionType impliesType (1 1) _string
  number impliesType (0 1) _integer
  customerID impliesType (1 1) _string
  portfolioID impliesType (0 1) _string
  stockISIN impliesType (1 1) _string
  valueChange impliesType (1 1) _float
  constraint impliesType (1 1) _string
  statisticsName impliesType (1 1) _string

axiom constraintConstraint
  nonFunctionalProperties
    dc#description hasValue "The type of constraint can only be '<' or '>' or '=>'
or '>=' or '=' "
  endNonFunctionalProperties
  definedBy !-
    ?x memberOf ExecuteIfStatistics and
    ?x[constraint hasValue ?type] and
    (
      ?type = "<" or
      ?type = ">" or
      ?type = "<=" or
      ?type = ">=" or
      ?type = "="
    ).

concept CheckStatistics
  nonFunctionalProperties
    dc#description hasValue "A request to check statistics for a given stock"
  endNonFunctionalProperties
  stockISIN impliesType (1 1) _string

concept ExecuteIfRatingChanges
  nonFunctionalProperties
    dc#description hasValue "A request to execute some action on a stock if its
rating reaches value specified in ratingValue attribute"
  endNonFunctionalProperties

```

```

actionType impliesType (1 1) _string
number impliesType (0 1) _integer
customerID impliesType (1 1) _string
portfolioID impliesType (0 1) _string
stockISIN impliesType (1 1) _string
issuer impliesType (1 1) _string
ratingValue impliesType (1 1) _string

concept Action
  nonFunctionalProperties
    dc#description hasValue "A concept representing an executable action, such as
BUY, SELL or ALERT"
  endNonFunctionalProperties
  actionType impliesType (1 1) _string
  number impliesType (0 1) _integer
  customerID impliesType (1 1) _string
  portfolioID impliesType (0 1) _string
  stockISIN impliesType (1 1) _string
  market impliesType (1 1) _integer

concept GetRelevantVariations
  nonFunctionalProperties
    dc#description hasValue "A request to get top variations for a given index"
  endNonFunctionalProperties
  indexISIN impliesType (1 1) _string
  marketCode impliesType (1 1) _integer

concept GetRecommendations
  nonFunctionalProperties
    dc#description hasValue "A request to get recommendations for a given stock"
  endNonFunctionalProperties
  stockISIN impliesType (1 1) _string
  marketCode impliesType (1 1) _integer

concept getRatings
  nonFunctionalProperties
    dc#description hasValue "A request to get ratings for a given stock"
  endNonFunctionalProperties
  stockISIN impliesType (1 1) _string
  marketCode impliesType (1 1) _integer

concept recommendationContainer
  items impliesType sm#Recommendation

concept stockContainer
  items impliesType sm#Stock

concept ratingContainer
  items impliesType sm#Rating

concept statisticsContainer
  items impliesType sm#Statistics

concept SendAlert
  nonFunctionalProperties
    dc#description hasValue "A request to send an alert with specified message"

```

```

endNonFunctionalProperties
channel impliesType (1 *) _string
contactData impliesType (1 *) _string
message impliesType (1 1) _string

concept Confirmation
nonFunctionalProperties
  dc#description hasValue "Confirmation after executing BUY/SELL action"
endNonFunctionalProperties
returnCode impliesType (1 1) _string
price impliesType (0 1) _string
currency impliesType (0 1) _string
message impliesType (0 1) _string

```

Listing 2 Stock Market Ontology

```

namespace { _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#"
,
  xsd _"http://www.w3c.org/2001/XMLSchema#" ,
  dc _"http://purl.org/dc/elements/1.1#" ,
  foaf _"http://xmlns.com/foaf/01/" ,
  wsm1 _"http://www.wsmo.org/2004/wsml#" ,
  dt _"http://www.wsmo.org/ontologies/dateTime#" ,
  cu
_"http://www.wsmo.org/2004/d3/d3.2/v0.1/20040628/resources/owlCurrencyMediator.wsml#"
,
  financial
_"http://users.isoco.net/~slosada/ontologies/bankinter/FinancialOntology.wsml#" }

ontology _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml"

concept Stock

  isPartOfcompany impliesType (1 1) _string
  nonFunctionalProperties
    dc#description hasValue "Represents one fraction of one specific company
ownership"
  endNonFunctionalProperties
  isContinuous impliesType (1 1) _boolean
  nonFunctionalProperties
    dc#description hasValue "Reflects whether or not the stock can be
sell/bought in the Continuous Market"
  endNonFunctionalProperties
  hasFaceValue impliesType (1 1) _decimal
  nonFunctionalProperties
    dc#description hasValue "The amount expressed in currency representing the
ownership of the company"
  endNonFunctionalProperties
  hasPriceValue impliesType (1 1) _decimal
  nonFunctionalProperties
    dc#description hasValue "The price of the last transaction for that stock
in an specific market"
  endNonFunctionalProperties
  hasStockType impliesType (1 1) _string
  nonFunctionalProperties
    dc#description hasValue "Specific types of rights in the company"

```

```

endNonFunctionalProperties
hasBestOffer impliesType (1 1) BestOffer
nonFunctionalProperties
  dc#description hasValue "The best price that can be found in the market at
this point of time. Two figures are available: best selling price and best buying
price."
endNonFunctionalProperties
hasBestBuyPrice impliesType (1 1) _decimal
nonFunctionalProperties
  dc#description hasValue ""
endNonFunctionalProperties
hasBestSellPrice impliesType (1 1) _decimal
nonFunctionalProperties
  dc#description hasValue ""
endNonFunctionalProperties
hasMaximum impliesType (1 1) _decimal
nonFunctionalProperties
  dc#description hasValue "The highest price in an specific period of time"
endNonFunctionalProperties
hasMinimum impliesType (1 1) _decimal
nonFunctionalProperties
  dc#description hasValue "The lowest price in an specific period of time"
endNonFunctionalProperties
hasDividend impliesType (1 1) _decimal
nonFunctionalProperties
  dc#description hasValue "The amount of the last profit distribution that
corresponds to a single stock"
endNonFunctionalProperties
hasISIN impliesType (1 1) _integer
nonFunctionalProperties
  dc#description hasValue "The unique identifier of a stock"
endNonFunctionalProperties
hasVariation impliesType (1 1) _decimal
nonFunctionalProperties
  dc#description hasValue "The variation of a stock"
endNonFunctionalProperties
hasDirection impliesType (1 1) _string
nonFunctionalProperties
  dc#description hasValue "As the variation does not specify the direction of
movement, this attribute indicates whether it is increasing, steady or decreasing"
endNonFunctionalProperties
hasRecommendation impliesType _string
hasStockMarket impliesType (1 1) StockMarket

concept StockVariation

hasVariation impliesType (1 1) _decimal
nonFunctionalProperties
  dc#description hasValue "The variation of the stock"
endNonFunctionalProperties
hasDirection impliesType (1 1) _string
nonFunctionalProperties
  dc#description hasValue "As the variation does not specify the direction of
movement, this attribute indicates whether it is increasing, steady or decreasing"
endNonFunctionalProperties
hasRelativeVariation impliesType (1 1) _decimal
nonFunctionalProperties

```

```

        dc#description hasValue "The relative variation of the stock"
    endNonFunctionalProperties
    hasMarket impliesType (1 1) _string
    nonFunctionalProperties
        dc#description hasValue "The name of the market relevant for the variation"
    endNonFunctionalProperties
    hasStock impliesType (1 1) _string
    nonFunctionalProperties
        dc#description hasValue "The identifier of the stock relevant for the
variation"
    endNonFunctionalProperties

concept StockMarket

    hasStocks impliesType (1 *) Stock
    nonFunctionalProperties
        dc#description hasValue "Stocks that can be sell/bought there."
    endNonFunctionalProperties
    hasIndexes impliesType (1 *) _string
    nonFunctionalProperties
        dc#description hasValue "Figures that reflect the overall trend of the
market or part of it"
    endNonFunctionalProperties
    hasName impliesType (1 1) _string
    nonFunctionalProperties
        dc#description hasValue "The name of the market"
    endNonFunctionalProperties
    hasCurrency impliesType (1 1) _string
    nonFunctionalProperties
        dc#description hasValue "The currency in which the stock prices are
expressed"
    endNonFunctionalProperties
    hasBrokers impliesType (1 *) _string
    nonFunctionalProperties
        dc#description hasValue "The persons/companies allowed to operate in the
market"
    endNonFunctionalProperties
    hasSessions impliesType (1 *) _string
    nonFunctionalProperties
        dc#description hasValue "The dates the market has been/will be opened"
    endNonFunctionalProperties
    hasCountry impliesType (1 1) _string
    nonFunctionalProperties
        dc#description hasValue "The country where the market is established"
    endNonFunctionalProperties
    hasQuotations impliesType (1 *) Quotation

concept BuySellOrder

    hasBuySellDate impliesType (1 1) _dateTime
    nonFunctionalProperties
        dc#description hasValue "The date and time in which the operation has been
performed"
    endNonFunctionalProperties
    hasNumberOfStockBuySell impliesType (1 1) _integer
    nonFunctionalProperties
        dc#description hasValue "Amount of stocks dealt in the operation"

```

```

endNonFunctionalProperties
hasISIN impliesType (1 1) _string
nonFunctionalProperties
  dc#description hasValue "The identifier of the stocks sell/bought"
endNonFunctionalProperties
isBuyOrSell impliesType (1 1) _string
nonFunctionalProperties
  dc#description hasValue "Determines the type of operation"
endNonFunctionalProperties
hasCurrency impliesType (1 1) _string
nonFunctionalProperties
  dc#description hasValue "The currency used to express the amount of the
operation. Both the transaction currency and the portfolio currency are required"
endNonFunctionalProperties
hasSession impliesType (1 1) Session
nonFunctionalProperties
  dc#description hasValue "The secuencial number of session"
endNonFunctionalProperties
isPerformed impliesType (1 1) _boolean
nonFunctionalProperties
  dc#description hasValue "Whether the order is performed or not"
endNonFunctionalProperties
hasDepositary impliesType (1 1) Depositary
nonFunctionalProperties
  dc#description hasValue "The person/company in charge of keeping safe the
stocks"
endNonFunctionalProperties
hasPortfolioNumber impliesType (1 1) Porfolio
nonFunctionalProperties
  dc#description hasValue "The account number (defined by the depositary)
where the stocks are kept"
endNonFunctionalProperties
hasUserID impliesType (1 1) User
nonFunctionalProperties
  dc#description hasValue "Determines the property of the shares"
endNonFunctionalProperties
hasMarket impliesType (1 1) StockMarket
nonFunctionalProperties
  dc#description hasValue "The market relevant for the order"
endNonFunctionalProperties

concept BestBuySellOrder subConceptOf { BuySellOrder, _BuySellOrder}

  executedInSession impliesType (1 1) Session
  nonFunctionalProperties
    dc#description hasValue "The secuencial number of session"
  endNonFunctionalProperties

concept ConditionedBuySellOrder subConceptOf { BuySellOrder, _BuySellOrder}

  hasExpirationDate impliesType (1 1) _dateTime
  nonFunctionalProperties
    dc#description hasValue "The date in which the order is cancelled if not
performed before"
  endNonFunctionalProperties

concept LastPrice

```

```

hasDate impliesType (1 1) _dateTime
nonFunctionalProperties
  dc#description hasValue "Date and time of historical values"
endNonFunctionalProperties
hasPriceValue impliesType (1 1) _decimal
nonFunctionalProperties
  dc#description hasValue "Historical values"
endNonFunctionalProperties

```

concept Session

```

hasDate impliesType (1 1) _dateTime
nonFunctionalProperties
  dc#description hasValue "The date of the session"
endNonFunctionalProperties
hasSessionValue impliesType (1 1) _decimal
nonFunctionalProperties
  dc#description hasValue "The amount of shares traded in one period of time"
endNonFunctionalProperties
hasClosingTime impliesType (1 1) _dateTime
nonFunctionalProperties
  dc#description hasValue "The time at what the market is closed"
endNonFunctionalProperties
hasOpeningTime impliesType (1 1) _dateTime
nonFunctionalProperties
  dc#description hasValue "The time at what the market is opened"
endNonFunctionalProperties
hasActualTime impliesType (1 1) _dateTime
hasQuotation impliesType _Quotation

```

concept Index

```

hasName impliesType (1 1) _string
nonFunctionalProperties
  dc#description hasValue "Index name (i.e.;Nasdaq, Dow Jones)"
endNonFunctionalProperties
hasInitialValue impliesType (1 1) _decimal
nonFunctionalProperties
  dc#description hasValue "The value of the index when a session starts"
endNonFunctionalProperties
hasActualValue impliesType (1 1) _decimal
nonFunctionalProperties
  dc#description hasValue "The latest value of the index"
endNonFunctionalProperties
hasIndexID impliesType (1 1) _string
nonFunctionalProperties
  dc#description hasValue "Index identification"
endNonFunctionalProperties
hasStockWeights impliesType (1 1) StockWeight
nonFunctionalProperties
  dc#description hasValue "The stocks that compose the index and weight in
index"
endNonFunctionalProperties
hasMarket impliesType (1 1) StockMarket
nonFunctionalProperties

```

