D3.5 SWF Use Case - Final Version 1.2

WSMO / SWF Working Draft 19 October 2004

This version:
http://www.deri.at/research/projects/swf/usecase/20041019/
Latest version:
http://www.deri.at/research/projects/swf/usecase/
Previous version:
http://www.deri.at/research/projects/swf/usecase/20041001/

Editors:
Michael Stollberg, DERI

Authors:
Michael Stollberg, DERI
Ioan Toma, DERI
Uwe Keller, DERI
Bernhard Keimel, Quarto
Peter Zugmann, Quarto

Abstract

This document specifies a use case for the Semantic Web Fred project, short SWF, in the domain of purchasing furniture. The aim of this use case is testing and validation as well as demonstrating the the SWF technology. The use case is open for reuse and apdaption within the SDK-Cluster working groups around WSMO.

Related Documents

WSMO Standard: WSMO D2 v0.2 Web Service Modeling Ontology (WSMO), last version at: http://www.wsmo.org/2004/d2/

WSMO Use Case: WSMO D3.2 v0.1 WSMO Use Case and Testing

SWF: all deliverables

Table of contents

1. Introduction

This document specifies a use case for the Semantic Web Fred project, short SWF. The aim is to test and validate the SWF technology, as well as defining a showcase for demonstrating the SWF technology. The use case is settled in the domain of purchasing furniture: we define buyers and sellers of whom each has a Goal (a buyer wants to purchase some furinture, and the seller wants to sell furniture); the goals of the participants can only be solved by a cooperation of a buyer and a seller.

Before we give a detailed overview of the use case in Section 2 and the models of SWF components in Section 3, we first briefly recall the objectives of SWF, the conceptual architecture, and the specification of the SWF components along with the SWF Component Description Language on basis of WSMO Standard [WSMO]. More detailed specification of the SWF technology can be found in the realted SWF Deliverables available at the SWF Website at http://www.deri.at/research/projects/swf/.

1.1. SWF Conceptual Architecture

The traditional approach that dominated IT-system design over decades is the request-provider model: a request carries the information of the service, invokes the service and collects the result of the service. In order to overcome the limitations in regard to support of the user perspective and reusability of services, the “rational agent approach” enforced in the AI-community introduces the notion of goals. The major motivation for this novel approach is that the user desire is **decoupled** from the resolving services, and the connection is **dynamically derived** during runtime by **intelligent mechanisms**. This allows **context-independent architectures** with **maximal reuse** of existing computational resources. The technological challenge for such systems is an expressive, well-defined description language and, on this basis, a coherent framework of mechanisms for goal resolution. This is basically the underlying idea of the Web Service Modeling Ontology [WSMO] – identifying Goals and Service, have declarative semantic descriptions of WSMO elements whereupon intelligent mechanisms work.

Semantic Web Fred realizes the WSMO approach, adding the notion of cooperative goal resolution. According to the paradigm of agents as autonomously acting entities in a software environment and with regard to task solving in real-world settings, more complex goals can only be achieved in cooperation as needed facilities might be held by different entities. Further, a cooperative goal resolution will only take place if it is profitable for all participants. In consequence, **symmetry** is a supplementary design paradigm of SWF: every acting entity in the system has Goals to be resolved as well as Services that provide the functionality the entity brings into a coopera-tive goal resolution. This is reflected in the SWF conceptual archtecture shown in Figure 1:

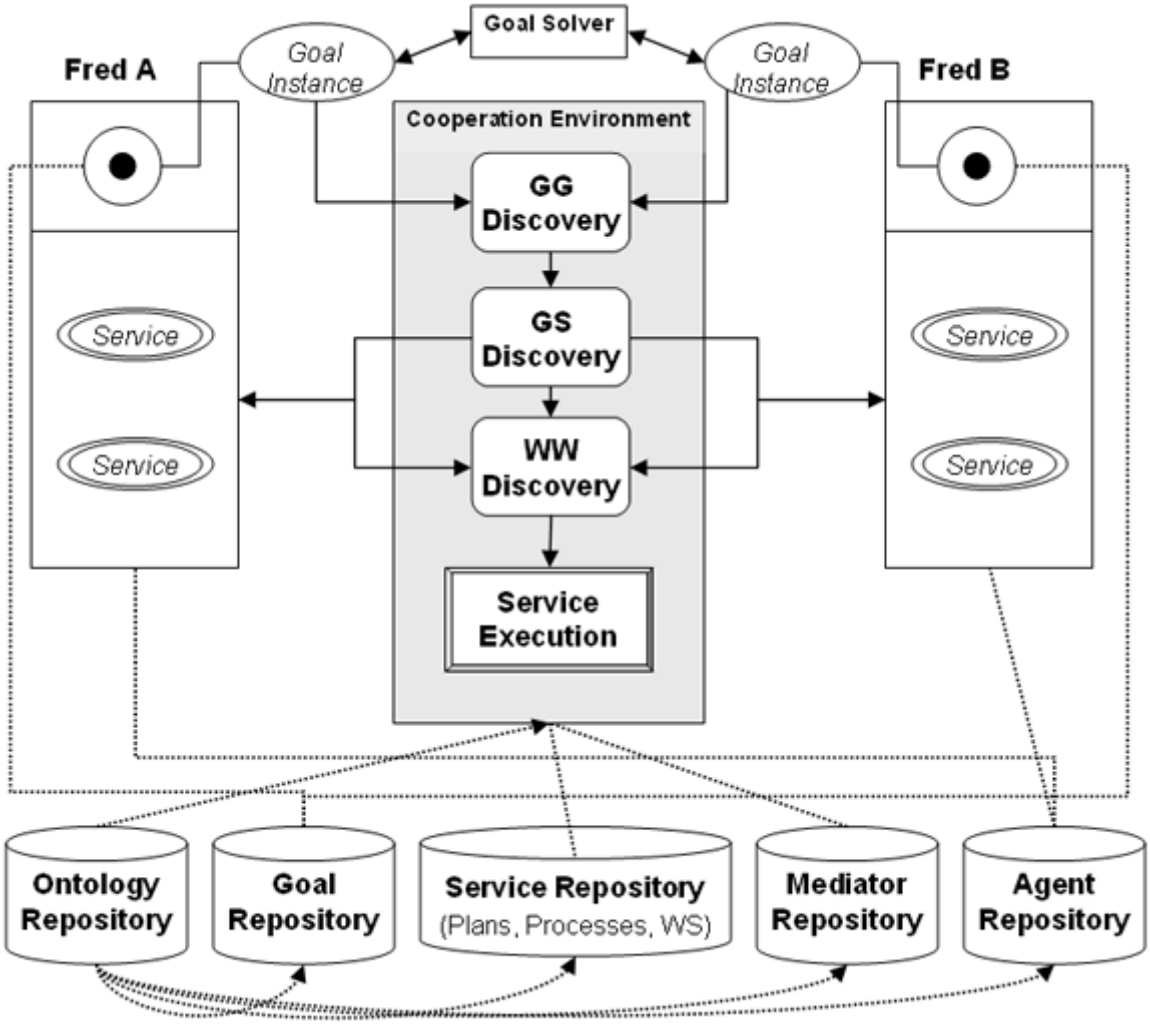

Figure 1. SWF Conceptual Architecture

The conceptual architecture illustrates the structure of Freds, the component repositories, and the architecture of the cooperative goal resolution environment. For explaining the overall functionality, we refer to the use specified in the remainder of this document : a buyer wants to buy a chair, and a seller wants to sell furniture (including chairs). These two goals can only be achieved in a cooperation between the buyer and seller, which is realized in SWF as follows.

Image that both parties are represented by Freds in the system (Fred A as the buyer, Fred B as the seller), and services for purchasing are available. At a certain point in time, both create Goal Instances (the ball on top of a Fred-box), indicating their desire to solve the Goal. Out of possibly several Freds in the system, so-called Goal-to-Goal Discovery (short: GG Discovery) detects Fred A and Fred B for cooperation by determining the compatibility of the Goal Instances. Then, A and B have to find suitable services that provide the specific functionalities needed for cooperation. For instance, Fred B as the seller needs a browsing facility for his product catalogue, ability to place a contract of purchase, and a facility for receiving payment; Fred A as the buyer needs the opposite facilities. This is performed by Goal-to-Service Discovery (short: GS Discovery) which detects appropriate services for each Fred in the service repository, similar to Web Service Discovery. In order to allow interaction

of the services for automated goal resolution, the services used by the cooperation partners have to be reconcilable with regard to their external behavior (the external visible business processes) and the expected messaging sequence. Thus, the third step in establishing cooperative goal resolution is Service-to-Service Discovery (short: WW Discovery, for societal reasons), establishing the behavioral compatibility of the services detected in GS Discovery. The result of the three discovery mechanisms is a Cooperation Contract that contains all information relevant for execution of the cooperation (the Freds, the Goals, and the Services to be used). Then, the cooperation partners are called into a Meeting Room (see above), wherein this contract is processed by the Service Execution component that handles all execution-related aspects (errors, compensation, etc.).

The goal resolution process is managed by the Goal Solver: it controls the discovery mechanisms, including iterations (e.g. re-call GG Discovery when GS Discovery fails) and rollbacks when the execution of services fails; it also monitors the resolution process of the Goal Instances of the cooperation partners. Although the goal resolution technology in SWF is designed for automated resolution of cooperative goals in particular, Goals that do not need cooperative resolution can be handled as well. Therefore, a Fred is called in a Singleton Meeting where the required services are executed without agent interaction.

1.2 SWF Components

The following briefly describes the SWF components along with the SWF Component Description Language on basis of WSMO Standard [WSMO]. Detailed specification of the SWF components are given in the realted SWF Deliverables available at the SWF Website at http://www.deri.at/research/projects/swf/.

Freds are software agents that act as an electronic representative on behalf of its owner. Goals can be assigned to a Fred for automated resolution, and a Fred has usage permissions for Services existing in the system. The FredBase is the agent runtime environment of the FRED system that comprises all management facilities for Freds. Interactions between Freds take place in Meeting Rooms, wherein all computational resources are available for Service Execution, see detailed explanations in [FRED Whitepaper].

Ontologies provide the formal semantics of the information used by all other components. SWF supports WSMO Ontologies. For internal management, ontologies are transformed into Java Objects in the FRED system, so-called Smart Objects. The expressiveness of Smart Objects for ontology representation is comparable to OWL-DL. The Smart Object technology comprises facilities for management, evolution, mismatch-handling, and persistent storage for ontology data.

Goals A Goal expresses a desire that a user delegates to a Fred for automated resolution. When assigned to a Fred, the information kept in the Goal is used to determine potential cooperation partners and the SWF Services needed to perform the resolution automatically. SWF distinguishes 2 different notion of of Goals:

- **Goal Template:** A Goal Template defines the ontological structure wherefrom Goal Instances are derived as concrete expressions of a desire. Goal Templates are pre-defined in the system, and a task assigned to a Fred is a

Goal Instances created out of an existing Goal Template.
A Goal Template is equivalent to WSMO Goals, described by the description elements for WSMO Goals: non-functional properties, imported ontologies, used mediators (OO Mediators), postcondition and effect. The listing shows the description elements of SWF Goal Templates, which are WSMO Goals; SWF Goal Templates can be interchanged as WSMO Goals with other WSMO-enabled applications.

Listing. Goal Template Description

```
entity goal
    nonFunctionalProperties ofType nonFunctionalProperties
    importOntologies ofTypeSet ontology
    usedMediators ofTypeSet {ooMediator, ggMediator}
    postConditions ofTypeSet axiom
    effects ofTypeSet axiom
```

- **Goal Instance**: A Goal Instance is the concrete expression of a desire, created by a Fred during runtime by instantiating a Goal Template (i.e. definition of specific values for certain attributes of the ontology objects and refinement of conditions defined in the Goal Template). A Goal Instance represents a task assigned to a Fred, and it gets resolved in the goal resolution process. It might take several cooperation meetings to resolve a Goal Instance. A Goal Instance inherits the descriptive information of the corresponding Goal Template, instantiates the ontological structures and refines the respective conditions, and carries additional information needed for managing the goal resolution process. The description elements for SWF Goal Instances are the following, the overall description structure of Goal Instances is given in the Listing below:
    - *Non-Functional Properties*: in addition to the WSMO Core Properties inherited from Goal Template: timeConstraints (time frame for resolution), resourceConstraints (preferred cooperation partners), goalResolutionConstraints (other user preferences).
    - *owner*: link to the Fred that owns the Goal Instance
    - *submission*: the information that will be handed over as input to a service.
    - *postcondition*: Goal Template postcondition, with optional refinements.
    - *effect*: Goal Template effects, with optional refinements.
    - *status*: resolution process information, possible states: open, processing, solved, cancelled, not solved.

Listing. Goal Instance Description

```
entity goalinstance
    instanceOf ofType goal
    nonFunctionalProperties ofType nonFunctionalProperties,
                                   timeConstraints,
                                   resourceConstraints,
                                   goalResolutionConstraints
    owner ofTypeSet Fred-ID
    submission ofTypeSet instance
    postConditions ofTypeSet axiom
    effects ofTypeSet axiom
    status ofType {open, processing, solved, cancelled, not solved}
```

Additionally, SWF defines **Coopertive Goals** that describes a Goal that has to be solved in cooperation of multiple partners. It defines compatible Goal Templates, of which Freds carry Goal Instances as partners in cooperative goal resolution. With regard to the use case, a Cooperative Goal 'purchase' has the compatible Goal Templates 'buy chair' and 'sell furniture'. Thus, a Cooperative Goal defines compatible cooperation roles on an ontological level. Cooperative Goals are described by the following elements:

- *Non-Functional Properties*: WSMO Core Properties.
- *compatible Goal Templates*: Goal Templates that are defined to be compatible.
- *cooperativeGoalConstraints*: conditions that resolve mismtaches between the compatible Goal Templates. This is realized by a WSMO GG Mediator.

Listing. Cooperative Goal Description

```
entity cooperativegoal
    nonFunctionalProperties ofType nonFunctionalProperties
    cooperativeGoalGroup ofTypeSet goal
    cooperativeGoalConstraints ofTypeSet ggMediator
```

Services The SWF Service Model is comprised of three different service types that can be used for automated goal resolution: (1) *Plans* are internal services implemented as Java programs, normally used for simple functionalities; (2) *Processes* are multi-step services wherein each activity can be resolved arbitrarily by a Goal or another Service, thus allow definition of complex, nested services; also (3) external Web Services can be used, which are executed remotely at their respective provider. All Services types in SWF are described by a common description scheme, and the resolution of the service type is left for service execution. SWF Services are described as WSMO Services by the following notions: *non-functional Properties* (Core Properties and Web Service specific properties as defined in WSMO), a *Capability* as a functional description (identical with WSMO Web Service Capabilities), and an *Interface* that specifies the behavior interfaces of a service needed for service usage. The Interface description of Services in SWF consits only of the Choreography Interface description as defined in [WSMO Choreography]; Service aggregation as addressed in WSMO within the notion of Orchestration is captured by the SWF process technology.

The following listing shows the description structure of SWF Services. Although completely described WSMO Web Services can be supported, only a subset of the description information are used in SWF:

- from the Web Service Interface, only the Choreography Interface definition is used in SWF
- for simplicity, SWF services only have 1 Choregraphy

this allows to specify a SWF Services description in a single file, with the structure defined in Listing below.

Listing. Service Description

```
entity webService
    nonFunctionalProperties ofType  wsNonFunctionalProperties
```

```
    importOntologies ofTypeSet ontology
    usedMediators ofTypeSet {ooMediator, wgMediator}
    capability ofType capability
      preconditions ofTypeSet axiom
      assumptions ofTypeSet axiom
      postconditions ofTypeSet axiom
      effects ofTypeSet axiom
    choreographyinterface ofType choreography
```

Mediators are used as to connect SWF components and resolve ouccuring heterogeneity. Therefore, WSMO Mediators are used. Development of mediation facilities is not addressed within SWF. Mediators used in SWF are WSMO Mediators, therefore described as WSMO Mediators with the Mediator description specification defined in WSMO Standard.

Listing. Mediator Description

```
entity mediator
    nonFunctionalProperties ofType nonFunctionalProperties
    importOntologies ofTypeSet ontology
    source ofTypeSet {ontology, goal, webService, mediator}
    target ofType {ontology, goal, webService, mediator}
    mediationService ofType {goal, webService, wwMediator}

entity ooMediator subEntityOf mediator
    source ofTypeSet {ontology, ooMediator}

entity ggMediator subEntityOf mediator
    usedMediators ofTypeSet ooMediator
    source ofTypeSet {goal, ggMediator}
    target ofType {goal, ggMediator}

entity wgMediator subEntityOf mediator
    usedMediators ofTypeSet ooMediator
    source ofType {webService, wgMediator}
    target ofType {goal, ggMediator}

entity wwMediator subEntityOf mediator
    usedMediators ofTypeSet ooMediator
    source ofType {webService, wwMediator}
    target ofType {webService, wwMediator}
```

SWF Repository is an UDDI registry which holds all SWF components, apart form Freds. It realizes a hybrid architecture in order to combines the benefits of central and peer-to-peer repositories. Therein, the non-functional properties of each component are mapped to equivalent UDDI information types and are stored centrally in the registry, while the detailed semantic descriptions are kept locally in a persistent repository at the respective owner. Publishing and retrieval is supported by the regular UDDI functionalities.
In order to support interchangeability, the SWF component repository is aligned with the WSMO Registry which has the same architecture [Herzog et al., 2004]. For concurrent publishing, SWF component descriptions are transformed into the corresponding WSMO component description, facilitated by the fact that the SWF uses WSML as the component description language.

2. Use Case Overview

The use case specified in this document is settled in the domain of purchasing furniture. In an SWF-enabled virtual, agent-based B2C marketplace, there are several Buyers that want to purchase pieces of furniture as well as Sellers that want to sell furniture. The marketplace has a national cover for Austria. The marketplace is realized in SWF, and all relevant SWF components are available in the system. This section explains the overall setting of the use case, and specifies the conceptual structure of the distinct SWF components as well as their interdependencies. The related models of each component are provided in Section 3.

The use case is designed with the intention to test, validate and demonstrate the SWF Mechansims for establishing cooperations between Freds for cooperative goal resolution. Therefore, we define different Goal Templates, Cooperative Goals, and SWF Services for buyers and sellers with regard to specific match-making scenarios within the distinct SWF Discoveres. The following explains the design of the specific SWF components in the use case.

2.1 Ontology

With regard to the basic design principle of modularized ontologies, we define three separate domain ontologies as the the terminology definitions for the use case. The following specifies the scope and the structure of the ontontologies.

2.1.1 Furnishing Ontology

This ontology describes the domain of furnitures as ususally can be found within rooms, and the relation between furniture and rooms; pieces of furniture are bought and sold by the participants of the virtual marketplace. The conceptual structure of this ontology is the following:

- taxonomy of furniture
- taxonomy of dwellings
- taxonomy of rooms
- taxonomy of realtions between furniture and rooms

2.1.2 Marketplace Ontology

This ontology contains all notions concerned with buyers and sellers as participants of the virtual marketplace, products, purchasing, payment and delivery. The markeplace ontology is inspired by the Purchase Ontology defined in [WSMO Use Case]; that Purchase Ontology is based on RosettaNet, thus concerned with B2B purchase orders. In constrast, the Marketplace Ontology is concerned with B2C purchasing; thus, we do not import the Purchase Ontology, but re-use some modelling constructs. The main building blocks of the Marketplace Ontology are:

- marketplace participants: defines buyers and seller with their contact / shipping / billing addresses

- product: describes products available in the marketplace
- purchase contract: defines a contract of purchase, which is the result of a successful purchase between a byuer and a seller
- payment method: specifies the payment methods accepted in the marketplace
- delivery: specifies types of delivery, i.e. how a buyer receives a purchased product

2.1.3 Location Ontology

This ontology describes locations and addresses with special regard to postal addresses in Austria (as the SWF virtual markeplace is settled in Austria). We use the Location Ontology as specified in [WSMO Use Case] with minor changes. The main contructs of this ontology are:

- taxonomy of locations (country, state, city, etc.)
- postal addresses located in a location

2.2 Goal Templates

The following specifies a set of pre-defined SWF Goal Templates whereof SWF Goal Instances can be created by marketplace participants and assigned to their distinct Freds. We define only a few Goal Templates in order to support the general desires a marketplace participant can have; when creating a Goal Instance, the notions defined in the Goal Template can be arbitrarily refined.

2.2.1 Buyer Goal Templates

It is assumed that the Buyers in the marketplace are private individuals that want to buy pieces of furniture.

Buyer Goal Template 1:

- **desire:** buying a single piece of furniture (furniture is the superconcept of all types of furniture), and get the furniture delivered to a ship address
- **postcondition:** a contract of purchase for a buyer for a specific piece of furniture, payment by creditcard
- **effect:** delivery of the purchased piece of furniture to a specific shipping address in Austria

Buyer Goal Template 2 (small revision of Buyer Goal Template 1):

- **desire:** purchasing a single piece of furniture, and get the furniture delivered (by drop ship oir by self collection)
- **postcondition:** a contract of purchase for a buyer for a specific piece of furniture, payment by ANY PAYMENT METHOD
- **effect:** delivery of the purchased piece of furniture to a specific shipping address in Austria, or self collection by the buyer

Buyer Goal Template 3:

- **desire:** get detailed product information for a specific piece of furniture without purchasing
- **postcondition:** all information on a specific piece of furniture which can be purchased as a product from a company or store
- **effect:** none

Buyer Goal Template 4:

- **desire:** buy a piece of furniture from a private seller, i.e. not from a company or store
- **postcondition:** a contract of purchase for a buyer for a specific piece of furniture from a seller who is a private person, payment method unspecified (means: all payment methods are accepted)
- **effect:** the buyer collects the bought piece of furniture himself (self collection as special type of delivery)

2.2.2 Seller Goal Templates

It is assumed that there 2 types of sellers in the marketplace:

1. comercial stores that provide serveral types of furniture and want to sell these
2. private sellers that want to sell specific pieces of furniture

Seller Goal Template 1:

- **desire:** sell a single piece of furniture, and deliver the furniture to the ship address of a buyer (in Austria only)
- **postcondition:** a contract of purchase for a seller for a specific piece of furniture, payment by any payment method
- **effect:** delivery of the purchased piece of furniture to the buyer's shipping address (restricted to Austria), or as self collection by the buyer

Seller Goal Template 2:

- **desire:** provide detailed information about products (pieces of furniture) available in a store
- **postcondition:** all information on a specific piece of furniture which can be purchased as a product from a company or store
- **effect:** none

Seller Goal Template 3:

- **desire:** a private seller sells a piece of furniture
- **postcondition:** a contract of purchase for a specific piece of furniture where the seller is a private person, payment method unspecified (means: all payment methods are accepted)

- **effect:** the buyer collects the bought piece of furniture himself (self collection)

2.3 Cooperative Goals

according to the structure of the Goal Templates defined above, we specify the following Cooperative Goals. The Cooperative Goals in this Use Case consist of 2 compatible Goal Templates only: one Buyer Goal Template and one Seller Goal Template, as a Buyer and a Seller have to interact in order to resolve their respective Goals. In general, a Cooperative Goal can specify multiple compatible Goal Templates, when multi-party collaborations are needed to resolve individual Goals in a cooperative manner.

Cooperative Goal 1:
* Buyer Goal Template 1
* Seller Goal Template 1

Cooperative Goal 2:
* Buyer Goal Template 2
* Seller Goal Template 1

Cooperative Goal 3:
* Buyer Goal Template 2
* Seller Goal Template 3

Cooperative Goal 4:
* Buyer Goal Template 3
* Seller Goal Template 2

Cooperative Goal 5:
* Buyer Goal Template 4
* Seller Goal Template 3

2.4 Services

The following specifies services for Buyers and Sellers in the marketplace. These are assumed to be pre-defined in the marketplace.

For specification of the Services in the following, we describe the overall functionality of the service in natural lanuage, and specify the content of the service description elements. These specifications are modeled in WSML in Section 3.5.

2.4.1 Buyer Services

The Buyer Services are assumed to be provided by the marketplace owner. Each Fred in the marketplace that is an electronic representative of a Buyer has usage permissions for these Services; services for buyers and sellers are distinguished in the serviceontology (see Section 3.5.1).

Buyer Service 1:

- **functionality:** buy a piece of furniture (i.e. receive a valid contract for purchase), and get the bought item delivered to a shipping address in Austria
- **SWF Service Type:** FRED Plan
- **capability**
    - **precondition:** required input is: [1] a piece of furniture to be bought, [2] buyer information (name, ship and bill address in Austria) , and [3] a payment method (types of payment methods are predefined in the ontology)
    - **assumption:** if payment method is credit card, the credit card has to be not expired
    - **postcondition:** a contract a purchase for the piece of furniture provided as input, with the buyer provided as input, and a payment method (all pre-defined payment methods are accepted)
    - **effect:** delivery of the purchased piece of furniture to the buyer's shipping address specified in the input, or via self collection by the buyer
- **choreography interface**
    1. Step: to be done
    2. Step

Buyer Service 2:

- **functionality:** gathers product information for pieces of funrinture offered in the market, and returns them by email to the requester who is a buyer
- **comment:** for demonstration purpose, there is a suitable set intances of furniture defined which sellers can offer a (sub)set of); for a more advaned setting, this service has to (a) query the databases of stores in the market, and (b) collect all furniture offered for sale by private sellers.
- **SWF Service Type:** FRED Plan
- **capability**
    - **precondition:** required input is: [1] the piece of furniture that product information are requested for, and [2] buyer information (name, email-address)
    - **assumption:** the buyer has to be registered as marketplace participant
    - **postcondition:** returns product information for all piece of furniture available as sale offers in the marketplace which match the furniture description provided as input; sends this information as email to the requester (the buyer provided as input)
    - **effect:** none
- **choreography interface**
    1. Step: to be done
    2. Step

Buyer Service 3:

- **functionality:** buy a piece of furniture from a private seller with self-collection as the delivery method
- **SWF Service Type:** FRED Plan
- **capability**
  - **precondition:** required input is: [1] the piece of furniture to be bought, [2] a buyer (name, bill address in austria)
  - **assumption:** there is at least 1 private seller in the markeplace who offers a piece of furniture that fits the desired furniture
  - **postcondition:** returns a contract of purchase between the buyer provided as input and a private seller for a piece of furniture that fits the desire; the accepted payment method are all those pre-defined in the ontology
  - **effect:** the dilvery method for the bought piece of furniture is self-collection
- **choreography interface**
  1. Step: to be done
  2. Step

2.4.2 Seller Services

There are 3 furniture stores registered as sellers in the markeplace: "Leiner" and "Kika" which are Austrian based furniture stores, and IKEA as an international furniture store. These stores provide their own services, for which only the service provider himself has usage permissions (NOTE: these services are only imaginary services for testing and demonstrating; there are no real purchases performed in the system, and this use case does not effect or display any real-world settings). Additionally, there are some seller services provided by the marketplace owner, for which every Fred that represents a seller has usage permissions.

Seller Service "Leiner":

- **functionality:** Leiner Web Service for purchasing a piece of furniture for all types of furniture offered by Leiner; the product is delivered either by the Leiner Delivery Service or via self-collection by the purchaser
- **provider / usage permission:** Leiner / only Leiner
- **SWF Service Type:** FRED Process
- **capability**
  - **precondition:** required input is: [1] piece of furniture to be sold, [2] Leiner as a seller (located in Austria), [3] accepted payment methods are credit card, invoice, or cash
  - **assumption:** the desired piece of furniture has to be available at Leiner
  - **postcondition:** returns a contract of purchase for a piece of furniture that fits the desire between the buyer provided as input and Leiner as the seller, with an accepted payment method (credit card or invoice)

    - **effect:** delivery of bought piece of furniture by the Leiner Delivery Service (not a Web Service), to buyer shipping addresses in Austria only; or the buyer gets the furniture by self-collection
- **choreography interface**
    1. Step: to be done
    2. Step

Seller Service "Kika":

- **functionality:** Kika Web Service for purchasing a piece of furniture for all types of furniture offered by Kika; the product is delivered either by the Kika Delivery Service or via self-collection by the purchaser
- **provider / usage permission:** Kika / only Kika
- **SWF Service Type:** FRED Process
- **capability**
    -
    - **precondition:** required input is: [1] piece of furniture to be sold, [2] Kika as a seller (located in Austria), [3] accepted payment methods are credit card, invoice, or cash
    - **assumption:** the desired piece of furniture has to be available at Kika
    - **postcondition:** returns a contract of purchase for a piece of furniture that fits the desire between the buyer provided as input and Kika as the seller, with an accepted payment method (credit card or invoice)
    - **effect:** delivery of bought piece of furniture by the Kika Delivery Service (not a Web Service), to buyer shipping addresses in Austria only; or the buyer gets the furniture by self-collection
- **choreography interface**
    1. Step: to be done
    2. Step

Seller Service "IKEA":

- **functionality:** IKEA Web Service for purchasing a piece of furniture for all types of furniture offered by IKEA; the product is delivered either by the IKEA Delivery Service or via self-collection by the purchaser
- **provider / usage permission:** IKEA / only IKEA
- **SWF Service Type:** FRED Process
- **capability**
    - **precondition:** required input is: [1] piece of furniture to be sold, [2] IKEA as a seller (located in Austria), [3] accepted payment methods are credit card, invoice, or cash
    - **assumption:** the desired piece of furniture has to be available at IKEA

    - **postcondition:** returns a contract of purchase for a piece of furniture that fits the desire between the buyer provided as input and IKEA as the seller, with an accepted payment method (credit card or invoice)
    - **effect:** delivery of bought piece of furniture by the IKEA Delivery Service (not a Web Service), to buyer shipping addresses in Austria only; or the buyer gets the furniture by self-collection
- **choreography interface**
    1. Step: to be done
    2. Step

Seller Service 1:

- **functionality:** general service for selling a piece of furniture, returning a contract of purchase with delivery of the product as an effect (any type of delivery). All specific features can be defined by the service user (i.e. the seller that uses this Service).
- **provider / usage permission:** marketplace owner / all 'Seller Freds'
- **SWF Service Type:** FRED Process
- **capability**
    - **precondition:** required input is: [1] piece of furniture to be sold, [2] a seller (registered marketplace participant, located in Austria), [3] accepted payment methods are credit card or invoice
    - **assumption:** the desired piece of furniture has to be available as a product offered by the seller
    - **postcondition:** returns a contract of purchase for a piece of furniture that fits the desire between a buyer and the seller provided as input, with accepted payments credit card or invoice.
    - **effect:** delivery of bought piece of furniture by the a Delivery Service (not a Web Service), to buyer shipping addresses in Austria only; or the buyer gets the furniture by self-collection
- **choreography interface**
    1. Step
    2. Step

Seller Service 2:

- **functionality:** general service for providing detailed information on a piece of furniture that can be bought in the marketplace
- **comment:** this service is implemented as a simple search of the 'marketplace product repository' which holds all pieces of furniture that can be purchased in the marketplace: each seller signs in for those instances of furniture that he wants to sell; so, restricted search on furniture that are offered by a specific seller or a group of sellers (e.g. private sellers) can be performed. In a more

elaborated version, this service searches (a) the product repositories of stores in the market, and (b) all furniture offered for sale by private sellers

- **provider / usage permission:** marketplace owner / all 'Seller Freds'
- **SWF Service Type:** FRED Plan
- **capability**
  - **precondition:** required input is: [1] the piece of furniture that product information are requested for, [2] the seller or type of seller (e.g. private seller only) whose offers to be searched for
  - **assumption:** if product information from a specific seller are requested, this seller has to be registered in the marketplace
  - **postcondition:** returns product information (incl. item, price, seller) for pieces of furniture that are existing as sales items in the market by the seller specified or all sellers of the seller type specified
  - **effect:** none
- **choreography interface**
  1. Step: to be done
  2. Step

Seller Service 3:

- **functionality:** general service for selling a piece of furniture; only private sellers are allowed to use this service. The service is provided by the markeplace owner as a generic "contract of purchase" service; the delivery type is always self-collection
- **provider / usage permission:** marketplace owner / all 'Seller Freds' that represent a private seller
- **SWF Service Type:** FRED Process
- **capability**
  - **precondition:** required input is: [1] piece of furniture to be sold , [2] a private seller (registered marketplace participant, located in Austria), [3] accepted payment methods all payment methods pre-defined in the ontology
  - **assumption:** [1] the desired piece of furniture has to be available in the marketplace, [2] if the payment method is credit card, then the credit card has to be valid (i.e. not expired)
  - **postcondition:** returns a contract of purchase for a piece of furniture that fits the desire between the buyer and the private seller provided as input, with the payment method provided as input
  - **effect:** the buyer gets the furniture by self-collection
- **choreography interface**
  1. Step: to be done
  2. Step

2.5 Mediators

Mediators in SWF are WSMO Mediators. We define the following Mediators in the Use Case, distinuished by the types of WSMO Mediators.

2.5.1 GG Mediators

With regard to the SWF Use Case resources defined above, we need 2 GG Mediators

GG Mediator 1:

- **description:** creates Buyer Goal Template 1 from Buyer Goal Template 2, with restricting the payment method to credit card
- **source:** Buyer Goal Template 2
- **target:** Buyer Goal Template 1
- **reduction:** the payment method is credit card only

GG Mediator 2:

- **description:** as a cooperationGoalContraint, it restricts the payment method to credit card between Buyer Goal Template 1 and Seller Goal Template 1
- **source:** Buyer Goal Template 1
- **target:** Seller Goal Template 1
- **reduction:** the payment method is credit card only

2.6 Participants

The number of particpants in the use case, i.e. Freds registered and participating in the marketplace, is unlimited. Each Fred represents a participant which is either a buyer or seller; the Goal Instances that are assigned to a Fred are derived from the existing Goal Templates; a Buyer Fred has usage permissions for all Buyer Services existing in the system, and a Seller Fred has usage permissions for specific Seller Services.

3. Use Case Models

This section provides the models of the use case elements as specified in the previous section. All SWF components are defined in WSML-Core [WSML], which is used as the modeling language withi SWF. The defualt namespace for the resources of this usecase is: http://swf.quarto.at/repository/.

3.1 Ontologies

As specified above, we define three modulraized ontologies as the terminology definitions for the use case:

1. "Furnishing Ontology" describes the domain of furnitures as ususally can be found within houses
2. "Marketplace Ontology" describes all ontology notions needed to describe elements concerning with products, purchasing contracts, buyers and seller, payment, and delivery in the virtual SWF marketplace
3. "Location Ontology" describes locations (such as continents, countries and cities and their interrelation)

The ontologies specified in the following are intended to be "real ontologies" in the sense that they describe the specific domain as a shared conceptualization in a sufficient manner. Aspart from the Furnishing Ontology, we re-use the ontologies specified in the WSMO Deliverable D3.2 Use Case and Testing [WSMO Use Case] with some minor changes as needed for our specific use case.

Listing. Furniture Ontology (WSML file)

```
/**
 * Furnishing Ontology
 */

namespace <<http://swf.quarto.at/repository/ontologies/furnishing.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    loc:<<http://swf.quarto.at/repository/ontologies/location.wsml#>>
    xsd:<<http://www.w3.org/2001/XMLSchema#>>

ontology <<http://swf.quarto.at/repository/ontologies/furnishing.wsml>>

nonFunctionalProperties
    dc:title hasValue "Furniture Ontology (simplified)"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {"Furniture", "Dewelling", "Accessories", "Room"}
    dc:description hasValue "describes different types of furniture"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValues {"Michael Stollberg", "Ioan Toma"}
    dc:date hasValue "2004-09-20"
    dc:type hasValue <<http://www.wsmo.org/2004/d2/v0.3/20040329/#ontos>>
    /**
     * ontologies are modeled as WSMO Ontologies
     */
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:coverage hasValue "project specific / general"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.9 $"
endNonFunctionalProperties


importedOntologies {<<http://swf.quarto.at/repository/ontologies/location.wsml>>}

/*****************************************************************************************
The Furnishing ontology consists of 4 major sub-ontologies:
        1. "Dwelling Ontology"
        2. "Room Ontology"
        3. "Furniture Ontology"
        4. "Accessories Ontology"
*****************************************************************************************/

/*****************************************************************************************
        Dwelling Ontology
*****************************************************************************************/
/**
 * concept definitions
 */
```

```
//top concept for every kind of dwelling

concept dwelling
    nonFunctionalProperties
        dc:description hasValue "general class - dwelling"
    endNonFunctionalProperties
    hasRooms ofType xsd:boolean

concept flat subConceptOf dwelling
    nonFunctionalProperties
        dc:description hasValue "flat subclass of dwelling"
    endNonFunctionalProperties

concept apartment subConceptOf flat
    nonFunctionalProperties
        dc:description hasValue "apartment subclass of flat"
    endNonFunctionalProperties

concept privateHouse subConceptOf dwelling
    nonFunctionalProperties
        dc:description hasValue "privateHouse subclass of dwelling"
    endNonFunctionalProperties


/*****************************************************************************************************
        Room Ontology
*****************************************************************************************************/
/**
 * concept definitions
 */
//top concept for every kind of room
concept room
    nonFunctionalProperties
        dc:description hasValue "general class - room"
    endNonFunctionalProperties
    height ofType xsd:integer
    sqm ofType xsd:float
    lengthOfWalls ofType set xsd:float
    numberOfWindows ofType xsd:integer
    numberOfDoors ofType xsd:integer

concept bathroom subConceptOf room
    nonFunctionalProperties
        dc:description hasValue "bathroom subclass of room"
    endNonFunctionalProperties

concept bedroom subConceptOf room
    nonFunctionalProperties
        dc:description hasValue "bedroom subclass of room"
    endNonFunctionalProperties

concept dinigroom subConceptOf room
    nonFunctionalProperties
        dc:description hasValue "diningroom subclass of room"
    endNonFunctionalProperties

concept livingroom subConceptOf room
    nonFunctionalProperties
        dc:description hasValue "livingroom subclass of room"
    endNonFunctionalProperties

concept homeOffice subConceptOf room
    nonFunctionalProperties
        dc:description hasValue "homeOffice subclass of room"
    endNonFunctionalProperties

concept kitchen subConceptOf room
    nonFunctionalProperties
        dc:description hasValue "kitchen subclass of room"
    endNonFunctionalProperties


/*****************************************************************************************************
        Furniture Ontology
*****************************************************************************************************/
/**
```

```
 * concept definitions
 */
//top concept for every kind of furniture
concept furniture
    nonFunctionalProperties
        dc:description hasValue "general class of furniture"
    endNonFunctionalProperties
    name ofType xsd:string
    width ofType xsd:integer
    height ofType xsd:integer
    depth ofType xsd:integer
    manufacturer ofType manufacturer
    material ofType set material
    color ofType xsd:string

//the entity who made the furniture; the seller concept is defined in SWFMO ontology
concept manufacturer
    nonFunctionalProperties
        dc:description hasValue "furniture manufacturer"
    endNonFunctionalProperties
    name ofType xsd:string
    address ofType loc:address

//the material the furniture is made off
concept material
    nonFunctionalProperties
        dc:description hasValue "material a piece of furniture is made of,
                                 includes material-type and quality"
    endNonFunctionalProperties
    type ofType xsd:string
    quality ofType xsd:integer


/*************************************************************************************************
 * BED hierarchy
*************************************************************************************************/
concept bed subConceptOf furniture
    nonFunctionalProperties
        dc:description hasValue "bed as a subclass of furniture"
    endNonFunctionalProperties
    numberOfPersons ofType xsd:integer

concept singleBed subConceptOf bed
    nonFunctionalProperties
        dc:description hasValue "a bed for only one person"
    endNonFunctionalProperties
    //restriction: only one person

concept doubleBed subConceptOf bed
    nonFunctionalProperties
        dc:description hasValue "a bed for two persons"
    endNonFunctionalProperties
    //restriction: two person

concept kingSizeBed subConceptOf bed
    nonFunctionalProperties
        dc:description hasValue "a huge bed"
    endNonFunctionalProperties

concept pencilePostBed subConceptOf bed
    nonFunctionalProperties
        dc:description hasValue "a bed with four slim posts; used alone or with a canopy"
    endNonFunctionalProperties

/*************************************************************************************************
 * STORAGE FURNITURE hierarchy
*************************************************************************************************/
concept storageFurniture subConceptOf furniture
    nonFunctionalProperties
        dc:description hasValue "kind of furniture used to store diferent thinghs"
    endNonFunctionalProperties
    storageCapacity ofType xsd:float

/**
 * CABINET hierarchy
 */
```

```
concept cabinet subConceptOf storageFurniture
    nonFunctionalProperties
        dc:description hasValue "cabinet as a subclass of furniture"
    endNonFunctionalProperties

concept fileCabinet subConceptOf cabinet
    nonFunctionalProperties
        dc:description hasValue "cabinet used to store documents in a office"
    endNonFunctionalProperties
    /**
     * todo specify the realtion fileCabinet and homeOffice
     */

/**
 * CHEST hierarchy
 */
concept chest subConceptOf storageFurniture
    nonFunctionalProperties
        dc:description hasValue "a box with a lid used especially for the safekeeping of belongings"
    endNonFunctionalProperties

concept blanketChest subConceptOf chest
    nonFunctionalProperties
        dc:description hasValue "a chest used for general storage, usually kept in the bedroom"
    endNonFunctionalProperties
    /**
     * todo specify the realtion between blanketChest and bedroom
     */

concept coffer subConceptOf chest
    nonFunctionalProperties
        dc:description hasValue "multi-functional traveling chest with handles and a domed lid but without feet, usually
made of oak"
    endNonFunctionalProperties

concept commode subConceptOf chest
    nonFunctionalProperties
        dc:description hasValue "french term for a chest-of-drawers"
    endNonFunctionalProperties

concept dresser subConceptOf chest
    nonFunctionalProperties
        dc:description hasValue "a low long chest of drawers used for storing clothes"
    endNonFunctionalProperties

/**
 * SHELF hierarchy
 */
concept shelf subConceptOf storageFurniture
    nonFunctionalProperties
        dc:description hasValue "a long seat with back and arms"
    endNonFunctionalProperties

concept bookshelf subConceptOf shelf
    nonFunctionalProperties
        dc:description hasValue "an open shelf for holding books"
    endNonFunctionalProperties
    shelvsNumber ofType xsd:integer
    isFlexible ofType xsd:boolean

/*****************************************************************************************
 * SEAT FURNITURE hierarchy
 *****************************************************************************************/
concept seat subConceptOf furniture
    nonFunctionalProperties
        dc:description hasValue "a seating accommodation"
    endNonFunctionalProperties

/**
 * CHAIR hierarchy
 */
concept chair subConceptOf seat
    nonFunctionalProperties
        dc:description hasValue "chair as a subclass of furniture"
    endNonFunctionalProperties
    numberOfLegs ofType xsd:integer
```

```
concept armChair subConceptOf chair
    nonFunctionalProperties
        dc:description hasValue "a chair with side structures to support the arms or elbows"
    endNonFunctionalProperties

concept fauteuil subConceptOf armChair
    nonFunctionalProperties
        dc:description hasValue "a French term for an armchair"
    endNonFunctionalProperties

concept gainsboroughChair subConceptOf armChair
    nonFunctionalProperties
        dc:description hasValue "deep armchair with an upholstered seat and back, padded open arms, and carved
decoration"
    endNonFunctionalProperties

concept seatRail subConceptOf chair
    nonFunctionalProperties
        dc:description hasValue "framework that supports the seat of a chair and holds the legs together"
    endNonFunctionalProperties

concept sideChair subConceptOf chair
    nonFunctionalProperties
        dc:description hasValue "a chair without arms, designed to stand against a wall"
    endNonFunctionalProperties

concept wingChair subConceptOf chair
    nonFunctionalProperties
        dc:description hasValue "a fully upholstered chair with wings at the sides to protect the sitter from drafts; also
known as a 'wing-back'"
    endNonFunctionalProperties

concept chaislounge subConceptOf chair //, sofa
    nonFunctionalProperties
        dc:description hasValue "long reclining chair/sofa"
    endNonFunctionalProperties

concept stool subConceptOf chair
    nonFunctionalProperties
        dc:description hasValue "a chair without back and arms, supproted by three or four legsor by a central
pedestral"
    endNonFunctionalProperties

/**
 * SOFA hierarchy
 */
concept sofa subConceptOf seat
    nonFunctionalProperties
        dc:description hasValue "a long seat with back and arms"
    endNonFunctionalProperties

concept clubSofa subConceptOf sofa
    nonFunctionalProperties
        dc:description hasValue "an upholstered piece of furniture whose arms are lower than its level back"
    endNonFunctionalProperties

concept sectional subConceptOf sofa
    nonFunctionalProperties
        dc:description hasValue "a sofa that has several segments, which may be used in combination or separately to
fit in a room"
    endNonFunctionalProperties

concept camelbackSofa subConceptOf sofa
    nonFunctionalProperties
        dc:description hasValue "sofa style characterized by a large symetrical rise or 'hump' located on the back"
    endNonFunctionalProperties

concept sofaBed subConceptOf sofa //, bed
    nonFunctionalProperties
        dc:description hasValue "a sofa that can be used as bed"
    endNonFunctionalProperties

/*************************************************************************************************
 * TABLE hierarchy
*************************************************************************************************/
```

```
concept table subConceptOf furniture
    nonFunctionalProperties
        dc:description hasValue "table as a subclass of furniture"
    endNonFunctionalProperties
    numberOfLegs ofType xsd:integer

concept cocktailTable subConceptOf table
    nonFunctionalProperties
        dc:description hasValue "a table positioned in front of the major seating units which provides a surface for
serving"
    endNonFunctionalProperties

concept console subConceptOf table
    nonFunctionalProperties
        dc:description hasValue "a table intended to stand against a wall, between windows"
    endNonFunctionalProperties

concept credenceTable subConceptOf table
    nonFunctionalProperties
        dc:description hasValue "a type of small table used for storing food before serving; generally a semi-circular
table with a hinged top"
    endNonFunctionalProperties

/**
 * TABLE - DESK hierarchy
 */
concept desk subConceptOf table
    nonFunctionalProperties
        dc:description hasValue "a table used for working"
    endNonFunctionalProperties
    /**
     * todo specify the realtion between desk and homeOffice
     */

concept bureau subConceptOf desk
    nonFunctionalProperties
        dc:description hasValue "a writing desk with a fall or cylinder front, enclosing a fitted interior, with drawers
below"
    endNonFunctionalProperties
    /**
     * todo specify the realtion between bureau and homeOffice
     */

concept pedestalDesk subConceptOf desk
    nonFunctionalProperties
        dc:description hasValue "a flat desk, usually with a leather top, that stands on two banks of drawers"
    endNonFunctionalProperties

/**
 * TABLE hierarchy (cont.)
 */
concept harvestTable subConceptOf table
    nonFunctionalProperties
        dc:description hasValue "a narrow rectangular table with hinged drop leaves"
    endNonFunctionalProperties

concept huntboard subConceptOf table
    nonFunctionalProperties
        dc:description hasValue "a light, portable sideboard used for serving food and drinks"
    endNonFunctionalProperties

concept pedestalTable subConceptOf table
    nonFunctionalProperties
        dc:description hasValue "a set of occasional tables that slide one beneath the other when not in use"
    endNonFunctionalProperties
    /**
     * todo specify that has allways ONE leg
     */

concept pembrokeTable subConceptOf table
    nonFunctionalProperties
        dc:description hasValue "a small two-flap table that stands on four legs"
    endNonFunctionalProperties
    /**
     * todo specify that has allways FOUR legs
     */
```

```
concept sofaTable subConceptOf table
    nonFunctionalProperties
        dc:description hasValue "a rectangular table with two hinged flaps at the ends designed to stand in front of a
sofa"
    endNonFunctionalProperties
    /**
     * todo specify the relation between this kind of table and soda concept
     */

concept teapoy subConceptOf table
    nonFunctionalProperties
        dc:description hasValue "a small piece of freestanding furniture designed for holding tea"
    endNonFunctionalProperties

concept tripodTable subConceptOf table
    nonFunctionalProperties
        dc:description hasValue "a small table with a round top supported by a three-legged pillar, originally made for
serving tea"
    endNonFunctionalProperties
    /**
     * todo specify that has allways THREE legs
     */

concept gatelegTable subConceptOf table
    nonFunctionalProperties
        dc:description hasValue "table with drop leaves and a movable leg, which can be swung into position to hold
the leaf upright or retracted to allow the leaf to remain down"
    endNonFunctionalProperties


/*************************************************************************************************
 * Accessories Ontology
*************************************************************************************************/
concept accessories
    nonFunctionalProperties
        dc:description hasValue "general class of accesories"
    endNonFunctionalProperties
    manufacturer ofType manufacturer
    material ofType set material
    color ofType xsd:string

/**
 * hygieneAccessories
 */
concept hygieneAccessories subConceptOf accessories
    nonFunctionalProperties
        dc:description hasValue "hygiene accesories"
    endNonFunctionalProperties

concept washingAccessories subConceptOf hygieneAccessories
    nonFunctionalProperties
        dc:description hasValue "washingAccesories subclass of hygieneAccessories"
    endNonFunctionalProperties
    height ofType xsd:integer
    hasFilter ofType xsd:boolean

concept shower subConceptOf washingAccessories
    nonFunctionalProperties
        dc:description hasValue "shower subclass of washingAccesories"
    endNonFunctionalProperties
    height ofType xsd:integer
    hasFilter ofType xsd:boolean

concept bathtube subConceptOf washingAccessories
    nonFunctionalProperties
        dc:description hasValue "bathtube subclass of washingAccesories"
    endNonFunctionalProperties
    capacity ofType xsd:float

concept jacuzii subConceptOf bathtube
    nonFunctionalProperties
        dc:description hasValue "jacuzzi subclass of bathtube"
    endNonFunctionalProperties
    hasSteam ofType xsd:boolean
```

```
concept toilet subConceptOf hygieneAccessories
    nonFunctionalProperties
        dc:description hasValue "toilet subclass of washingAccesories"
    endNonFunctionalProperties

concept washingMachine subConceptOf hygieneAccessories
    nonFunctionalProperties
        dc:description hasValue "washingMachine subclass of washingAccesories"
    endNonFunctionalProperties

concept dryer subConceptOf hygieneAccessories
    nonFunctionalProperties
        dc:description hasValue "dryer subclass of washingAccesories"
    endNonFunctionalProperties

/**
 * cookingAccessories
 */
concept cookingAccessories subConceptOf accessories
    nonFunctionalProperties
        dc:description hasValue "cooking accesories"
    endNonFunctionalProperties

concept kitchenSink subConceptOf cookingAccessories
    nonFunctionalProperties
        dc:description hasValue "kitchenSink subclass of cookingAccessories"
    endNonFunctionalProperties
    numberOfBasins ofType xsd:integer

concept oven subConceptOf cookingAccessories
    nonFunctionalProperties
        dc:description hasValue "oven subclass of cookingAccessories"
    endNonFunctionalProperties
    energySource ofType energysource //electric, gas, dual
    maxTemperature ofType xsd:float
    /*
     * todo put an restriction that the temperature can not be less that 0 or more that 300 C
     */
    hasSelfCleaning ofType xsd:boolean

concept microwaveOven subConceptOf oven
    nonFunctionalProperties
        dc:description hasValue "microwaveoven subclass of oven"
    endNonFunctionalProperties

concept dishwasher subConceptOf cookingAccessories
    nonFunctionalProperties
        dc:description hasValue "dishwasher subclass of cookingAccessories"
    endNonFunctionalProperties

concept mixer subConceptOf cookingAccessories
    nonFunctionalProperties
        dc:description hasValue "mixer subclass of cookingAccessories"
    endNonFunctionalProperties

concept toaster subConceptOf cookingAccessories
    nonFunctionalProperties
        dc:description hasValue "toaster subclass of cookingAccessories"
    endNonFunctionalProperties

concept cookingRecipient subConceptOf cookingAccessories
    nonFunctionalProperties
        dc:description hasValue "cookingRecipient subclass of cookingAccessories"
    endNonFunctionalProperties
    volum ofType xsd:float

concept pot subConceptOf cookingRecipient
    nonFunctionalProperties
        dc:description hasValue "pot subclass of cookingRecipient"
    endNonFunctionalProperties

concept pan subConceptOf cookingRecipient
    nonFunctionalProperties
        dc:description hasValue "pan subclass of cookingRecipient"
    endNonFunctionalProperties
```

```
/**
 * lightingAccessories
 */
concept lightingAccessories subConceptOf accessories
    nonFunctionalProperties
        dc:description hasValue "lighting accesories"
    endNonFunctionalProperties

concept lamp subConceptOf lightingAccessories
    nonFunctionalProperties
        dc:description hasValue "lamp subclass of lightingAccessories"
    endNonFunctionalProperties
    power ofType xsd:integer                    //measure the consum (watt)
    brightness ofType xsd:integer          //measure the brightness (lumen)

concept floorLamp subConceptOf lamp
    nonFunctionalProperties
        dc:description hasValue "foorLamp subclass of lamp"
    endNonFunctionalProperties

concept ceilingLamp subConceptOf lamp
    nonFunctionalProperties
        dc:description hasValue "ceilingLamp subclass of lamp"
    endNonFunctionalProperties

concept tableLamp subConceptOf lamp
    nonFunctionalProperties
        dc:description hasValue "tableLamp subclass of lamp"
    endNonFunctionalProperties

concept wallLamp subConceptOf lamp
    nonFunctionalProperties
        dc:description hasValue "wallLamp subclass of lamp"
    endNonFunctionalProperties


/**
 * communicationAccessories
 */
concept communicationAccessories subConceptOf accessories
    nonFunctionalProperties
        dc:description hasValue "communication accesories"
    endNonFunctionalProperties

concept phone subConceptOf communicationAccessories
    nonFunctionalProperties
        dc:description hasValue "phone subclass of communicationAccessories"
    endNonFunctionalProperties
    number ofType xsd:string

concept mobilephone subConceptOf phone
    nonFunctionalProperties
        dc:description hasValue "mobilephone subclass of phone"
    endNonFunctionalProperties

concept fax subConceptOf communicationAccessories
    nonFunctionalProperties
        dc:description hasValue "fax subclass of communicationAccessories"
    endNonFunctionalProperties
    number ofType xsd:string

/**
 * entertainmentAccessories
 */
concept entertainmentAccessories subConceptOf accessories
    nonFunctionalProperties
        dc:description hasValue "entertainment accesories"
    endNonFunctionalProperties

concept televisor subConceptOf entertainmentAccessories
    nonFunctionalProperties
        dc:description hasValue "televisor subclass of     accesories"
    endNonFunctionalProperties
    isColor ofType xsd:boolean
    diagonal ofType xsd:float
```

```
concept radio subConceptOf entertainmentAccessories
    nonFunctionalProperties
        dc:description hasValue "radio subclass of     accesories"
    endNonFunctionalProperties

/*****************************************************************************************
 * realtions definitions
*****************************************************************************************/

/*****************************************************************************************
 * R2F - room2furniture relations
*****************************************************************************************/
relation room2furniture
    nonFunctionalProperties
        dc:description hasValue "general relation between a room and the furniture that might be in the room"
    endNonFunctionalProperties
    paramRoom ofType room
    paramFurniture ofType furniture

/**
 * bathroom2furniture
 */
relation bathroom2furniture
    nonFunctionalProperties
        dc:description hasValue "relation between bathroom and the furniture that might be in the bathroom"
    endNonFunctionalProperties
    paramBathroom ofType bathroom
    paramFurniture ofType bathroomFurniture

concept bathroomFurniture
    nonFunctionalProperties
        dc:description hasValue "set of furniture that can be found in the bathroom"
    endNonFunctionalProperties
    shelf ofType shelf
    cabinet ofType cabinet
    stool ofType stool

/**
 * bedroom2furniture
 */
relation bedroom2furniture
    nonFunctionalProperties
        dc:description hasValue "relation between bedroom and the furniture that might be in the bedroom"
    endNonFunctionalProperties
    paramBedroom ofType bedroom
    paramFurniture ofType bedroomFurniture

concept bedroomFurniture
    nonFunctionalProperties
        dc:description hasValue "set of furniture that can be found in the bedroom"
    endNonFunctionalProperties
    shelf ofType shelf
    bed ofType bed

/**
 * dinigroom2furniture
 */
relation dinigroom2furniture
    nonFunctionalProperties
        dc:description hasValue "relation between dinigroom and the furniture that might be in the dinigroom"
    endNonFunctionalProperties
    paramDinigroom ofType dinigroom
    paramFurniture ofType diningroomFurniture

concept diningroomFurniture
    nonFunctionalProperties
        dc:description hasValue "set of furniture that can be found in the diningroom"
    endNonFunctionalProperties
    table ofType table
    chair ofType chair

/**
 * livingroom2furniture
 */
relation livingroom2furniture
    nonFunctionalProperties
```

```
            dc:description hasValue "relation between livingroom and the furniture that might be in the livingroom"
        endNonFunctionalProperties
        paramLivingroom ofType livingroom
        paramFurniture ofType livingroomFurniture

    concept livingroomFurniture
        nonFunctionalProperties
            dc:description hasValue "set of furniture that can be found in the livingroom"
        endNonFunctionalProperties
        table ofType table
        sofa ofType sofa

    /**
     * homeOffice2furniture
     */
    relation homeOffice2furniture
        nonFunctionalProperties
            dc:description hasValue "relation between homeOffice and the furniture that might be in the homeOffice"
        endNonFunctionalProperties
        paramHomeOffice ofType homeOffice
        paramFurniture ofType homeOfficeFurniture

    concept homeOfficeFurniture
        nonFunctionalProperties
            dc:description hasValue "set of furniture that can be found in the home office"
        endNonFunctionalProperties
        desk ofType desk
        chair ofType chair

    /**
     * kitchen2furniture
     */
    relation kitchen2furniture
        nonFunctionalProperties
            dc:description hasValue "relation between kitchen and the furniture that might be in the kitchen"
        endNonFunctionalProperties
        paramKitchen ofType kitchen
        paramFurniture ofType kitchenFurniture

    concept kitchenFurniture
        nonFunctionalProperties
            dc:description hasValue "set of furniture that can be found in the kitchen"
        endNonFunctionalProperties
        table ofType table
        shelf ofType shelf

    /*************************************************************************************************
     * R2A - room2accessories relations
    *************************************************************************************************/
    relation room2accessories
        nonFunctionalProperties
            dc:description hasValue "general relation between a room and the accessories that might be in the room"
        endNonFunctionalProperties
        paramRoom ofType room
        paramAccessories ofType accessories

    /**
     * bathroom2accessories
     */
    relation bathroom2accessories
        nonFunctionalProperties
            dc:description hasValue "relation between bathroom and the accessories that might be in the bathroom"
        endNonFunctionalProperties
        paramBathroom ofType bathroom
        paramAccessories ofType bathroomAccessories

    concept bathroomAccessories
        nonFunctionalProperties
            dc:description hasValue "set of accessories that can be found in the bathroom"
        endNonFunctionalProperties
        toilet ofType toliet
        washingAccessories ofType washingAccessories
        lightingAccessories ofType lightingAccessories

    /**
     * bedroom2accessories
```

```
 */
relation bedroom2accessories
    nonFunctionalProperties
        dc:description hasValue "relation between bedroom and the accessories that might be in the bedroom"
    endNonFunctionalProperties
    paramBedroom ofType bedroom
    paramAccessories ofType bedroomAccessories

concept bedroomAccessories
    nonFunctionalProperties
        dc:description hasValue "set of accessories that can be found in the bedroom"
    endNonFunctionalProperties
    lightingAccessories ofType lightingAccessories

/**
 * dinigroom2accessories
 */
relation dinigroom2accessories
    nonFunctionalProperties
        dc:description hasValue "relation between dinigroom and the accessories that might be in the dinigroom"
    endNonFunctionalProperties
    paramDinigroom ofType dinigroom
    paramAccessories ofType diningroomAccessories

concept diningroomAccessories
    nonFunctionalProperties
        dc:description hasValue "set of accessories that can be found in the diningroom"
    endNonFunctionalProperties
    lightingAccessories ofType lightingAccessories

/**
 * livingroom2Accessories
 */
relation livingroom2accessories
    nonFunctionalProperties
        dc:description hasValue "relation between livingroom and the accessories that might be in the livingroom"
    endNonFunctionalProperties
    paramLivingroom ofType livingroom
    paramAccessories ofType livingroomAccessories

concept livingroomAccessories
    nonFunctionalProperties
        dc:description hasValue "set of accessories that can be found in the livingroom"
    endNonFunctionalProperties
    lightingAccessories ofType lightingAccessories
    entertainmentAccessories ofType entertainmentAccessories

/**
 * homeOffice2accessories
 */
relation homeOffice2accessories
    nonFunctionalProperties
        dc:description hasValue "relation between homeOffice and the accessories that might be in the homeOffice"
    endNonFunctionalProperties
    paramHomeOffice ofType homeOffice
    paramAccessories ofType homeOfficeAccessories

concept homeOfficeAccessories
    nonFunctionalProperties
        dc:description hasValue "set of accessories that can be found in the home office"
    endNonFunctionalProperties
    lamp ofType lamp
    communicationAccessories ofType communicationAccessories

/**
 * kitchen2accessories
 */
relation kitchen2accessories
    nonFunctionalProperties
        dc:description hasValue "relation between kitchen and the accessories that might be in the kitchen"
    endNonFunctionalProperties
    paramKitchen ofType kitchen
    paramAccessories ofType kitchenAccessories

concept kitchenAccessories
    nonFunctionalProperties
```

```
        dc:description hasValue "set of accessories that can be found in the kitchen"
    endNonFunctionalProperties
    cookingAccessories ofType cookingAccessories
```

The "Marketplace Ontology" defines all aspects products, purchasing contracts, buyers and seller, payment, and delivery in the virtual SWF marketplace. This ontology is partially based on the Purchase Ontology developed in the WSMO Use Case, which is a generic ontology that specifies aspects around purchasing with regard to ebXML. However, this generic Purchase Ontology is far too complex for the purpose of our use case on the one hand, and, on the other, it misses certain aspects needed with the SWF markeplace. The Marketplace therefore restricts the Purchase Ontology [in fact, this should be done by an OO Mediator - to be defined later, when there is a specification of how to define OO Mediators in WSMO], and extends it by additional aspects needed in within the SWF Marketplace.