```

        dc#description hasValue "The market to what the index refers"
    endNonFunctionalProperties
    hasTopVariations impliesType (10 10) Stock
    nonFunctionalProperties
        dc#description hasValue "The stocks with top ten variations in the index"
    endNonFunctionalProperties

concept IndexSession

    hasIndex impliesType (1 1) Index
    nonFunctionalProperties
        dc#description hasValue "Weighted average of a number of stock prices"
    endNonFunctionalProperties
    hasIndexSessionValue impliesType (1 1) _decimal
    nonFunctionalProperties
        dc#description hasValue "Value of index in a session"
    endNonFunctionalProperties

concept StockWeight

    hasWeight impliesType (1 1) _decimal
    nonFunctionalProperties
        dc#description hasValue "Weight of stock in index"
    endNonFunctionalProperties
    hasIndex impliesType (1 1) Index
    nonFunctionalProperties
        dc#description hasValue "Weighted average of a number of stock prices"
    endNonFunctionalProperties
    hasStock impliesType (1 1) Stock
    nonFunctionalProperties
        dc#description hasValue "Stotk that refernces the index"
    endNonFunctionalProperties

concept Depositary

    hasDepositaryID impliesType (1 1) _string
    nonFunctionalProperties
        dc#description hasValue "The identificacion of the Depositary"
    endNonFunctionalProperties
    CIF impliesType (1 1) _string
    hasCommisions impliesType (1 *) _string
    nonFunctionalProperties
        dc#description hasValue "The commissions the Depositary applies"
    endNonFunctionalProperties
    hasBroker impliesType (1 1) Broker
    nonFunctionalProperties
        dc#description hasValue "The broker the depopsitary works for"
    endNonFunctionalProperties

concept Broker

    CIF impliesType (1 1) _string
    nonFunctionalProperties
        dc#description hasValue ""
    endNonFunctionalProperties
    hasNonOfficialTaxes impliesType (1 *) _string

```

```

nonFunctionalProperties
  dc#description hasValue ""
endNonFunctionalProperties
hasStockMarkets impliesType (1 *) StockMarket
nonFunctionalProperties
  dc#description hasValue ""
endNonFunctionalProperties

concept Rating

hasRatingValue impliesType (1 1) _string
nonFunctionalProperties
  dc#description hasValue ""
endNonFunctionalProperties
hasIssuer impliesType (1 1) _string
nonFunctionalProperties
  dc#description hasValue ""
endNonFunctionalProperties
hasStockISIN impliesType (1 1) _string

concept Statistics

hasStatisticsValue impliesType (1 1) _float
nonFunctionalProperties
  dc#description hasValue ""
endNonFunctionalProperties
hasName impliesType (1 1) _string
nonFunctionalProperties
  dc#description hasValue ""
endNonFunctionalProperties
hasStockISIN impliesType (1 1) _string

concept Portfolio

hasPortFolioID impliesType (1 1) _string
nonFunctionalProperties
  dc#description hasValue "The number (assigned by the Depository) where the
stocks are deposited"
endNonFunctionalProperties
hasStocks impliesType (1 *) StockPortfolio
nonFunctionalProperties
  dc#description hasValue "Identificaction of the Stocks deposited in a
Portfolio"
endNonFunctionalProperties
hasUserID impliesType (1 1) financial#User
nonFunctionalProperties
  dc#description hasValue "Identification of the client owner of the stocks
deposited"
endNonFunctionalProperties
hasDepository impliesType (1 1) Depository
nonFunctionalProperties
  dc#description hasValue "Identification of the depository"
endNonFunctionalProperties
hasAssociatedAccount impliesType (1 1) financial#SavingAccount
nonFunctionalProperties
  dc#description hasValue "Account asociated to this portfolio"

```

```

endNonFunctionalProperties
hasCustomer impliesType (1 1) financial#Customer
nonFunctionalProperties
  dc#description hasValue "Identification of the client owner of the stocks
deposited"
endNonFunctionalProperties

concept StockMaxValue

  initPeriod impliesType (1 1) _dateTime
  nonFunctionalProperties
    dc#description hasValue ""
  endNonFunctionalProperties
  endPeriod impliesType (1 1) _dateTime
  nonFunctionalProperties
    dc#description hasValue ""
  endNonFunctionalProperties
  value impliesType (1 1) _decimal
  nonFunctionalProperties
    dc#description hasValue ""
  endNonFunctionalProperties

concept Dividend

  amount impliesType (1 1) _decimal
  nonFunctionalProperties
    dc#description hasValue ""
  endNonFunctionalProperties
  date impliesType _dateTime
  nonFunctionalProperties
    dc#description hasValue ""
  endNonFunctionalProperties

concept StockPortfolio

  hasStock impliesType (1 1) Stock
  nonFunctionalProperties
    dc#description hasValue "The stock"
  endNonFunctionalProperties
  stocksNumber impliesType (1 1) _integer
  nonFunctionalProperties
    dc#description hasValue "Number of the stocks"
  endNonFunctionalProperties

concept Quotation

  stockMarket impliesType (1 1) StockMarket
  stock impliesType (1 1) Stock
  date impliesType (1 1) _dateTime
  volume impliesType (1 1) _double

concept Recommendation

  hasMarket impliesType (1 1) _integer
  hasEntity impliesType (1 1) _string
  hasRecommendationValue impliesType (1 1) _string

```

```
hasStockISIN impliesType (1 1) _string
```

3.2 Goals

The following listings provide semantic descriptions of goals describing users' wishes specified in the use case instances.

Listing 3 Goal UCI1

```
wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalVariationDependentAction.
wsml#",
  dc _"http://purl.org/dc/elements/11#",
  foaf _"http://xmlns.com/foaf/01/",
  xsd _"http://www.w3c.org/2001/XMLSchema#",
  wsml _"http://www.wsmo.org/2004/wsml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"}

goal
_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalVariationDependentAction.w
sml"

nfp
  dc#title hasValue "Goal for variation dependent actions"
  dc#type hasValue _"http://www.wsmo.org/2004/d2#goals"
  dc#description hasValue "If the value of a specified stock belongs to the top
five variations on a specified market, then the system execute some action"
  dc#contributor hasValue ""
  dc#date hasValue _date(2005,11,24)
  dc#format hasValue "text/plain"
  dc#language hasValue "en-US"
  dc#rights hasValue _"http://www.isoco.com/privacy.html"
  wsml#version hasValue "$Revision: 03 $"
endnfp

importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml
#"}

capability
_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalVariationDependentAction#c
apability"

sharedVariables {?marketISIN, ?stockISIN}
/*
  ?marketISIN - the name of the market
  ?stockISIN - the identifier of the stock
*/
```

```

precondition
  nfp
    dc#description hasValue "The stock specified by the user has to belong to the
top five variations on a specified market"
  endnfp
  definedBy
    ?stock [hasName hasValue ?stockName,
            hasISIN hasValue ?stockISIN] memberOf sm#Stock and
    ?market [hasISIN hasValue ?marketISIN,
            hasTopVariations hasValue ?stock] memberOf sm#StockMarket .

postcondition
  nfp
    dc#description hasValue "Result of the web service is execution of the
desired action"
  endnfp
  definedBy
    ?action[actionType hasValue ?actionType,
            number hasValue ?number,
            customerID hasValue ?customerID,
            portfolioID hasValue ?portfolioID,
            stockISIN hasValue ?stockISIN,
            market hasValue ?marketISIN] memberOf smp#action .

```

Listing 4 Goal UCI2

```

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalValueDependentAction2.wsm
l#",
  dc _"http://purl.org/dc/elements/11#",
  foaf _"http://xmlns.com/foaf/01/",
  xsd _"http://www.w3c.org/2001/XMLSchema#",
  wsml _"http://www.wsmo.org/2004/wsml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#" }

goal
_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalValueDependentAction2.wsm
l#"

  nfp
    dc#title hasValue "Goal for value dependent actions 2"
    dc#type hasValue _"http://www.wsmo.org/2004/d2#goals"
    dc#description hasValue "A Goal to execute an action on a stock if its value
rises"
    dc#contributor hasValue ""
    dc#date hasValue _date(2005,11,24)
    dc#format hasValue "text/plain"
    dc#language hasValue "en-US"
    dc#rights hasValue _"http://www.isoco.com/privacy.html"
    wsml#version hasValue "$Revision: 03 $"

```

```

endnfp

importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
    _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml
#"}

capability
_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalValueDependentAction2#capa
bility"

sharedVariables {?stockName, ?marketISIN, ?stockISIN}
/*
    ?stockName - the name of the stock
    ?marketISIN - the name of the market
    ?stockISIN - the identifier of the stock
*/

precondition
nfp
    dc:description hasValue "The value of the stock specified by the user has to
rise"
endnfp
definedBy
    ?stock [hasName hasValue ?stockName,
            hasISIN hasValue ?stockISIN] memberOf sm#Stock and
    ?market [hasISIN hasValue ?marketISIN] memberOf sm#StockMarket .
    ?stockVariation [hasStock hasValue ?stock,
                    hasMarket hasValue ?market,
                    hasRelativeVariation hasValue
?relativeVariation] memberOf sm#StockVariation and
    ?relativeVariation > 0 .

postcondition
nfp
    dc:description hasValue "Result of the web service is execution of the
desired action"
endnfp
definedBy
    ?action[actionType hasValue ?actionType,
            number hasValue ?number,
            customerID hasValue ?customerID,
            portfolioID hasValue ?portfolioID,
            stockISIN hasValue ?stockISIN,
            market hasValue ?marketISIN] memberOf smp#action .

```

Listing 5 Goal UCI3

```

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalRecommendationDependentAc
tion.wsml#",
    dc _"http://purl.org/dc/elements/11#",
    foaf _"http://xmlns.com/foaf/01/",
    xsd _"http://www.w3c.org/2001/XMLSchema#",

```

```

wsml _"http://www.wsmo.org/2004/wsml#",
sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"}

goal
_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalRecommendationDependentAction.wsml"

nfp
  dc#title hasValue "Goal for recommendation dependent actions"
  dc#type hasValue _"http://www.wsmo.org/2004/d2#goals"
  dc#description hasValue "A Goal to execute an action on a stock based on its recommendations"
  dc#contributor hasValue ""
  dc#date hasValue _date(2005,11,24)
  dc#format hasValue "text/plain"
  dc#language hasValue "en-US"
  dc#rights hasValue _"http://www.isoco.com/privacy.html"
  wsml#version hasValue "$Revision: 03 $"
endnfp

importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"}

capability
_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalRecommendationDependentAction#capability"
  sharedVariables {?marketISIN, ?stockISIN, ?actionType}
  /*
  ?marketISIN - the name of the market
  ?stockISIN - the identifier of the stock
  ?actionType - BUY or SELL
  */

precondition
  nfp
    dc#description hasValue "All recommendations regarding stock specified by the user have to advise some action"
  endnfp
  definedBy
    ?stock [hasName hasValue ?stockName,
      hasISIN hasValue ?stockISIN] memberOf sm#Stock and
    ?market [hasISIN hasValue ?marketISIN] memberOf sm#StockMarket and
    forall {?x} (?stock [hasRecommendation hasValue ?x] and ?x = ?actionType).

postcondition
  nfp
    dc#description hasValue "Result of the web service is execution of the desired action"
  endnfp
  definedBy
    ?action[actionType hasValue ?actionType,

```

```

number hasValue ?number,
customerID hasValue ?customerID,
portfolioID hasValue ?portfolioID,
stockISIN hasValue ?stockISIN,
market hasValue ?marketISIN] memberOf smp#action .

```

Listing 6 Goal UCI4

```

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace
{
  _"http://users.isoco.net/~slosada/ontologies/bankinter/GoalValueDependentAction1.wsm
  l#",
  dc _"http://purl.org/dc/elements/11#",
  foaf _"http://xmlns.com/foaf/01/",
  xsd _"http://www.w3c.org/2001/XMLSchema#",
  wsml _"http://www.wsmo.org/2004/wsml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  smp
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"
}

goal
_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalValueDependentAction1.wsm
l#"

nfp
  dc#title hasValue "Goal for value dependent action 1"
  dc#type hasValue _"http://www.wsmo.org/2004/d2#goals"
  dc#description hasValue "A Goal to execute an action on a stock based on its
  value"
  dc#contributor hasValue ""
  dc#date hasValue _date(2005,11,24)
  dc#format hasValue "text/plain"
  dc#language hasValue "en-US"
  dc#rights hasValue _"http://www.isoco.com/privacy.html"
  wsml#version hasValue "$Revision: 03 $"
endnfp

importsOntology {
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml
  #"}

capability
_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalValueDependentAction1#capa
bility"

sharedVariables {?stock, ?stockName, ?marketISIN, ?stockISIN}
/*
  ?stock - instance of the relevant stock
  ?stockName - the name of the stock
  ?marketISIN - the name of the market
  ?stockISIN - the identifier of the stock
*/

```

```

precondition
  nfp
    dc:description hasValue "The value of the stock specified by the user has to
fall more than specified user's value"
  endnfp
  definedBy
    ?stock [hasName hasValue ?stockName,
            hasISIN hasValue ?stockISIN] memberOf sm#Stock and
    ?market [hasISIN hasValue ?marketISIN] memberOf sm#StockMarket .
    ?stockVariation [hasStock hasValue ?stock,
                    hasMarket hasValue ?market,
                    hasRelativeVariation hasValue
?relativeVariation] memberOf sm#StockVariation and
    ?relativeVariation < ?userValue .

postcondition
  nfp
    dc:description hasValue "Result of the web service is execution of the
desired action"
  endnfp
  definedBy
    ?action[actionType hasValue ?actionType,
            number hasValue ?number,
            customerID hasValue ?customerID,
            portfolioID hasValue ?portfolioID,
            stockISIN hasValue ?stockISIN,
            market hasValue ?marketISIN] memberOf smp#action .

```

Listing 7 Goal UCI5

```

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalRatingDependentAction.wsm
l#",
  dc _"http://purl.org/dc/elements/11#",
  foaf _"http://xmlns.com/foaf/01/",
  xsd _"http://www.w3c.org/2001/XMLSchema#",
  wsml _"http://www.wsmo.org/2004/wsml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"}

goal
_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalRatingDependentAction.wsm
l#"

nfp
  dc#title hasValue "Goal for rating dependent actions"
  dc#type hasValue _"http://www.wsmo.org/2004/d2#goals"
  dc#description hasValue "A Goal to execute an action on a stock based on its
rating"
  dc#contributor hasValue ""
  dc#date hasValue _date(2005,11,14)
  dc#format hasValue "text/plain"
  dc#language hasValue "en-US"

```

```

dc#rights hasValue _"http://www.isoco.com/privacy.html"
wsm1#version hasValue "$Revision: 01 $"
endnfp

importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsm1#",
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsm1#"}

capability
_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalRatingDependentAction#capability"
  sharedVariables {?stockISIN, ?market}

precondition
  nfp
  dc#description hasValue "The rating the stock reaches the value specified by the user"
  endnfp
  definedBy
    ?stock[hasName hasValue ?stockName,
      hasISIN hasValue ?stockISIN,
      hasStockMarket hasValue ?market] memberOf sm#Stock and
    ?rating[hasStock hasValue ?stock,
      hasIssuer hasValue ?issuer,
      hasRatingValue hasValue ?ratingValue] memberOf sm#Rating.

postcondition
  nfp
  dc#description hasValue "Result of the web service is execution of the desired action"
  endnfp
  definedBy
    ?action[actionType hasValue ?actionType,
      number hasValue ?number,
      customerID hasValue ?customerID,
      portfolioID hasValue ?portfolioID,
      stockISIN hasValue ?stockISIN,
      market hasValue ?market] memberOf smp#action .

```

Listing 8 Goal UCI6

```

wsm1Variant _"http://www.wsmo.org/wsm1/wsm1-syntax/wsm1-rule"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalStatisticsDependentAction.wsm1#",
  dc _"http://purl.org/dc/elements/11#",
  foaf _"http://xmlns.com/foaf/01/",
  xsd _"http://www.w3c.org/2001/XMLSchema#",
  wsm1 _"http://www.wsmo.org/2004/wsm1#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsm1#",

```

```

smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"}

goal
_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalStatisticsDependentAction.
wsml"

nfp
  dc#title hasValue "Goal for statistics dependent actions"
  dc#type hasValue _"http://www.wsmo.org/2004/d2#goals"
  dc#description hasValue "A Goal to execute an action on a stock based on its
statistics"
  dc#contributor hasValue ""
  dc#date hasValue _date(2005,11,14)
  dc#format hasValue "text/plain"
  dc#language hasValue "en-US"
  dc#rights hasValue _"http://www.isoco.com/privacy.html"
  wsml#version hasValue "$Revision: 01 $"
endnfp

importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml
#"}

capability
_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalStatisticsDependentAction#
capability"
  sharedVariables {?stockISIN, ?marketISIN}

precondition
  nfp
    dc#description hasValue "Value of some stock statistics is bigger than the
value specified by the user"
  endnfp
  definedBy
    ?stock[hasName hasValue ?stockName,
      hasISIN hasValue ?stockISIN,
      hasStockMarket hasValue ?market] memberOf sm#Stock and
    ?market[hasISIN hasValue ?marketISIN] and
    ?statistics[hasStock hasValue ?stock,
      hasName hasValue ?statName,
      hasStatisticsValue hasValue ?value] memberOf sm#Statistics and
    ?value > ?userValue
    .

postcondition
  nfp
    dc#description hasValue "Result of the web service is execution of the
desired action"
  endnfp
  definedBy
    ?action[actionType hasValue ?actionType,
      number hasValue ?number,
      customerID hasValue ?customerID,

```

```

portfolioID hasValue ?portfolioID,
stockISIN hasValue ?stockISIN,
market hasValue ?marketISIN] memberOf smp#action .

```

Listing 9 Goal UCI7

```

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace
{
  _"http://users.isoco.net/~slosada/ontologies/bankinter/GoalSelectAndExecute.wsml#",
  dc _"http://purl.org/dc/elements/11#",
  foaf _"http://xmlns.com/foaf/01/",
  xsd _"http://www.w3c.org/2001/XMLSchema#",
  wsml _"http://www.wsmo.org/2004/wsml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  smp
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"
}

goal
_"http://users.isoco.net/~slosada/ontologies/bankinter/GoalSelectAndExecute.wsml"

  nfp
    dc#title hasValue "Goal for select and execute orders"
    dc#type hasValue _"http://www.wsmo.org/2004/d2#goals"
    dc#description hasValue "A Goal for selecting stocks and executing on them Buy or
Sell orders"
    dc#contributor hasValue ""
    dc#date hasValue _date(2005,11,14)
    dc#format hasValue "text/plain"
    dc#language hasValue "en-US"
    dc#rights hasValue _"http://www.isoco.com/privacy.html"
    wsml#version hasValue "$Revision: 01 $"
  endnfp

  importsOntology {
    _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
    _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml
#"}

  capability
  _"http://users.isoco.net/~slosada/ontologies/bankinter/GoalSelectAndExecute#capabilit
y"

  //no precondition

  postcondition
  nfp
    dc#description hasValue "Result of the web service is execution of the
desired action on the stock which has the highest variation"
  endnfp
  definedBy
  ?stock[hasISIN hasValue ?stockISIN,
hasVariation hasValue ?variation] memberOf sm#Stock and

```

```
(forall ?stocks (?stocks[hasVariation hasValue ?variations] memberOf sm#Stock
and      (?variations =< ?variation)))
and
?action[actionType hasValue ?actionType,
         number hasValue ?number,
         customerID hasValue ?customerID,
         portfolioID hasValue ?portfolioID,
         stockISIN hasValue ?stockISIN,
         market hasValue ?market] memberOf smp#action .
```

3.3 Web Services

The following listings present semantic descriptions of Web Services. The use case instances 2 and 4 can be realized with one Semantic Web Service.

Listing 10 Web Service UCI1

```
wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WSVariationDependentAction.wsml#",
  dc _"http://purl.org/dc/elements/11#",
  foaf _"http://xmlns.com/foaf/01/",
  xsd _"http://www.w3c.org/2001/XMLSchema#",
  wsml _"http://www.wsmo.org/2004/wsml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  fin
_"http://users.isoco.net/~slosada/ontologies/bankinter/FinancialOntology.wsml#"}

webService
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSVariationDependentAction.wsml"

nfp
  dc#title hasValue "Web Service for variation dependent actions"
  dc#type hasValue _"http://www.wsmo.org/2004/d2#webservice"
  dc#description hasValue "If the stock value belongs to the top five variations, then the system execute some action"
  dc#contributor hasValue ""
  dc#date hasValue _date(2005,11,16)
  dc#format hasValue "text/plain"
  dc#language hasValue "en-US"
  dc#rights hasValue _"http://www.isoco.com/privacy.html"
  wsml#version hasValue "$Revision: 01 $"
endnfp

importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#"}
importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/FinancialOntology.wsml#"}

```

```

capability
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSVariationDependentAction#capability"

sharedVariables {?stockISIN, ?marketName, ?direction, ?action, ?stock, ?value,
?stocksNumber, ?customer, ?portfolio, ?market, ?savingAccount, ?number, ?balance}
/*
    ?stockISIN - the identifier of the stock
    ?marketName - the name of the relevant market
    ?direction - indicates whether the user is interested in increasing variation,
decreasing variation
                    or it is not important to him (_#)
    ?action - SELL | BUY | ALERT
    ?stock - the relevant stock
    ?value - value of a single relevant stock
    ?stocksNumber - number of relevant stocks currently in portfolio
    ?customer - the customer relevant for the operation
    ?portfolio - the portfolio, from which the action is to be performed
    ?market - market for the relevant stock
    ?savingAccount - the account of the customer
    ?number - number of stocks to be bought or sold by the user
    ?balance - balance on the account
*/

precondition
nfp
    dc#description hasValue "The system has to know the portfolio ID, from which
to execute action, and the user, to check if the portfolio belongs to the user. The
stock name, number of them and market have to be known to the system, if it is going
to check whether it belongs to the top five variations and then do something with it.
The action has to be specified as well. The stock value has to belong to top five
variations. If not, no action is performed. In case of SELL action user has to
possess at least the specified number of stocks in his portfolio. In case of BUY
action, the value of stocks the user intends to buy must not be bigger than the
balance on user's account"
endnfp
definedBy
    ?stock [hasISIN hasValue ?stockISIN] memberOf sm#Stock and
    ?market [hasName hasValue ?marketName,
                hasTopVariations hasValue ?stock] memberOf sm#StockMarket and
    ?stock [hasStockMarket hasValue ?market,
                hasPriceValue hasValue ?value] and
    ?direction memberOf {"increasing", "", "decreasing"} and
    ?stockVariation [hasStock hasValue ?stock,
                    hasMarket hasValue ?market] memberOf
sm#StockVariation and
    ?action memberOf {"BUY", "SELL", "ALERT"} and
    ?customer memberOf fin#Customer and
    ?portfolio [hasPortFolioID hasValue ?portfolioID,
                hasCustomer hasValue ?customer,
                hasAssociatedAccount hasValue ?savingAccount] memberOf
sm#Portfolio and
    ?savingAccount memberOf fin#SavingAccount and
    (
    (
    /*
    In case of SELL action user has to possess at least the specified number of
stocks in his portfolio.
    */

```

```

?action = "SELL" and
?portfolio [hasStocks hasValue ?stockPorfolio] and
    ?stockPortfolio [hasStocks hasValue ?stock,
                    stocksNumber hasValue ?stocksNumber] and
        ?stocksNumber >= ?number
    ) or
(
/*
In case of ALERT no additional precondition is assumed
*/
?action = "ALERT"
) or
(
/*
In case of BUY action, the value of stocks the user intends to buy must not be
bigger than the balance on user's account
*/
?action = "BUY" and
?savingAccount [balanceAccount hasValue ?balance] and
?balance >= (?number * ?value)
)
).

postcondition
    nfp
        dc#description hasValue "In case of SELL, the user deposes of the number of
stocks from the portfolio. The value is added to his savings account. In case of BUY,
the number of stocks is added to portfolio. The value is subtracted from the savings
account"
    endnfp
    definedBy
    (
/*
In case of SELL, the user deposes of the number of stocks from the portfolio.
The value is added to his savings account.
*/
?stockPortfolio [stocksNumber hasValue (?stocksNumber - ?number)] and
?savingAccount [balanceAccount hasValue (?balance + (?number * ?value))]
) or
(
/*
In case of BUY, the number of stocks is added to portfolio.
The value is subtracted from the savings account
*/
?stockPortfolio [stocksNumber hasValue (?stocksNumber + ?number)] and
?savingAccount [balanceAccount hasValue (?balance - (?number * ?value))]
) .

```

Listing 11 Web Service UCI2 and UCI4

```

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WSValueDependentAction.wsml#"
,
    dc _"http://purl.org/dc/elements/11#",

```

```

foaf _"http://xmlns.com/foaf/01/",
xsd _"http://www.w3c.org/2001/XMLSchema#",
wsml _"http://www.wsmo.org/2004/wsml#",
sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
fin
_"http://users.isoco.net/~slosada/ontologies/bankinter/FinancialOntology.wsml#"}

webService
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSValueDependentAction.wsml"

nfp
  dc#title hasValue "Web Service for value dependent actions"
  dc#type hasValue _"http://www.wsmo.org/2004/d2#webservice"
  dc#description hasValue "If the value of some stock changes then execute BUY |
SELL | ALERT"
  dc#contributor hasValue ""
  dc#date hasValue _date(2005,11,17)
  dc#format hasValue "text/plain"
  dc#language hasValue "en-US"
  dc#rights hasValue _"http://www.isoco.com/privacy.html"
  wsml#version hasValue "$Revision: 01 $"
endnfp

  importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#"
  importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/FinancialOntology.wsml#"

  capability
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSValueDependentAction#capabil
ity"

  sharedVariables {?stockISIN, ?stock, ?marketName, ?market, ?customer, ?portfolioID,
?stockPortfolio, ?action, ?changeDirection, ?relativeVariation, ?valueChange,
?number, ?savingAccount, ?stocksNumber, ?value, ?balance}

/*
?stockISIN - the identifier of the relevant stock
?stock - the relevant stock
?marketName - the name of the relevant market
?market - the relevant market
?customer - customer relevant for the transaction
?portfolioID - the identifier of the customer's portfolio
?stockPortfolio - the relevant stock in the portfolio of the customer
?action - desired action BUY | SELL | ALERT
?changeDirection - direction of the value's change; one of <, >, <=, >=
?relativeVariation - the actual change of value of the given stock
?valueChange - the value to compare the actual change with
?number - number of stocks to be bought / sold
?savingAccount - the account for value transfer
?stocksNumber - the initial number of stocks in the portfolio
?value - the initial price value of a single stock
?balance - balance on the savings account
*/

precondition
  nfp
    dc#description hasValue "The necessary information to be provided to the

```

system includes: stock, market, kind of action, customer, portfolio. The stock has to belong to the specified market and the portfolio to the customer. The relative variation of the stock has to satisfy the condition provided by the customer. In case of SELL action user has to possess at least the specified number of stocks in his portfolio. In case of BUY action, the value of stocks the user intends to buy must not be bigger than the balance on user's account"