Listing. Marketplace Ontology (WSML file)

```
/**
 * SWF Marketplace Ontology
 */

namespace <<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    furn:<<http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    loc:<<http://swf.quarto.at/repository/ontologies/location.wsml#>>
    xsd:<<http://www.w3.org/2001/XMLSchema#>>


ontology <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>


nonFunctionalProperties
    dc:title hasValue "SWF Marketplace Ontology"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {"Marketplace Participants", "Buyer", "Seller", "Product", "Purchase", "Payment", "Delivery"}
    dc:description hasValue "describes notions concerned with purchasing in the SWF Marketplace"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma"
    dc:date hasValue "2004-09-20"
    dc:type hasValue <<http://www.wsmo.org/2004/d2/v0.3/20040329/#ontos>>
    /**
     * ontologies are modeled as WSMO Ontologies
     */
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://www.wsmo.org/2004/d3/d3.2/v0.1/20040719/resources/purchase>>,
        <<http://www.daml.ecs.soton.ac.uk/ont/currency.daml>>}
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.7 $"
endNonFunctionalProperties


importedOntologies {
    <<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://swf.quarto.at/repository/ontologies/location.wsml>>}

/**
 * conceptDefinitions
*/

/**
 * Marketplace Participants
 * defines buyers and sellers with their contact / shipping / billing addresses
*/

concept marketplaceParticipant
```

```
        nonFunctionalProperties
            dc:description hasValue "superconcept for all marketplace participants"
        endNonFunctionalProperties
        marketplaceParticipantID ofType xsd:string
        name ofType xsd:string
        email ofType xsd:string
        telephone ofType xsd:string
        fax ofType xsd:string

    concept buyer subConceptOf marketplaceParticipant
        nonFunctionalProperties
            dc:description hasValue "marketplace participant that wants to buy"
        endNonFunctionalProperties
        billToAddress ofType loc:address
        shipToAddress ofType loc:address
        hasPaymentMethod ofType set paymentMethod

    concept seller subConceptOf marketplaceParticipant
        nonFunctionalProperties
            dc:description hasValue "marketplace participant that wants to sell"
        endNonFunctionalProperties
        contactAddress ofType loc:address
        acceptsPaymentMethod ofType set paymentMethod

    concept company subConceptOf seller
        nonFunctionalProperties
            dc:description hasValue "comcercial selling store"
        endNonFunctionalProperties
        companyNumber ofType xsd:string
        website ofType xsd:string

    concept privateSeller subConceptOf seller
        nonFunctionalProperties
            dc:description hasValue "private seller"
        endNonFunctionalProperties


    /**
     * Product
     * defines properties of products in the marketplace
     * for this use case, products are piece of furniture
    */

    concept product
        nonFunctionalProperties
            dc:description hasValue "product available for purchase (is a piece of furniture)"
        endNonFunctionalProperties
        item ofType furn:furniture
        price ofType price
        provider ofType seller

    concept price
        nonFunctionalProperties
            dc:description hasValue "specifies monetary amount and currency of the price of a product.
             A price includes taxes. "
        endNonFunctionalProperties
        amount ofType xsd:float
        currency ofType currency

    concept currency
        nonFunctionalProperties
            dc:description hasValue "currency definition in accordance to
             http://www.daml.ecs.soton.ac.uk/ont/currency.daml
             when there is an OO Mediator existent, this will be used"
        endNonFunctionalProperties
        name ofType xsd:string
        code ofType xsd:string

    /**
     * Purchase Contract
     * defines properties of a contract of purchase for a product between a buyer and a seller
    */

    concept purchaseContract
        nonFunctionalProperties
            dc:description hasValue "defines a contract of purchase for a product between a buyer and a seller.
```

```
                Establishing purchase contracts is the main functionality of the marketplace; a purchase contract
                is restricted to one seller and one buyer for one product."
        endNonFunctionalProperties
        purchaseContractID ofType xsd:integer
        purchaseItem ofType product
        buyer ofType buyer
        seller ofType seller
        purchasePayment ofType paymentMethod
        contractDate ofType dt:date


    /**
     * Payment
     * defines different types of payment methods supported in the marketplace
    */

    concept paymentMethod
        nonFunctionalProperties
            dc:description hasValue "superconcept of payment methods"
        endNonFunctionalProperties

    concept creditCard subConceptOf paymentMethod
        nonFunctionalProperties
            dc:description hasValue "payment method credit card"
        endNonFunctionalProperties
        type ofType creditCardType
        number ofType xsd:string
        holder ofType xsd:string
        expMonth ofType dt:monthOfYear
        expYear ofType dt:year

    concept invoice subConceptOf paymentMethod
        nonFunctionalProperties
            dc:description hasValue "payment method invoice"
        endNonFunctionalProperties
        invoiceNumber ofType xsd:string
        sellerAccount ofType account
        dueDate ofType dt:date

    concept check subConceptOf paymentMethod
        nonFunctionalProperties
            dc:description hasValue "payment method check"
        endNonFunctionalProperties
        checkNumber ofType xsd:integer
        drawer ofType buyer
        receiver ofType seller
        drawerAccount ofType account

    concept cash subConceptOf paymentMethod
        nonFunctionalProperties
            dc:description hasValue "payment method cash"
        endNonFunctionalProperties
        payer ofType buyer
        receiver ofType seller

    // realted concepts

    concept creditCardType
        nonFunctionalProperties
            dc:description hasValue "specifies type of credit card"
        endNonFunctionalProperties

    concept account
        nonFunctionalProperties
            dc:description hasValue "specifies a bank account"
        endNonFunctionalProperties
        bank ofType bank
        owner ofType xsd:string
        accountNumber ofType xsd:integer

    concept bank
        nonFunctionalProperties
            dc:description hasValue "specifies a bank (financial institute)"
        endNonFunctionalProperties
        bankName ofType xsd:string
        bankAddress ofType loc:address
```

```
    bankIdentifierCode ofType xsd:string


/**
 * Delivery
 * defines different types of delivery methods supported in the marketplace
*/

concept delivery
    nonFunctionalProperties
        dc:description hasValue "superconcept of delivery methods.
            delivery is a main function of the marketplace in addition to purchase contracting."
    endNonFunctionalProperties
    deliveryItem ofType product

concept dropShip subConceptOf delivery
    nonFunctionalProperties
        dc:description hasValue "delivery directly to the buyer shipping address by a delivery service"
    endNonFunctionalProperties
    sender ofType seller
    receiver ofType buyer
    carrier ofType dropShipCarrier

concept selfCollection subConceptOf delivery
    nonFunctionalProperties
        dc:description hasValue "buyer collects purchased "
    endNonFunctionalProperties

// additional concepts

concept dropShipCarrier
    nonFunctionalProperties
        dc:description hasValue "a company that provides a drop ship delivery service"
    endNonFunctionalProperties
    name ofType xsd:string
    companyNumber ofType xsd:string
    contactaddress ofType loc:address
    transportBy ofType transportationMean
    deliveryCoverage ofType loc:location

concept transportationMean
    nonFunctionalProperties
        dc:description hasValue "mean of transportation used by a drop ship delivery service"
    endNonFunctionalProperties
```

The Location Ontology specified within the WSMO Use is also a general Ontology for describing locations. We use this ontology in our use case without any changes. Just for completeness of the use cas emodels and for ease of use, the following Listing shows the ontology.

Listing. Location Ontology (WSML file)

```
/**
 * Location Ontology
 */

namespace <<http://swf.quarto.at/repository/location.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    xsd:<<http://www.w3.org/2001/XMLSchema#>>
    wsmoloc:<<http://www.wsmo.org/2004/d3/d3.2/v0.1/20040628/resources/loc.wsml#>>

ontology <<http://swf.quarto.at/repository/location.wsml>>

nonFunctionalProperties
    dc:title hasValue "Location"
    dc:creator hasValue "DERI International"
    dc:subject hasValues {"Location", "Country", "State", "City", "Address"}
    dc:description hasValue "describes notions of location and postal addresses with special focus on Austria"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Ruben Lara, Axel Polleres, Michael Stollberg, Ioan Toma"
```

```
        dc:date hasValue "2004-09-20"
        dc:type hasValue <<http://www.wsmo.org/2004/d2/v0.3/20040329/#ontos>>
        /**
         * ontologies are modeled as WSMO Ontologies
         */
        dc:format hasValue "text/html"
        dc:language hasValue "en-US"
        dc:relation hasValues {<<http://www.daml.org/2001/09/countries/fips-10-4-ont>>,
                <<http://www.daml.org/2001/09/countries/iso-3166-ont>>,
                <<http://www.daml.org/2003/09/factbook/factbook-ont>>,
                <<http://www.daml.org/2001/02/geofile/geofile-ont>>,
                <<http://daml.umbc.edu/ontologies/ittalks/address>>,
                <<http://www.daml.ecs.soton.ac.uk/ont/currency.daml>>}
        dc:coverage hasValue "general, specfically Austria"
        dc:rights hasValue <<http://www.deri.org/privacy.html>>
        version hasValue "$Revision: 1.6 $"
    endNonFunctionalProperties

    importedOntologies {<<http://www.wsmo.org/2004/d3/d3.2/v0.1/20040719/resources/loc.wsml>>}

    /**
     * CONCEPT DEFINITIONS
     * NOTE: in the following, we provide the the subset of ontological notions defined in the
     * Location Ontology defined in the WSMO Use Case that will be used for modelling notions of
     * location and address within the SWF USe Case
    */

        concept location
            nonFunctionalProperties
                dc:description hasValue "superconcept for all notions of spatial locations"
                dc:realtion hasValues {<<wsmoloc:location>>}
            endNonFunctionalProperties

        concept geographicalLocation subConceptOf location
            nonFunctionalProperties
                dc:description hasValue "general notion of a geographical location"
                dc:realtion hasValues {<<wsmoloc:geographicalLocation>>}
            endNonFunctionalProperties
            name ofType xsd:string
            longitude ofType wsmoloc:longitude
                comment: defined in http://www.daml.org/2001/02/geofile/geofile-dt.xsd#longitude
            latitude ofType wsmoloc:latitude
                comment: defined in http://www.daml.org/2001/02/geofile/geofile-dt.xsd#latitude

        concept continent subConceptOf geographicalLocation

        concept country subConceptOf geographicalLocation
            nonFunctionalProperties
                dc:description hasValue "describes a country"
                dc:relation hasValues {<<wsmoloc:country>>}
            endNonFunctionalProperties
            inContinent ofType continent
            population ofType xsd:integer
            extension ofType xsd:integer
            isoCode ofType xsd:string
                comment: this specifies the ISO 3166 Country Code

        concept state subConceptOf geographicalLocation
            nonFunctionalProperties
                dc:description hasValue "a state, especially to describe countries in Austria"
                dc:relation hasValues {<<wsmoloc:state>>}
            endNonFunctionalProperties
            inCountry ofType country
            population ofType xsd:integer
            extension ofType xsd:integer

        concept city subConceptOf geographicalLocation
            nonFunctionalProperties
                dc:description hasValue "City"
            endNonFunctionalProperties
            inCountry ofType contry
            inState ofType state

        concept address subConceptOf ad:address
            nonFunctionalProperties
                dc:description hasValue "Extended address, adding more details to
```

```
            city, state and country"
        endNonFunctionalProperties
        street ofType xsd:string
        number ofType xsd:string
        city ofType city
        zip ofType xsd:integer
```

3.2 Goal Templates

The following specfies pre-defined Goal Templates for buyers and sellers as described above.

3.2.1 Buyer Goal Templates

Listing. Buyer Goal Template 1 (WSML file)

```
/**
 * Buyer Goal Template 1
 */

namespace <<http://swf.quarto.at/repository/GoalTemplates/BuyFurnitureCreditCardGoal.1.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    furn:<<http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    loc:<<http://swf.quarto.at/repository/ontologies/location.wsml#>>

goal <<http://swf.quarto.at/repository/GoalTemplates/BuyFurnitureCreditCardGoal.1.wsml>>

nonFunctionalProperties
    dc:title hasValue "Buyer Goal Template 1"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:buyer, swfmo:purchaseContract, furn:furniture, swfmo:dropShip, swfmo:creditCard}
    dc:description hasValue "Goal Template for buying one piece of furniture"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-20"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
            <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
            <<http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.1 $"
endNonFunctionalProperties


importedOntologies {
    <<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
    <<http://swf.quarto.at/repository/ontologies/location.wsml>>}

comment: no mediators are used


postcondition
 axiom bgs1Postcondition
    nonFunctionalProperties
        dc:description hasValue "a contract of purchase for a buyer for a specific piece of furniture, payment by
creditcard"
    endNonFunctionalProperties
    definedBy
        exists ?PCID(?PCID memberOf xsd:integer) and
        exists ?PurchaseItem(?PurchaseItem[
```

```
            item hasValue ?PurchaseFurniture
        ] memberOf swfmo:product) and
        exists ?PurchaseFurniture(?PurchaseFurniture memberOf furn:furniture) and
        exists ?Buyer(?Buyer[
                billToAddress hasValue ?Address,
                shipToAddress hasValue ?Address,
                hasPaymentMethod hasValues {?CreditCard}
        ] memberOf swfmo:buyer) and
        exists ?Address(?Address[
                city.inCountry hasValue austria
        ] memberOf loc:address) and
        exists ?CreditCard(?CreditCard memberOf swfmo:creditCard) and
    ?X[
            purchaseContractID hasValue ?PCID,
            purchaseItem hasValue ?PurchaseItem,
            buyer hasValue ?Buyer,
            purchasePayment hasValue ?CreditCard
        ] memberOf swfmo:purchaseContract .


effect
 axiom bgs1Effect
    nonFunctionalProperties
        dc:description hasValue "direct delivery of the purchased piece of furniture to a specific shipping address in
Austria"
    endNonFunctionalProperties
    definedBy
        exists ?Item(?Item memberOf swfmo:product) and
        exists ?Buyer(?Buyer[
             shipToAddress hasValue ?Address
        ] memberOf swfmo:buyer) and
        exists ?Address(?Address[
            city.inCountry hasValue austria
        ] memberOf loc:address) and
        ?X[
             deliveryItem hasValue ?Item,
             receiver hasValue ?Buyer
        ] memberOf swfmo:dropShip .
```

Listing. Buyer Goal Template 2 (WSML file)

```
/**
 * Buyer Goal Template 2
 */

namespace <<http://swf.quarto.at/repository/GoalTemplates/BuyFurnitureAnyPaymentGoal.1.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    furn:<<http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    loc:<<http://swf.quarto.at/repository/ontologies/location.wsml#>>

goal <<http://swf.quarto.at/repository/GoalTemplates/BuyFurnitureAnyPaymentGoal.1.wsml>>

nonFunctionalProperties
    dc:title hasValue "Buyer Goal Template 2"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:buyer, swfmo:purchaseContract, furn:furniture, swfmo:dropShip}
    dc:description hasValue "buying a single piece of furniture with any payment method"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-20"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {
            <<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
            <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
            <<http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.2 $"
endNonFunctionalProperties
```

```
importedOntologies {
    <<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
    <<http://swf.quarto.at/repository/ontologies/location.wsml>>}

comment: no mediators are used


postcondition
 axiom bgs2Postcondition
    nonFunctionalProperties
        dc:description hasValue "a contract of purchase for a specific piece of furniture, payment by any payment
method"
    endNonFunctionalProperties
    definedBy
        exists ?PCID(?PCID memberOf xsd:integer) and
        exists ?PurchaseItem(?PurchaseItem[
            item hasValue ?PurchaseFurniture,
            provider hasValue ?Seller
        ] memberOf swfmo:product) and
        exists ?PurchaseFurniture(?PurchaseFurniture memberOf furn:furniture) and
        exists ?Buyer(?Buyer[
                billToAddress hasValue ?Address,
                shipToAddress hasValue ?Address,
                hasPaymentMethod hasValues {?Payment}
        ] memberOf swfmo:buyer) and
        exists ?Address(?Address[
                city.inCountry hasValue austria
        ] memberOf loc:address) and
        exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
     ?X[
            purchaseContractID hasValue ?PCID,
            purchaseItem hasValue ?PurchaseItem,
            buyer hasValue ?Buyer,
            purchasePayment hasValue ?Payment
        ] memberOf swfmo:purchaseContract .


effect
 axiom bgs2Effect
    nonFunctionalProperties
        dc:description hasValue "direct delivery of the purchased piece of furniture to a specific
            buyer shipping address in Austria, or self collection by the buyer"
    endNonFunctionalProperties
    definedBy
        exists ?Item(?Item memberOf swfmo:product) and
        exists ?Buyer(Buyer[
            shipToAddress hasValue ?BuyerAddress
        ] memberOf swfmo:buyer) and
        exists ?BuyerAddress(?BuyerAddress[
            city.inCountry hasValue austria
        ] memberOf loc:address) and
        exists ?Carrier(?Carrier[
            deliverCoverage hasValue austria
        ] memberOf swfmo:dropShipCarrier) and
    (?X[
             deliveryItem hasValue ?Item,
             receiver hasValue ?Buyer,
             carrier hasValue ?Carrier
             ] memberOf swfmo:dropShip
     )
     or
     (?X[
             deliveryItem hasValue ?Item
     ] memberOf swfmo:selfCollection) .
```

Listing. Buyer Goal Template 3 (WSML file)

```
/**
 * Buyer Goal Template 3
 */

namespace <<http://swf.quarto.at/repository/GoalTemplates/QueryFurnitureRequestGoal.1.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
```

```
    furn:<<http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>

goal <<http://swf.quarto.at/repository/GoalTemplates/QueryFurnitureRequestGoal.1.wsml>>

nonFunctionalProperties
    dc:title hasValue "Buyer Goal Template 3"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:buyer, swfmo:product, furn:furniture}
    dc:description hasValue "get detailed product information for a specific piece of furniture without purchasing"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-20"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {
            <<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
            <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
            <<http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.2 $"
endNonFunctionalProperties


importedOntologies{
    <<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>}

comment: no mediators are used


postcondition
 axiom bgs3Postcondition
    nonFunctionalProperties
        dc:description hasValue "product information for a specific piece of furniture
                    provided by a specific seller or seller group"
    endNonFunctionalProperties
    definedBy
        exists ?FurnitureSearched(?FurnitureSearched memberOf furn:furniture) and
        exists ?Provider(?Provider memberOf swfmo:seller) and
        ?X[
            item hasValue ?FurnitureSearched,
            provider hasValue ?Provider
         ] memberOf swfmo:product .


effect
 axiom bgs3Effect
    nonFunctionalProperties
        dc:description hasValue "no effect"
    endNonFunctionalProperties
    definedBy
     ?X .
comment: this means there is no effect - however this is modelled in the end.
```

Listing. Buyer Goal Template 4 (WSML file)

```
/**
 * Buyer Goal Template 4
 */

namespace
<<http://swf.quarto.at/repository/GoalTemplates/BuyFurniturePrivateGoal.1.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    furn:<<http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    loc:<<http://swf.quarto.at/repository/ontologies/location.wsml#>>

goal <<http://swf.quarto.at/repository/GoalTemplates/BuyFurniturePrivateGoal.1.wsml>>

nonFunctionalProperties
    dc:title hasValue "Buyer Goal Template 4"
```

```
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {furn:furniture, swfmo:purchase, swfmo:privateSeller, swfmo:delivery}
    dc:description hasValue "Goal Template for buying one piece of furniture from a private seller"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-20"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {
            <<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
            <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
            <<http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.2 $"
endNonFunctionalProperties


importedOntologies{
    <<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
    <<http://swf.quarto.at/repository/ontologies/location.wsml>>}

comment: no mediators are used


postcondition
 axiom bgs4Postcondition
    nonFunctionalProperties
        dc:description hasValue "a contract of purchase for a specific piece of furniture with a private seller, payment
by any payment method"
    endNonFunctionalProperties
    definedBy
        exists ?PCID(?PCID memberOf xsd:integer) and
        exists ?PurchaseItem(?PurchaseItem[
            item hasValue ?PurchaseFurniture,
            provider hasValue ?PrivateSeller
        ] memberOf swfmo:product) and
        exists ?PurchaseFurniture(?PurchaseFurniture memberOf furn:furniture) and
        exists ?Buyer(?Buyer[
                hasPaymentMethod hasValues {?BuyerPayment}
        ] memberOf swfmo:buyer) and
        exists ?BuyerPayment(?BuyerPayment memberOf swfmo:paymentMethod) and
        exists ?PrivateSeller(?PrivateSeller[
             acceptsPaymentMethod hasValues {?SellerPayment}
        ] memberOf swfmo:privateSeller) and
        exists ?SellerPayment(?SellerPayment memberOf swfmo:paymentMethod) and
        exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
        ((?Payment) implies
                ((?Payment memberOf ?BuyerPayment) and
                (?Payment memberOf ?SellerPayment)))
        and
     ?X[
            purchaseContractID hasValue ?PCID,
            purchaseItem hasValue ?PurchaseItem,
            buyer hasValue ?Buyer,
            seller hasValue ?PrivateSeller,
            purchasePayment hasValue ?Payment
        ] memberOf swfmo:purchaseContract .


effect
 axiom bgs4Effect
    nonFunctionalProperties
        dc:description hasValue "direct delivery of the purchased piece of furniture to a specific shipping address in
Austria"
    endNonFunctionalProperties
    definedBy
     exists ?Item(?Item memberOf swfmo:product) and
     ?X[
                deliveryItem hasValue ?Item
        ] memberOf swfmo:selfCollection .
```

3.2.2 Seller Goal Templates

Listing. Seller Goal Template 1 (WSML file)

```
/**
 * Seller Goal Template 1
 */

namespace <<http://swf.quarto.at/repository/GoalTemplates/SellFurnitureGeneralGoal.1.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    furn:<<http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    loc:<<http://swf.quarto.at/repository/ontologies/location.wsml#>>

goal <<http://swf.quarto.at/repository/GoalTemplates/SellFurnitureGeneralGoal.1.wsml>>

nonFunctionalProperties
    dc:title hasValue "Seller Goal Template 1"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:seller, swfmo:purchaseContract, furn:furniture, swfmo:dropShip}
    dc:description hasValue "Goal Template for selling one piece of furniture"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-20"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
            <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
            <<http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.3 $"
endNonFunctionalProperties


importedOntologies {
    <<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
    <<http://swf.quarto.at/repository/ontologies/location.wsml>>}


postcondition
 axiom sgs1Postcondition
    nonFunctionalProperties
        dc:description hasValue "a contract of purchase for a seller for a specific piece of furniture, payment by
creditcard"
    endNonFunctionalProperties
    definedBy
        exists ?PCID(?PCID memberOf xsd:integer) and
        exists ?PurchaseItem(?PurchaseItem[
            item hasValue ?PurchaseFurniture,
            provider hasValue ?Seller
        ] memberOf swfmo:product) and
        exists ?PurchaseFurniture(?PurchaseFurniture memberOf furn:furniture) and
        exists ?Seller(?Seller[
                contactAddress hasValue ?Address,
                acceptsPaymentMethod hasValues {?Payment}
        ] memberOf swfmo:seller) and
        exists ?Address(?Address[
                city.inCountry hasValue austria
        ] memberOf loc:address) and
        exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
        ?X[
            purchaseContractID hasValue ?PCID,
            purchaseItem hasValue ?PurchaseItem,
            seller hasValue ?Seller,
            purchasePayment hasValue ?Payment
        ] memberOf swfmo:purchaseContract .


effect
 axiom sgs1Effect
    nonFunctionalProperties
```

```
        dc:description hasValue "direct delivery of the purchased piece of furniture to a specific shipping address in
Austria"
    endNonFunctionalProperties
    definedBy
        exists ?Item(?Item memberOf swfmo:product) and
        exists ?Seller(?Seller[
            contactAddress hasValue ?ContactAddress
        ] memberOf swfmo:seller) and
        exists ?ContactAddress(?ContactAddress[
            city.inCountry hasValue austria
        ] memberOf loc:address) and
        exists ?Buyer(Buyer[
            shipToAddress hasValue ?BuyerAddress
        ] memberOf swfmo:buyer) and
        exists ?BuyerAddress(?BuyerAddress[
            city.inCountry hasValue austria
        ] memberOf loc:address) and
        exists ?Carrier(?Carrier[
            deliverCoverage hasValue austria
        ] memberOf swfmo:dropShipCarrier) and
    (?X[
            deliveryItem hasValue ?Item,
            sender hasValue ?Seller,
            receiver hasValue ?Buyer,
            carrier hasValue ?Carrier
        ] memberOf swfmo:dropShip
    ) or
    (?X[
            deliveryItem hasValue ?Item
        ] memberOf swfmo:selfCollection) .
```

Listing. Seller Goal Template 2 (WSML file)

```
/**
 * Seller Goal Template 2
 */

namespace <<http://swf.quarto.at/repository/GoalTemplates/QueryFurnitureProviderGoal.1.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    furn:<<http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>

goal <<http://www.fredlife.com/wsmo/resources/goals/QueryFurnitureProviderGoal.1.wsml>>

nonFunctionalProperties
    dc:title hasValue "Seller Goal Template 2"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:seller, swfmo:product, furn:furniture}
    dc:description hasValue "provide detailed product information for a specific piece of furniture without purchasing"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-20"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
        <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
        <<http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.2 $"
endNonFunctionalProperties


importedOntologies {
    <<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
    <<http://swf.quarto.at/repository/ontologies/location.wsml>>}

comment: no mediators are used
```

```
postcondition
 axiom sgs2Postcondition
     nonFunctionalProperties
         dc:description hasValue "provide information of all products offered in the marketplace"
     endNonFunctionalProperties
     definedBy
         exists ?FurnitureSearched(?FurnitureSearched memberOf furn:furniture) and
         exists ?Provider(?Provider memberOf swfmo:seller) and
         ?X[
              item hasValue ?FurnitureSearched,
              provider hasValue ?Provider
          ] memberOf swfmo:product .


effect
 axiom sgs2Effect
     nonFunctionalProperties
         dc:description hasValue "no effect"
     endNonFunctionalProperties
     definedBy
      ?X .
comment: this means there is no effect - however this is modelled in the end.
```

Listing. Seller Goal Template 3 (WSML file)

```
/**
 * Seller Goal Template 3
 */

namespace <<http://swf.quarto.at/repository/GoalTemplates/SellFurniturePrivateGoal.1.wsml#>>
     dc:<<http://purl.org/dc/elements/1.1#>>
     wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
     furn:<<http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
     swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>


goal <<http://swf.quarto.at/repository/GoalTemplates/SellFurniturePrivateGoal1.wsml>>

nonFunctionalProperties
     dc:title hasValue "Seller Goal Template 3"
     dc:creator hasValue "SWF Project"
     dc:subject hasValues {swfmo:privateSeller, swfmo:purchaseContract, furn:furniture, swfmo:selfCollection}
     dc:description hasValue "Goal Template for a private seller for selling one piece of furniture"
     dc:publisher hasValue "SWF Project"
     dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
     dc:date hasValue "2004-09-20"
     dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
     dc:format hasValue "text/html"
     dc:language hasValue "en-US"
     dc:relation hasValues {<<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
          <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
          <<http://swf.quarto.at/repository/ontologies/location.wsml>>}
     dc:coverage hasValue "SWF virtual marketplace"
     dc:rights hasValue <<http://www.deri.org/privacy.html>>
     version hasValue "$Revision: 1.2 $"
endNonFunctionalProperties


importedOntologies {
     <<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
     <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
     <<http://swf.quarto.at/repository/ontologies/location.wsml>>}

comment: no mediators are used


postcondition
 axiom sgs3Postcondition
     nonFunctionalProperties
```

```
        dc:description hasValue "a contract of purchase for a seller for a specific piece of furniture, payment by
creditcard"
    endNonFunctionalProperties
    definedBy
        exists ?PCID(?PCID memberOf xsd:integer) and
        exists ?PurchaseItem(?PurchaseItem[
            item hasValue ?PurchaseFurniture,
            provider hasValue ?PrivateSeller
        ] memberOf swfmo:product) and
        exists ?PurchaseFurniture(?PurchaseFurniture memberOf furn:furniture) and
        exists ?PrivateSeller(?PrivateSeller[
                contactAddress hasValue ?Address,
                acceptsPaymentMethod hasValues {?Payment}
        ] memberOf swfmo:privateSeller) and
        exists ?Address(?Address[
                city.inCountry hasValue austria
        ] memberOf loc:address) and
        exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
        ?X[
            purchaseContractID hasValue ?PCID,
            purchaseItem hasValue ?PurchaseItem,
            seller hasValue ?PrivateSeller,
            purchasePayment hasValue ?Payment
        ] memberOf swfmo:purchaseContract .


effect
 axiom sgs3Effect
    nonFunctionalProperties
        dc:description hasValue "direct delivery of the purchased piece of furniture to a specific shipping address in
Austria"
    endNonFunctionalProperties
    definedBy
        exists ?Item(?Item memberOf swfmo:product) and
        ?X[
                deliveryItem hasValue ?Item
         ] memberOf swfmo:selfCollection .
```

3.3 Cooperative Goals

This section specifies compatible Goal Templates in Cooperative Goals. Cooperative Goals are defined in an Ontology; the schema defines the description elements of Cooperative Goals, the instances specify the Cooperative Goals of this Use Case as defined above.