```

endnfp
definedBy
    ?stock [hasISIN hasValue ?stockISIN] memberOf sm#Stock and
    ?market [hasName hasValue ?marketName] memberOf sm#StockMarket and
    ?stock [hasStockMarket hasValue ?market,
            hasPriceValue hasValue ?value] and
    ?action memberOf {"BUY", "SELL", "ALERT"} and
    ?customer memberOf fin#Customer and
    ?portfolio [hasPortFolioID hasValue ?portfolioID,
               hasCustomer hasValue ?customer] memberOf sm#Portfolio
and
    ?stockVariation [hasStock hasValue ?stock,
                    hasMarket hasValue ?market,
                    hasRelativeVariation hasValue
?relativeVariation] memberOf sm#StockVariation and
    (
        ((?changeDirection = "<") and (?relativeVariation < ?valueChange)) or
        ((?changeDirection = "<=") and (?relativeVariation =< ?valueChange)) or
        ((?changeDirection = ">") and (?relativeVariation > ?valueChange)) or
        ((?changeDirection = ">=") and (?relativeVariation >= ?valueChange))
    ) and
    (
        (
            /*
            In case of SELL action user has to possess at least the specified number of
            stocks in his portfolio.
            */
            ?action = "SELL" and
            ?portfolio [hasStocks hasValue ?stockPorfolio] and
                ?stockPortfolio [hasStocks hasValue ?stock,
                               stocksNumber hasValue ?stocksNumber] and
                ?stocksNumber >= ?number
            ) or
        (
            /*
            In case of ALERT no additional precondition is assumed
            */
            ?action = "ALERT"
            ) or
        (
            /*
            In case of BUY action, the value of stocks the user intends to buy must not be
            bigger than the balance on user's account
            */
            ?action = "BUY" and
            ?portfolio [hasAssociatedAccount hasValue ?savingAccount] and
            ?savingAccount [balanceAccount hasValue ?balance] memberOf fin#SavingAccount and
            ?balance >= (?number * ?value)
            )
    ).

postcondition

```

```

nfp
  dc:description hasValue "In case of SELL, the user deposes of the number of
stocks from the portfolio. The value is added to his savings account. In case of BUY,
the number of stocks is added to portfolio. The value is subtracted from the savings
account"
  endnfp
  definedBy
  (
  /*
  In case of SELL, the user deposes of the number of stocks from the portfolio.
  The value is added to his savings account.
  */
  ?stockPortfolio [stocksNumber hasValue (?stocksNumber - ?number)] and
  ?savingsAccount [balanceAccount hasValue (?balance + (?number * ?value))]
  ) or
  (
  /*
  In case of BUY, the number of stocks is added to portfolio.
  The value is subtracted from the savings account
  */
  ?stockPortfolio [stocksNumber hasValue (?stocksNumber + ?number)] and
  ?savingsAccount [balanceAccount hasValue (?balance - (?number * ?value))]
  ).

```

Listing 12 Web Service UCI3

```

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WSRecommendationDependentActi
on.wsml#",
  dc _"http://purl.org/dc/elements/11#",
  foaf _"http://xmlns.com/foaf/01/",
  xsd _"http://www.w3c.org/2001/XMLSchema#",
  wsmo _"http://www.wsmo.org/2004/wsml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  fin
_"http://users.isoco.net/~slosada/ontologies/bankinter/FinancialOntology.wsml#"}

webService
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSRecommendationDependentActio
n.wsml"

nfp
  dc:title hasValue "Web Service for recommendation dependent actions"
  dc:type hasValue _"http://www.wsmo.org/2004/d2#webservice"
  dc:description hasValue "Execute BUY | SELL if recomendation of some | all
entities is BUY | SELL"
  dc:contributor hasValue ""
  dc:date hasValue _date(2005,11,18)
  dc:format hasValue "text/plain"
  dc:language hasValue "en-US"
  dc:rights hasValue _"http://www.isoco.com/privacy.html"
  wsmo:version hasValue "$Revision: 01 $"
  endnfp

```

```

    importsOntology {
    _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#"
    importsOntology {
    _"http://users.isoco.net/~slosada/ontologies/bankinter/FinancialOntology.wsml#"

    capability
    _"http://users.isoco.net/~slosada/ontologies/bankinter/WSRecommendationDependentActio
n#capability"

    sharedVariables {?stockISIN, ?stock, ?marketName, ?market, ?customer, ?portfolioID,
?stockPortfolio, ?action, ?changeDirection, ?relativeVariation, ?valueChange,
?number, ?savingAccount, ?stocksNumber, ?value, ?balance}

    /*
    ?stockISIN - the identifier of the relevant stock
    ?stock - the relevant stock
    ?marketName - the name of the relevant market
    ?market - the relevant market
    ?customer - customer relevant for the transaction
    ?portfolioID - the identifier of the customer's portfolio
    ?stockPortfolio - the relevant stock in the portfolio of the customer
    ?action - desired action BUY | SELL | ALERT
    ?quantifier - some | all - regarding the recommendations necessary to execute
action
    ?number - number of stocks to be bought / sold
    ?savingAccount - the account for value transfer
    ?stocksNumber - the initial number of stocks in the portfolio
    ?value - the initial price value of a single stock
    ?balance - balance on the savings account
    */

    precondition
    nfp
    dc#description hasValue "The necessary information to be provided to the
system includes: stock, market, kind of action, customer, portfolio, quantifier -
some | all regarding the recommendations. The stock has to belong to the specified
market and the portfolio to the customer. In case of SELL action user has to possess
at least the specified number of stocks in his portfolio. In case of BUY action, the
value of stocks the user intends to buy must not be bigger than the balance on user's
account"
    endnfp
    definedBy
    ?stock [hasISIN hasValue ?stockISIN] memberOf sm#Stock and
    ?market [hasName hasValue ?marketName] memberOf sm#StockMarket and
    ?stock [hasStockMarket hasValue ?market,
            hasPriceValue hasValue ?value] and
    ?action memberOf {"BUY", "SELL"} and
    ?customer memberOf fin#Customer and
    ?portfolio [hasPortFolioID hasValue ?portfolioID,
               hasCustomer hasValue ?customer] memberOf sm#Portfolio
and
    ?quantifier memberOf {"some", "all"} and
    (
    (
    /* case of "some" */
    (?quantifier = "some") and (exists {?x} (?stock [hasRecommendation hasValue
?x] and ?x = ?action))
    ) or
    )

```

```

    (
      /* case of "all" */
      (?quantifier = "all") and (forall {?x} (?stock [hasRecommendation hasValue
?x] and ?x = ?action))
    )
  ) and
  (
    (
      /*
      In case of SELL action user has to possess at least the specified number of
      stocks in his portfolio.
      */
      ?action = "SELL" and
      ?portfolio [hasStocks hasValue ?stockPorfolio] and
      ?stockPortfolio [hasStocks hasValue ?stock,
      stocksNumber hasValue ?stocksNumber] and
      ?stocksNumber >= ?number
    ) or
    (
      /*
      In case of BUY action, the value of stocks the user intends to buy must not be
      bigger than the balance on user's account
      */
      ?action = "BUY" and
      ?portfolio [hasAssociatedAccount hasValue ?savingAccount] and
      ?savingAccount [balanceAccount hasValue ?balance] memberOf fin#SavingAccount and
      ?balance >= (?number * ?value)
    )
  ).

postcondition
  nfp
  dc#description hasValue "In case of SELL, the user deposes of the number of
stocks from the portfolio. The value is added to his savings account. In case of BUY,
the number of stocks is added to portfolio. The value is subtracted from the savings
account"
  endnfp
  definedBy
  (
    /*
    In case of SELL, the user deposes of the number of stocks from the portfolio.
    The value is added to his savings account.
    */
    ?stockPortfolio [stocksNumber hasValue (?stocksNumber - ?number)] and
    ?savingAccount [balanceAccount hasValue (?balance + (?number * ?value))]
  ) or
  (
    /*
    In case of BUY, the number of stocks is added to portfolio.
    The value is subtracted from the savings account
    */
    ?stockPortfolio [stocksNumber hasValue (?stocksNumber + ?number)] and
    ?savingAccount [balanceAccount hasValue (?balance - (?number * ?value))]
  ).

```

Listing 13 Web Service UCI5

```

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace
{
  _"http://users.isoco.net/~slosada/ontologies/bankinter/WSRatingDependentAction.wsml#"
  ,
  dc _"http://purl.org/dc/elements/11#" ,
  foaf _"http://xmlns.com/foaf/01/" ,
  xsd _"http://www.w3c.org/2001/XMLSchema#" ,
  wsml _"http://www.wsmo.org/2004/wsml#" ,
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#" ,
  fin
  _"http://users.isoco.net/~slosada/ontologies/bankinter/FinancialOntology.wsml#"
}

webService
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSRatingDependentAction.wsml"

  nfp
    dc#title hasValue "Web Service for rating dependent actions"
    dc#type hasValue _"http://www.wsmo.org/2004/d2#webservice"
    dc#description hasValue "WS to execute some action if the rating of some stock
changes"
    dc#contributor hasValue ""
    dc#date hasValue _date(2005,11,16)
    dc#format hasValue "text/plain"
    dc#language hasValue "en-US"
    dc#rights hasValue _"http://www.isoco.com/privacy.html"
    wsml#version hasValue "$Revision: 01 $"
  endnfp

  importsOntology {
    _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#"
  }
  importsOntology {
    _"http://users.isoco.net/~slosada/ontologies/bankinter/FinancialOntology.wsml#"
  }

  capability
  _"http://users.isoco.net/~slosada/ontologies/bankinter/WSRatingDependentAction#capabi
lity"

  sharedVariables {?stockISIN, ?marketName, ?action, ?stock, ?value, ?stocksNumber,
?stockVariation, ?customer, ?portfolio, ?market, ?newStocksNumber, ?savingAccount}
  /*
  ?stockISIN - the identifier of the stock
  ?marketName - the name of the relevant market
  ?action - SELL | BUY | ALERT
  ?customer - the customer relevant for the operation
  ?portfolioID - identifier of the portfolio, from which the action is to be
performed
  ?number - number of stocks to be bought or sold by the user
  */

  precondition
    nfp
      dc#description hasValue "The system has to know the portfolio ID, from which
to execute action, and the user, to check if the portfolio belongs to the user. The

```

stock name, number of them and market have to be known to the system, if it is going to check whether its rating has changed and then do something with it. The action has to be specified as well. The stock value has to belong to top five variations. If not, no action is performed. In case of SELL action user has to possess at least the specified number of stocks in his portfolio. In case of BUY action, the value of stocks the user intends to buy must not be bigger than the balance on user's account"

```

endnfp
definedBy
  ?stock [hasISIN hasValue ?stockISIN] memberOf sm#Stock and
  ?market [hasName hasValue ?marketName] memberOf sm#StockMarket and
  ?stock [hasStockMarket hasValue ?market,
          hasPriceValue hasValue ?value] and
  ?action memberOf {"BUY", "SELL", "ALERT"} and
  ?customer memberOf fin#Customer and
  ?portfolio [hasPortFolioID hasValue ?portfolioID,
              hasCustomer hasValue ?customer] memberOf sm#Portfolio
and
  ?rating[hasStock hasValue ?stock,
          hasIssuer hasValue ?issuer,
          hasRatingValue hasValue ?ratingValue] memberOf sm#Rating and
  (
    (
      /*
      In case of SELL action user has to possess at least the specified number of
      stocks in his portfolio.
      */
      ?action = "SELL" and
      ?portfolio [hasStocks hasValue ?stockPorfolio] and
        ?stockPortfolio [hasStocks hasValue ?stock,
                        stocksNumber hasValue ?stocksNumber] and
        ?stocksNumber >= ?number
    ) or
    (
      /*
      In case of ALERT no additional precondition is assumed
      */
      ?action = "ALERT"
    ) or
    (
      /*
      In case of BUY action, the value of stocks the user intends to buy must not be
      bigger than the balance on user's account
      */
      ?action = "BUY" and
      ?portfolio [hasAssociatedAccount hasValue ?savingAccount] and
      ?savingAccount [balanceAccount hasValue ?balance] memberOf fin#SavingAccount and
      ?balance >= (?number * ?value)
    )
  ).

postcondition
  nfp
  dc#description hasValue "In case of SELL, the user deposes of the number of
  stocks from the portfolio. The value is added to his savings account. In case of BUY,
  the number of stocks is added to portfolio. The value is subtracted from the savings
  account"
endnfp
definedBy

```

```
(
/*
In case of SELL, the user disposes of the number of stocks from the portfolio.
The value is added to his savings account.
*/
?stockPortfolio [stocksNumber hasValue (?stocksNumber - ?number)] and
?savingAccount [balanceAccount hasValue (?balance + (?number * ?value))]
) or
(
/*
In case of BUY, the number of stocks is added to portfolio.
The value is subtracted from the savings account
*/
?stockPortfolio [stocksNumber hasValue (?stocksNumber + ?number)] and
?savingAccount [balanceAccount hasValue (?balance - (?number * ?value))]
) .
```

Listing 14 Web Service UCI6