Listing. Cooperative Knowledge Ontology(WSML file)

```
/**
 * SWF Use Case Cooperative Knowledge Ontology
 */

namespace <<http://swf.quarto.at/repository/GoalCooperation/cooperativeknowledgeontology.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    xsd:<<http://www.w3.org/2001/XMLSchema#>>

ontology <<http://swf.quarto.at/repository/GoalCooperation/cooperativeknowledgeontology.wsml>>


nonFunctionalProperties
    dc:title hasValue "SWF Use Case Cooperative Knowledge Ontology "
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {"Goals", "compatible", "cooperation roles"}
    dc:description hasValue "describes the schema for cooperative goal knowledge to specify cooperative goals"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValues {"Reinhold Herzog", "Berhard Keimel", "Michael Stollberg"}
    dc:date hasValue "2004-09-21"
    dc:type hasValue <<http://www.wsmo.org/2004/d2/v0.3/20040329/#ontos>>
    /**
```

```
        * ontologies are modeled as WSMO Ontologies
        */
       dc:format hasValue "text/html"
       dc:language hasValue "en-US"
       dc:relation hasValues {<<ndCooperationKnowledge1.0.wsml>>}
       dc:coverage hasValue "project specific "
       dc:rights hasValue <<http://www.deri.org/privacy.html>>
       version hasValue "$Revision: 1.5 $"
endNonFunctionalProperties


// CONCEPTS

concept cooperativeGoalGroup
     nonFunctionalProperties
          dc:description hasValue "a cooperativeGoalGroup defines compatible Goal Schemas( that are Goal Schemas
                that potential cooperation partners can carry Goal Instances of), along with constraints which are
modelled
                by WSMO GG Mediators in order to resolve mismatches between compatible Goal Schemas that are not
compatible
                a priori."
     endNonFunctionalProperties

     hasCooperativeGoal ofType set Goal
     hasCooperativeGoalConstraint ofType set GGMediator


/**
 * INSTANCES
 * these are the Cooperative Goals defined in the SWF Use Case
*/

instance BuyerSellerCC memberOf cooperativeGoalGroup
     hasCooperativeGoal hasValues {
          <<http://swf.quarto.at/repository/GoalCooperation/BuyFurnitureCredidCardGoal.1.wsml>>,
          <<http://swf.quarto.at/repository/GoalCooperation/SellFurnitureGeneralGoal.1.wsml>>}
     hasCooperativeGoalConstraint hasValues {
     <<cooperativegoal1ggMediator.wsml>>}

instance BuyerSellerGeneral memberOf cooperativeGoalGroup
     hasCooperativeGoal hasValues {
          <<http://swf.quarto.at/repository/GoalCooperation/BuyFurnitureAnyPaymentGoal.1.wsml>>,
          <<http://swf.quarto.at/repository/GoalCooperation/SellFurnitureGeneralGoal.1.wsml>>}

instance BuyerSellerGeneralPrivate memberOf cooperativeGoalGroup
     hasCooperativeGoal hasValues {
          <<http://swf.quarto.at/repository/GoalCooperation/BuyFurnitureAnyPaymentGoal.1.wsml>>,
          <<http://swf.quarto.at/repository/GoalCooperation/SellFurniturePrivateGoal.1.wsml>>}

instance QueryFurniture memberOf cooperativeGoalGroup
     hasCooperativeGoal hasValues {
          <<http://swf.quarto.at/repository/GoalCooperation/QueryFurnitureRequestGoal.1.wsml>>,
          <<http://swf.quarto.at/repository/GoalCooperation/QueryFurnitureProviderGoal.1.wsml>>}

instance BuyerSellerPrivate memberOf cooperativeGoalGroup
     hasCooperativeGoal hasValues {
          <<http://swf.quarto.at/repository/GoalCooperation/BuyFurniturePrivateGoal.1.wsml>>,
          <<http://swf.quarto.at/repository/GoalCooperation/SellFurniturePrivateGoal.1.wsml>>}
```

3.4 Goal Instances

The following specifies examples for Goal Instanaces for buyers and selllers which are assigned to Freds for automated resolution in the use case. Other Goal Instances can be created from the Goal Templates defined above.

For this Use Case, we only specify a few Goal Instances for every Goal Template described above, in order to showcase how Goal Instances look like. In the prototype

implementation, Goal Instances are defined as WSMO Goals; the additional constructs defined for SWF Goal Instances are handled in the FRED Goal Instance technology.

The concrete data used in the Goal Instance definitions refer to the instances defined in the SWF Use Case Knowledge Base defined in Section 3.7

3.4.1 Buyer Goal Instances

Listing. Buyer Goal Instance 1 (WSML file)

```
/**
 * Buyer Goal Instance 1
 *
 * for the SWF prototype, Goal Instance are modelled as WSMO Goals,
 * and managed as 'Active WSMO Goals' in the system
 *
 *
 * the owner of a Goal Instance is defined in the corresponding FRED Goal Instance
 * the same holds for the status information
 *
 * Goal Instances do not import ontologies or use mediators;
 * all domain terminology is inherited from the Goal Template
 * Goal Instance notions only refine (= no extensions) Goal Template notions
*/


namespace <<http://swf.quarto.at/repository/ActiveGoals/111#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    kb:<<http://swf.quarto.at/repository/ontologies/kb.wsml#>>

goal <<http://swf.quarto.at/repository/ActiveGoals/111.wsml>>

nonFunctionalProperties
    dc:title hasValue "Buyer Goal Instance 1"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:buyer, swfmo:purchaseContract, swfmo:dropShip, swfmo:creditCard, kb:f6}
    dc:description hasValue "Goal Instance from Buyer Goal Template 1 for buying a chair"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-21"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
        comment: for the prototype, Goal Instances are described as WSMO Goals
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://swf.quarto.at/repository/GoalTemplates/BuyFurnitureCredidCardGoal.1.wsml>>}
        comment: the value of the is specified as the Goal Template that the Goal Instance is a instance of
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.4 $"
endNonFunctionalProperties


/**
 * the notions of submission are now incorporated in the postcondontion and effects

submission bgi1Submission
    nonFunctionalProperties
        dc:description hasValue "the information to be submitted as input to a service are:
                    [1] a piece of furniture to be bought,
                    [2] buyer information (name, bill address in Europe, ship address in Austria, payment information),
                    [3] payment methods that the buyer has."
    endNonFunctionalProperties
    definedBy
        // refers to f6 in SWF Use Case KB
        instance PurchaseFurniture memberOf furn:chair
            height hasValue 90
            material hasValues {wood}
            color hasValue "red"
            numberOfLegs hasValue 4
```

```
        instance MichaelStollberg memberOf swfmo:buyer
            marketplaceParticipantID hasValue "buyer1234"
            name hasValue "Michael Stollberg"
            email hasValue "michael.stollberg@deri.org"
            telephone hasValue "+43-512-507-6479"
            fax hasValue "+43-512-507-9872"
            billToAddress hasValue MSAddress
            shipToAddress hasValue MSAddress
            hasPaymentMethod hasValues {MSCreditCard, MSCash}

        instance MSAddress memberOf loc:address
            street hasValue "Technikerstrasse"
            number hasValue "13"
            city hasValue innsbruck
            zip hasValue 6020

        instance MSCreditCard memberOf swf:creditCard
            type hasValue MasterCard
            number hasValue "123456789"
            holder hasValue "Michael Stollberg"
            expMonth hasValue 08
            expYear hasValue 2007

*/


postcondition
 axiom bgi1Postcondition
    nonFunctionalProperties
        dc:description hasValue "a contract of purchase for the Buyer for a chair (f6 in SWF Use Case KB), payment by
creditcard"
    endNonFunctionalProperties
    definedBy
        exists ?PCID(?PCID memberOf xsd:integer) and
        exists ?PurchaseItem(?PurchaseItem[
            item hasValue ?PurchaseFurniture
        ] memberOf swfmo:product) and
        exists ?PurchaseFurniture(?PurchaseFurniture[
            height hasValue 90,
            material hasValues {wood},
            color hasValue "red",
            numberOfLegs hasValue 4
        ] memberOf furn:chair) and
        ?X[
            purchaseContractID hasValue ?PCID,
            purchaseItem hasValue ?PurchaseItem,
            buyer hasValue kb:MichaelStollberg,
            purchasePayment hasValue kb:MSCreditCard
        ] memberOf swfmo:purchaseContract .


effect
 axiom bgi1Effect
    nonFunctionalProperties
        dc:description hasValue "direct delivery of the purchased piece of furniture to a specific shipping address in
Austria"
    endNonFunctionalProperties
    definedBy
        exists ?Item(?Item[
             item hasValue ?PurchaseFurniture
        ] memberOf swfmo:product) and
        exists ?PurchaseFurniture(?PurchaseFurniture[
            height hasValue 90,
            material hasValues {wood},
            color hasValue "red",
            numberOfLegs hasValue 4
        ] memberOf furn:chair) and
        ?X[
                deliveryItem hasValue ?Item,
                receiver hasValue kb:MichaelStollberg
        ] memberOf swfmo:dropShip .
```

Listing. Buyer Goal Instance 2 (WSML file)

```
/**
 * Buyer Goal Instance 2
 *
 * for the SWF prototype, Goal Instance are modelled as WSMO Goals,
 * and managed as 'Active WSMO Goals' in the system
 *
 *
 * the owner of a Goal Instance is defined in the corresponding FRED Goal Instance
 * the same holds for the status information
 *
 * Goal Instances do not import ontologies or use mediators;
 * all domain terminology is inherited from the Goal Template
 * Goal Instance notions only refine (= no extensions) Goal Template notions
*/


namespace <<http://swf.quarto.at/repository/ActiveGoals/112#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    kb:<<http://swf.quarto.at/repository/ontologies/kb.wsml#>>

goal <<http://swf.quarto.at/repository/ActiveGoals/112.wsml>>

nonFunctionalProperties
    dc:title hasValue "Buyer Goal Instance 2"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:buyer, swfmo:purchaseContract, furn:desk, swfmo:dropShip, swfmo:creditCard}
    dc:description hasValue "Goal Instance from Buyer Goal Template 2 for buying a bed"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-29"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
        comment: for the prototype, Goal Instances are described as WSMO Goals
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://swf.quarto.at/repository/GoalTemplates/BuyFurnitureAnyPaymentGoal.1.wsml>>}
        comment: the value of the is specified as the Goal Template that the Goal Instance is a instance of
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.3 $"
endNonFunctionalProperties


/**
 * the notions of submission are now incorporated in the postcondontion and effects

submission bgi2Submission
    nonFunctionalProperties
        dc:description hasValue "the information to be submitted as input to a service are:
                [1] a piece of furniture to be bought,
                [2] buyer information (name, bill address in Europe, ship address in Austria, payment information),
                [3] payment methods that the buyer has."
    endNonFunctionalProperties
    definedBy
        // refers to SWF Use Case KB f19, a double bed
        instance PurchaseFurniture memberOf furn:doubleBed
            width hasValue 140
            material hasValues {steel}
            color hasValue "white"


        instance MichaelStollberg memberOf swfmo:buyer
            marketplaceParticipantID hasValue "buyer1234"
            name hasValue "Michael Stollberg"
            email hasValue "michael.stollberg@deri.org"
            telephone hasValue "+43-512-507-6479"
            fax hasValue "+43-512-507-9872"
            billToAddress hasValue MSAddress
            shipToAddress hasValue MSAddress
            hasPaymentMethod hasValues {MSCreditCard, MSCash}

        instance MSAddress memberOf loc:address
            street hasValue "Technikerstrasse"
            number hasValue "13"
            city hasValue innsbruck
```

```
            zip hasValue 6020

        instance MSCreditCard memberOf swf:creditCard
            type hasValue MasterCard
            number hasValue "123456789"
            holder hasValue "Michael Stollberg"
            expMonth hasValue 08
            expYear hasValue 2007

        instance MSCash memberOf swfmo:cash
            payer hasValue MichaelStollberg

*/


postcondition
 axiom bgi2Postcondition
    nonFunctionalProperties
        dc:description hasValue "a contract of purchase for the Buyer for a specific piece of furniture, payment by
creditcard"
    endNonFunctionalProperties
    definedBy
        exists ?PCID(?PCID memberOf xsd:integer) and
        exists ?PurchaseItem(?PurchaseItem[
            item hasValue ?PurchaseFurniture
        ] memberOf swfmo:product) and
        exists ?PurchaseFurniture(?PurchaseFurniture[
            width hasValue 140,
            material hasValues {steel},
            color hasValue "white"
        ] memberOf furn:doubleBed) and
        exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
        (?Payment implies (?Payment=kb:MSCreditCard or ?Payment=kb:MSCash)) and
        ?X[
            purchaseContractID hasValue ?PCID,
            purchaseItem hasValue ?PurchaseItem,
            buyer hasValue kb:MichaelStollberg,
            purchasePayment hasValue ?Payment
        ] memberOf swfmo:purchaseContract .


effect
 axiom bgi2Effect
    nonFunctionalProperties
        dc:description hasValue "direct delivery of the purchased piece of furniture to a specific shipping address in
Austria,
            or self collection by the purchaser "
    endNonFunctionalProperties
    definedBy
        exists ?Item(?Item[
            item hasValue ?PurchaseFurniture
        ] memberOf swfmo:product) and
        exists ?PurchaseFurniture(?PurchaseFurniture[
            width hasValue 140,
            material hasValues {steel},
            color hasValue "white"
        ] memberOf furn:doubleBed) and
    (?X[
                deliveryItem hasValue ?Item,
                receiver hasValue kb:MichaelStollberg
            ] memberOf swfmo:dropShip)
        or
        (?X[
                deliveryItem hasValue ?Item
             ] memberOf swfmo:selfCollection) .
```

Listing. Buyer Goal Instance 3 (WSML file)

```
/**
 * Buyer Goal Instance 3
 *
 * for the SWF prototype, Goal Instance are modelled as WSMO Goals,
 * and managed as 'Active WSMO Goals' in the system
 *
 *
```

```
 * the owner of a Goal Instance is defined in the corresponding FRED Goal Instance
 * the same holds for the status information
 *
 * Goal Instances do not import ontologies or use mediators;
 * all domain terminology is inherited from the Goal Template
 * Goal Instance notions only refine (= no extensions) Goal Template notions
*/


namespace <<http://swf.quarto.at/repository/ActiveGoals/113#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    kb:<<http://swf.quarto.at/repository/ontologies/kb.wsml#>>

goal <<http://swf.quarto.at/repository/ActiveGoals/113.wsml>>

nonFunctionalProperties
    dc:title hasValue "Buyer Goal Instance 3"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:buyer, swfmo:product, furn:storageFurniture}
    dc:description hasValue "Goal Instance from Buyer Goal Template 3 for gathering product information
            about storage furniture, offered by all sellers in the marketplace"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-30"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
        comment: for the prototype, Goal Instances are described as WSMO Goals
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://swf.quarto.at/repository/GoalTemplates/QueryFurnitureRequestGoal.1.wsml>>}
        comment: the value of the is specified as the Goal Template that the Goal Instance is a instance of
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.3 $"
endNonFunctionalProperties


/**
 * the notions of submission are now incorporated in the postcondontion and effects

submission bgi3Submission
    nonFunctionalProperties
        dc:description hasValue "the information to be submitted as input to a service are:
                [1] a type of furniture product information is required for,
                [2] the buyer who is requesting information,
                [3] the seller of type of sellers that product infomration is requested for"
    endNonFunctionalProperties
    definedBy
        instance FurnitureSearched memberOf furn:storageFurniture
            material hasValues {wood}

        instance DieterFensel memberOf swfmo:buyer
            marketplaceParticipantID hasValue "buyer65245"
            name hasValue "Dieter Fensel"
            email hasValue "dieter.fensel@deri.org"
            telephone hasValue "+43-512-507-6488"
            fax hasValue "+43-512-507-9872"
            billToAddress hasValue DFAddress
            shipToAddress hasValue DFAddress

        instance DFAddress memberOf loc:address
            street hasValue "Technikerstrasse"
            number hasValue "13"
            city hasValue innsbruck
            zip hasValue 6020

*/


postcondition
 axiom bgi3Postcondition
    nonFunctionalProperties
        dc:description hasValue "a contract of purchase for the Buyer for a specific piece of furniture, payment by
creditcard"
    endNonFunctionalProperties
```

```
        definedBy
         exists ?FurnitureSearched(?FurnitureSearched[
                material hasValues {wood}
         ] memberOf furn:storageFurniture) and
         exists ?Seller(?Seller memberOf swfmo:seller) and
         ?X[
              item hasValue ?FurnitureSearched,
              provider hasValue ?Seller
         ] memberOf swfmo:product .


    effect
     axiom bgi3Effect
        nonFunctionalProperties
            dc:description hasValue "direct delivery of the purchased piece of furniture to a specific shipping address in
    Austria"
        endNonFunctionalProperties
        definedBy
         ?X .

    comment: means there isnt an effect
```

Listing. Buyer Goal Instance 4 (WSML file)

```
    /**
     * Buyer Goal Instance 4
     *
     * for the SWF prototype, Goal Instance are modelled as WSMO Goals,
     * and managed as 'Active WSMO Goals' in the system
     *
     *
     * the owner of a Goal Instance is defined in the corresponding FRED Goal Instance
     * the same holds for the status information
     *
     * Goal Instances do not import ontologies or use mediators;
     * all domain terminology is inherited from the Goal Template
     * Goal Instance notions only refine (= no extensions) Goal Template notions
    */


    namespace <<http://swf.quarto.at/repository/ActiveGoals/114#>>
        dc:<<http://purl.org/dc/elements/1.1#>>
        wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
        swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
        kb:<<http://swf.quarto.at/repository/ontologies/kb.wsml#>>

    goal <<http://swf.quarto.at/repository/ActiveGoals/114.wsml>>

    nonFunctionalProperties
        dc:title hasValue "Buyer Goal Instance 1"
        dc:creator hasValue "SWF Project"
        dc:subject hasValues {swfmo:buyer, swfmo:purchaseContract, swfmo:paymentMethod, kb:f5, swfmo:dropShip}
        dc:description hasValue "Goal Instance from Buyer Goal Template 4 for buying a bookshelf"
        dc:publisher hasValue "SWF Project"
        dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
        dc:date hasValue "2004-09-29"
        dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
            comment: for the prototype, Goal Instances are described as WSMO Goals
        dc:format hasValue "text/html"
        dc:language hasValue "en-US"
        dc:relation hasValues {<<http://swf.quarto.at/repository/GoalTemplates/BuyFurniturePrivateGoal.1.wsml>>}
            comment: the value of the is specified as the Goal Template that the Goal Instance is a instance of
        dc:coverage hasValue "SWF virtual marketplace"
        dc:rights hasValue <<http://www.deri.org/privacy.html>>
        version hasValue "$Revision: 1.3 $"
    endNonFunctionalProperties


    /**
     * the notions of submission are now incorporated in the postcondontion and effects

    submission bgi4Submission
        nonFunctionalProperties
            dc:description hasValue "the information to be submitted as input to a service are:
                        [1] a piece of furniture to be bought,
```

```
                    [2] buyer information (name, bill address in Europe, ship address in Austria, payment information),
                    [3] payment methods that the buyer has."
        endNonFunctionalProperties
        definedBy
            // refers to SWF Use Case KB f5, a book shelf offered as product by a private seller
            instance PurchaseFurniture memberOf furn:bookShelf
                depth hasValue 110
                width hasValue 30
                material hasValues {wood}
                isFlexible hasValue false

            instance loanToma memberOf swfmo:buyer
                marketplaceParticipantID hasValue "buyer9876"
                name hasValue "loan Toma"
                email hasValue "ioan.toma@deri.org"
                telephone hasValue "+43-512-507-6461"
                fax hasValue "+43-512-507-9872"
                billToAddress hasValue ITAddress
                shipToAddress hasValue ITAddress
                hasPaymentMethod hasValues {ITCreditCard, ITCash, ITCheck}

            instance ITAddress memberOf loc:address
                street hasValue "Technikerstrasse"
                number hasValue "13"
                city hasValue innsbruck
                zip hasValue 6020

            instance ITCreditCard memberOf swfmo:creditCard
                type hasValue Visa
                number hasValue "237671"
                holder hasValue "loan Toma"
                expMonth hasValue 06
                expYear hasValue 2006

            instance ITCash memberOf swfmo:cash
                payer hasValue loanToma

            instance ITCheck memberOf swfmo:check
                drawer hasValue loanToma
                drawerAccount hasValue ITAccount

            instance ITAccount memberOf swfmo:account
                bank hasValue HypoTirol
                owner hasValue "loan Toma"
                accountNumber hasValue      17111345


    */


    postcondition
     axiom bgi4Postcondition
        nonFunctionalProperties
            dc:description hasValue "a contract of purchase for the Buyer for a specific piece of furniture, payment by
    creditcard"
        endNonFunctionalProperties
        definedBy
            exists ?PCID(?PCID memberOf xsd:integer) and
            exists ?PurchaseItem(?PurchaseItem[
                item hasValue ?PurchaseFurniture,
                provider hasValue ?PrivateSeller
            ] memberOf swfmo:product) and
            exists ?PurchaseFurniture(?PurchaseFurniture[
                depth hasValue 110,
                width hasValue 30,
                material hasValues {wood},
                isFlexible hasValue false
            ] memberOf furn:bookShelf) and
            exists ?PrivateSeller(?PrivateSeller memberOf swfmo:privateSeller) and
            exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
            (?Payment implies (?Payment=kb:ITCreditCard
                    or ?Payment=kb:ITCash
                    or ?Payment=kb:ITCheck) )
            and
            ?X[
                purchaseContractID hasValue ?PCID,
```

```
            purchaseItem hasValue ?PurchaseItem,
            buyer hasValue loanToma,
            seller hasValue ?PrivateSeller,
            purchasePayment hasValue ?Payment
        ] memberOf swfmo:purchaseContract .


effect
 axiom bgi4Effect
    nonFunctionalProperties
        dc:description hasValue "direct delivery of the purchased piece of furniture to a specific shipping address in
Austria"
    endNonFunctionalProperties
    definedBy
        exists ?Item(?Item[
            item hasValue PurchaseFurniture,
            provider hasValue ?PrivateSeller
        ] memberOf swfmo:product) and
        exists ?PurchaseFurniture(?PurchaseFurniture[
            depth hasValue 100,
            width hasValue 30,
            material hasValues {wood},
            isFlexible hasValue false
        ] memberOf furn:bookShelf) and
        exists ?PrivateSeller(?PrivateSeller memberOf swfmo:privateSeller) and
        ?X[
            deliveryItem hasValue ?Item
        ] memberOf swfmo:selfCollection .
```

3.4.2 Seller Goal Instances

The following defines Seller Goal Instances, thus Goals that sellers in the marketplace assign to their Freds.

Listing. Seller Goal Instance 1 - for IKEA (WSML file)

```
/**
 * Seller Goal Instance 1 (for IKEA)
 *
 * for the SWF prototype, Goal Instance are modelled as WSMO Goals,
 * and managed as 'Active WSMO Goals' in the system
 *
 *
 * the owner of a Goal Instance is defined in the corresponding FRED Goal Instance
 * the same holds for the status information
 *
 * Goal Instances do not import ontologies or use mediators;
 * all domain terminology is inherited from the Goal Template
 * Goal Instance notions only refine (= no extensions) Goal Template notions
*/


namespace <<http://swf.quarto.at/repository/ActiveGoals/121.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    kb:<<http://swf.quarto.at/repository/ontologies/kb.wsml#>>

goal <<http://swf.quarto.at/repository/ActiveGoals/121.wsml>>

nonFunctionalProperties
    dc:title hasValue "Seller Goal Instance 1"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {kb:IKEA, swfmo:purchaseContract, furn:furniture, swfmo:delivery, swfmo:paymentMethod}
    dc:description hasValue "Goal Instance from private IKEA as a seller for selling a piece of furniture"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-21"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
```

```
    dc:relation hasValues {<<http://swf.quarto.at/repository/GoalTemplates/SellFurnitureGeneralGoal.1.wsml>>}
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.3 $"
endNonFunctionalProperties

/**
    * the notions of submission are now incorporated in the postcondontion and effects

submission sgi1Submission
    nonFunctionalProperties
        dc:description hasValue "the information to be submitted as input to a service are:
                [1] a piece of furniture to be sold,
                [2] IKEA information as a seller,
                [3] payment methods that IKEA accepts
                [4] the IKEA Drop Ship Delivery Service."
    endNonFunctionalProperties
    definedBy

        comment: all products with provider = IKEA are considered

        instance IKEA memberOf swfmo:company
            marketplaceParticipantID hasValue "sellerIKEA"
            name hasValue "IKEA"
            email hasValue "office@ikea.at"
            telephone hasValue "+43-1-123456"
            fax hasValue "+43-1-123456"
            contactAddress hasValue IKEAAddress
            acceptsPaymentMethod hasValues {IKEACreditCard, IKEAInvoice, IKEACash}
            companyNumber hasValue "xyz"
            website hasValue "www.ikea.at"

        instance IKEAAddress memberOf loc:address
            street hasValue "am DEZ"
            number hasValue "3"
            city hasValue innsbruck
            zip hasValue 6020

        instance IKEACreditCard memberOf swf:creditCard
            type hasValue (MasterCard or Visa)

        instance IKEAInvoice memberOf swf:invoice
            receiver hasValue IKEA

        instance IKEACash memberOf swf:cash
            receiver hasValue IKEA


        instance IKEADeliveryService memberOf swfmo:dropShipCarrier
            name hasValue "IKEA Delivery Service"
            companyNumber hasValue "IKEAds001"
            contactaddress hasValue IKEAAddress
            transportBy hasValue truck
            deliveryCoverage hasValue austria
*/

postcondition
 axiom sgi1Postcondition
    nonFunctionalProperties
        dc:description hasValue "a contract of purchase for IKEA for a specific piece of furniture"
    endNonFunctionalProperties
    definedBy
        exists ?PCID(?PCID memberOf xsd:integer) and
        exists ?PurchaseItem(?PurchaseItem[
            item hasValue ?PurchaseFurniture,
            provider hasValue IKEA
        ] memberOf swfmo:product) and
        exists ?PurchaseFurniture(?PurchaseFurniture memberOf furn:furniture) and
        exists ?Buyer(?Buyer[
            hasPaymentMethod hasValues {?Payment}
        ] memberOf swfmo:buyer) and
        exists ?BuyerPayment(?BuyerPayment memberOf swfmo:paymentMethod) and
        exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
        ((?Payment) implies
             ((?Payment memberOf ?BuyerPayment) and
                 (?Payment memberOf IKEA.acceptsPaymentMethod)))
```

```
        and
        ?X[
                purchaseContractID hasValue ?PCID,
                purchaseItem hasValue ?PurchaseItem,
                buyer hasValue ?Buyer,
                seller hasValue IKEA,
                purchasePayment hasValue ?Payment
        ] memberOf swfmo:purchaseContract .


effect
 axiom sgi1Effect
    nonFunctionalProperties
        dc:description hasValue "self collection of the sold piece of furniture by the buyer"
    endNonFunctionalProperties
    definedBy
        exists ?Item(?Item[
              provider hasValue IKEA
        ] memberOf swfmo:product) and
        exists ?Buyer(?Buyer[
             shipToAddress.city.inCountry hasValue austria
        ] memberOf swfmo:buyer) and
    (?X[
              deliveryItem hasValue ?Item,
              sender hasValue IKEA,
              receiver hasValue ?Buyer,
              carrier hasValue IKEADeliveryService
         ] memberOf swfmo:dropShip
     ) or
     (?X[
              deliveryItem hasValue ?Item
            ] memberOf swfmo:selfCollection) .
```

Listing. Seller Goal Instance 1.2 - for Leiner (WSML file)

```
/**
 * Seller Goal Instance 1.2 (for Leiner)
 *
 * for the SWF prototype, Goal Instance are modelled as WSMO Goals,
 * and managed as 'Active WSMO Goals' in the system
 *
 *
 * the owner of a Goal Instance is defined in the corresponding FRED Goal Instance
 * the same holds for the status information
 *
 * Goal Instances do not import ontologies or use mediators;
 * all domain terminology is inherited from the Goal Template
 * Goal Instance notions only refine (= no extensions) Goal Template notions
*/


namespace
<<http://swf.quarto.at/repository/ActiveGoals/1212.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    kb:<<http://swf.quarto.at/repository/ontologies/kb.wsml#>>

goal <<http://swf.quarto.at/repository/ActiveGoals/1212.wsml>>

nonFunctionalProperties
    dc:title hasValue "Seller Goal Instance 1.2"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {kb:LEINER, swfmo:purchaseContract, furn:furniture, swfmo:delivery, swfmo:paymentMethod}
    dc:description hasValue "Goal Instance from private Leiner as a seller for selling a piece of furniture"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-21"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
```

```
    dc:relation hasValues {<<http://swf.quarto.at/repository/GoalTemplates/SellFurnitureGeneralGoal.1.wsml>>}
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.3 $"
endNonFunctionalProperties

/**
    * the notions of submission are now incorporated in the postcondontion and effects

submission sgi12Submission
    nonFunctionalProperties
        dc:description hasValue "the information to be submitted as input to a service are:
                [1] a piece of furniture to be sold,
                [2] Leiner information as a seller,
                [3] payment methods that Leiner accepts
                [4] the Leiner Drop Ship Delivery Service."
    endNonFunctionalProperties
    definedBy

     comment: all products with provider = IKEA are considered

        instance LEINER memberOf swfmo:company
            marketplaceParticipantID hasValue "sellerLeiner"
            name hasValue "Leiner"
            email hasValue "office@leiner.at"
            telephone hasValue "+43-662-63970"
            fax hasValue "+43-662-620843"
            contactAddress hasValue LeinerAddress
            acceptsPaymentMethod hasValues {LeinerCreditCard, LeinerInvoice, LeinerCash}
            companyNumber hasValue "abc"
            website hasValue "www.leiner.at"

        instance LeinerAddress memberOf loc:address
            street hasValue "Alpenstraße"
            number hasValue "1"
            city hasValue salzburgStadt
            zip hasValue 6020

        instance LeinerCreditCard memberOf swf:creditCard
            type hasValue (MasterCard or Visa)

        instance LeinerInvoice memberOf swf:invoice
            receiver hasValue Leiner

        instance LeinerCash memberOf swf:cash
            receiver hasValue Leiner


        instance LeinerDeliveryService memberOf swfmo:dropShipCarrier
            name hasValue "Leiner Delivery Service"
            companyNumber hasValue "Leinerds001"
            contactaddress hasValue LeinerAddress
            transportBy hasValue truck
            deliveryCoverage hasValue austria
*/

postcondition
 axiom sgi12Postcondition
    nonFunctionalProperties
        dc:description hasValue "a contract of purchase for Leiner for a specific piece of furniture"
    endNonFunctionalProperties
    definedBy
        exists ?PCID(?PCID memberOf xsd:integer) and
        exists ?PurchaseItem(?PurchaseItem[
            item hasValue ?PurchaseFurniture,
            provider hasValue LEINER
        ] memberOf swfmo:product) and
        exists ?PurchaseFurniture(?PurchaseFurniture memberOf furn:furniture) and
        exists ?Buyer(?Buyer[
            hasPaymentMethod hasValues {?BuyerPayment}
        ] memberOf swfmo:buyer) and
        exists ?BuyerPayment(?BuyerPayment memberOf swfmo:paymentMethod) and
        exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
        ((?Payment) implies
             ((?Payment memberOf ?BuyerPayment) and
                 (?Payment memberOf LEINER.acceptsPaymentMethod)))
```

```
        and
        ?X[
                purchaseContractID hasValue ?PCID,
                purchaseItem hasValue ?PurchaseItem,
                buyer hasValue ?Buyer,
                seller hasValue LEINER,
                purchasePayment hasValue ?Payment
        ] memberOf swfmo:purchaseContract .


effect
 axiom sgi12Effect
    nonFunctionalProperties
        dc:description hasValue "self collection of the sold piece of furniture by the buyer"
    endNonFunctionalProperties
    definedBy
        exists ?Item(?Item[
             provider hasValue LEINER
        ] memberOf swfmo:product) and
        exists ?Buyer(?Buyer[
            shipToAddress.city.inCountry hasValue austria
        ] memberOf swfmo:buyer) and
    (?X[
             deliveryItem hasValue ?Item,
             sender hasValue LEINER,
             receiver hasValue ?Buyer,
             carrier hasValue LeinerDeliveryService
         ] memberOf swfmo:dropShip
     ) or
     (?X[
             deliveryItem hasValue ?Item
            ] memberOf swfmo:selfCollection) .
```

Listing. Seller Goal Instance 1.3 - for Kika (WSML file)

```
/**
 * Seller Goal Instance 1.3
 *
 * for the SWF prototype, Goal Instance are modelled as WSMO Goals,
 * and managed as 'Active WSMO Goals' in the system
 *
 *
 * the owner of a Goal Instance is defined in the corresponding FRED Goal Instance
 * the same holds for the status information
 *
 * Goal Instances do not import ontologies or use mediators;
 * all domain terminology is inherited from the Goal Template
 * Goal Instance notions only refine (= no extensions) Goal Template notions
*/


namespace
<<http://swf.quarto.at/repository/ActiveGoals/1213.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>


goal <<http://swf.quarto.at/repository/ActiveGoals/1213.wsml>>

nonFunctionalProperties
    dc:title hasValue "Seller Goal Instance 1.3"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:company, swfmo:purchaseContract, furn:furniture, swfmo:delivery,
swfmo:paymentMethod}
    dc:description hasValue "Goal Instance from private Kika as a seller for selling a piece of furniture"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-21"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
    dc:format hasValue "text/html"
```

```
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://swf.quarto.at/repository/GoalTemplates/SellFurnitureGeneralGoal.1.wsml>>}
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.3 $"
endNonFunctionalProperties

/**
    * the notions of submission are now incorporated in the postcondontion and effects

submission sgi13Submission
    nonFunctionalProperties
        dc:description hasValue "the information to be submitted as input to a service are:
                [1] a piece of furniture to be sold,
                [2] Kika information as a seller,
                [3] payment methods that Kika accepts
                [4] the Kika Drop Ship Delivery Service."
    endNonFunctionalProperties
    definedBy
        instance Kika memberOf swfmo:company
            marketplaceParticipantID hasValue "sellerKika"
            name hasValue "Kika"
            email hasValue "office@kika.at"
            telephone hasValue "+43-1-2033539"
            fax hasValue "+43-1-2033539"
            contactAddress hasValue KikaAddress
            acceptsPaymentMethod hasValues {KikaCreditCard, KikaInvoice, KikaCash}
            companyNumber hasValue "kika"
            website hasValue "www.kika.at"

        instance KikaAddress memberOf loc:address
            street hasValue "Donaustadtstraße"
            number hasValue "1"
            city hasValue wienStadt
            zip hasValue 1220

        instance KikaCreditCard memberOf swf:creditCard
            type hasValue (MasterCard or Visa)

        instance KikaInvoice memberOf swf:invoice
            receiver hasValue Kika

        instance KikaCash memberOf swf:cash
            receiver hasValue Kika


        instance KikaDeliveryService memberOf swfmo:dropShipCarrier
            name hasValue "Kika Delivery Service"
            companyNumber hasValue "Kikads001"
            contactaddress hasValue KikaAddress
            transportBy hasValue truck
            deliveryCoverage hasValue austria
*/

postcondition
 axiom sgi13Postcondition
    nonFunctionalProperties
        dc:description hasValue "a contract of purchase for Kika for a specific piece of furniture"
    endNonFunctionalProperties
    definedBy
        exists ?PCID(?PCID memberOf xsd:integer) and
        exists ?PurchaseItem(?PurchaseItem[
            item hasValue ?PurchaseFurniture,
            provider hasValue KIKA
        ] memberOf swfmo:product) and
        exists ?PurchaseFurniture(?PurchaseFurniture memberOf furn:furniture) and
        exists ?Buyer(?Buyer[
            hasPaymentMethod hasValues {?BuyerPayment}
        ] memberOf swfmo:buyer) and
        exists ?BuyerPayment(?BuyerPayment memberOf swfmo:paymentMethod) and
        exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
        ((?Payment) implies
             ((?Payment memberOf ?BuyerPayment) and
                 (?Payment memberOf KIKA.acceptsPaymentMethod)))
        and
        ?X[
```

```
                purchaseContractID hasValue ?PCID,
                purchaseItem hasValue ?PurchaseItem,
                buyer hasValue ?Buyer,
                seller hasValue KIKA,
                purchasePayment hasValue ?Payment
        ] memberOf swfmo:purchaseContract .


effect
 axiom sgi13Effect
    nonFunctionalProperties
        dc:description hasValue "self collection of the sold piece of furniture by the buyer"
    endNonFunctionalProperties
    definedBy
        exists ?Item(?Item[
            provider hasValue KIKA
        ] memberOf swfmo:product) and
        exists ?Buyer(?Buyer[
            shipToAddress.city.inCountry hasValue austria
        ] memberOf swfmo:buyer) and
    (?X[
            deliveryItem hasValue ?Item,
            sender hasValue KIKA,
            receiver hasValue ?Buyer,
            carrier hasValue GermanParcel
        ] memberOf swfmo:dropShip
     ) or
     (?X[
            deliveryItem hasValue ?Item
           ] memberOf swfmo:selfCollection) .
```

Listing. Seller Goal Instance 2 - providing product information (WSML file)

```
/**
 * Seller Goal Instance 2
 *
 * this is the goal instance to provide product information; it is unclear who is the owner of this:
 * - if it is a single seller, then each seller needs such a Goal Instance with status open all the time
 * - it could be the Marketplace Owner, with a permanent Goal to provide product information to the buyers
 *
 * for the SWF prototype, Goal Instance are modelled as WSMO Goals,
 * and managed as 'Active WSMO Goals' in the system
 *
 *
 * the owner of a Goal Instance is defined in the corresponding FRED Goal Instance
 * the same holds for the status information
 *
 * Goal Instances do not import ontologies or use mediators;
 * all domain terminology is inherited from the Goal Template
 * Goal Instance notions only refine (= no extensions) Goal Template notions
*/


namespace <<http://swf.quarto.at/repository/ActiveGoals/122#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    kb:<<http://swf.quarto.at/repository/ontologies/kb.wsml#>>

goal <<http://swf.quarto.at/repository/ActiveGoals/122.wsml>>

nonFunctionalProperties
    dc:title hasValue "Seller Goal Instance 2"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:seller, swfmo:product, furn:furniture}
    dc:description hasValue "Goal Instance from Buyer Seller Template 2 for providing product information
            about all products available in the marketplace that have a specific type of furniture as sell item."
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
```

```
        dc:date hasValue "2004-09-30"
        dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
            comment: for the prototype, Goal Instances are described as WSMO Goals
        dc:format hasValue "text/html"
        dc:language hasValue "en-US"
        dc:relation hasValues {<<http://swf.quarto.at/repository/GoalTemplates/QueryFurnitureProviderGoal.1.wsml>>}
            comment: the value of the is specified as the Goal Template that the Goal Instance is a instance of
        dc:coverage hasValue "SWF virtual marketplace"
        dc:rights hasValue <<http://www.deri.org/privacy.html>>
        version hasValue "$Revision: 1.2 $"
    endNonFunctionalProperties


    /**
     * the notions of submission are now incorporated in the postcondontion and effects

    submission sgi2Submission
        nonFunctionalProperties
            dc:description hasValue "the information to be submitted as input to a service are:
                        [1] a type of furniture product information is required for,
                        [2] the buyer who is requesting information,
                        [3] the seller of type of sellers that product infomration is requested for"
        endNonFunctionalProperties
        definedBy
            instance FurnitureSearched memberOf furn:furniture

            instance aSeller memberOf swfmo:seller


    */


    postcondition
     axiom sgi2Postcondition
        nonFunctionalProperties
            dc:description hasValue "a contract of purchase for the Buyer for a specific piece of furniture, payment by
    creditcard"
        endNonFunctionalProperties
        definedBy
         exists ?FurnitureSearched(?FurnitureSearched memberOf furn:furniture) and
         exists ?Seller(?Seller memberOf swfmo:seller) and
         ?X[
              item hasValue ?FurnitureSearched,
              provider hasValue ?Seller
         ] memberOf swfmo:product .


    effect
     axiom sgi2Effect
        nonFunctionalProperties
            dc:description hasValue "direct delivery of the purchased piece of furniture to a specific shipping address in
    Austria"
        endNonFunctionalProperties
        definedBy
         ?X .

    comment: means there isnt an effect
```

Listing. Seller Goal Instance 3 - for a Private Seller (WSML file)

```
    /**
     * Seller Goal Instance 3 (for Private Seller)
     *
     * for the SWF prototype, Goal Instance are modelled as WSMO Goals,
     * and managed as 'Active WSMO Goals' in the system
     *
     *
     * the owner of a Goal Instance is defined in the corresponding FRED Goal Instance
     * the same holds for the status information
```

```
 *
 * Goal Instances do not import ontologies or use mediators;
 * all domain terminology is inherited from the Goal Template
 * Goal Instance notions only refine (= no extensions) Goal Template notions
*/


namespace <<http://swf.quarto.at/repository/ActiveGoals/123.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    kb:<<http://swf.quarto.at/repository/ontologies/kb.wsml#>>

goal <<http://swf.quarto.at/repository/ActiveGoals/123.wsml>>

nonFunctionalProperties
    dc:title hasValue "Seller Goal Instance 2"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {kb:doubleBed, swfmo:privateSeller, swfmo:purchaseContract, swfmo:paymentMethod,
swfmo:selfCollection}
    dc:description hasValue "Goal Instance from private seller for selling a double bed"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-21"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://swf.quarto.at/repository/GoalTemplates/SellFurniturePrivateGoal.1.wsml>>}
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.2 $"
endNonFunctionalProperties

/**
    * the notions of submission are now incorporated in the postcondontion and effects

submission sgi3Submission
    nonFunctionalProperties
        dc:description hasValue "the information to be submitted as input to a service are:
                    [1] a piece of furniture to be sold,
                    [2] private seller information (name, contact address in Austria),
                    [3] payment methods that the private seller accepts."
    endNonFunctionalProperties
    definedBy
        // refers to SWF Use Case KB f19
        instance SellFurniture memberOf furn:doubleBed
            depth hasValue 200
            width hasValue 140
            material hasValues {steel, polyester}
            color hasValue "white"

        instance UKProductPrice memberOf swfmo:price
            amount hasValue 50.00
            currency hasValue euro

        instance UKProduct memberOf swfmo:product
            item hasValue UKProduct
            price hasValue UKProductPrice
            provider hasValue UweKeller

        instance UweKeller memberOf swfmo:privateSeller
            marketplaceParticipantID hasValue "seller9876"
            name hasValue "Uwe Keller"
            email hasValue "uwe.keller@deri.org"
            telephone hasValue "+43-512-507-6468"
            fax hasValue "+43-512-507-9872"
            billToAddress hasValue UKAddress
            shipToAddress hasValue UKAddress
            acceptsPaymentMethod hasValues {UKCheck, UKCash}

        instance UKAddress memberOf loc:address
            street hasValue "Goethestrasse"
            number hasValue "10"
            city hasValue innsbruck
            zip hasValue 6020
```

```
        instance UKCheck memberOf swfmo:check
            receiver hasValue UweKeller

        instance UKCash memberOf swfmo:cash
            receiver hasValue UweKeller
*/


postcondition
 axiom sgi3Postcondition
    nonFunctionalProperties
        dc:description hasValue "a contract of purchase for the Prviate Seller for a specific piece of furniture"
    endNonFunctionalProperties
    definedBy
        exists ?PCID(?PCID memberOf xsd:integer) and
        exists ?PurchaseItem(?PurchaseItem[
            item hasValue kb:UKProduct,
            provider hasValue kb:UweKeller
        ] memberOf swfmo:product) and
        exists ?Buyer(?Buyer[
            hasPaymentMethod hasValues {?BuyerPayment}
        ] memberOf swfmo:buyer) and
        exists ?BuyerPayment(?BuyerPayment memberOf swfmo:paymentMethod) and
        exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
        ((?Payment) implies
            ((?Payment memberOf ?BuyerPayment) and
                (?Payment memberOf UweKeller.acceptsPaymentMethod)))
        and
        ?X[
                purchaseContractID hasValue ?PCID,
                purchaseItem hasValue kb:UKProduct,
                buyer hasValue ?Buyer,
                seller hasValue kb:UweKeller,
                purchasePayment hasValue ?Payment
        ] memberOf swfmo:purchaseContract .


effect
 axiom sgi3Effect
    nonFunctionalProperties
        dc:description hasValue "self collection of the sold piece of furniture by the buyer"
    endNonFunctionalProperties
    definedBy
        ?X[
            deliveryItem hasValue kb:UKProduct
        ] memberOf swfmo:selfCollection .
```

Listing. Seller Goal Instance 3.2 - for a Private Seller (WSML file)

```
/**
 * Seller Goal Instance 3.2 (for Prviate Seller)
 *
 * for the SWF prototype, Goal Instance are modelled as WSMO Goals,
 * and managed as 'Active WSMO Goals' in the system
 *
 *
 * the owner of a Goal Instance is defined in the corresponding FRED Goal Instance
 * the same holds for the status information
 *
 * Goal Instances do not import ontologies or use mediators;
 * all domain terminology is inherited from the Goal Template
 * Goal Instance notions only refine (= no extensions) Goal Template notions
*/


namespace
<<http://swf.quarto.at/repository/ActiveGoals/1232.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
```

```
        swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
        kb:<<http://swf.quarto.at/repository/ontologies/kb.wsml#>>

    goal <<http://swf.quarto.at/repository/ActiveGoals/1232.wsml>>

    nonFunctionalProperties
        dc:title hasValue "Seller Goal Instance 3.2"
        dc:creator hasValue "SWF Project"
        dc:subject hasValues {kb:bookShelf, swfmo:privateSeller, swfmo:purchaseContract, swfmo:selfCollection,
swfmo:creditCard}
        dc:description hasValue "Goal Instance from private seller for selling a book shelf"
        dc:publisher hasValue "SWF Project"
        dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
        dc:date hasValue "2004-09-30"
        dc:type hasValue <<http://www.wsmo.org/2004/d2#goals>>
        dc:format hasValue "text/html"
        dc:language hasValue "en-US"
        dc:relation hasValues {<<http://swf.quarto.at/repository/GoalTemplates/SellFurniturePrivateGoal.1.wsml>>}
        dc:coverage hasValue "SWF virtual marketplace"
        dc:rights hasValue <<http://www.deri.org/privacy.html>>
        version hasValue "$Revision: 1.3 $"
    endNonFunctionalProperties

    /**
        * the notions of submission are now incorporated in the postcondontion and effects

    submission sgi32Submission
        nonFunctionalProperties
            dc:description hasValue "the information to be submitted as input to a service are:
                        [1] a piece of furniture to be sold,
                        [2] private seller information (name, contact address in Austria),
                        [3] payment methods that the private seller accepts."
        endNonFunctionalProperties
        definedBy
            // refers to SWF Use Case KB f5
            instance aBookShelf memberOf furn:bookShelf
                depth hasValue 110
                width hasValue 30
                material hasValues {wood}
                color hasValue "brown"
                shelvsNumber hasValue 5
                isFlexible hasValue false

            instance MSBookShelfPrice memberOf swfmo:price
                amount hasValue 19.00
                currency hasValue euro

            instance MSBookShelf memberOf swfmo:product
                item hasValue aBookShelf
                price hasValue MSBookShelfPrice
                provider hasValue MichaelStollbergSeller

            instance MichaelStollbergSeller memberOf swfmo:privateSeller
                marketplaceParticipantID hasValue "buyer1234"
                name hasValue "Michael Stollberg"
                email hasValue "michael.stollberg@deri.org"
                telephone hasValue "+43-512-507-6479"
                fax hasValue "+43-512-507-9872"
                billToAddress hasValue MSSellerAddress
                shipToAddress hasValue MSSellerAddress
                acceptsPaymentMethod hasValues {MSAccCheck, MSAccCash}

            instance MSSellerAddress memberOf loc:address
                street hasValue "Landseestrasse"
                number hasValue "3a"
                city hasValue innsbruck
                zip hasValue 6020

            instance MSAccCheck memberOf swfmo:check
                receiver hasValue MichaelStollbergSeller

            instance MSAccCash memberOf swfmo:cash
                receiver hasValue MichaelStollbergSeller
    */
```

```
postcondition
 axiom sgi32Postcondition
    nonFunctionalProperties
        dc:description hasValue "a contract of purchase for the Private Seller for a specific piece of furniture"
    endNonFunctionalProperties
    definedBy
        exists ?PCID(?PCID memberOf xsd:integer) and
        exists ?PurchaseItem(?PurchaseItem[
            item hasValue kb:MSBookShelf,
            provider hasValue kb:MichaelStollbergSeller
        ] memberOf swfmo:product) and
        exists ?Buyer(?Buyer[
            hasPaymentMethod hasValues {?BuyerPayment}
        ] memberOf swfmo:buyer) and
        exists ?BuyerPayment(?BuyerPayment memberOf swfmo:paymentMethod) and
        exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
        ((?Payment) implies
            ((?Payment memberOf ?BuyerPayment) and
                (?Payment memberOf MichaelStollbergSeller.acceptsPaymentMethod)))
        and
        ?X[
                purchaseContractID hasValue ?PCID,
                purchaseItem hasValue kb:MSBookShelf,
                buyer hasValue ?Buyer,
                seller hasValue kb:MichaelStollbergSeller,
                purchasePayment hasValue ?Payment
        ] memberOf swfmo:purchaseContract .


effect
 axiom sgi32Effect
    nonFunctionalProperties
        dc:description hasValue "self collection of the sold piece of furniture by the buyer"
    endNonFunctionalProperties
    definedBy
        ?X[
            deliveryItem hasValue kb:MSBookShelf
        ] memberOf swfmo:selfCollection .
```

3.5 Services

This section specifies the SWF services available virtual marketplace of the use case, as specified above.

3.5.1 Service Ontology

This ontology defines the classes of services used within the SWF Use Case, distingishing buyer- and seller services. This ontology is used within GS Discovery for ensuring that only buyerservices are detected for buyers, and only sellerservices for sellers. The reason for defining an ontology for this distinction and not to incorporate this into the matchmaking for GS Discovery is that the intended user client for a service is not related to the object of interest matchmaking - only the latter is considered for inference-based matchmaking. The rationale for this with further explanations is provided in

Listing. Service Ontology (WSML file)

```
/**
 * SWF Use Case Service Ontology
 *
 * defines services classes, used to distingish buyer- / seller services
 */
```

```
namespace <<http://swf.quarto.at/repository/ontologies/serviceontology.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    xsd:<<http://www.w3.org/2001/XMLSchema#>>

ontology <<http://swf.quarto.at/repository/ontologies/serviceontology.wsml>>


nonFunctionalProperties
    dc:title hasValue "SWF Use Case Service Ontology "
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {"Services", "services classes"}
    dc:description hasValue "defines classes of services, to distingish services for buyers / sellers"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValues {"Michael Stollberg", "Berhard Keimel"}
    dc:date hasValue "2004-10-18"
    dc:type hasValue <<http://www.wsmo.org/2004/d2/v0.3/20040329/#ontos>>
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://swf.quarto.at/repository/services/>>}
    dc:coverage hasValue "project specific"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.2 $"
endNonFunctionalProperties


// CONCEPTS

concept swfUseCaseService
    nonFunctionalProperties
        dc:description hasValue "general concept for all SWF Services"
    endNonFunctionalProperties
    id ofType dc:identifier

concept buyerservice subConceptOf swfUseCaseService
    nonFunctionalProperties
        dc:description hasValue "concept of all buyer services"
    endNonFunctionalProperties

concept sellerservice subConceptOf swfUseCaseService
    nonFunctionalProperties
        dc:description hasValue "concept of all seller services"
    endNonFunctionalProperties


/**
 * INSTANCES
 * these are the services defined in the SWF Use Case
*/

// Buyer Services
instance buyerservice1 memberOf buyerservice
    id hasValue <<http://swf.quarto.at/repository/services/buyerservice1.wsml#>>

instance buyerservice2 memberOf buyerservice
    id hasValue <<http://swf.quarto.at/repository/services/buyerservice2.wsml#>>

instance buyerservice1 memberOf buyerservice
    id hasValue <<http://swf.quarto.at/repository/services/buyerservice3.wsml#>>


// Seller Services
instance sellerservice1 memberOf sellerservice
    id hasValue <<http://swf.quarto.at/repository/services/sellerservice1.wsml#>>

instance sellerservice2 memberOf sellerservice
    id hasValue <<http://swf.quarto.at/repository/services/sellerservice2.wsml#>>

instance sellerservice3 memberOf sellerservice
    id hasValue <<http://swf.quarto.at/repository/services/sellerservice3.wsml#>>


instance sellerserviceIKEA memberOf sellerservice
    id hasValue <<http://swf.quarto.at/repository/services/sellerserviceIKEA.wsml#>>
```

```
instance sellerserviceKika memberOf sellerservice
    id hasValue <<http://swf.quarto.at/repository/services/sellerserviceKika.wsml#>>

instance sellerserviceLeiner memberOf sellerservice
    id hasValue <<http://swf.quarto.at/repository/services/sellerserviceLeiner.wsml#>>
```

3.5.2 Buyer Services

Listing. Buyer Service 1 (WSML file)

```
/**
 * Buyer Service 1
 */


namespace <<http://http://swf.quarto.at/repository/services/buyerservice1.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    furn:<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    swfmo:<<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    loc:<<http://http://swf.quarto.at/repository/ontologies/location.wsml#>>

webservice <<http://http://swf.quarto.at/repository/services/buyergoalservice1.wsml>>

nonFunctionalProperties
    dc:title hasValue "Buyer Service 1"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:buyer, swfmo:purchaseContract, furn:furniture, swfmo:delivery}
    dc:description hasValue "general marketplace buyer service for purchasing one piece of furniture"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-20"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#webservice>>
        comment: SWF Services are modeled as WSMO Web Services
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "loc:austria"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.9 $"
endNonFunctionalProperties


importedOntologies{
    <<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}


capability bs1Capability
        precondition
            axiom bs1Precondition
                nonFunctionalProperties
                    dc:description hasValue "required input is:
                    [1] a piece of furniture to be bought,
                    [2] buyer information (name, bill address in Europe, ship address in Austria, payment information),
                    [3] a payment method (types of payment methods are predefined in the ontology)."
                endNonFunctionalProperties
                definedBy
                 ?InputItem memberOf furn:furniture
                 and ?InputBuyer[
                      markeplaceParticipantID hasValue ?ID,
                      name hasValue ?X,
                      telephone hasValue ?Y,
                      fax hasValue ?Z,
                      billToAddress hasValue ?BillToAddress,
                      shipToAddress hasValue ?ShipToAddress,
                      hasPaymentMethod hasValues {?Payment}
                 ] memberOf swfmo:buyer
                 and ?BillToAddress[
```

```
                city.inCountry hasValue austria
            ] memberOf loc:address
            and ?ShipToAddress[
                city.inCountry hasValue austria
            ] memberOf loc:address
            and ?Payment memberOf swfmo:paymentMethod .


        assumption
            axiom bs1Assumption
                nonFunctionalProperties
                    dc:description hasValue "if payment method is credit card, the credit card has to be not expired"
                endNonFunctionalProperties
                definedBy
                    (?BuyerCreditCard memberOf swfpo:creditCard) implies
                    ((currentDate.date.year < BuyerCreditCard.expYear) or
                        ((currentDate.date.year = BuyerCreditCard.expYear) and
                                ((currentDate.date.monthOfyear < BuyerCreditCard.expMonth) or
                                 (currentDate.date.monthOfyear = BuyerCreditCard.expMonth))
                        )
                    ) .


        postcondition
            axiom bs1Postcondition
                nonFunctionalProperties
                    dc:description hasValue "a contract a purchase for the piece of furniture
                        provided as input, with the buyer provided as input, and a payment method (
                        all pre-defined payment methods are accepted)"
                endNonFunctionalProperties
                definedBy
                    exists ?PCID(?PCID memberOf xsd:integer) and
                    exists ?PurchaseItem(?PurchaseItem[
                        item hasValue ?PurchaseFurniture,
                        provider hasValue ?Seller
                    ] memberOf swfmo:product) and
                    exists ?PurchaseFurniture(?PurchaseFurniture memberOf furn:furniture) and
                    exists ?Buyer(?Buyer[
                        billToAddress hasValue ?Address,
                        shipToAddress hasValue ?Address,
                        hasPaymentMethod hasValues {?BuyerPayment}
                    ] memberOf swfmo:buyer) and
                    exists ?Address(?Address[
                        city.inCountry hasValue austria
                    ] memberOf loc:address) and
                    exists ?BuyerPayment(?BuyerPayment memberOf swfmo:paymentMethod) and
                    exists ?Seller(?Seller[
                        acceptsPaymentMethod hasValues {?SellerPayment}
                    ] memberOf furn:seller) and
                    exists ?SellerPayment(?SellerPayment memberOf swfmo:paymentMethod) and
                    exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
                    ((?Payment) implies
                        ((?Payment memberOf ?BuyerPayment) and
                            (?Payment memberOf ?SellerPayment)))
                    and
                    ?X[
                        purchaseContractID hasValue ?PCID,
                        purchaseItem hasValue ?PurchaseItem,
                        buyer hasValue ?Buyer,
                        purchasePayment hasValue ?Payment
                    ] memberOf swfmo:purchaseContract .


        effect
            axiom bs1Effect
                nonFunctionalProperties
                    dc:description hasValue "delivery of the purchased piece of furniture to
                        the shipping address of the buyer as specified in the input, or via
                        self collection by the buyer"
                endNonFunctionalProperties
                definedBy
                    exists ?Item(?Item memberOf swfmo:product) and
                    exists ?Buyer(Buyer[
                        shipToAddress hasValue ?BuyerAddress
                    ] memberOf swfmo:buyer) and
                    exists ?BuyerAddress(?BuyerAddress[
```

```
                    city.inCountry hasValue austria
                ] memberOf loc:address) and
                exists ?Carrier(?Carrier[
                    deliverCoverage hasValue austria
                ] memberOf swfmo:dropShipCarrier) and
            (?X[
                    deliveryItem hasValue ?Item,
                    receiver hasValue ?Buyer,
                    carrier hasValue ?Carrier
                ] memberOf swfmo:dropShip
            )
            or
            (?X[
                    deliveryItem hasValue ?Item
                ] memberOf swfmo:selfCollection) .


interface bs1Interface
    nonFunctionalProperties
        dc:description hasValue "defines the Choreography of Buyer Service 1"
    endNonFunctionalProperties
```

Listing. Buyer Service 2 (WSML file)

```
/**
 * Buyer Service 2
 */


namespace <<http://http://swf.quarto.at/repository/services/buyerservice2.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    furn:<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    swfmo:<<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    loc:<<http://http://swf.quarto.at/repository/ontologies/location.wsml#>>

webservice <<http://http://swf.quarto.at/repository/services/buyergoalservice2.wsml>>

nonFunctionalProperties
    dc:title hasValue "Buyer Service 2"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:buyer, swfmo:product, furn:furniture}
    dc:description hasValue "gathers product information for pieces of furniture offered in the market,
        and returns them by email to the requester who is a buyer"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-30"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#webservice>>
        comment: SWF Services are modeled as WSMO Web Services
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "loc:austria"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.6 $"
endNonFunctionalProperties


importedOntologies{
    <<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}


capability bs2Capability
        precondition
            axiom bs2Precondition
                nonFunctionalProperties
                    dc:description hasValue "required input is:
                    [1] the piece of furniture that product information are requested for,
                    [2] buyer information (name, email-address),
                    [3] a seller of type of seller that should be the provider of the product."
                endNonFunctionalProperties
```

```
                definedBy
                 ?Furniture memberOf furn:furniture
                 and ?InputBuyer[
                      name hasValue ?X,
                      email hasValue ?Y
                 ] memberOf swfmo:buyer
                 and ?Seller memberOf swfmo:seller .


        assumption
            axiom bs2Assumption
                nonFunctionalProperties
                    dc:description hasValue "the buyer has to be registered as marketplace participant"
                endNonFunctionalProperties
                definedBy
                    ?Buyer[
                        marketplaceParticipantID hasValue id
                    ] memberOf swfmo:buyer .


        postcondition
            axiom bs2Postcondition
                nonFunctionalProperties
                    dc:description hasValue "product information for a specific piece of furniture
                        provided by a specific seller or seller group "
                endNonFunctionalProperties
                definedBy
                    exists ?FurnitureSearched(?FurnitureSearched memberOf furn:furniture) and
                    exists ?Provider(?Provider memberOf swfmo:seller) and
                    ?X[
                        item hasValue ?FurnitureSearched,
                        provider hasValue ?Provider
                    ] memberOf swfmo:product .


        effect
            axiom bs2Effect
                nonFunctionalProperties
                    dc:description hasValue "there is no effect"
                endNonFunctionalProperties
                definedBy
                 ?X .
        comment: this means there is no effect - however this is modelled in the end.


    interface bs2Interface
        nonFunctionalProperties
            dc:description hasValue "defines the Choreography of Buyer Service 2"
        endNonFunctionalProperties
```