```
wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WSStatisticsDependentAction.w
sml#",
  dc _"http://purl.org/dc/elements/11#",
  foaf _"http://xmlns.com/foaf/01/",
  xsd _"http://www.w3c.org/2001/XMLSchema#",
  wsml _"http://www.wsmo.org/2004/wsml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  fin
_"http://users.isoco.net/~slosada/ontologies/bankinter/FinancialOntology.wsml#"}

webService
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSStatisticsDependentAction.w
sml"

nfp
  dc#title hasValue "Web Service for statistics dependent actions"
  dc#type hasValue _"http://www.wsmo.org/2004/d2#webservice"
  dc#description hasValue "WS to execute some action on the stock if some
statistics concerning this stock reach a specified value"
  dc#contributor hasValue ""
  dc#date hasValue _date(2005,11,16)
  dc#format hasValue "text/plain"
  dc#language hasValue "en-US"
  dc#rights hasValue _"http://www.isoco.com/privacy.html"
  wsml#version hasValue "$Revision: 01 $"
endnfp

importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#"}
importsOntology {
```

```

_"http://users.isoco.net/~slosada/ontologies/bankinter/FinancialOntology.wsml#"}

    capability
    _"http://users.isoco.net/~slosada/ontologies/bankinter/WSStatisticsDependentAction#ca
    pability"

    sharedVariables {?stockISIN, ?stock, ?marketName, ?market, ?customer, ?portfolioID,
    ?stockPortfolio, ?action, ?relativeVariation, ?number, ?savingAccount, ?stocksNumber,
    ?value, ?balance}

    /*
    ?stockISIN - the identifier of the relevant stock
    ?stock - the relevant stock
    ?marketName - the name of the relevant market
    ?market - the relevant market
    ?customer - customer relevant for the transaction
    ?portfolioID - the identifier of the customer's portfolio
    ?stockPortfolio - the relevant stock in the portfolio of the customer
    ?action - desired action BUY | SELL | ALERT
    ?relativeVariation - the actual change of value of the given stock
    ?number - number of stocks to be bought / sold
    ?savingAccount - the account for value transfer
    ?stocksNumber - the initial number of stocks in the portfolio
    ?value - the initial price value of a single stock
    ?balance - balance on the savings account
    ?statisticsName - name of the relevant statistics
    ?valueChange - the reference value provided by the user
    ?constraint - one of =, >, <, <=, >=
    */

    precondition
        nfp
            dc#description hasValue "The necessary information to be provided to the
            system includes: stock, market, kind of action, customer, portfolio. The stock has to
            belong to the specified market and the portfolio to the customer. The statistics of
            the stock has to satisfy the condition provided by the customer. In case of SELL
            action user has to possess at least the specified number of stocks in his portfolio.
            In case of BUY action, the value of stocks the user intends to buy must not be bigger
            than the balance on user's account"
            endnfp
            definedBy
                ?stock [hasISIN hasValue ?stockISIN] memberOf sm#Stock and
                ?market [hasName hasValue ?marketName] memberOf sm#StockMarket and
                ?stock [hasStockMarket hasValue ?market,
                    hasPriceValue hasValue ?value] and
                ?action memberOf {"BUY", "SELL", "ALERT"} and
                ?customer memberOf fin#Customer and
                ?portfolio [hasPortFolioID hasValue ?portfolioID,
                    hasCustomer hasValue ?customer] memberOf sm#Portfolio
            and
                ?statistics [hasStock hasValue ?stock,
                    hasName hasValue ?statisticsName,
                    hasStatisticsValue hasValue ?statValue] memberOf
            sm#Statistics and
            (
                ((?constraint = "<") and (?statValue < ?valueChange)) or
                ((?constraint = "<=") and (?statValue <= ?valueChange)) or
                ((?constraint = ">") and (?statValue > ?valueChange)) or
                ((?constraint = ">=") and (?statValue >= ?valueChange)) or

```

```

    ((?constraint = "=") and (?statValue = ?valueChange))
  ) and
  (
    (
      /*
      In case of SELL action user has to possess at least the specified number of
      stocks in his portfolio.
      */
      ?action = "SELL" and
      ?portfolio [hasStocks hasValue ?stockPorfolio] and
        ?stockPortfolio [hasStocks hasValue ?stock,
          stocksNumber hasValue ?stocksNumber] and
          ?stocksNumber >= ?number
    ) or
    (
      /*
      In case of ALERT no additional precondition is assumed
      */
      ?action = "ALERT"
    ) or
    (
      /*
      In case of BUY action, the value of stocks the user intends to buy must not be
      bigger than the balance on user's account
      */
      ?action = "BUY" and
      ?portfolio [hasAssociatedAccount hasValue ?savingAccount] and
      ?savingAccount [balanceAccount hasValue ?balance] memberOf fin#SavingAccount and
      ?balance >= (?number * ?value)
    )
  ).

postcondition
  nfp
    dc#description hasValue "In case of SELL, the user depose of the number of
    stocks from the portfolio. The value is added to his savings account. In case of BUY,
    the number of stocks is added to portfolio. The value is subtracted from the savings
    account"
  endnfp
  definedBy
    (
      /*
      In case of SELL, the user depose of the number of stocks from the portfolio.
      The value is added to his savings account.
      */
      ?stockPortfolio [stocksNumber hasValue (?stocksNumber - ?number)] and
      ?savingAccount [balanceAccount hasValue (?balance + (?number * ?value))]
    ) or
    (
      /*
      In case of BUY, the number of stocks is added to portfolio.
      The value is subtracted from the savings account
      */
      ?stockPortfolio [stocksNumber hasValue (?stocksNumber + ?number)] and
      ?savingAccount [balanceAccount hasValue (?balance - (?number * ?value))]
    ).

```

Listing 15 Web Service UCI7

```

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WSSelectAndExecute.wsml#",
  dc _"http://purl.org/dc/elements/11#",
  foaf _"http://xmlns.com/foaf/01/",
  xsd _"http://www.w3c.org/2001/XMLSchema#",
  wsml _"http://www.wsmo.org/2004/wsml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  fin
_"http://users.isoco.net/~slosada/ontologies/bankinter/FinancialOntology.wsml#"}

webService
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSSelectAndExecute.wsml"

  nfp
    dc#title hasValue "Web Service for select and execute orders"
    dc#type hasValue _"http://www.wsmo.org/2004/d2#webservice"
    dc#description hasValue "Web Service for selecting stocks and executing on them
Buy or Sell orders"
    dc#contributor hasValue ""
    dc#date hasValue _date(2005,11,16)
    dc#format hasValue "text/plain"
    dc#language hasValue "en-US"
    dc#rights hasValue _"http://www.isoco.com/privacy.html"
    wsml#version hasValue "$Revision: 01 $"
  endnfp

  importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#"}
  importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/FinancialOntology.wsml#"}

  capability
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSSelectAndExecute#capability"

  sharedVariables {?stockISIN, ?action, ?stock, ?value, ?stocksNumber, ?number,
?balance}
  /*
    ?stockISIN - the identifier of the stock
    ?action - SELL | BUY | ALERT
    ?stock - the relevant stock
    ?value - value of a single relevant stock
    ?stocksNumber - number of relevant stocks currently in portfolio
    ?number - number of stocks to be bought or sold by the user
    ?balance - balance on the account
  */

  precondition
    nfp
      dc#description hasValue "The stock has the highest variation on the market."

```

The user has more money than the value of the stocks he wants to buy, or more stocks than the number he wants to sell"

```

endnfp
definedBy
  ?stock [hasISIN hasValue ?stockISIN] memberOf sm#Stock and
  ?market [hasISIN hasValue ?marketISIN] memberOf sm#StockMarket and
  ?stock [hasStockMarket hasValue ?market,
          hasPriceValue hasValue ?value,
          hasVariation hasValue ?variation] and
  (forall ?stocks (?stocks[hasVariation hasValue ?variations,
                          hasStockMarket hasValue ?market]
memberOf sm#Stock
and      (?variations =< ?variation))) and
  ?action memberOf {"BUY", "SELL"} and
  ?customer memberOf fin#Customer and
  ?portfolio [hasPortFolioID hasValue ?portfolioID,
              hasCustomer hasValue ?customer,
              hasAssociatedAccount hasValue ?savingAccount] memberOf
sm#Portfolio and
  ?savingAccount memberOf fin#SavingAccount and
  (
  (
  /*
  In case of SELL action user has to possess at least the specified number of
  stocks in his portfolio.
  */
  ?action = "SELL" and
  ?portfolio [hasStocks hasValue ?stockPorfolio] and
    ?stockPortfolio [hasStocks hasValue ?stock,
                    stocksNumber hasValue ?stocksNumber] and
    ?stocksNumber >= ?number
  ) or
  (
  /*
  In case of BUY action, the value of stocks the user intends to buy must not be
  bigger than the balance on user's account
  */
  ?action = "BUY" and
  ?savingAccount [balanceAccount hasValue ?balance] and
  ?balance >= (?number * ?value)
  )
  ).

postcondition
  nfp
  dc#description hasValue "In case of SELL, the user deposes of the number of
  stocks from the portfolio. The value is added to his savings account. In case of BUY,
  the number of stocks is added to portfolio. The value is subtracted from the savings
  account"
  endnfp
  definedBy
  (
  /*
  In case of SELL, the user deposes of the number of stocks from the portfolio.
  The value is added to his savings account.
  */
  ?stockPortfolio [stocksNumber hasValue (?stocksNumber - ?number)] and
  ?savingAccount [balanceAccount hasValue (?balance + (?number * ?value))]

```

```

) or
(
/*
    In case of BUY, the number of stocks is added to portfolio.
    The value is subtracted from the savings account
*/
?stockPortfolio [stocksNumber hasValue (?stocksNumber + ?number)] and
?savingAccount [balanceAccount hasValue (?balance - (?number * ?value))]
) .

```

3.4 Interfaces

According to [6], the interface of a Web Service should contain choreography, which describes the communication pattern that allows one to consume the functionality of Web Service and orchestration, which describes how the overall functionality of the Web Service is achieved by means of cooperation of different Web Services. At the moment of writing this deliverable the orchestration had not been specified, therefore we limit ourselves to providing the choreography part of the interfaces.

The orchestration part for the interfaces will be written to be included in the prototype application. This deliverable will be then updated to include orchestration descriptions.

Listing 16 Interface UC11

```

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WSVariationDependentActionInt
erface.wsml#" ,
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#" ,
  smp _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#" ,
  WS _"http://users.isoco.net/~slosada/ontologies/bankinter/WSVariationDependentAction.wsm
l#" ,
  QoS-eBanking _"http://www.example.org/QoS-eBanking"}

webService
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSVariationDependentAction.wsm
l"

interface WSVariationDependentActionInterface

  importsOntology {_"http://www.example.org/QoS-eBanking"}
  nonFunctionalProperties

/*
    dc:description hasValue "QoS properties of paid stock market
information service."

    QoSParameter availabilityLevelForPaidService
      instance paidServiceAvailability memberOf QoS-eBanking#Availability
      hasMeanValue hasValue _double("0.999999")
      hasBaseNumberOfRequestsToEvaluateAvailability
hasValue _integer("100,000")

```

```

        QoSParameter dataFreshnessForPaidService
            instance paidServiceDataFressNess memberOf
QoSeBanking#DataFreshness
                hasMeanValue hasValue _double("10.0")
                hasStandardDeviation hasValue _double("1.0")
                hasMeasurementUnit hasValue QoSeBanking#minute

        QoSParameter capacityForPaidService
            instance paidServiceCapacity memberOf QoSeBanking#MaximumCapacity
                hasMeanValue hasValue _double("10,000")
                hasMeasurementUnit hasValue QoS-eBanking#requestsPerSecond

        QoSParameter responseTimeForPaidService
            instance paidServiceCapacity memberOf QoSeBanking#MaximumCapacity
                hasMeanValue hasValue _double("1.5")
                hasStandardDeviation hasValue _double("0.25")
                hasMeasurementUnit hasValue QoSeBanking#second
*/

    endNonFunctionalProperties

choreography WSVariationDependentActionChoreography
    stateSignature
        importsOntology
(_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#" )
    in {
        smp#ExecuteIfInTopFiveVariations
    }

    out {
        smp#Confirmation
    }

guardedTransition WSVariationDependentActionChoreographyRules

    forAll {?request} with (
?request memberOf smp#ExecuteIfInTopFiveVariations
    )

    do
    add( _#[[] memberOf smp#Confirmation)
    endForAll

```

Listing 17 Interfaces UCI2 and UCI4

```

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WValueDependentActionInterfa
ce.wsml#",
    sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
    smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#",

```

```

WS
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSValueDependentAction.wsml#",
  QoS-eBanking _"http://www.example.org/QoS-eBanking"}

webService
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSValueDependentAction.wsml"

interface WSValueDependentActionInterface

  importsOntology {_"http://www.example.org/QoS-eBanking"}
  nonFunctionalProperties

/*
    dc:description hasValue "QoS properties of paid stock market
information service."

    QoSParameter availabilityLevelForPaidService
      instance paidServiceAvailability memberOf QoS-eBanking#Availability
      hasMeanValue hasValue _double("0.999999")
      hasBaseNumberOfRequestsToEvaluateAvailability
hasValue _integer("100,000")

    QoSParameter dataFreshnessForPaidService
      instance paidServiceDataFressNess memberOf QoS-
eBanking#DataFreshness
      hasMeanValue hasValue _double("10.0")
      hasStandardDeviation hasValue _double("1.0")
      hasMeasurementUnit hasValue QoS-eBanking#minute

    QoSParameter capacityForPaidService
      instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
      hasMeanValue hasValue _double("10,000")
      hasMeasurementUnit hasValue QoS-eBanking#requestsPerSecond

    QoSParameter responseTimeForPaidService
      instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
      hasMeanValue hasValue _double("1.5")
      hasStandardDeviation hasValue _double("0.25")
      hasMeasurementUnit hasValue QoS-eBanking#second

*/

  endNonFunctionalProperties

choreography WSValueDependentActionChoreography
  stateSignature
    importsOntology
    {_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
    _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"}
    in {
      smp#ExecuteIfValueChanges,
      smp#ExecuteIfValueRises
    }

    out {
      smp#Confirmation
    }

```

```
guardedTransition WSValueDependentActionChoreographyRules
```

```

    forAll {?request} with (
      ?request memberOf smp#ExecuteIfValueChanges or
      ?request memberOf smp#ExecuteIfValueRises
    )

    do
      add( _#[[] memberOf smp#Confirmation)
    endForAll

```

Listing 18 Interface UCI3

```

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WSRecommendationsDependentActi
ionInterface.wsml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#",
  WS
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSRecommendationsDependentActi
on.wsml#",
  QoS-eBanking _"http://www.example.org/QoS-eBanking"}

webService
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSRecommendationsDependentActi
on.wsml"

interface WSRecommendationsDependentActionInterface

  importsOntology {_"http://www.example.org/QoS-eBanking"}
  nonFunctionalProperties

  dc:description hasValue "QoS properties of paid stock market
information service."
/*
  QoSParameter availabilityLevelForPaidService
    instance paidServiceAvailability memberOf QoS-eBanking#Availability
      hasMeanValue hasValue _double("0.999999")
      hasBaseNumberOfRequestsToEvaluateAvailability
hasValue _integer("100,000")

  QoSParameter dataFreshnessForPaidService
    instance paidServiceDataFressNess memberOf QoS-
eBanking#DataFreshness
      hasMeanValue hasValue _double("10.0")
      hasStandardDeviation hasValue _double("1.0")
      hasMeasurementUnit hasValue QoS-eBanking#minute

  QoSParameter capacityForPaidService
    instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
      hasMeanValue hasValue _double("10,000")
      hasMeasurementUnit hasValue QoS-eBanking#requestsPerSecond

```

```

        QoSParameter responseTimeForPaidService
            instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
                hasMeanValue hasValue _double("1.5")
                hasStandardDeviation hasValue _double("0.25")
                hasMeasurementUnit hasValue QoS-eBanking#second
    */

    endNonFunctionalProperties

choreography WSRecommendationsDependentActionChoreography
    stateSignature
        importsOntology
        (_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
        _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#" )
        in {
            smp#ExecuteIfRecommendations
        }

        out {
            smp#Confirmation
        }

guardedTransition WSRecommendationsDependentActionChoreographyRules

    forAll {?request} with (
        ?request memberOf smp#ExecuteIfRecommendations
    )

    do
        add( _#[[] memberOf smp#Confirmation)
    endForAll

```

Listing 19 Interface UCI5

```

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WSRatingDependentActionInterf
ace.wsml#",
    sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
    smp
    _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#",
    WS
    _"http://users.isoco.net/~slosada/ontologies/bankinter/WSRatingDependentAction.wsml#"
    ,
    QoSeBanking _"http://www.example.org/QoS-eBanking"}

webService
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSRatingDependentAction.wsml"

interface WSRatingDependentActionInterface

importsOntology {_"http://www.example.org/QoS-eBanking"}

```

```

nonFunctionalProperties

    dc#description hasValue "QoS properties of paid stock market
information service."
/*
    QoSParameter availabilityLevelForPaidService
        instance paidServiceAvailability memberOf QoS-eBanking#Availability
            hasMeanValue hasValue _double("0.999999")
            hasBaseNumberOfRequestsToEvaluateAvailability
hasValue _integer("100,000")

    QoSParameter dataFreshnessForPaidService
        instance paidServiceDataFressNess memberOf QoS-
eBanking#DataFreshness
            hasMeanValue hasValue _double("10.0")
            hasStandardDeviation hasValue _double("1.0")
            hasMeasurementUnit hasValue QoS-eBanking#minute

    QoSParameter capacityForPaidService
        instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
            hasMeanValue hasValue _double("10,000")
            hasMeasurementUnit hasValue QoS-eBanking#requestsPerSecond

    QoSParameter responseTimeForPaidService
        instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
            hasMeanValue hasValue _double("1.5")
            hasStandardDeviation hasValue _double("0.25")
            hasMeasurementUnit hasValue QoS-eBanking#second
*/
endNonFunctionalProperties

choreography WSRatingDependentActionChoreography
stateSignature
    importsOntology
    ("http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
    "http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#")
    in {
        smp#ExecuteIfRatingChanges
    }

    out {
        smp#Confirmation
    }

guardedTransition WSRatingDependentActionChoreographyRules

    forAll {?request} with (
    ?request memberOf smp#ExecuteIfRatingChanges
    )

    do
    add( _#[[] memberOf smp#Confirmation)
    endForAll

```

Listing 20 Interface UCI6

```

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WSStatisticsDependentActionIn
terface.wsml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#",
  WS
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSStatisticsDependentAction.ws
ml#",
  QoS-eBanking _"http://www.example.org/QoS-eBanking"}

webService
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSStatisticsDependentAction.ws
ml"

interface WSStatisticsDependentActionInterface

  importsOntology {_"http://www.example.org/QoS-eBanking"}
  nonFunctionalProperties

/*
      dc#description hasValue "QoS properties of paid stock market
information service."

      QoSParameter availabilityLevelForPaidService
        instance paidServiceAvailability memberOf QoS-eBanking#Availability
        hasMeanValue hasValue _double("0.999999")
        hasBaseNumberOfRequestsToEvaluateAvailability
hasValue _integer("100,000")

      QoSParameter dataFreshnessForPaidService
        instance paidServiceDataFressNess memberOf QoS-
eBanking#DataFreshness
        hasMeanValue hasValue _double("10.0")
        hasStandardDeviation hasValue _double("1.0")
        hasMeasurementUnit hasValue QoS-eBanking#minute

      QoSParameter capacityForPaidService
        instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
        hasMeanValue hasValue _double("10,000")
        hasMeasurementUnit hasValue QoS-eBanking#requestsPerSecond

      QoSParameter responseTimeForPaidService
        instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
        hasMeanValue hasValue _double("1.5")
        hasStandardDeviation hasValue _double("0.25")
        hasMeasurementUnit hasValue QoS-eBanking#second
*/

  endNonFunctionalProperties

choreography WSStatisticsDependentActionChoreography
  stateSignature
    importsOntology
{_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",

```

```

_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"
  in {
    smp#ExecuteIfStatistics
  }

  out {
    smp#Confirmation
  }

guardedTransition WSStatisticsDependentActionChoreographyRules

  forall {?request} with (
    ?request memberOf smp#ExecuteIfStatistics
  )

  do
  add( _#[[] memberOf smp#Confirmation)
endForAll

```

Listing 21 Interface UCI7

```

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WSSelectAndExecuteInterface.w
sml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#",
  WS
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSSelectAndExecute.wsml#",
  QoS-eBanking _"http://www.example.org/QoS-eBanking"}

webService
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSSelectAndExecute.wsml"

interface WSSelectAndExecuteInterface

  importsOntology {_"http://www.example.org/QoS-eBanking"}
  nonFunctionalProperties

  dc:description hasValue "QoS properties of paid stock market
information service."
/*
  QoSParameter availabilityLevelForPaidService
    instance paidServiceAvailability memberOf QoS-eBanking#Availability
    hasMeanValue hasValue _double("0.999999")
    hasBaseNumberOfRequestsToEvaluateAvailability
hasValue _integer("100,000")

  QoSParameter dataFreshnessForPaidService
    instance paidServiceDataFressNess memberOf QoS-
eBanking#DataFreshness
    hasMeanValue hasValue _double("10.0")
    hasStandardDeviation hasValue _double("1.0")

```

```

        hasMeasurementUnit hasValue QoS-eBanking#minute

    QoSParameter capacityForPaidService
        instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
        hasMeanValue hasValue _double("10,000")
        hasMeasurementUnit hasValue QoS-eBanking#requestsPerSecond

    QoSParameter responseTimeForPaidService
        instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
        hasMeanValue hasValue _double("1.5")
        hasStandardDeviation hasValue _double("0.25")
        hasMeasurementUnit hasValue QoS-eBanking#second
*/

    endNonFunctionalProperties

choreography WSSelectAndExecuteChoreography
    stateSignature
        importsOntology
        {_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"}
        in {
            smp#SelectAndExecute
        }

        out {
            smp#Confirmation
        }

guardedTransition WSSelectAndExecuteChoreographyRules

    forAll {?request} with (
        ?request memberOf smp#SelectAndExecute
    )

    do
        add( _#[] memberOf smp#Confirmation)
    endForAll

```

4 CONCLUSIONS

This document provides WSMO descriptions of semantic web services as planned for the second eBanking application. We have specified the goals, web services, choreography interfaces and ontologies. For the development of the application we will still need to specify the orchestration part of the interfaces.

The content of this deliverable may be used for testing the technical architecture of the DIP project and is therefore applicable for WP 1, 2, 3, 4, 5 and 6.

REFERENCES

- [1] Stollberg, M. et al. (2005). *D3.2 v0.2 WSMO Use Case Modeling and Testing*. Available at: <http://www.wsmo.org/TR/d3/d3.2/v0.2/20050413/>

-
- [2] He, H.; Haas, H.; Orchard, D. (2004). *Web Services Architecture Usage Scenarios*. Available at: <http://www.w3.org/TR/2004/NOTE-ws-arch-scenarios-20040211/>
- [3] Losada Alonso, S. et. al. (2005). *Specification of application 2*. DIP Deliverable 10.6.
- [4] Martínez Montes, M. et. al. (2004). *Financial Ontology*. DIP Deliverable 10.3.
- [5] Losada Alonso, S. et. al. (2005). *Financial Ontology*. DIP Deliverable 10.7.
- [6] Feier, C.; Domingue, J. (2005). *D3.Iv0.1 WSMO Primer*. Available at: <http://www.wsmo.org/TR/d3/d3.1/v0.1/>
- [7] Krummenacher, R; Hepp, M; Polleres, A; Bussler, D; Fensel, D (2005). *WWW or What Is Wrong with Web Services*. Proceedings of the 2005 IEEE European Conference on Web Services (IEEE ECOWS 2005), November 14-16, Växjö, Sweden.

ANNEX I

The following WSMML documents describe Web Services which are composed to achieve the functionality required by the user.

```
wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"  
  
namespace {_"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetRatings.wsml#",  
  dc _"http://purl.org/dc/elements/11#",  
  foaf _"http://xmlns.com/foaf/01/",
```

```

xsd _"http://www.w3c.org/2001/XMLSchema#",
wsml _"http://www.wsmo.org/2004/wsml#",
sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"

webService _"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetRatings.wsml"

  nfp
    dc#title hasValue "Web Service WSgetRatings"
    dc#type hasValue _"http://www.wsmo.org/2004/d2#webservice"
    dc#description hasValue "WS to get a container of ratings for a given stock"
    dc#contributor hasValue {"Darek Kleczek", "Silvestre Losada"}
    dc#date hasValue _date(2005,12,21)
    dc#format hasValue "text/plain"
    dc#language hasValue "en-US"
    dc#rights hasValue _"http://www.isoco.com/privacy.html"
    wsml#version hasValue "$Revision: 01 $"
    wsml#endpointDescription hasValue
    _"https://aia.ebankinter.com/wsBrokerService/#wsdl.service(BrokerService)"
    endnfp

    importsOntology {
    _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#"
    importsOntology {
    _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"
    }

    capability
    _"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetRatings#capability"

    sharedVariables {?request}

    /*
    */

    precondition
      nfp
        dc#description hasValue "There has to be a request to get the ratings"
        endnfp
        definedBy
          ?request memberOf smp#GetRatings.

    postcondition
      nfp
        dc#description hasValue "Returns a container consisting of ratings for a given
stock"
        endnfp
        definedBy
          (?request[stockISIN hasValue stockISIN] memberOf smp#GetRatings implies
exists ?container (?container memberOf smp#ratingContainer and
forall {?item} (?container[items hasValue ?item] implies ?item[hasStockISIN
hasValue ?stockISIN] memberOf sm#Rating))).

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetRatingsInterface.wsml#",
sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#",
WS _"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetRatings.wsml#",
QoS-eBanking _"http://www.example.org/QoS-eBanking"}

webService _"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetRatings.wsml"

interface WSgetRatingsInterface

  importsOntology {_"http://www.example.org/QoS-eBanking"}

```

```

nonFunctionalProperties

    dc:description hasValue "QoS properties of paid stock market information
service."