Listing. Buyer Service 3 (WSML file)

```
/**
 * Buyer Service 3
 */


namespace <<http://http://swf.quarto.at/repository/services/buyerservice3.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    furn:<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    swfmo:<<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    loc:<<http://http://swf.quarto.at/repository/ontologies/location.wsml#>>

webservice <<http://http://swf.quarto.at/repository/services/buyergoalservice3.wsml>>

nonFunctionalProperties
    dc:title hasValue "Buyer Service 3"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:buyer, swfmo:purchaseContract, furn:furniture, swfmo:delivery}
    dc:description hasValue "general marketplace buyer service for purchasing one piece of
        furniture from a private seller"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-20"
```

```
    dc:type hasValue <<http://www.wsmo.org/2004/d2#webservice>>
        comment: SWF Services are modeled as WSMO Web Services
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "loc:austria"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.6 $"
endNonFunctionalProperties


importedOntologies{
    <<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}


capability bs3Capability
        precondition
            axiom bs1Precondition
                nonFunctionalProperties
                    dc:description hasValue "required input is:
                    [1] the piece of furniture to be bought,
                    [2] buyer information (name, payment information)
                    [3] a payment method (types of payment methods are predefined in the ontology)."
                endNonFunctionalProperties
                definedBy
                 ?InputItem memberOf furn:furniture
                 and ?InputBuyer[
                      name hasValue ?X,
                      telephone hasValue ?Y,
                      fax hasValue ?Z,
                      hasPaymentMethod hasValues {?Payment}
                 ] memberOf swfmo:buyer
                 and ?Payment memberOf swfmo:paymentMethod .


        assumption
            axiom bs3Assumption
                nonFunctionalProperties
                    dc:description hasValue "if payment method is credit card, the credit card has to be not expired"
                endNonFunctionalProperties
                definedBy
                    ?PrivateSeller[
                         marketplaceParticipantID hasValue ?X,
                         productLine hasValues {?Y}
                    ] memberOf swfmo:privateSeller
                    and (?InputItem implies (?InputItem memberOf ?Y)) .


        postcondition
            axiom bs3Postcondition
                nonFunctionalProperties
                    dc:description hasValue "a contract a purchase for the piece of furniture
                        provided as input, with the buyer provided as input, and a payment method (
                        all pre-defined payment methods are accepted)"
                endNonFunctionalProperties
                definedBy
                    exists ?PCID(?PCID memberOf xsd:integer) and
                    exists ?PurchaseItem(?PurchaseItem[
                        item hasValue ?PurchaseFurniture,
                        provider hasValue ?PrivateSeller
                    ] memberOf swfmo:product) and
                    exists ?PurchaseFurniture(?PurchaseFurniture memberOf furn:furniture) and
                    exists ?Buyer(?Buyer[
                        hasPaymentMethod hasValues {?BuyerPayment}
                    ] memberOf swfmo:buyer) and
                    exists ?BuyerPayment(?BuyerPayment memberOf swfmo:paymentMethod) and
                    exists ?PrivateSeller(?PrivateSeller[
                        acceptsPaymentMethod hasValues {?SellerPayment}
                    ] memberOf swfmo:privateSeller) and
                    exists ?SellerPayment(?SellerPayment memberOf swfmo:paymentMethod) and
                    exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
                    ((?Payment) implies
```

```
                    ((?Payment memberOf ?BuyerPayment) and
                        (?Payment memberOf ?SellerPayment)))
                and
            ?X[
                purchaseContractID hasValue ?PCID,
                purchaseItem hasValue ?PurchaseItem,
                buyer hasValue ?Buyer,
                seller hasValue ?PrivateSeller,
                purchasePayment hasValue ?Payment
            ] memberOf swfmo:purchaseContract .


    effect
        axiom bs3Effect
            nonFunctionalProperties
                dc:description hasValue "delivery of the purchased piece of furniture to
                    the shipping address of the buyer as specified in the input "
            endNonFunctionalProperties
            definedBy
                exists ?Item(?Item[
                    item hasValue PurchaseFurniture,
                    provider hasValue ?PrivateSeller
                ] memberOf swfmo:product) and
                exists ?PurchaseFurniture(?PurchaseFurniture memberOf furn:furniture) and
                exists ?PrivateSeller(?PrivateSeller memberOf swfmo:privateSeller) and
                ?X[
                        deliveryItem hasValue ?Item
                    ] memberOf swfmo:selfCollection .


interface bs3Interface
    nonFunctionalProperties
        dc:description hasValue "defines the Choreography of Buyer Service 1"
    endNonFunctionalProperties
```

3.5.3 Seller Services

Listing. Seller Service Leiner (WSML file)

```
/**
 * Seller Service Leiner
 */


namespace <<http://http://swf.quarto.at/repository/services/sellerserviceLeiner.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    furn:<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    swfmo:<<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    loc:<<http://http://swf.quarto.at/repository/ontologies/location.wsml#>>

webservice <<http://http://swf.quarto.at/repository/services/sellerserviceLeiner.wsml>>

nonFunctionalProperties
    dc:title hasValue "Seller Service Leiner"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:seller, swfmo:purchaseContract, furn:furniture, swfmo:delivery}
    dc:description hasValue "Leiner seller service for selling a piece of furniture"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-20"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#webservice>>
        comment: SWF Services are modeled as WSMO Web Services
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "loc:austria"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.6 $"
endNonFunctionalProperties
```

```
importedOntologies {
    <<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}


capability ssLeinerCapability
        precondition
            axiom ssLeinerPrecondition
                nonFunctionalProperties
                    dc:description hasValue "required input is:
                    [1] a piece of furniture to be sold,
                    [2] Leiner information as seller (registered marketplace participant, located in Austria, accepted
payment),
                    [3] possible payment for the purcahse contract is credit card or invoice."
                endNonFunctionalProperties
                definedBy
                    ?InputItem memberOf furn:furniture
                     and ?InputSeller[
                         marketplaceParticipantID hasValue "IKEA",
                         name hasValue "IKEA",
                         telephone hasValue "1234",
                         fax hasValue "1234",
                         contactAddress hasValue IKEAAddress,
                         acceptsPaymentMethod hasValues {?Payment}
                     ] memberOf swfmo:company
                     and IKEAAddress memberOf loc:address
                     and ((?Payment memberOf swfmo:creditcard)
                         or (?Payment memberOf swfmo:invoice)
                         or (?Payment memberOf swfmo:cash)) .


        assumption
            axiom ssLeinerAssumption
                nonFunctionalProperties
                    dc:description hasValue "the desired piece of furniture has to be available as a product offered by
the seller"
                endNonFunctionalProperties
                definedBy
                    ?Product[
                        item hasValue ?InputItem,
                        provider hasValue Leiner
                    ] memberOf swfmo:product
                    and ?InputItem memberOf furn:furniture
                    and Leiner memberOf swfmo:company .


        postcondition
            axiom ssLeinerPostcondition
                nonFunctionalProperties
                    dc:description hasValue "returns a contract of purchase for a piece
                        of furniture that fits the desire between a buyer and the seller
                        provided as input, with accepted payments credit card or invoice. "
                endNonFunctionalProperties
                definedBy
                    exists ?PCID(?PCID memberOf xsd:integer) and
                    exists ?PurchaseItem(?PurchaseItem[
                        item hasValue ?PurchaseFurniture,
                        provider hasValue LEINER
                    ] memberOf swfmo:product) and
                    exists ?PurchaseFurniture(?PurchaseFurniture memberOf furn:furniture) and
                    exists ?Buyer(?Buyer[
                        hasPaymentMethod hasValues {?BuyerPayment}
                    ] memberOf swfmo:buyer) and
                    exists ?BuyerPayment(?BuyerPayment memberOf swfmo:paymentMethod) and
                    exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
                    ((?Payment) implies
                        ((?Payment memberOf ?BuyerPayment) and
                            (?Payment memberOf LEINER.acceptsPaymentMethod)))
                    and
                    ?X[
                        purchaseContractID hasValue ?PCID,
                        purchaseItem hasValue ?PurchaseItem,
                        buyer hasValue ?Buyer,
```

```
                        seller hasValue LEINER,
                        purchasePayment hasValue ?Payment
                    ] memberOf swfmo:purchaseContract .


        effect
            axiom ssLeinerEffect
                nonFunctionalProperties
                    dc:description hasValue "delivery of the purchased piece of furniture to
                        the buyer shipping address specified in the input, or via self collection
                        by the buyer"
                endNonFunctionalProperties
                definedBy
                    exists ?Item(?Item[
                        provider hasValue LEINER
                    ] memberOf swfmo:product) and
                    exists ?Buyer(?Buyer[
                        shipToAddress.city.inCountry hasValue austria
                    ] memberOf swfmo:buyer) and
                    (?X[
                        deliveryItem hasValue ?Item,
                        sender hasValue LEINER,
                        receiver hasValue ?Buyer,
                        carrier hasValue LeinerDeliveryService
                    ] memberOf swfmo:dropShip)
                or
                    (?X[
                        deliveryItem hasValue ?Item
                    ] memberOf swfmo:selfCollection) .


interface ssLeinerInterface
    nonFunctionalProperties
        dc:description hasValue "defines the Choreography of Seller Service 1"
    endNonFunctionalProperties
```

Listing. Seller Service Kika (WSML file)

```
/**
 * Seller Service Kika
 */


namespace <<http://http://swf.quarto.at/repository/services/sellerserviceKika.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    furn:<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    swfmo:<<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    loc:<<http://http://swf.quarto.at/repository/ontologies/location.wsml#>>

webservice <<http://http://swf.quarto.at/repository/services/sellerserviceKika.wsml>>

nonFunctionalProperties
    dc:title hasValue "Seller Service Kika"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:seller, swfmo:purchaseContract, furn:furniture, swfmo:delivery}
    dc:description hasValue "Kika seller service for selling a piece of furniture"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-20"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#webservice>>
        comment: SWF Services are modeled as WSMO Web Services
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "loc:austria"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.6 $"
endNonFunctionalProperties
```

```
importedOntologies {
    <<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}


capability ssKikaCapability
        precondition
            axiom ssKikaPrecondition
                nonFunctionalProperties
                    dc:description hasValue "required input is:
                    [1] a piece of furniture to be sold,
                    [2] Kika information as seller (registered marketplace participant, located in Austria, accepted
payment),
                    [3] possible payment for the purcahse contract is credit card or invoice."
                endNonFunctionalProperties
                definedBy
                    ?InputItem memberOf furn:furniture
                     and ?InputSeller[
                         marketplaceParticipantID hasValue "Kika",
                         name hasValue "Kika",
                         telephone hasValue "2345",
                         fax hasValue "2345",
                         contactAddress hasValue KikaAddress,
                         acceptsPaymentMethod hasValues {?Payment}
                     ] memberOf swfmo:company
                     and KikaAddress memberOf loc:address
                     and ((?Payment memberOf swfmo:creditcard)
                         or (?Payment memberOf swfmo:invoice)
                         or (?Payment memberOf swfmo:cash)) .


        assumption
            axiom ssKikaAssumption
                nonFunctionalProperties
                    dc:description hasValue "the desired piece of furniture has to be available as a product offered by
the seller"
                endNonFunctionalProperties
                definedBy
                    ?Product[
                        item hasValue ?InputItem,
                        provider hasValue Kika
                    ] memberOf swfmo:product
                    and ?InputItem memberOf furn:furniture
                    and Kika memberOf swfmo:company .


        postcondition
            axiom ssKikaPostcondition
                nonFunctionalProperties
                    dc:description hasValue "returns a contract of purchase for a piece
                        of furniture that fits the desire between a buyer and the seller
                        provided as input, with accepted payments credit card or invoice. "
                endNonFunctionalProperties
                definedBy
                    exists ?PCID(?PCID memberOf xsd:integer) and
                    exists ?PurchaseItem(?PurchaseItem[
                        item hasValue ?PurchaseFurniture,
                        provider hasValue KIKA
                    ] memberOf swfmo:product) and
                    exists ?PurchaseFurniture(?PurchaseFurniture memberOf furn:furniture) and
                    exists ?Buyer(?Buyer[
                        hasPaymentMethod hasValues {?BuyerPayment}
                    ] memberOf swfmo:buyer) and
                    exists ?BuyerPayment(?BuyerPayment memberOf swfmo:paymentMethod) and
                    exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
                    ((?Payment) implies
                        ((?Payment memberOf ?BuyerPayment) and
                            (?Payment memberOf KIKA.acceptsPaymentMethod)))
                    and
                    ?X[
                        purchaseContractID hasValue ?PCID,
                        purchaseItem hasValue ?PurchaseItem,
                        buyer hasValue ?Buyer,
```

```
                        seller hasValue KIKA,
                        purchasePayment hasValue ?Payment
                    ] memberOf swfmo:purchaseContract .


        effect
            axiom ssKikaEffect
                nonFunctionalProperties
                    dc:description hasValue "delivery of the purchased piece of furniture to
                        the buyer shipping address specified in the input, or via self collection
                        by the buyer"
                endNonFunctionalProperties
                definedBy
                    exists ?Item(?Item[
                        provider hasValue KIKA
                    ] memberOf swfmo:product) and
                    exists ?Buyer(?Buyer[
                        shipToAddress.city.inCountry hasValue austria
                    ] memberOf swfmo:buyer) and
                    (?X[
                        deliveryItem hasValue ?Item,
                        sender hasValue KIKA,
                        receiver hasValue ?Buyer,
                        carrier hasValue GermanParcel
                    ] memberOf swfmo:dropShip)
                or
                 (?X[
                        deliveryItem hasValue ?Item
                    ] memberOf swfmo:selfCollection) .


interface ssKikaInterface
    nonFunctionalProperties
        dc:description hasValue "defines the Choreography of Seller Service 1"
    endNonFunctionalProperties
```

Listing. Seller Service IKEA (WSML file)

```
/**
 * Seller Service IKEA
 */


namespace <<http://http://swf.quarto.at/repository/services/sellerserviceIKEA.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    furn:<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    swfmo:<<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    loc:<<http://http://swf.quarto.at/repository/ontologies/location.wsml#>>

webservice <<http://http://swf.quarto.at/repository/services/sellerserviceIKEA.wsml>>

nonFunctionalProperties
    dc:title hasValue "Seller Service IKEA"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:seller, swfmo:purchaseContract, furn:furniture, swfmo:delivery}
    dc:description hasValue "IKEA seller service for selling a piece of furniture"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-20"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#webservice>>
        comment: SWF Services are modeled as WSMO Web Services
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "loc:austria"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.6 $"
```

```
endNonFunctionalProperties


importedOntologies {
    <<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}


capability sslKEACapability
        precondition
            axiom sslKEAPrecondition
                nonFunctionalProperties
                    dc:description hasValue "required input is:
                    [1] a piece of furniture to be sold,
                    [2] IKEA information as seller (registered marketplace participant, located in Austria, accepted
payment),
                    [3] possible payment for the purchase contract is credit card or invoice."
                endNonFunctionalProperties
                definedBy
                    ?InputItem memberOf furn:furniture
                     and ?InputSeller[
                         marketplaceParticipantID hasValue "IKEA",
                         name hasValue "IKEA",
                         telephone hasValue "3456",
                         fax hasValue "3456",
                         contactAddress hasValue IKEAAddress,
                         acceptsPaymentMethod hasValues {?Payment}
                     ] memberOf swfmo:company
                     and IKEAAddress memberOf loc:address
                     and ((?Payment memberOf swfmo:creditcard)
                         or (?Payment memberOf swfmo:invoice)
                         or (?Payment memberOf swfmo:cash)) .


        assumption
            axiom sslKEAAssumption
                nonFunctionalProperties
                    dc:description hasValue "the desired piece of furniture has to be available as a product offered by
the seller"
                endNonFunctionalProperties
                definedBy
                    ?Product[
                        item hasValue ?InputItem,
                        provider hasValue IKEA
                    ] memberOf swfmo:product
                    and ?InputItem memberOf furn:furniture
                    and IKEA memberOf swfmo:company .


        postcondition
            axiom sslKEAPostcondition
                nonFunctionalProperties
                    dc:description hasValue "returns a contract of purchase for a piece
                        of furniture that fits the desire between a buyer and the seller
                        provided as input, with accepted payments credit card or invoice. "
                endNonFunctionalProperties
                definedBy
                    exists ?PCID(?PCID memberOf xsd:integer) and
                    exists ?PurchaseItem(?PurchaseItem[
                        item hasValue ?PurchaseFurniture,
                        provider hasValue IKEA
                    ] memberOf swfmo:product) and
                    exists ?PurchaseFurniture(?PurchaseFurniture memberOf furn:furniture) and
                    exists ?Buyer(?Buyer[
                        hasPaymentMethod hasValues {?BuyerPayment}
                    ] memberOf swfmo:buyer) and
                    exists ?BuyerPayment(?BuyerPayment memberOf swfmo:paymentMethod) and
                    exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
                    ((?Payment) implies
                        ((?Payment memberOf ?BuyerPayment) and
                            (?Payment memberOf IKEA.acceptsPaymentMethod)))
                    and
                    ?X[
                        purchaseContractID hasValue ?PCID,
                        purchaseItem hasValue ?PurchaseItem,
```

```
                        buyer hasValue ?Buyer,
                        seller hasValue IKEA,
                        purchasePayment hasValue ?Payment
                    ] memberOf swfmo:purchaseContract .


        effect
            axiom sslKEAEffect
                nonFunctionalProperties
                    dc:description hasValue "delivery of the purchased piece of furniture to
                        the buyer shipping address specified in the input, or via self collection
                        by the buyer"
                endNonFunctionalProperties
                definedBy
                    exists ?Item(?Item[
                        provider hasValue IKEA
                    ] memberOf swfmo:product) and
                    exists ?Buyer(?Buyer[
                        shipToAddress.city.inCountry hasValue austria
                    ] memberOf swfmo:buyer) and
                    (?X[
                        deliveryItem hasValue ?Item,
                        sender hasValue IKEA,
                        receiver hasValue ?Buyer,
                        carrier hasValue IKEADeliveryService
                    ] memberOf swfmo:dropShip)
                or
                 (?X[
                        deliveryItem hasValue ?Item
                    ] memberOf swfmo:selfCollection) .


 interface sslKEAInterface
     nonFunctionalProperties
            dc:description hasValue "defines the Choreography of Seller Service 1"
     endNonFunctionalProperties
```

Listing. Seller Service 1 (WSML file)

```
/**
 * Seller Service 1
 */


namespace <<http://http://swf.quarto.at/repository/services/sellerservice1.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    furn:<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    swfmo:<<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    loc:<<http://http://swf.quarto.at/repository/ontologies/location.wsml#>>

webservice <<http://http://swf.quarto.at/repository/services/sellerservice1.wsml>>

nonFunctionalProperties
    dc:title hasValue "Seller Service 1"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:seller, swfmo:purchaseContract, furn:furniture, swfmo:delivery}
    dc:description hasValue "general marketplace seller service for selling a piece of furniture"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-20"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#webservice>>
        comment: SWF Services are modeled as WSMO Web Services
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "loc:austria"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
```

```
    version hasValue "$Revision: 1.9 $"
endNonFunctionalProperties


importedOntologies {
    <<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}


capability ss1Capability
        precondition
            axiom ss1Precondition
                nonFunctionalProperties
                    dc:description hasValue "required input is:
                    [1] a piece of furniture to be sold,
                    [2] seller information (registered marketplace participant, located in Austria, accepted payment),
                    [3] possible payment for the purcahse contract is credit card or invoice."
                endNonFunctionalProperties
                definedBy
                 ?InputItem memberOf furn:furniture
                 and ?InputSeller[
                      marketplaceParticipantID hasValue ?X1,
                      name hasValue ?X2,
                      telephone hasValue ?X3,
                      fax hasValue ?X4,
                      contactAddress hasValue ?ContactAddress,
                      acceptsPaymentMethod hasValues {?Payment}
                 ] memberOf swfmo:seller
                 and ?ContactAddress[
                      city.inCountry hasValue ?Y
                 ] memberOf loc:address
                 and ?Y[
                      inContinent hasValue europe
                 ] memberOf loc:country
                 and ((?Payment memberOf swfmo:creditcard)
                              or (?Payment memberOf swfmo:invoice)) .


        assumption
            axiom ss1Assumption
                nonFunctionalProperties
                    dc:description hasValue "the desired piece of furniture has to be available as a product offered by
the seller"
                endNonFunctionalProperties
                 definedBy
                    ?Product[
                        item hasValue ?InputItem,
                        provider hasValue ?InputSeller
                    ] memberOf swfmo:product
                    and ?InputItem memberOf furn:furniture
                    and ?InputSeller memberOf swfmo:seller .


        postcondition
            axiom ss1Postcondition
                nonFunctionalProperties
                    dc:description hasValue "returns a contract of purchase for a piece
                         of furniture that fits the desire between a buyer and the seller
                         provided as input, with accepted payments credit card or invoice. "
                endNonFunctionalProperties
                definedBy
                    exists ?PCID(?PCID memberOf xsd:integer) and
                    exists ?PurchaseItem(?PurchaseItem[
                        item hasValue ?PurchaseFurniture,
                        provider hasValue ?Seller
                    ] memberOf swfmo:product) and
                    exists ?PurchaseFurniture(?PurchaseFurniture memberOf furn:furniture) and
                    exists ?Buyer(?Buyer[
                        hasPaymentMethod hasValues {?BuyerPayment}
                    ] memberOf swfmo:buyer) and
                    exists ?BuyerPayment(?BuyerPayment memberOf swfmo:paymentMethod) and
                    exists ?Seller(?Seller[
                         contactAddress hasValue ?Address,
                         acceptsPaymentMethod hasValues {?SellerPayment}
                    ] memberOf swfmo:seller) and
```

```
                    exists ?Address(?Address[
                        city.inCountry hasValue austria
                    ] memberOf loc:address) and
                    exists ?SellerPayment(?SellerPayment memberOf swfmo:paymentMethod) and
                    exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
                    ((?Payment) implies
                        ((?Payment memberOf ?BuyerPayment) and
                            (?Payment memberOf ?SellerPayment)))
                    and
                    ?X[
                        purchaseContractID hasValue ?PCID,
                        purchaseItem hasValue ?PurchaseItem,
                        seller hasValue ?Seller,
                        purchasePayment hasValue ?Payment
                    ] memberOf swfmo:purchaseContract .


        effect
            axiom ss1Effect
                nonFunctionalProperties
                    dc:description hasValue "delivery of the purchased piece of furniture to
                        the shipping address of the buyer specified in the input, alternatively self collection by buyer"
                endNonFunctionalProperties
                definedBy
                    exists ?Item(?Item memberOf swfmo:product) and
                    exists ?Seller(?Seller[
                        contactAddress hasValue ?ContactAddress
                    ] memberOf swfmo:seller) and
                    exists ?ContactAddress(?ContactAddress[
                        city.inCountry hasValue austria
                    ] memberOf loc:address) and
                    exists ?Buyer(Buyer[
                        shipToAddress hasValue ?BuyerAddress
                    ] memberOf swfmo:buyer) and
                    exists ?BuyerAddress(?BuyerAddress[
                        city.inCountry hasValue austria
                    ] memberOf loc:address) and
                    exists ?Carrier(?Carrier[
                        deliverCoverage hasValue austria
                    ] memberOf swfmo:dropShipCarrier) and
                (?X[
                        deliveryItem hasValue ?Item,
                        sender hasValue ?Seller,
                        receiver hasValue ?Buyer,
                        carrier hasValue ?Carrier
                    ] memberOf swfmo:dropShip
                )
                or
                (?X[
                        deliveryItem hasValue ?Item
                    ] memberOf swfmo:selfCollection) .


    interface ss1Interface
        nonFunctionalProperties
            dc:description hasValue "defines the Choreography of Seller Service 1"
        endNonFunctionalProperties
```

Listing. Seller Service 2 (WSML file)

```
/**
 * Seller Service 2
 */


namespace <<http://http://swf.quarto.at/repository/services/sellerservice2.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    furn:<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    swfmo:<<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    loc:<<http://http://swf.quarto.at/repository/ontologies/location.wsml#>>
```

```
webservice <<http://http://swf.quarto.at/repository/services/sellerservice2.wsml>>

nonFunctionalProperties
    dc:title hasValue "Seller Service 2"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:seller, swfmo:product, furn:furniture}
    dc:description hasValue "general marketplace seller service for providing product information for specific types of
furniture"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-30"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#webservice>>
        comment: SWF Services are modeled as WSMO Web Services
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "loc:austria"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.6 $"
endNonFunctionalProperties


importedOntologies {
    <<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}


capability ss2Capability
        precondition
            axiom ss2Precondition
                nonFunctionalProperties
                    dc:description hasValue "required input is:
                    [1] the piece of furniture that product information are requested for,
                    [2] the seller or type of seller (e.g. private seller only) whose offers to be searched for."
                endNonFunctionalProperties
                definedBy
                 ?InputItem memberOf furn:furniture
                 and ?InputSeller memberOf swfmo:seller .


        assumption
            axiom ss2Assumption
                nonFunctionalProperties
                    dc:description hasValue "the desired piece of furniture has to be available as a product offered by
the seller"
                endNonFunctionalProperties
                definedBy
                ?Product[
                    item hasValue ?InputItem,
                    provider hasValue ?InputSeller
                ] memberOf swfmo:product
                and ?InputItem memberOf furn:furniture
                and ?InputSeller memberOf swfmo:seller .


        postcondition
            axiom ss2Postcondition
                nonFunctionalProperties
                    dc:description hasValue "provides detailed product information for a piece of furniture
                        offered by a specific seller or seller group"
                endNonFunctionalProperties
                definedBy
                    exists ?FurnitureSearched(?FurnitureSearched memberOf furn:furniture) and
                    exists ?Provider(?Provider memberOf swfmo:seller) and
                    ?X[
                        item hasValue ?FurnitureSearched,
                        provider hasValue ?Provider
                    ] memberOf swfmo:product .


        effect
            axiom ss2Effect
```

```
                nonFunctionalProperties
                    dc:description hasValue "no effect"
                endNonFunctionalProperties
                definedBy
                 ?X .
                 comment: means there is no effect


 interface ss2Interface
     nonFunctionalProperties
             dc:description hasValue "defines the Choreography of Seller Service 1"
     endNonFunctionalProperties
```

Listing. Seller Service 3 (WSML file)

```
/**
 * Seller Service 3
 */


namespace <<http://http://swf.quarto.at/repository/services/sellerservice3.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    furn:<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    swfmo:<<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>
    loc:<<http://http://swf.quarto.at/repository/ontologies/location.wsml#>>

webservice <<http://http://swf.quarto.at/repository/services/sellerservice3.wsml>>

nonFunctionalProperties
    dc:title hasValue "Seller Service 3"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:privateSeller, swfmo:purchaseContract, furn:furniture, swfmo:delivery}
    dc:description hasValue "general marketplace seller service for selling a piece of furniture
        for private sellers only"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-20"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#webservice>>
        comment: SWF Services are modeled as WSMO Web Services
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "loc:austria"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.6 $"
endNonFunctionalProperties


importedOntologies {
    <<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
    <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}


capability ss3Capability
        precondition
            axiom ss3Precondition
                nonFunctionalProperties
                    dc:description hasValue "required input is:
                    [1] a piece of furniture to be sold,
                    [2] private seller information (registered marketplace participant, located in Austria, accepted
payment),
                    [3] possible payment for the purcahse contract is credit card or invoice."
                endNonFunctionalProperties
                definedBy
                 ?InputItem memberOf furn:furniture
                 and ?InputSeller[
                     marketplaceParticipantID hasValue ?X1,
```

```
                    name hasValue ?X2,
                    telephone hasValue ?X3,
                    fax hasValue ?X4,
                    contactAddress hasValue ?ContactAddress,
                    acceptsPaymentMethod hasValues {?SellerPayment}
                ] memberOf swfmo:privateSeller
                and ?ContactAddress[
                    city.inCountry hasValue austria
                ] memberOf loc:address
                and ?Payment memberOf swfmo:paymentMethod
                and ((?Payment) implies (?Payment memberOf ?SellerPayment)) .


        assumption
            axiom ss3Assumption
                nonFunctionalProperties
                    dc:description hasValue "the desired piece of furniture has to be available as a product offered by
the seller"
                endNonFunctionalProperties
                definedBy
                    ?Product[
                        item hasValue ?InputItem,
                        provider hasValue ?PrivateSeller
                    ] memberOf swfmo:product
                    and ?InputItem memberOf furn:furniture
                    and ?PrivateSeller memberOf PrivateSeller .


        postcondition
            axiom ss3Postcondition
                nonFunctionalProperties
                    dc:description hasValue "returns a contract of purchase for a piece
                        of furniture that fits the desire between a buyer and the seller
                        provided as input, with accepted payments credit card or invoice. "
                endNonFunctionalProperties
                definedBy
                    exists ?PCID(?PCID memberOf xsd:integer) and
                    exists ?PurchaseItem(?PurchaseItem[
                        item hasValue ?PurchaseFurniture,
                        provider hasValue ?PrivateSeller
                    ] memberOf swfmo:product) and
                    exists ?PurchaseFurniture(?PurchaseFurniture memberOf furn:furniture) and
                    exists ?Buyer(?Buyer[
                        hasPaymentMethod hasValues {?BuyerPayment}
                    ] memberOf swfmo:buyer) and
                    exists ?BuyerPayment(?BuyerPayment memberOf swfmo:paymentMethod) and
                    exists ?PrivateSeller(?PrivateSeller[
                        contactAddress hasValue ?SellerAddress,
                        acceptsPaymentMethod hasValues {?SellerPayment}
                    ] memberOf swfmo:privateSeller) and
                    exists ?SellerAddress(?SellerAddress[
                        city.inCountry hasValue austria
                    ] memberOf loc:address) and
                    exists ?SellerPayment(?SellerPayment memberOf swfmo:paymentMethod) and
                    exists ?Payment(?Payment memberOf swfmo:paymentMethod) and
                    ((?Payment) implies
                        ((?Payment memberOf ?BuyerPayment) and
                            (?Payment memberOf ?SellerPayment)))
                    and
                ?X[
                    purchaseContractID hasValue ?PCID,
                    purchaseItem hasValue ?PurchaseItem,
                    buyer hasValue ?Buyer,
                    seller hasValue ?PrivateSeller,
                    purchasePayment hasValue ?Payment
                ] memberOf swfmo:purchaseContract .


        effect
            axiom ss3Effect
                nonFunctionalProperties
                    dc:description hasValue "delivery of the purchased piece of furniture to
                        the shipping address of the buyer as specified in the input "
                endNonFunctionalProperties
                definedBy
                    exists ?Item(?Item memberOf swfmo:product) and
```

```
                    ?X[
                        deliveryItem hasValue ?Item
                    ] memberOf swfmo:selfCollection .


interface ss3Interface
    nonFunctionalProperties
        dc:description hasValue "defines the Choreography of Seller Service 1"
    endNonFunctionalProperties
```

3.6 Mediators

This section specifies the WSMO Mediators used with the SWF Use Case

3.6.1 GG Mediators

WSMO GG Mediators used with the SWF Use Case.