    QoSParameter availabilityLevelForPaidService
        instance paidServiceAvailability memberOf QoS-eBanking#Availability
        hasMeanValue hasValue _double("0.999999")
        hasBaseNumberOfRequestsToEvaluateAvailability
hasValue _integer("100,000")

    QoSParameter dataFreshnessForPaidService
        instance paidServiceDataFressNess memberOf QoS-eBanking#DataFreshness
        hasMeanValue hasValue _double("10.0")
        hasStandardDeviation hasValue _double("1.0")
        hasMeasurementUnit hasValue QoS-eBanking#minute

    QoSParameter capacityForPaidService
        instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
        hasMeanValue hasValue _double("10,000")
        hasMeasurementUnit hasValue QoS-eBanking#requestsPerSecond

    QoSParameter responseTimeForPaidService
        instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
        hasMeanValue hasValue _double("1.5")
        hasStandardDeviation hasValue _double("0.25")
        hasMeasurementUnit hasValue QoS-eBanking#second

endNonFunctionalProperties

choreography WSgetRatingsChoreography
    stateSignature
        importsOntology
        ("http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
        _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"}
        in {
            smp#GetRatings withGrounding
WS#wSDL.interfaceMessageReference(Service1Soap/getRatings/In)
        }
        out {
            smp#ratingContainer withGrounding
WS#wSDL.interfaceMessageReference(Service1Soap/getRatings/Out)
        }

guardedTransition WSgetRatingsChoreographyRules

    forall {?request} with (
        ?request memberOf smp#GetRatings
    )

    do
    add( _#[] memberOf smp#ratingContainer)
endForAll

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetRecommendations.wsml#",
dc _"http://purl.org/dc/elements/11#",
foaf _"http://xmlns.com/foaf/01/",
xsd _"http://www.w3c.org/2001/XMLSchema#",
wsml _"http://www.wsmo.org/2004/wsml#",
sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"}

webService
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetRecommendations.wsml"

```

```

nfp
dc#title hasValue "Web Service WSgetRecommendations"
dc#type hasValue _"http://www.wsmo.org/2004/d2#webservice"
dc#description hasValue "WS to get a container of recommendations for a given
stock"
dc#contributor hasValue {"Darek Kleczek", "Silvestre Losada"}
dc#date hasValue _date(2005,12,21)
dc#format hasValue "text/plain"
dc#language hasValue "en-US"
dc#rights hasValue _"http://www.isoco.com/privacy.html"
wsml#version hasValue "$Revision: 01 $"
wsml#endpointDescription hasValue
_"https://aia.ebankinter.com/wsBrokerService/#wsdl.service(BrokerService)"
endnfp

importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#"}
importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"}

capability
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetRecommendations#capability"

sharedVariables {?request}

/*
*/

precondition
nfp
dc#description hasValue "There has to be a request to get the recommendations"
endnfp
definedBy
?request memberOf smp#GetRecommendations.

postcondition
nfp
dc#description hasValue "Returns a container consisting of recommendations for
a given stock"
endnfp
definedBy
(?request[stockISIN hasValue stockISIN] memberOf smp#GetRecommendations implies
exists ?container (?container memberOf smp#recommendationContainer and
forall {?item} (?container[items hasValue ?item] implies ?item[hasStockISIN
hasValue ?stockISIN] memberOf sm#Recommendation))).

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetRecommendationsInterface.w
sml#",
sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#",
WS
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetRecommendations.wsml#",
QoS-eBanking _"http://www.example.org/QoS-eBanking"}

webService
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetRecommendations.wsml"

interface WSgetRecommendationsInterface

importsOntology {_"http://www.example.org/QoS-eBanking"}
nonFunctionalProperties

dc#description hasValue "QoS properties of paid stock market information
service."

```

```

    QoSParameter availabilityLevelForPaidService
      instance paidServiceAvailability memberOf QoS-eBanking#Availability
      hasMeanValue hasValue _double("0.999999")
      hasBaseNumberOfRequestsToEvaluateAvailability
hasValue _integer("100,000")

    QoSParameter dataFreshnessForPaidService
      instance paidServiceDataFressNess memberOf QoS-eBanking#DataFreshness
      hasMeanValue hasValue _double("10.0")
      hasStandardDeviation hasValue _double("1.0")
      hasMeasurementUnit hasValue QoS-eBanking#minute

    QoSParameter capacityForPaidService
      instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
      hasMeanValue hasValue _double("10,000")
      hasMeasurementUnit hasValue QoS-eBanking#requestsPerSecond

    QoSParameter responseTimeForPaidService
      instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
      hasMeanValue hasValue _double("1.5")
      hasStandardDeviation hasValue _double("0.25")
      hasMeasurementUnit hasValue QoS-eBanking#second

  endNonFunctionalProperties

choreography WSgetRecommendationsChoreography
  stateSignature
    importsOntology
  ("http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#")
  in {
    smp#GetRecommendations withGrounding
  WS#wSDL.interfaceMessageReference(Service1Soap/getRecommendations/In)
  }
  out {
    smp#recommendationContainer withGrounding
  WS#wSDL.interfaceMessageReference(Service1Soap/getRecommendations/Out)
  }

guardedTransition WSgetRecommendationsChoreographyRules

  forall {?request} with (
    ?request memberOf smp#GetRecommendations
  )

  do
  add( _#[[] memberOf smp#recommendationContainer)
  endForAll

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetStatistics.wsml#",
  dc _"http://purl.org/dc/elements/11#",
  foaf _"http://xmlns.com/foaf/01/",
  xsd _"http://www.w3c.org/2001/XMLSchema#",
  wsml _"http://www.wsmo.org/2004/wsml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  smp
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"}

webService _"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetStatistics.wsml"

  nfp
  dc#title hasValue "Web Service WSgetStatistics"
  dc#type hasValue _"http://www.wsmo.org/2004/d2#webservice"
  dc#description hasValue "WS to get a container of statistics for a given stock"
  dc#contributor hasValue {"Darek Kleczek", "Silvestre Losada"}
  dc#date hasValue _date(2005,12,21)

```

```

dc#format hasValue "text/plain"
dc#language hasValue "en-US"
dc#rights hasValue _"http://www.isoco.com/privacy.html"
wsml#version hasValue "$Revision: 01 $"
wsml#endpointDescription hasValue
_"https://aia.ebankinter.com/wsBrokerService/#wsdl.service(BrokerService)"
endnfp

importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#"}
importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"}

capability
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetStatistics#capability"

sharedVariables {?request}

/*
*/

precondition
nfp
dc#description hasValue "There has to be a request to get the statistics"
endnfp
definedBy
?request memberOf smp#GetStatistics.

postcondition
nfp
dc#description hasValue "Returns a container consisting of statistics for a
given stock"
endnfp
definedBy
(?request[stockISIN hasValue stockISIN] memberOf smp#GetStatistics implies
exists ?container (?container memberOf smp#StatisticsContainer and
forall {?item} (?container[items hasValue ?item] implies ?item[hasStockISIN
hasValue ?stockISIN] memberOf sm#Statistics))).

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetStatisticsInterface.wsml#"
,
sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#" ,
smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#" ,
WS _"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetStatistics.wsml#" ,
QoS-eBanking _"http://www.example.org/QoS-eBanking"}

webService _"http://users.isoco.net/~slosada/ontologies/bankinter/WSgetStatistics.wsml"

interface WSgetStatisticsInterface

importsOntology {_"http://www.example.org/QoS-eBanking"}
nonFunctionalProperties

dc#description hasValue "QoS properties of paid stock market information
service."

QoSParameter availabilityLevelForPaidService
instance paidServiceAvailability memberOf QoS-eBanking#Availability
hasMeanValue hasValue _double("0.99999")
hasBaseNumberOfRequestsToEvaluateAvailability
hasValue _integer("100,000")

QoSParameter dataFreshnessForPaidService
instance paidServiceDataFreshness memberOf QoS-eBanking#DataFreshness
hasMeanValue hasValue _double("10.0")

```

```

        hasStandardDeviation hasValue _double("1.0")
        hasMeasurementUnit hasValue QoS-eBanking#minute

    QoSParameter capacityForPaidService
        instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
        hasMeanValue hasValue _double("10,000")
        hasMeasurementUnit hasValue QoS-eBanking#requestsPerSecond

    QoSParameter responseTimeForPaidService
        instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
        hasMeanValue hasValue _double("1.5")
        hasStandardDeviation hasValue _double("0.25")
        hasMeasurementUnit hasValue QoS-eBanking#second

endNonFunctionalProperties

choreography WSgetStatisticsChoreography
    stateSignature
        importsOntology
        ("http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
        "http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#")
    in {
        smp#GetStatistics withGrounding
    WS#wSDL.interfaceMessageReference(Service1Soap/getStatistics/In)
    }
    out {
        smp#StatisticsContainer withGrounding
    WS#wSDL.interfaceMessageReference(Service1Soap/getStatistics/Out)
    }

guardedTransition WSgetStatisticsChoreographyRules

    forall {?request} with (
        ?request memberOf smp#GetStatistics
    )

    do
        add( _#[] memberOf smp#StatisticsContainer)
    endforall

wsmlVariant "http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace
{
    "http://users.isoco.net/~slosada/ontologies/bankinter/WSPerformBuySell.wsml#",
    dc "http://purl.org/dc/elements/1.1#",
    foaf "http://xmlns.com/foaf/0.1/",
    xsd "http://www.w3c.org/2001/XMLSchema#",
    wsml "http://www.wsmo.org/2004/wsml#",
    sm "http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
    smp
    "http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#",
    fin "http://users.isoco.net/~slosada/ontologies/bankinter/FinancialOntology.wsml#"
}

webService
"http://users.isoco.net/~slosada/ontologies/bankinter/WSPerformBuySell.wsml"

nfp
    dc#title hasValue "Web Service WSPerformBuySell"
    dc#type hasValue "http://www.wsmo.org/2004/d2#webservice"
    dc#description hasValue "Executes Buy or Sell action"
    dc#contributor hasValue {"Darek Kleczek", "Silvestre Losada"}
    dc#date hasValue _date(2005,12,21)
    dc#format hasValue "text/plain"
    dc#language hasValue "en-US"
    dc#rights hasValue "http://www.isoco.com/privacy.html"
    wsml#version hasValue "$Revision: 01 $"
    wsml#endpointDescription hasValue
    "https://aia.ebankinter.com/wsBrokerService/#wSDL.service(BrokerService)"
endnfp

```

```

importsOntology {
_ "http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#"
importsOntology {
_ "http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"
importsOntology {
_ "http://users.isoco.net/~slosada/ontologies/bankinter/FinancialOntology.wsml#"

capability
_ "http://users.isoco.net/~slosada/ontologies/bankinter/WSPerformBuySell#capability"

sharedVariables {?action}

/*
*/

precondition
nfp
dc#description hasValue "There is an action of type BUY or SELL to execute"
endnfp
definedBy
?action [type hasValue "BUY"] memberOf smp#action or
?action[type hasValue "SELL"] memberOf smp#action.

postcondition
nfp
dc#description hasValue "Buy action implies that customer has more stocks in
portfolio and less money on account"
endnfp
definedBy
?action[type hasValue "BUY",
number hasValue ?number,
customerID hasValue ?customerID,
portfolioID hasValue ?portfolioID,
stockISIN hasValue ?stockISIN,
market hasValue ?market] memberOf smp#action and
?portfolio [hasPortFolioID hasValue ?portfolioID,
hasStocks hasValue ?stockPortfolio,
hasAssociatedAccount hasValue ?savingAccount] memberOf
sm#Portfolio and
?savingAccount [balanceAccount hasValue ?balance] memberOf fin#SavingAccount and
?stock [hasISIN hasValue ?stockISIN,
hasPriceValue hasValue ?value] memberOf sm#Stock and
?stockPortfolio [hasStocks hasValue ?stock,
stocksNumber hasValue ?stocksNumber] implies

?stockPortfolio [stocksNumber hasValue (?stocksNumber + ?number)] and
?savingAccount [balanceAccount hasValue (?balance - (?number * ?value))].

postcondition
nfp
dc#description hasValue "Sell action implies that customer has less stocks in
portfolio and more money on account"
endnfp
definedBy
?action[type hasValue "SELL",
number hasValue ?number,
customerID hasValue ?customerID,
portfolioID hasValue ?portfolioID,
stockISIN hasValue ?stockISIN,
market hasValue ?market] memberOf smp#action and
?portfolio [hasPortFolioID hasValue ?portfolioID,
hasStocks hasValue ?stockPortfolio,
hasAssociatedAccount hasValue ?savingAccount] memberOf
sm#Portfolio and
?savingAccount [balanceAccount hasValue ?balance] memberOf fin#SavingAccount and
?stock [hasISIN hasValue ?stockISIN,
hasPriceValue hasValue ?value] memberOf sm#Stock and
?stockPortfolio [hasStocks hasValue ?stock,
stocksNumber hasValue ?stocksNumber] implies

```

```

?stockPortfolio [stocksNumber hasValue (?stocksNumber - ?number)] and
?savingAccount [balanceAccount hasValue (?balance + (?number * ?value))].

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"

namespace
{_"http://users.isoco.net/~slosada/ontologies/bankinter/WSPerformBuySellInterface.wsml#"
,
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#" ,
  smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#" ,
  WS _"http://users.isoco.net/~slosada/ontologies/bankinter/WSPerformBuySell.wsml#" ,
  QoS-eBanking _"http://www.example.org/QoS-eBanking" }

webService
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSPerformBuySell.wsml"

interface WSPerformBuySellInterface

  importsOntology {_"http://www.example.org/QoS-eBanking" }
  nonFunctionalProperties

    dc:description hasValue "QoS properties of paid stock market information
service."

    QoSParameter availabilityLevelForPaidService
      instance paidServiceAvailability memberOf QoS-eBanking#Availability
      hasMeanValue hasValue _double("0.999999")
      hasBaseNumberOfRequestsToEvaluateAvailability
hasValue _integer("100,000")

    QoSParameter capacityForPaidService
      instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
      hasMeanValue hasValue _double("10,000")
      hasMeasurementUnit hasValue QoS-eBanking#requestsPerSecond

    QoSParameter responseTimeForPaidService
      instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
      hasMeanValue hasValue _double("1.5")
      hasStandardDeviation hasValue _double("0.25")
      hasMeasurementUnit hasValue QoS-eBanking#second

  endNonFunctionalProperties

choreography WSPerformBuySellChoreography
  stateSignature
    importsOntology
    (_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#" ,
    _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#" )
  in {
    smp#Action withGrounding
    WS#wSDL.interfaceMessageReference(BrokerService/performBuySell/In)
  }
  out {
    smp#Confirmation withGrounding
    WS#wSDL.interfaceMessageReference(BrokerService/performBuySell/Out)
  }

guardedTransition WSPerformBuySellChoreographyRules

  forall {?request} with (
  ?request[type hasValue "BUY"] memberOf smp#Action or
  ?request[type hasValue "SELL"] memberOf smp#Action
  )

  do
  add( _#[] memberOf smp#Confirmation)
  endForAll

```

```

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace
{
  _"http://users.isoco.net/~slosada/ontologies/bankinter/WSRelevantVariations.wsml#",
  dc _"http://purl.org/dc/elements/11#",
  foaf _"http://xmlns.com/foaf/01/",
  xsd _"http://www.w3c.org/2001/XMLSchema#",
  wsml _"http://www.wsmo.org/2004/wsml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  smp
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"
}

webservice
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSRelevantVariations.wsml"

  nfp
    dc#title hasValue "Web Service WSRelevantVariations"
    dc#type hasValue _"http://www.wsmo.org/2004/d2#webservice"
    dc#description hasValue "Retrieves the 5 bigger increases and 5 bigger decreases
for a given index"
    dc#contributor hasValue {"Darek Kleczek", "Silvestre Losada"}
    dc#date hasValue _date(2005,12,21)
    dc#format hasValue "text/plain"
    dc#language hasValue "en-US"
    dc#rights hasValue _"http://www.isoco.com/privacy.html"
    wsml#version hasValue "$Revision: 01 $"
    wsml#endpointDescription hasValue
_"https://aia.ebankinter.com/wsBrokerService/#wsdl.service(BrokerService)"
  endnfp

  importsOntology {
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#"
  importsOntology {
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"
  }

  capability
  _"http://users.isoco.net/~slosada/ontologies/bankinter/WSRelevantVariations#capability"

  sharedVariables {?request}

  /*
  */

  precondition
    nfp
      dc#description hasValue "There is a request to get the relavant variations of
an index"
      endnfp
      definedBy
        ?request memberOf smp#GetRelevantVariations.

  postcondition
    nfp
      dc#description hasValue "The service returns a container with 5 biggest
increases and 5 biggest decreases of an index"
      endnfp
      definedBy
        (?request[indexISIN hasValue ?indexISIN] memberOf smp#GetRelevantVariations implies
          (exists ?container (?container memberOf smp#stockContainer and
            forall ?item (?container[items hasValue ?item] implies ((?item memberOf
sm#Stock) and (?index[hasTopVariations hasValue ?item] memberOf sm#Index))))
          )).

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"

namespace
{
  _"http://users.isoco.net/~slosada/ontologies/bankinter/WSRelevantVariationsInterface.w
sml#",

```

```

sm    _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#",
WS
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSRelevantVariations.wsml#",
QoS-eBanking _"http://www.example.org/QoS-eBanking"}

webService
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSRelevantVariations.wsml"

interface WSRelevantVariationsInterface

    importsOntology {_"http://www.example.org/QoS-eBanking"}
    nonFunctionalProperties

        dc#description hasValue "QoS properties of paid stock market information
service."

        QoSParameter availabilityLevelForPaidService
            instance paidServiceAvailability memberOf QoS-eBanking#Availability
            hasMeanValue hasValue _double("0.999999")
            hasBaseNumberOfRequestsToEvaluateAvailability
hasValue _integer("100,000")

        QoSParameter dataFreshnessForPaidService
            instance paidServiceDataFressNess memberOf QoS-eBanking#DataFreshness
            hasMeanValue hasValue _double("10.0")
            hasStandardDeviation hasValue _double("1.0")
            hasMeasurementUnit hasValue QoS-eBanking#minute

        QoSParameter capacityForPaidService
            instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
            hasMeanValue hasValue _double("10,000")
            hasMeasurementUnit hasValue QoS-eBanking#requestsPerSecond

        QoSParameter responseTimeForPaidService
            instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
            hasMeanValue hasValue _double("1.5")
            hasStandardDeviation hasValue _double("0.25")
            hasMeasurementUnit hasValue QoS-eBanking#second

    endNonFunctionalProperties

choreography WSRelevantVariationsChoreography
    stateSignature
    importsOntology
    (_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#")
    in {
        smp#GetRelevantVariations withGrounding
WS#wSDL.interfaceMessageReference(Service1Soap/relavantVariations/In)
    }
    out {
        smp#stockContainer withGrounding
WS#wSDL.interfaceMessageReference(Service1Soap/relavantVariations/Out)
    }

guardedTransition WSRelevantVariationsChoreographyRules

    forAll {?request} with (
        ?request memberOf smp#GetRelevantVariations
    )

    do
        add( _#[] memberOf smp#stockContainer)
    endForAll

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

```

```

namespace
{
  _"http://users.isoco.net/~slosada/ontologies/bankinter/WSsearchValues.wsml#",
  dc _"http://purl.org/dc/elements/11#",
  foaf _"http://xmlns.com/foaf/01/",
  xsd _"http://www.w3c.org/2001/XMLSchema#",
  wsml _"http://www.wsmo.org/2004/wsml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  smp
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"
}

webService _"http://users.isoco.net/~slosada/ontologies/bankinter/WSsearchValues.wsml"

  nfp
    dc#title hasValue "Web Service WSsearchValues"
    dc#type hasValue _"http://www.wsmo.org/2004/d2#webservice"
    dc#description hasValue "Returns data about a stock given its name and market"
    dc#contributor hasValue {"Darek Kleczek", "Silvestre Losada"}
    dc#date hasValue _date(2005,12,21)
    dc#format hasValue "text/plain"
    dc#language hasValue "en-US"
    dc#rights hasValue _"http://www.isoco.com/privacy.html"
    wsml#version hasValue "$Revision: 01 $"
    wsml#endpointDescription hasValue
    _"https://aia.ebankinter.com/wsBrokerService/#wsdl.service(BrokerService)"
    endnfp

    importsOntology {
    _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#"
    importsOntology {
    _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"
    }

    capability
    _"http://users.isoco.net/~slosada/ontologies/bankinter/WSsearchValues#capability"

    sharedVariables {?request}

    /*
    */

    precondition
      nfp
        dc#description hasValue "There exists a request to check data of a stock"
        endnfp
        definedBy
          ?request memberOf smp#CheckStock.

    postcondition
      nfp
        dc#description hasValue "The service returns an instance of a stock with
attribute data"
        endnfp
        definedBy
          (?request[stockName hasValue ?stockName] memberOf smp#CheckStock implies
exists ?stock (?stock[isPartOfcompany hasValue ?stockName] memberOf
sm#Stock)).

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"

namespace
{
  _"http://users.isoco.net/~slosada/ontologies/bankinter/WSsearchValuesInterface.wsml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  smp
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#",
  WS _"http://users.isoco.net/~slosada/ontologies/bankinter/WSsearchValues.wsml#",
  QoS-eBanking _"http://www.example.org/QoS-eBanking"
}

webService _"http://users.isoco.net/~slosada/ontologies/bankinter/WSsearchValues.wsml"

interface WSsearchValuesInterface

```

```

importsOntology {"http://www.example.org/QoS-eBanking"}
nonFunctionalProperties

  dc:description hasValue "QoS properties of paid stock market information
service."

  QoSParameter availabilityLevelForPaidService
    instance paidServiceAvailability memberOf QoS-eBanking#Availability
    hasMeanValue hasValue _double("0.999999")
    hasBaseNumberOfRequestsToEvaluateAvailability
hasValue _integer("100,000")

  QoSParameter dataFreshnessForPaidService
    instance paidServiceDataFressNess memberOf QoS-eBanking#DataFreshness
    hasMeanValue hasValue _double("10.0")
    hasStandardDeviation hasValue _double("1.0")
    hasMeasurementUnit hasValue QoS-eBanking#minute

  QoSParameter capacityForPaidService
    instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
    hasMeanValue hasValue _double("10,000")
    hasMeasurementUnit hasValue QoS-eBanking#requestsPerSecond

  QoSParameter responseTimeForPaidService
    instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
    hasMeanValue hasValue _double("1.5")
    hasStandardDeviation hasValue _double("0.25")
    hasMeasurementUnit hasValue QoS-eBanking#second

endNonFunctionalProperties

choreography WSsearchValuesChoreography
  stateSignature
    importsOntology
  ("http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"}
  in {
    smp#CheckStock withGrounding
WS#wSDL.interfaceMessageReference(BrokerServiceSoap/searchValue/In)
  }
  out {
    sm#Stock withGrounding
WS#wSDL.interfaceMessageReference(BrokerServiceSoap/searchValue/Out)
  }

guardedTransition WSsearchValuesChoreographyRules

  forAll {?request} with (
    ?request memberOf smp#CheckStock
  )

  do
    add( _#[[] memberOf sm#Stock)
  endForAll

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace {"http://users.isoco.net/~slosada/ontologies/bankinter/WSSendAlert.wsml#",
  dc _"http://purl.org/dc/elements/11#",
  foaf _"http://xmlns.com/foaf/01/",
  xsd _"http://www.w3c.org/2001/XMLSchema#",
  wsmo _"http://www.wsmo.org/2004/wsml#",
  sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
  smp
  _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"}

webService _"http://users.isoco.net/~slosada/ontologies/bankinter/WSSendAlert.wsml"

```

```

nfp
dc#title hasValue "Web Service WSSendAlert"
dc#type hasValue _"http://www.wsmo.org/2004/d2#webservice"
dc#description hasValue "Executes Buy or Sell action"
dc#contributor hasValue {"Darek Kleczek", "Silvestre Losada"}
dc#date hasValue _date(2005,12,21)
dc#format hasValue "text/plain"
dc#language hasValue "en-US"
dc#rights hasValue _"http://www.isoco.com/privacy.html"
wsml#version hasValue "$Revision: 01 $"
wsml#endpointDescription hasValue
_"https://aia.ebankinter.com/wsBrokerService/#wsdl.service(BrokerService)"
endnfp

importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#"}
importsOntology {
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#"}

capability
_"http://users.isoco.net/~slosada/ontologies/bankinter/WSSendAlert#capability"

/*
No shared variables and postcondition for this Web Service
*/

precondition
nfp
dc#description hasValue "A request to send an alert has to exist"
endnfp
definedBy
?request memberOf smp#SendAlert.

wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"

namespace
{"_http://users.isoco.net/~slosada/ontologies/bankinter/WSSendAlertInterface.wsml#",
sm _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
smp
_"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#",
WS _"http://users.isoco.net/~slosada/ontologies/bankinter/WSSendAlert.wsml#",
QoS-eBanking _"http://www.example.org/QoS-eBanking"}

webService _"http://users.isoco.net/~slosada/ontologies/bankinter/WSSendAlert.wsml"

interface WSSendAlertInterface

importsOntology {"_http://www.example.org/QoS-eBanking"}
nonFunctionalProperties

dc#description hasValue "QoS properties of paid stock market information
service."

QoSParameter availabilityLevelForPaidService
instance paidServiceAvailability memberOf QoS-eBanking#Availability
hasMeanValue hasValue _double("0.999999")

hasBaseNumberOfRequestsToEvaluateAvailability hasValue _integer("100,000")

QoSParameter capacityForPaidService
instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
hasMeanValue hasValue _double("10,000")
hasMeasurementUnit hasValue QoS-eBanking#requestsPerSecond

QoSParameter responseTimeForPaidService
instance paidServiceCapacity memberOf QoS-eBanking#MaximumCapacity
hasMeanValue hasValue _double("1.5")
hasStandardDeviation hasValue _double("0.25")

```

```

hasMeasurementUnit hasValue QoS-eBanking#second

endNonFunctionalProperties

choreography WSSendAlertChoreography
stateSignature
importsOntology
( _"http://users.isoco.net/~slosada/ontologies/bankinter/StockMarket.wsml#",
_ "http://users.isoco.net/~slosada/ontologies/bankinter/StockMarketProcess.wsml#" }
in {
    smp#SendAlert withGrounding
WS#wSDL.interfaceMessageReference(Service1Soap/sendAlert/In)
}

out {
    smp#Confirmation withGrounding
WS#wSDL.interfaceMessageReference(Service1Soap/sendAlert/Out)
}

guardedTransition WSSendAlertChoreographyRules

forall {?request} with (
?request memberOf smp#SendAlert
)

do
add( _#[[] memberOf smp#Confirmation)
endforall

```