NOTE: WSMO GG Mediators are under construction; currently, the notion of "Reduction" is not for GG Mediators. The following models are intended as a suggestion on how to use Reduction within GG Mediators.

Listing. GG Mediator 1 (WSML file)

```
/**
 * ggMediator between BGS2, BGS1
*/

namespace <<http://http://swf.quarto.at/repository/mediators/ggMediator1.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    bgs1:<<http://http://swf.quarto.at/repository/GoalTemplates/buyergoalschema1.wsml>>
    bgs2:<<http://http://swf.quarto.at/repository/GoalTemplates/buyergoalschema2.wsml>>

ggMediator <<http://http://swf.quarto.at/repository/mediators/ggMediator1.wsml>>

nonFunctionalProperties
    dc:title hasValue "GG Mediator between BGS2, BGS1"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:buyer, swfmo:paymentMethod}
    dc:description hasValue "states that BGS1 is derived from BGS2"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-20"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#mediators>>
        comment: described as WSMO-Mediators
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
            <<http://http://swf.quarto.at/ontologies/swfmo.wsml>>,
            <<http://http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.3 $"
endNonFunctionalProperties

comment: no imported ontologies / used mediators as these are
inherited from the source and target

source <<http://http://swf.quarto.at/repository/GoalTemplates/buyergoalschema2.wsml>>

target <<http://http://swf.quarto.at/repository/GoalTemplates/buyergoalschema1.wsml>>

reduction
```

```
axiom ggM1Redcution
    nonFunctionalProperties
        dc:description hasValue "restricts the paymentMethod to creditcard"
    endNonFunctionalProperties
    definedBy
    (?X memberOf swfmo:paymentMethod
            and bgs2:postcondition.purchasecontract[
                ?PurchasePayment hasValue ?X
            ] memberOf swfmo:purchaseContract)
     implies
    (?Y memberOf swfmo:creditcard
            and bgs1:postcondition.purchasecontract[
                ?PurchasePayment hasValue ?Y
            ] memberOf swfmo:purchaseContract) .
```

Listing. GG Mediator 2 (WSML file)

```
/**
 * GGMediator for Cooperative Goal 1
*/

namespace <<http://swf.quarto.at/repository/mediators/cooperativegoal1ggMediator.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    bgs1:<<http://swf.quarto.at/repository/GoalTemplates/buyergoalschema1.wsml>>
    sgs1:<<http://swf.quarto.at/repository/GoalTemplates/sellergoalschema1.wsml>>

ggMediator <<http://swf.quarto.at/repository/mediators/cooperativegoal1ggMediator.wsml>>

nonFunctionalProperties
    dc:title hasValue "GG Mediator for Cooperative Goal 1"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {swfmo:buyer, swfmo:paymentMethod}
    dc:description hasValue "restricts paymentMethod to credit card payment only"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma, Uwe Keller"
    dc:date hasValue "2004-09-20"
    dc:type hasValue <<http://www.wsmo.org/2004/d2#mediators>>
        comment: described as WSMO-Mediators
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
            <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
            <<http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.3 $"
endNonFunctionalProperties

comment: no imported ontologies / used mediators as these are inherited from the source and target

source <<http://swf.quarto.at/repository/GoalTemplates/buyergoalschema1.wsml>>

target <<http://swf.quarto.at/repository/GoalTemplates/sellergoalschema1.wsml>>

reduction
 axiom cg1Redcution
    nonFunctionalProperties
        dc:description hasValue "restricts the paymentMethod to creditcard"
    endNonFunctionalProperties
    definedBy
    ((?X memberOf swfmo:creditCard
            and bgs1:postcondition.purchasecontract[
                ?PurchasePayment hasValue ?X
            ] memberOf swfmo:purchaseContract)
     and
     (?Y memberOf swfmo:paymentMethod
            and sgs1:postcondition.purchasecontract[
                ?PurchasePayment hasValue ?Y
            ] memberOf swfmo:purchaseContract))
    implies (?Y memberOf swfmo:creditcard) .
```

3.6 SWF Use Case Knowledge Base

This section pre-defines instances for the SWF Use Case.

Listing. SWF Use Case Knowledge Base (WSML file)

```
/**
 * SWF Use Case Knowledge Base
 * this is a collection of pre-defined data for the SWF Use Case
*/

namespace <<http://swf.quarto.at/repository/ontologies/kb.wsml#>>
    dc:<<http://purl.org/dc/elements/1.1#>>
    wsml:<<http://www.wsmo.org/2004/d16/d16.1/v0.2/20040418#>>
    furn:<<http://swf.quarto.at/repository/ontologies/furniture.wsml#>>
    loc:<<http://swf.quarto.at/repository/ontologies/location.wsml#>>
    swfmo:<<http://swf.quarto.at/repository/ontologies/swfmo.wsml#>>

ontology <<http://swf.quarto.at/repository/ontologies/kb.wsml>>

nonFunctionalProperties
    dc:title hasValue "SWF Use Case Knowledge Base"
    dc:creator hasValue "SWF Project"
    dc:subject hasValues {furn:furniture, swfmo:marketplaceParticipant, swfmo:product, swfmo:paymentMethod,
loc:address, loc:city, loc:country}
    dc:description hasValue "describes pre-defined instance data for the SWF Use Case"
    dc:publisher hasValue "SWF Project"
    dc:contributor hasValue "Michael Stollberg, Ioan Toma"
    dc:date hasValue "2004-10-13"
    dc:type hasValue <<http://www.wsmo.org/2004/d2/v0.3/20040329/#ontos>>
    dc:format hasValue "text/html"
    dc:language hasValue "en-US"
    dc:relation hasValues {<<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
    <<http://swf.quarto.at/repository/ontologies/location.wsml>>}
    dc:coverage hasValue "SWF virtual marketplace"
    dc:rights hasValue <<http://www.deri.org/privacy.html>>
    version hasValue "$Revision: 1.7 $"
endNonFunctionalProperties

importedOntologies {
    <<http://swf.quarto.at/repository/ontologies/furniture.wsml>>,
    <<http://swf.quarto.at/repository/ontologies/swfmo.wsml>>,
    <<http://swf.quarto.at/repository/ontologies/location.wsml>>}


/**
 * Furniture
 */

/*****************************************************************************************
 * Material instances
*****************************************************************************************/
instance wood memberOf furn:material

instance silk memberOf furn:material

instance down memberOf furn:material

instance polyester memberOf furn:material

instance polywood memberOf furn:material

instance veneer memberOf furn:material

instance cotton memberOf furn:material
```

```
instance leather memberOf furn:material

instance foil memberOf furn:material

instance glass memberOf furn:material

instance lacquer memberOf furn:material

/*****************************************************************************************
 * BED instances
*****************************************************************************************/
instance f2 memberOf furn:singleBed
            depth hasValue 200
            width hasValue 90
            material hasValues {wood}
            color hasValue "black"

instance f18 memberOf furn:singleBed
            depth hasValue 200
            width hasValue 100
            material hasValues {wood, polyester}
            color hasValue "white"

instance f19 memberOf furn:doubleBed
            depth hasValue 200
            width hasValue 140
            material hasValues {steel, polyester}
            color hasValue "white"

instance f20 memberOf furn:kingSizeBed
            depth hasValue 250
            width hasValue 200
            material hasValues {steel, polyester}
            color hasValue "blue"

/*****************************************************************************************
 * STORAGE FURNITURE instances
*****************************************************************************************/
instance f1 memberOf furn:storageFurniture
            material hasValues {wood}

/**
 * CABINET instances
 */
instance f15 memberOf furn:cabinet
            depth hasValue 195
            width hasValue 89
            height hasValue 46
            material hasValues {veneer, lacquer, glass}
            color hasValue "brown"
            shelvsNumber hasValue 6
            isFlexible hasValue false

instance f16 memberOf furn:cabinet
            depth hasValue 110
            width hasValue 100
            height hasValue 40
            material hasValues {veneer, lacquer, wood, glass}
            color hasValue "brown"
            shelvsNumber hasValue 4
            isFlexible hasValue false

/**
 * CHEST instances
 */
instance f17 memberOf furn:chest
            depth hasValue 100
            width hasValue 80
            height hasValue 40
            material hasValues {wood, polyester}
            color hasValue "gray"
            storageCapacity hasValue 100

/**
 * SHELF instances
```

```
 */
instance f5 memberOf furn:bookShelf
        depth hasValue 110
        width hasValue 30
        material hasValues {wood}
        color hasValue "brown"
        shelvsNumber hasValue 5
        isFlexible hasValue false

instance f13 memberOf furn:shelf
        depth hasValue 93
        width hasValue 20
        height hasValue 66
        material hasValues {foil}
        color hasValue "white"
        shelvsNumber hasValue 2
        isFlexible hasValue false

instance f14 memberOf furn:shelf
        depth hasValue 69
        width hasValue 42
        height hasValue 30
        material hasValues {wood}
        color hasValue "brown"
        shelvsNumber hasValue 4
        isFlexible hasValue false


/*************************************************************************************************
 * SEAT FURNITURE instances
*************************************************************************************************/
/**
 * CHAIR instances
 */
instance f6 memberOf furn:chair
        depth hasValue 49
        width hasValue 42
        height hasValue 90
        material hasValues {wood}
        color hasValue "red"
        numberOfLegs hasValue 4

instance f7 memberOf furn:chair
        depth hasValue 56
        width hasValue 51
        height hasValue 85
        material hasValues {steel, polyester, polywood}
        color hasValue "silver"
        numberOfLegs hasValue 4

instance f8 memberOf furn:chair
        depth hasValue 81
        width hasValue 58
        height hasValue 53
        material hasValues {lacquer, veneer}
        color hasValue "blue"
        numberOfLegs hasValue 4

/*
 * ARMCHAIR instances
 */
instance f4 memberOf furn:armChair
        depth hasValue 110
        width hasValue 100
        material hasValues {silk, down, wood}
        color hasValue "magneta"
        numberOfLegs hasValue 4

instance f9 memberOf furn:armChair
        depth hasValue 72
        width hasValue 80
        height hasValue 78
        material hasValues {wood, cotton, poyester}
        color hasValue "white"
        numberOfLegs hasValue 4
```

```
instance f12 memberOf furn:chaislounge
        depth hasValue 91
        width hasValue 76
        height hasValue 41
        material hasValues {cotton, polyester, wood}
        color hasValue "black"

/**
 * SOFA isntances
 */
instance f10 memberOf furn:sofaBed
        depth hasValue 219
        width hasValue 96
        height hasValue 84
        material hasValues {wood, cotton, poyester}
        color hasValue "red"
        numberOfLegs hasValue 4

instance f11 memberOf furn:clubSofa
        depth hasValue 180
        width hasValue 88
        height hasValue 69
        material hasValues {leather, steel}
        color hasValue "blak"
        numberOfLegs hasValue 4

/*****************************************************************************************************
 * TABLE instances
*****************************************************************************************************/
instance diningTable memberOf furn:table
instance kitchenTable memberOf furn:table

instance f21 memberOf furn:pedestalTable
        depth hasValue 60
        width hasValue 60
        height hasValue 63
        material hasValues {polyester}
        color hasValue "white"

instance f22 memberOf furn:table
        depth hasValue 135
        width hasValue 74
        height hasValue 74
        material hasValues {wood}
        color hasValue "gray"
        numberOfLegs hasValue 4

/**
 * DESK isntances
 */
instance f3 memberOf furn:desk
        depth hasValue 120
        width hasValue 90
        material hasValues {wood}
        color hasValue "black"
        numberOfLegs hasValue 4


/**
 * Products
 */
/*****************************************************************************************************
 * Products from private sellers
*****************************************************************************************************/
instance UKProduct memberOf swfmo:product
        item hasValue f19
        price hasValue UKProductPrice
        provider hasValue UweKeller
instance UKProductPrice memberOf swfmo:price
        amount hasValue 50.00
        currency hasValue euro

instance MSBookShelf memberOf swfmo:product
        item hasValue f5
        price hasValue MSBookShelfPrice
        provider hasValue MichaelStollbergSeller
```

```
instance MSBookShelfPrice memberOf swfmo:price
        amount hasValue 19.00
        currency hasValue euro

/*****************************************************************************************
 * Product Catalogue IKEA
*****************************************************************************************/
/*
 * IKEA Chairs
 */
instance STEFANChair memberOf swfmo:product
        item hasValue f6
        price hasValue IKEAPrice1
        provider hasValue IKEA
instance IKEAPrice1 memberOf swfmo:price
        amount hasValue 19.99
        currency hasValue euro

instance SIXTENChair memberOf swfmo:product
        item hasValue f7
        price hasValue IKEAPrice2
        provider hasValue IKEA
instance IKEAPrice2 memberOf swfmo:price
        amount hasValue 49
        currency hasValue euro

instance TRASSENTChair memberOf swfmo:product
        item hasValue f8
        price hasValue IKEAPrice3
        provider hasValue IKEA
instance IKEAPrice3 memberOf swfmo:price
        amount hasValue 99
        currency hasValue euro

/*
 * IKEA ArmChairs
 */
instance TULLSTAArmChair memberOf swfmo:product
        item hasValue f9
        price hasValue IKEAPrice4
        provider hasValue IKEA
instance IKEAPrice4 memberOf swfmo:price
        amount hasValue 99
        currency hasValue euro

/*
 * IKEA Chaislong
 */
instance TYLOSANDChaislongue memberOf swfmo:product
        item hasValue f12
        price hasValue IKEAPrice5
        provider hasValue IKEA
instance IKEAPrice5 memberOf swfmo:price
        amount hasValue 389
        currency hasValue euro

/*
 * IKEA Sofas
 */
instance NILSBYSofaBed memberOf swfmo:product
        item hasValue f10
        price hasValue IKEAPrice6
        provider hasValue IKEA
instance IKEAPrice6 memberOf swfmo:price
        amount hasValue 259
        currency hasValue euro

instance KILPPANSofaBed memberOf swfmo:product
        item hasValue f11
        price hasValue IKEAPrice7
        provider hasValue IKEA
instance IKEAPrice7 memberOf swfmo:price
        amount hasValue 499
        currency hasValue euro

/*
```

```
 * IKEA Shelf
 */
instance ROBINShelf memberOf swfmo:product
        item hasValue f13
        price hasValue IKEAPrice8
        provider hasValue IKEA
instance IKEAPrice8 memberOf swfmo:price
        amount hasValue 19.99
        currency hasValue euro

instance SLAKTShelf memberOf swfmo:product
        item hasValue f14
        price hasValue IKEAPrice9
        provider hasValue IKEA
instance IKEAPrice9 memberOf swfmo:price
        amount hasValue 19.99
        currency hasValue euro

/*
 * IKEA Cabinets
 */
instance FORSHEDCabinet memberOf swfmo:product
        item hasValue f15
        price hasValue IKEAPrice10
        provider hasValue IKEA
instance IKEAPrice10 memberOf swfmo:price
        amount hasValue 379.00
        currency hasValue euro

instance ROSFORSCabinet memberOf swfmo:product
        item hasValue f16
        price hasValue IKEAPrice11
        provider hasValue IKEA
instance IKEAPrice11 memberOf swfmo:price
        amount hasValue 299.00
        currency hasValue euro

/*
 * IKEA Chests
 */
instance ANEBODAChest memberOf swfmo:product
        item hasValue f17
        price hasValue IKEAPrice12
        provider hasValue IKEA
instance IKEAPrice12 memberOf swfmo:price
        amount hasValue 69.00
        currency hasValue euro

/*
 * IKEA Beds
 */
instance TOVIKSingleBed memberOf swfmo:product
        item hasValue f18
        price hasValue IKEAPrice13
        provider hasValue IKEA
instance IKEAPrice13 memberOf swfmo:price
        amount hasValue 99.00
        currency hasValue euro

instance NORESUNDDoubleBed memberOf swfmo:product
        item hasValue f19
        price hasValue IKEAPrice14
        provider hasValue IKEA
instance IKEAPrice14 memberOf swfmo:price
        amount hasValue 199.00
        currency hasValue euro

instance MORKEDALKingSizeBed memberOf swfmo:product
        item hasValue f20
        price hasValue IKEAPrice15
        provider hasValue IKEA
instance IKEAPrice15 memberOf swfmo:price
        amount hasValue 299.00
        currency hasValue euro

/*
```

```
 * IKEA Tables
 */
instance SANDSKARSPedestalTable memberOf swfmo:product
        item hasValue f21
        price hasValue IKEAPrice16
        provider hasValue IKEA
instance IKEAPrice16 memberOf swfmo:price
        amount hasValue 29.00
        currency hasValue euro

instance NORDENTable memberOf swfmo:product
        item hasValue f22
        price hasValue IKEAPrice17
        provider hasValue IKEA
instance IKEAPrice17 memberOf swfmo:price
        amount hasValue 79.00
        currency hasValue euro


/*****************************************************************************************
 * Product Catalogue LEINER
*****************************************************************************************/
/*
 * LEINER Chairs
 */
instance LEIChair memberOf swfmo:product
        item hasValue f6
        price hasValue LEINERPrice1
        provider hasValue LEINER
instance LEINERPrice1 memberOf swfmo:price
        amount hasValue 24.99
        currency hasValue euro

instance SENTChair memberOf swfmo:product
        item hasValue f8
        price hasValue LEINERPrice2
        provider hasValue LEINER
instance LEINERPrice2 memberOf swfmo:price
        amount hasValue 80
        currency hasValue euro

/*
 * LEINER ArmChairs
 */
instance MASEArmChair memberOf swfmo:product
        item hasValue f9
        price hasValue LEINERPrice3
        provider hasValue LEINER
instance LEINERPrice3 memberOf swfmo:price
        amount hasValue 120
        currency hasValue euro

/*
 * LEINER Chaislong
 */
instance MITAChaislongue memberOf swfmo:product
        item hasValue f12
        price hasValue LEINERPrice4
        provider hasValue LEINER
instance LEINERPrice4 memberOf swfmo:price
        amount hasValue 200
        currency hasValue euro

/*
 * LEINER Sofas
 */
instance DEMETSofaBed memberOf swfmo:product
        item hasValue f10
        price hasValue LEINERPrice5
        provider hasValue LEINER
instance LEINERPrice5 memberOf swfmo:price
        amount hasValue 250
        currency hasValue euro

/*
 * LEINER Shelf
```

```
 */
instance HOODShelf memberOf swfmo:product
        item hasValue f13
        price hasValue LEINERPrice6
        provider hasValue LEINER
instance LEINERPrice6 memberOf swfmo:price
        amount hasValue 35.00
        currency hasValue euro

/*
 * LEINER Cabinets
 */
instance MINISTCabinet memberOf swfmo:product
        item hasValue f15
        price hasValue LEINERPrice7
        provider hasValue LEINER
instance LEINERPrice7 memberOf swfmo:price
        amount hasValue 400.00
        currency hasValue euro

instance FORSCabinet memberOf swfmo:product
        item hasValue f16
        price hasValue LEINERPrice8
        provider hasValue LEINER
instance LEINERPrice8 memberOf swfmo:price
        amount hasValue 250.00
        currency hasValue euro

/*
 * LEINER Chests
 */
instance ABAChest memberOf swfmo:product
        item hasValue f17
        price hasValue LEINERPrice9
        provider hasValue LEINER
instance LEINERPrice9 memberOf swfmo:price
        amount hasValue 100.00
        currency hasValue euro

/*
 * LEINER Beds
 */
instance VIKISingleBed memberOf swfmo:product
        item hasValue f18
        price hasValue LEINERPrice10
        provider hasValue LEINER
instance LEINERPrice10 memberOf swfmo:price
        amount hasValue 200.00
        currency hasValue euro

instance HENRYKingSizeBed memberOf swfmo:product
        item hasValue f20
        price hasValue LEINERPrice11
        provider hasValue LEINER
instance LEINERPrice11 memberOf swfmo:price
        amount hasValue 350.00
        currency hasValue euro

/*
 * LEINER Tables
 */
instance SANDSKARSPedestalTable memberOf swfmo:product
        item hasValue f21
        price hasValue LEINERPrice12
        provider hasValue LEINER
instance LEINERPrice12 memberOf swfmo:price
        amount hasValue 35.00
        currency hasValue euro

instance DEDALTable memberOf swfmo:product
        item hasValue f22
        price hasValue LEINERPrice13
        provider hasValue LEINER
instance LEINERPrice13 memberOf swfmo:price
        amount hasValue 90.00
        currency hasValue euro
```

```
/*********************************************************************************************
 * Product Catalogue KIKA
 *********************************************************************************************/
/*
 * KIKA Chairs
 */
instance FANChair memberOf swfmo:product
        item hasValue f6
        price hasValue KIKAPrice1
        provider hasValue KIKA
instance KIKAPrice1 memberOf swfmo:price
        amount hasValue 30.99
        currency hasValue euro

instance TRENTChair memberOf swfmo:product
        item hasValue f8
        price hasValue KIKAPrice2
        provider hasValue KIKA
instance KIKAPrice2 memberOf swfmo:price
        amount hasValue 89
        currency hasValue euro

/*
 * KIKA ArmChairs
 */
instance KIKArmChair memberOf swfmo:product
        item hasValue f9
        price hasValue KIKAPrice3
        provider hasValue KIKA
instance KIKAPrice3 memberOf swfmo:price
        amount hasValue 79
        currency hasValue euro

/*
 * KIKA Chaislong
 */
instance TITIChaislongue memberOf swfmo:product
        item hasValue f12
        price hasValue KIKAPrice4
        provider hasValue KIKA
instance KIKAPrice4 memberOf swfmo:price
        amount hasValue 500
        currency hasValue euro

/*
 * KIKA Sofas
 */
instance TOMSofaBed memberOf swfmo:product
        item hasValue f10
        price hasValue KIKAPrice5
        provider hasValue KIKA
instance KIKAPrice5 memberOf swfmo:price
        amount hasValue 300
        currency hasValue euro


/*
 * KIKA Shelf
 */
instance MAHOShelf memberOf swfmo:product
        item hasValue f14
        price hasValue KIKAPrice6
        provider hasValue KIKA
instance KIKAPrice6 memberOf swfmo:price
        amount hasValue 39.99
        currency hasValue euro

/*
 * KIKA Cabinets
 */
instance SHEDCabinet memberOf swfmo:product
        item hasValue f15
        price hasValue KIKAPrice7
        provider hasValue KIKA
```

```
instance KIKAPrice7 memberOf swfmo:price
        amount hasValue 250.00
        currency hasValue euro

instance FORSCabinet memberOf swfmo:product
        item hasValue f16
        price hasValue KIKAPrice8
        provider hasValue KIKA
instance KIKAPrice8 memberOf swfmo:price
        amount hasValue 319.00
        currency hasValue euro

/*
 * KIKA Chests
 */
instance TODAChest memberOf swfmo:product
        item hasValue f17
        price hasValue KIKAPrice9
        provider hasValue KIKA
instance KIKAPrice9 memberOf swfmo:price
        amount hasValue 99.00
        currency hasValue euro

/*
 * KIKA Beds
 */
instance VIKINGSingleBed memberOf swfmo:product
        item hasValue f18
        price hasValue KIKAPrice10
        provider hasValue KIKA
instance KIKAPrice10 memberOf swfmo:price
        amount hasValue 109.00
        currency hasValue euro

instance SUNDDDoubleBed memberOf swfmo:product
        item hasValue f19
        price hasValue KIKAPrice11
        provider hasValue KIKA
instance KIKAPrice11 memberOf swfmo:price
        amount hasValue 220.00
        currency hasValue euro

/*
 * KIKA Tables
 */
instance SAMPTable memberOf swfmo:product
        item hasValue f22
        price hasValue KIKAPrice12
        provider hasValue KIKA
instance KIKAPrice12 memberOf swfmo:price
        amount hasValue 69.00
        currency hasValue euro

/*****************************************************************************************************
 * Marketplace Participants
*****************************************************************************************************/
/*
 * Ioan Toma      - buyer
 */
    instance IoanToma memberOf swfmo:buyer
        marketplaceParticipantID hasValue "buyer9876"
        name hasValue "Ioan Toma"
        email hasValue "ioan.toma@deri.org"
        telephone hasValue "+43-512-507-6461"
        fax hasValue "+43-512-507-9872"
        billToAddress hasValue ITAddress
        shipToAddress hasValue ITAddress
        hasPaymentMethod hasValues {ITCreditCard, ITCash, ITCheck}

    instance ITAddress memberOf loc:address
        street hasValue "Technikerstrasse"
        number hasValue "13"
        city hasValue innsbruck
        zip hasValue 6020

    instance ITCreditCard memberOf swfmo:creditCard
```

```
        type hasValue Visa
        number hasValue "237671"
        holder hasValue "Ioan Toma"
        expMonth hasValue 06
        expYear hasValue 2006

    instance ITCash memberOf swfmo:cash
        payer hasValue IoanToma

    instance ITCheck memberOf swfmo:check
        drawer hasValue IoanToma
        drawerAccount hasValue ITAccount

    instance ITAccount memberOf swfmo:account
        bank hasValue HypoTirol
        owner hasValue "Ioan Toma"
        accountNumber hasValue 17111345

    instance HypoTirol memberOf swfmo:bank
        bankName hasValue "Hypo Bank Tirol"
        bankAddress hasValue HypoAddress
        bankIdentifierCode hasValue "57000"

    instance HypoAddress memberOf loc:address
        street hasValue "Am Innrain"
        number hasValue "48"
        city hasValue innsbruck
        zip hasValue 6020

/*
 * Dieter Fensel      - buyer
 */
    instance DieterFensel memberOf swfmo:buyer
        marketplaceParticipantID hasValue "buyer65245"
        name hasValue "Dieter Fensel"
        email hasValue "dieter.fensel@deri.org"
        telephone hasValue "+43-512-507-6488"
        fax hasValue "+43-512-507-9872"
        billToAddress hasValue DFAddress
        shipToAddress hasValue DFAddress

    instance DFAddress memberOf loc:address
        street hasValue "Technikerstrasse"
        number hasValue "13"
        city hasValue innsbruck
        zip hasValue 6020

/*
 * Michael Stollberg - buyer
 */
    instance MichaelStollberg memberOf swfmo:buyer
        marketplaceParticipantID hasValue "buyer1234"
        name hasValue "Michael Stollberg"
        email hasValue "michael.stollberg@deri.org"
        telephone hasValue "+43-512-507-6479"
        fax hasValue "+43-512-507-9872"
        billToAddress hasValue MSAddress
        shipToAddress hasValue MSAddress
        hasPaymentMethod hasValues {MSCreditCard, MSCash}

    instance MSAddress memberOf loc:address
        street hasValue "Technikerstrasse"
        number hasValue "13"
        city hasValue innsbruck
        zip hasValue 6020

    instance MSCreditCard memberOf swf:creditCard
        type hasValue MasterCard
        number hasValue "123456789"
        holder hasValue "Michael Stollberg"
        expMonth hasValue 08
        expYear hasValue 2007

    instance MSCash memberOf swfmo:cash
        payer hasValue MichaelStollberg
```

```
/*
 * Michael Stollberg - private seller
 */
        instance MichaelStollbergSeller memberOf swfmo:privateSeller
            marketplaceParticipantID hasValue "seller1234"
            name hasValue "Michael Stollberg"
            email hasValue "michael.stollberg@deri.org"
            telephone hasValue "+43-512-507-6479"
            fax hasValue "+43-512-507-9872"
            contactAddress hasValue MSSellerAddress
            acceptsPaymentMethod hasValues {MSAccCheck, MSAccCash}

        instance MSSellerAddress memberOf loc:address
            street hasValue "Landseestrasse"
            number hasValue "3a"
            city hasValue innsbruck
            zip hasValue 6020

        instance MSAccCheck memberOf swfmo:check
            receiver hasValue MichaelStollbergSeller

        instance MSAccCash memberOf swfmo:cash
            receiver hasValue MichaelStollbergSeller

/*
 * Uwe Keller - private seller
 */
        instance UweKeller memberOf swfmo:privateSeller
            marketplaceParticipantID hasValue "seller9876"
            name hasValue "Uwe Keller"
            email hasValue "uwe.keller@deri.org"
            telephone hasValue "+43-512-507-6468"
            fax hasValue "+43-512-507-9872"
            contactAddress hasValue UKAddress
            hasPaymentMethod hasValues {UKCheck, UKCash}

        instance UKAddress memberOf loc:address
            street hasValue "Goethestrasse"
            number hasValue "10"
            city hasValue innsbruck
            zip hasValue 6020

        instance UKCheck memberOf swfmo:check
            receiver hasValue UweKeller

        instance UKCash memberOf swfmo:cash
            receiver hasValue UweKeller

/*
 * IKEA - seller company
 */
        instance IKEA memberOf swfmo:company
            marketplaceParticipantID hasValue "sellerIKEA"
            name hasValue "IKEA"
            email hasValue "office@ikea.at"
            telephone hasValue "+43-1-123456"
            fax hasValue "+43-1-123456"
            contactAddress hasValue IKEAAddress
            acceptsPaymentMethod hasValues {IKEACreditCard, IKEAInvoice, IKEACash}
            companyNumber hasValue "xyz"
            website hasValue "www.ikea.at"

        instance IKEAAddress memberOf loc:address
            street hasValue "am DEZ"
            number hasValue "3"
            city hasValue innsbruck
            zip hasValue 6020

        instance IKEACreditCard memberOf swf:creditCard
            type hasValue ?IKEAAcceptedCreditCards

            ?IKEAAcceptedCreditCards hasValue MasterCard
            ?IKEAAcceptedCreditCards hasValue Visa
        comment: means IKEA accepts mastercard and visa as credit cards

        instance IKEAInvoice memberOf swf:invoice
```

```
            receiver hasValue IKEA

        instance IKEACash memberOf swf:cash
            receiver hasValue IKEA

    /*
     * LEINER - seller company
     */
        instance LEINER memberOf swfmo:company
            marketplaceParticipantID hasValue "sellerLEINER"
            name hasValue "Leiner"
            email hasValue "office@leiner.at"
            telephone hasValue "+43-662-63970"
            fax hasValue "+43-662-620843"
            contactAddress hasValue LeinerAddress
            acceptsPaymentMethod hasValues {LEINERCreditCard, LEINERInvoice, LEINERCash}
            companyNumber hasValue "abc"
            website hasValue "www.leiner.at"

        instance LEINERAddress memberOf loc:address
            street hasValue "Alpenstraße"
            number hasValue "1"
            city hasValue salzburgStadt
            zip hasValue 5020

        instance LEINERCreditCard memberOf swf:creditCard
            type hasValue ?LEINERAcceptedCreditCards
            ?LEINERAcceptedCreditCards hasValue MasterCard
            ?LEINERAcceptedCreditCards hasValue Visa
        comment: means LEINER accepts mastercard and visa as credit cards

        instance LEINERInvoice memberOf swf:invoice
            receiver hasValue LEINER

        instance LEINERCash memberOf swf:cash
            receiver hasValue LEINER

    /*
     * KIKA - seller company
     */
        instance KIKA memberOf swfmo:company
            marketplaceParticipantID hasValue "sellerKika"
            name hasValue "Kika"
            email hasValue "office@kika.at"
            telephone hasValue "+43-1-2033539"
            fax hasValue "+43-1-2033539"
            contactAddress hasValue KIKAAddress
            acceptsPaymentMethod hasValues {KIKACreditCard, KIKAInvoice, KIKACash}
            companyNumber hasValue "kika"
            website hasValue "www.kika.at"

        instance KIKAAddress memberOf loc:address
            street hasValue "Donaustadtstraße"
            number hasValue "1"
            city hasValue wienStadt
            zip hasValue 1220

        instance KIKACreditCard memberOf swf:creditCard
            type hasValue ?KIKAAcceptedCreditCards
            ?KIKAAcceptedCreditCards hasValue MasterCard
            ?KIKAAcceptedCreditCards hasValue Visa
        comment: means KIKA accepts mastercard and visa as credit cards

        instance KIKAInvoice memberOf swf:invoice
            receiver hasValue KIKA

        instance KIKACash memberOf swf:cash
            receiver hasValue KIKA


    /*********************************************************************************************
     * Locations
    *********************************************************************************************/
        instance europe memberOf loc:continent
            name hasValue "Europe"
```

```
        instance austria memberOf loc:country
            name hasValue "Austria"
            inContinent hasValue europe
            isoCode hasValue "AT"

        instance germany memberOf loc:country
            name hasValue "Germany"
            inContinent hasValue europe
            isoCode hasValue "DE"

    /*
     * the 9 States of Austria
     */
        instance burgenland memberOf loc:state
            name hasValue "Burgenland"
            inCountry hasValue austria

        instance kaernten memberOf loc:state
            name hasValue "Kaernten"
            inCountry hasValue austria

        instance niederoesterreich memberOf loc:state
            name hasValue "Niederoesterreich"
            inCountry hasValue austria

        instance oberoesterreich memberOf loc:state
            name hasValue "Oberoesterreich"
            inCountry hasValue austria

        instance salzburg memberOf loc:state
            name hasValue "salzburg"
            inCountry hasValue austria

        instance steiermark memberOf loc:state
            name hasValue "Steiermark"
            inCountry hasValue austria

        instance tirol memberOf loc:state
            name hasValue "Tirol"
            inCountry hasValue austria

        instance vorarlberg memberOf loc:state
            name hasValue "Vorarlberg"
            inCountry hasValue austria

        instance wien memberOf loc:state
            name hasValue "Wien"
            inCountry hasValue austria

    /*
     * pre-defined Austrian cities
     */
        instance innsbruck memberOf loc:city
            name hasValue "Innsbruck"
            inCountry hasValue autria
            inState hasValue tirol

        instance salzburgStadt memberOf loc:city
            name hasValue "Salzburg"
            inCountry hasValue autria
            inState hasValue salzburg

        instance munich memberOf loc:city
            name hasValue "Muenchen"
            inCountry hasValue germany

        instance wienStadt memberOf loc:city
            name hasValue "Wien"
            inCountry hasValue autria
            inState hasValue wien

    /*
     * Drop Ship Deliverers
     */
    instance GermanParcel memberOf swfmo:dropShipCarrier
                name hasValue "German Parcel"
```

```
        companyNumber hasValue "gp"
        contactaddress hasValue GPAddress
        transportBy hasValue truck
        deliveryCoverage hasValue europe

instance GPAddress memberOf loc:address
        street hasValue "Franz-Joseph-Strasse"
        number hasValue "17"
        city hasValue munich
        zip hasValue 80801

instance IKEADeliveryService memberOf swfmo:dropShipCarrier
        name hasValue "IKEA Delivery Service"
        companyNumber hasValue "IKEAds001"
        contactaddress hasValue IKEAAddress
        transportBy hasValue truck
        deliveryCoverage hasValue austria

instance LeinerDeliveryService memberOf swfmo:dropShipCarrier
        name hasValue "Leiner Delivery Service"
        companyNumber hasValue "Leinerds001"
        contactaddress hasValue LeinerAddress
        transportBy hasValue truck
        deliveryCoverage hasValue austria

// transportation means

instance truck memberOf swfmo:transportationMean

instance train memberOf swfmo:transportationMean

instance plane memberOf swfmo:transportationMean

/*
 * currencies - only Euro, as the markeptlace is limited to Austria
 */
instance euro memberOf swfmo:curreny
 name hasValue "Euro"
 code hasValue "EUR"

/*
 * payment methods
 */
instance MasterCard memberOf swfmo:creditCardType

instance Visa memberOf swfmo:creditCardType

instance AmercianExpress memberOf swfmo:creditCardType
```

4. SWF Discoveres: Matchmaking Overview

A major asset of the use case specified above is realistic cooperations of Freds can be determined by matchmaking within the SWF Discoverers. The following gives a brief overview of the architecture of the discoverers and explains how the separate components are realized. Furthermore, it provides overviews of the intended matchmaking for establishing reasonable cooperations, and the testing results of distinct matchmakers which "prove" that the modeling of the resources is correct on the one hand, and, on the other, that the matchmakers work correctly. The SWF compoments for GG Discovery, GS Discovery, and WW Discovery that work on the resources specified in the use case are specified in [SWF Deliverable D4]. That deliverable specifies the SWF architecture and the specific discoverers in detail; these issues - which are the far more interesting ones - are omitted from this document with respect to length and readability.

4.1 GG Discoverer

As the first step in the cooperative goal resolution, GG Discovery detects potential cooperation partners by determining the compatibility of their respective Goal Instances. The compatibility is given when the objects of interest (i.e. the Goal postconditions) match, and when the cooperation roles of the Goal Instance owners are compatible.

Figure 2 shows the structure of GG Discovery. A *Cooperative Goal* defines compatible Goal Templates, thus specifying compatible cooperation roles and constituting a pre-selection of possible cooperation partners at design time. Optionally, a WSMO GG Mediator can be defined that resolves mismatches between Goal Schemas that are not compatible a priori. For checking the compatibility of the objects of interest of Goal Instances created from compatible Goal Templates, the ontology objects defined in the Goal Instances have to be either the same or be a superset or subset of each other. The matchmaking between Goal Instances is realized in the FOL-theorem-prover VAMPIRE [Riazanov and Voronkov, 2002] by checking whether the Goal Instance of the initiating Fred (Fred *A* is the figure) logically entails the Goal Instances of Goal Schemas that are compatible to the one of Fred *A*.

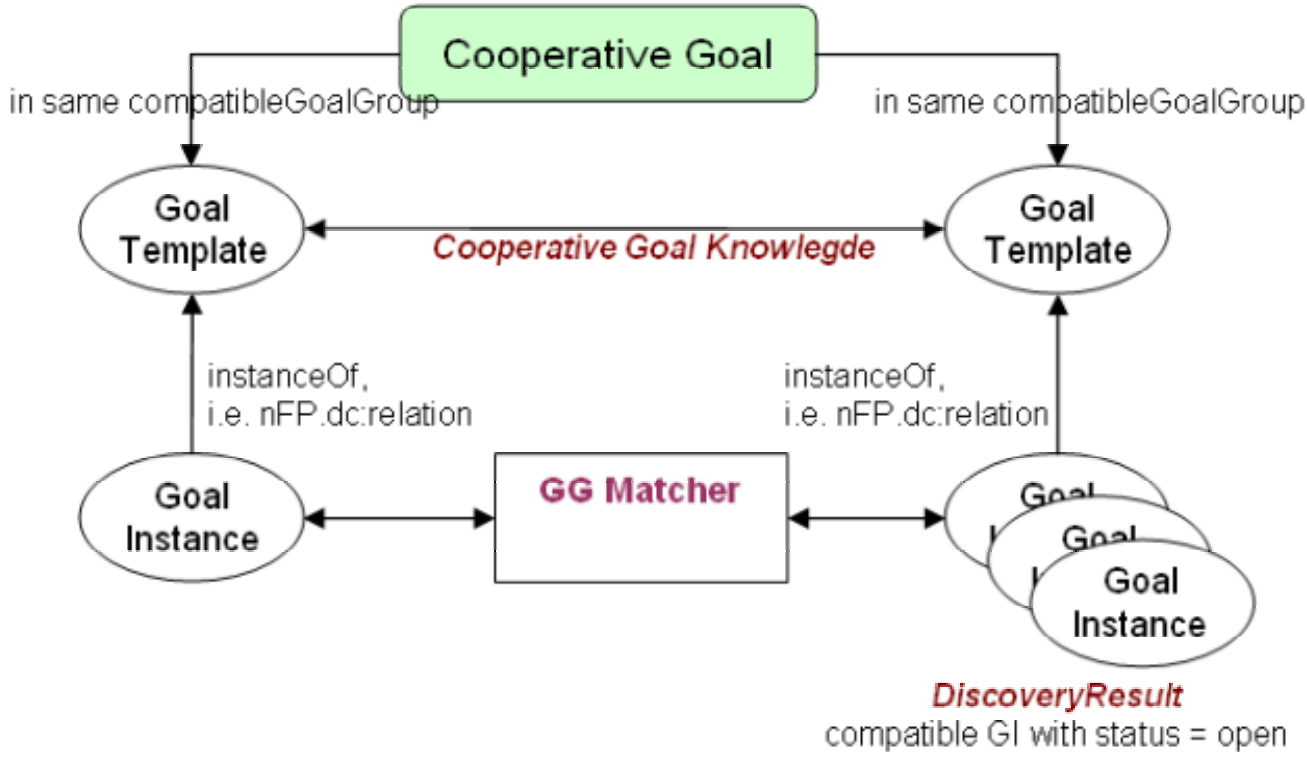

Figure 2. GG Discovery Overview

The Cooperative Goal Knowledge is modelled within the Cooperative Goal Ontology (see Section 3.3); this ontology is queried via an API for, so that all Goal Templates that are in the same 'compatibleGoalGroup' are retrieved. For all Goal Instances that are instanceOf a Goal Template that is compatible to the Goal Template of the Goal Instance that is provided within the DiscoveryRequest, the object of interest has to be

matched. This is done within the GGMatcher, which determines the final DiscoveryResult of the GG Discoverer.

The following table shows the matching resources that are required to establish reasonable cooperations. We use the fowolling abbreviations for the table:

- GT = Goal Template
- GI = Goal Instance
- S = Service
- for a 'Buyer Resource', the resource has a prefix 'B'; for a 'Seller Resource' the prefix 'S'.

for Goal Instances, we put the file name in brackets as well, according to the resources in Section 3.4.

Table: GGMatcher requested matches

Resources	BGI 1 (111)	BGI 2 (112)	BGI 3 (113)	BGI 4 (114)	SGI IKEA (121)	SGI LEINER (1212)	SGI KIKA (1213)	SGI 2 (122)	SGI priv. 1 (123)	SGI priv. 2 (1232)
BGI 1 (111)					X	X	X			
BGI 2 (112)					X	X	X		X	
BGI 3 (113)								X		
BGI 4 (114)										X
SGI IKEA (121)	X	X								
SGI LEINER (1212)	X	X								
SGI KIKA (1213)	X	X								
SGI 2(122)			X							
SGI priv. 1 (123)		X								
SGI priv. 2 (1232)				X						

According to these requirements for reasonable cooperations as a GG Discovery Result, we include the test scripts for realization of the GGMatcher in Vampire which show resources, the matching result togther with the evaluation according to the requirements specified in the table above. The first file is for so called 'purchase goals', and the second one for so called 'query goals'; GI 3 (113) and SGI 2 (122) are query goals (dealing with product information), while all others are purchase goals.

As different matching patterns are used, we need different test files. These testscripts have been run & executed with Vampire version 6 as available in the SWF CVS / bin/ (Linux version only).

Listing. GG Matcher Results for Purchase Goals

```
%------------------------------------------------------------------------
% File     : testGGMatcherPC.p
% Domain   : SWF Use Case GS Discovery Preselector Matchmaking Test
% Problem  : tests all combinations for Purchasing Goals
% Version  : 1.2
%------------------------------------------------------------------------

include('../ontologies/onto.ax').
include('../ontologies/kb.ax').
include('../ontologies/universe.ax').

% Buyer Goal Instance 1
%include('../goals/goalinstances/111.ax').
% all Seller Goal Instances
%include('../goals/goalinstances/121.ax').      %YES    CORRECT
%include('../goals/goalinstances/1212.ax').     %YES    CORRECT
%include('../goals/goalinstances/1213.ax').     %YES    CORRECT
%include('../goals/goalinstances/122.ax').      %NO     CORRECT
%include('../goals/goalinstances/123.ax').      %NO     CORRECT
%include('../goals/goalinstances/1232.ax').     %NO     CORRECT

% Buyer Goal Instance 2
%include('../goals/goalinstances/112.ax').
% all Seller Goal Instances
%include('../goals/goalinstances/121.ax').      %YES    CORRECT
%include('../goals/goalinstances/1212.ax').     %YES    CORRECT
%include('../goals/goalinstances/1213.ax').     %YES    CORRECT
%include('../goals/goalinstances/122.ax').      %NO     CORRECT
%include('../goals/goalinstances/123.ax').      %YES    CORRECT
%include('../goals/goalinstances/1232.ax').     %NO     CORRECT


% Buyer Goal Instance 4
%include('../goals/goalinstances/114.ax').
% all Seller Goal Instances
%include('../goals/goalinstances/121.ax').      %NO     CORRECT
%include('../goals/goalinstances/1212.ax').     %NO     CORRECT
%include('../goals/goalinstances/1213.ax').     %NO     CORRECT
%include('../goals/goalinstances/122.ax').      %NO     CORRECT
%include('../goals/goalinstances/123.ax').      %NO     CORRECT
%include('../goals/goalinstances/1232.ax').     %YES    CORRECT


%--------------------------------
%testing of seller gola intances matching not needed because of Intersection Match
%--------------------------------

% Seller Goal Instance 1
%include('../goals/goalinstances/121.ax').      %YES    CORRECT
% all Buyer Goal Instances
%include('../goals/goalinstances/111.ax').
%include('../goals/goalinstances/112.ax').
%include('../goals/goalinstances/113.ax').
%include('../goals/goalinstances/114.ax').

% Seller Goal Instance 1.2
%include('../goals/goalinstances/1212.ax').
% all Buyer Goal Instances
%include('../goals/goalinstances/111.ax').
%include('../goals/goalinstances/112.ax').
%include('../goals/goalinstances/113.ax').
%include('../goals/goalinstances/114.ax').

% Seller Goal Instance 1.3
%include('../goals/goalinstances/1213.ax').
% all Buyer Goal Instances
%include('../goals/goalinstances/111.ax').
%include('../goals/goalinstances/112.ax').
%include('../goals/goalinstances/113.ax').
```

```
%include('../goals/goalinstances/114.ax').

% Seller Goal Instance 3
%include('../goals/goalinstances/123.ax').
% all Buyer Goal Instances
%include('../goals/goalinstances/111.ax').
%include('../goals/goalinstances/112.ax').
%include('../goals/goalinstances/113.ax').
%include('../goals/goalinstances/114.ax').

% Seller Goal Instance 3.2
%include('../goals/goalinstances/1232.ax').
% all Buyer Goal Instances
%include('../goals/goalinstances/111.ax').
%include('../goals/goalinstances/112.ax').
%include('../goals/goalinstances/113.ax').
%include('../goals/goalinstances/114.ax').


%----------------------------------------------------------------------------
% Proof Obligation GG Discoverer for PurchaseContracts Goal Instances:
% - Intersection for postconditions
% - Intersection for effects

input_formula(po, conjecture,(
? [X] : ( goalpostcondition(X) & goalpostcondition2(X) ) &
? [Y] : ( goaleffect(Y) & goaleffect2(Y) )
)).
%----------------------------------------------------------------------------
```

4.2 GS Discoverer

The second step in the resolution process is GS Discovery which detects suitable services for automated goal resolution separately for each cooperation partner identified in GG Discovery. Analogous to Web Service Discovery, GS Discovery returns a set of services that a partner can use for automated goal resolution and thus realizes the WSMO approach for Web Service Discovery [Keller et al., 2004].

Figure 3 shows the structure of GS Discovery for Fred *A* as a potential cooperation partner. The proof obligation for matchmaking between the Goal Instance of Fred *A* and Service Capabilities as functional descriptions of available Services is that under consideration of all Ontologies and Mediators used in the Goal and the Capability description, if the submission of the Goal Instance satisfies the precondition and assumption of the Capability, and if the Goal Instance postcondition implies the Capability postcondition as well as if the Goal effects imply the Capability effects, then that the Service matches the Goal. For resolving partial Goal-Capability matches into exact matches, a WSMO WG Mediator defines the required reduction similar to the usage of GG Mediators in GG Discovery.

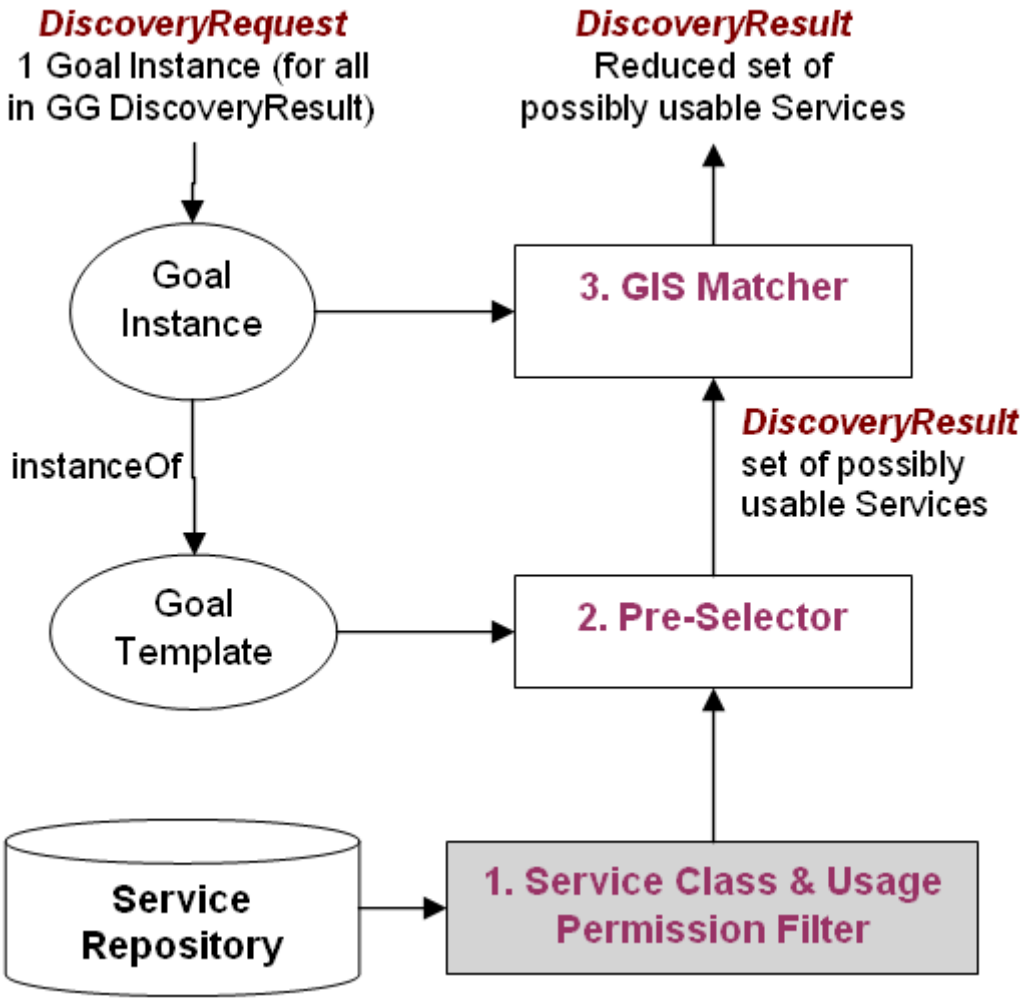

Figure 3. GS Discovery Overview

As shown in the figure, the GS Discoverer consists of three components:

1. Service Class and Usage Permission Filter: This pre-filters services to be matched according to the Service Ontology (see Section 3.5.1) - i.e. for a Buyer Goal Instance only buyerservices will be matched; additionally, potential usabale services can be pre-filtered according to usage permissions of Freds (e.g. The LEINER-Fred is not allowed to use the IKEA Service)
2. the Pre-Selector matches the Goal Template of the Goal Instances provided as DiscoveryRequest with those services available in the Repository that are valid for the Pre-Filter
3. the GIS Matcher matches the Goal Instance with the DiscoveryResult of the Pre-Selector, determining the final GS DiscoveryResult.

While the Pre-Filter works via APIs on the Service Ontology, the Pre-Selector and the GIS Matcher are implemented as matchmakers with VAMPIRE.

4.2.1 Pre-Selector Matchmaking

As for the GG Matcher, the following table shows the matchmaking requirements for the GS Pre-Selector.

Table: GGMatcher requested matches

Services \ Goal Templates	BGT 1	BGT 2	BGT 3	BGT 4	SGT 1	SGT 2	SGT 3
BS 1	X	X		X			
BS 2			X				
BS 3		X		X			
SS IKEA					X		
SS LEINER					X		
SS KIKA					X		
SS 1					X		
SS 2						X	
SS 3					X		X

The following listings shows the testing results for the Pre-Selector as realized and tested with VAMPIRE; here also, we have separate testing files for purchase- and query goals.

Listing. Pre-Selector Results for Purchase Goals

```
%--------------------------------------------------------------------------
% File     : testGSPreselectorPC.p
% Domain   : SWF Use Case GG Discovery Matchmaking Test
% Problem  : tests all combinations for GG Discovery for Purchasing Goals
% Version  : 1.2
%--------------------------------------------------------------------------

include('../ontologies/onto.ax'). include('../ontologies/kb.ax').
include('../ontologies/universe.ax').

% Buyer Goal Template 1
%include('../goals/goaltemplates/BuyFurnitureCreditCardGoal.ax').
% all relevant Services
%include('../services/buyerservice1.ax').      %YES    CORRECT
%include('../services/buyerservice2.ax').      %NO     CORRECT
%include('../services/buyerservice3.ax').      %NO    CORRECT

%include('../services/sellerservice1.ax').     %YES    WRONG, handled by service ontology pre-filter
%include('../services/sellerserviceIKEA.ax').  %YES    WRONG, handled by service ontology pre-filter
%include('../services/sellerservice3.ax').     %NO    CORRECT

% Buyer Goal Template 2
%include('../goals/goaltemplates/BuyFurnitureAnyPaymentGoal.ax').
% all relevant Services
%include('../services/buyerservice1.ax').      %YES    CORRECT
%include('../services/buyerservice2.ax').      %NO     CORRECT
%include('../services/buyerservice3.ax').      %YES    CORRECT

%include('../services/sellerservice1.ax').     %YES    WRONG, handled by service ontology pre-filter
%include('../services/sellerserviceIKEA.ax').  %YES    WRONG, handled by service ontology pre-filter
%include('../services/sellerservice3.ax').     %YES    WRONG, handled by service ontology pre-filter


% Buyer Template 4
%include('../goals/goaltemplates/BuyFurniturePrivateGoal.ax').
% all relevant Services
%include('../services/buyerservice1.ax').      %YES    CORRECT
%include('../services/buyerservice2.ax').      %NO     CORRECT
%include('../services/buyerservice3.ax').      %YES    CORRECT
```

```
%include('../services/sellerservice1.ax').      %YES    WRONG, handled by service ontology pre-filter
%include('../services/sellerserviceIKEA.ax').   %NO     CORRECT
%include('../services/sellerservice3.ax').      %YES    WRONG, handled by service ontology pre-filter

%-----------------------------------------------

% Seller Template 1
%include('../goals/goaltemplates/SellFurnitureGeneralGoal.ax').
% all relevant Services
%include('../services/sellerservice1.ax').      %YES    CORRECT
%include('../services/sellerservice2.ax').      %NO     CORRECT
%include('../services/sellerserviceIKEA.ax').   %YES    CORRECT
%include('../services/sellerserviceLEINER.ax'). %YES    CORRECT
%include('../services/sellerserviceKIKA.ax').   %YES    CORRECT
%include('../services/sellerservice3.ax').      %YES    CORRECT

%include('../services/buyerservice1.ax').       %YES    WRONG, handled by service ontology pre-filter
%include('../services/buyerservice3.ax').       %YES    WRONG, handled by service ontology pre-filter

% Seller Template 3
include('../goals/goaltemplates/SellFurniturePrivateGoal.ax').
% all relevant Services
%include('../services/sellerservice1.ax').      %NO     CORRECT
%include('../services/sellerservice2.ax').      %NO     CORRECT
%include('../services/sellerserviceIKEA.ax').   %NO     CORRECT
%include('../services/sellerserviceLEINER.ax'). %NO     CORRECT
%include('../services/sellerserviceKIKA.ax').   %NO     CORRECT
%include('../services/sellerservice3.ax').      %YES    CORRECT

%include('../services/buyerservice1.ax').       %YES    WRONG, handled by service ontology pre-filter
%include('../services/buyerservice3.ax').       %YES    WRONG, handled by service ontology pre-filter


%-------------------------------------------------------------------------
% Proof Obligation GG Discoverer for PurchaseContracts Goal Instances:
% - Intersection for postconditions
% - Intersection for effects

input_formula(po, conjecture,(
? [X] : ( goalpostcondition(X) & wspostcondition(X) )
&
? [Y] : ( goaleffect(Y) & wseffect(Y) )
)).
%-------------------------------------------------------------------------
```

Listing. Pre-Selector Results for Query Goals

```
%-------------------------------------------------------------------------
% File     : testGSPreselectorQ.p
% Domain   : SWF Use Case GG Discovery Matchmaking Test
% Problem  : tests all combinations for GG Discovery for Querying Goals
% Version  : 1.2
%-------------------------------------------------------------------------

include('../ontologies/onto.ax').
include('../ontologies/kb.ax').
include('../ontologies/universe.ax').

% Buyer Goal Template 3
%include('../goals/goaltemplates/QueryFurnitureRequestGoal.ax').
% all relevant services
%include('../services/buyerservice2.ax').       %YES     CORRECT
%include('../services/sellerservice2.ax').      %NO      WRONG, handled by service ontology pre-filter
%include('../services/buyerservice1.ax').       %NO      CORRECT

% Seller Goal Template 3
%include('../goals/goaltemplates/QueryFurnitureProviderGoal.ax').
% all relevant services
%include('../services/sellerservice2.ax').      %YES     CORRECT
%include('../services/buyerservice2.ax').       %NO      WRONG, handled by service ontology pre-filter
%include('../services/buyerservice1.ax').       %NO      CORRECT
```

```
%include('../services/sellerservice1.ax').      %NO      CORRECT

%------------------------------------------------------------------------
% Proof Obligation GG Discoverer for PurchaseContracts Goal Instances:
% - PlugIn for postconditions
% - Exact for effects

input_formula(po, conjecture,(
! [X] : ( goalpostcondition(X) => wspostcondition(X) ) &
! [Y] : ( goaleffect(Y) <=> wseffect(Y) )
)).
%------------------------------------------------------------------------
```

4.2.2 GIS Matcher Matchmaking

The following table shows the matchmaking requirements for the GIS Matcher in the GS Discoverer GS Pre-Selector.

Table: GIS Matcher requested matches

Resources	BGI 1 (111)	BGI 2 (112)	BGI 3 (113)	BGI 4 (114)	SGI IKEA (121)	SGI LEINER (1212)	SGI KIKA (1213)	SGI 2 (122)	SGI priv. 1 (123)	SGI priv. 2 (1232)
BS 1	X	X								
BS 2			X							
BS 3		X		X						
SS IKEA					X					
SS LEINER						X				
SS KIKA							X			
SS 1					X				X	X
SS 2								X		
SS 3									X	X

The testing scripts below for purchase- and for query goals show the matchmaking results as realized with VAMPIRE.

Listing. GIS Matcher Results for Purchase Goals

```
%------------------------------------------------------------------------
% File     : testGISMatcherPC.p
% Domain   : SWF Use Case GG Discovery Matchmaking Test
% Problem  : tests all combinations for GG Discovery for Purchasing Goals
% Version  : 1.2
%------------------------------------------------------------------------

include('../ontologies/onto.ax').
include('../ontologies/kb.ax').
include('../ontologies/universe.ax').

% Buyer Goal Instance 1
%include('../goals/goalinstances/111.ax').
% all relevant Services
%include('../services/buyerservice1.ax').      %YES    CORRECT
```

```
%include('../services/buyerservice2.ax').       %NO     CORRECT
%include('../services/buyerservice3.ax').       %NO     CORRECT

%include('../services/sellerservice1.ax').      %YES    WRONG, handled by service ontology pre-filter
%include('../services/sellerserviceIKEA.ax').   %NO     CORRECT
%include('../services/sellerservice3.ax').      %NO     CORRECT

% Buyer Goal Instance 2
%include('../goals/goalinstances/112.ax').
% all relevant Services
%include('../services/buyerservice1.ax').       %YES    CORRECT
%include('../services/buyerservice2.ax').       %NO     CORRECT
%include('../services/buyerservice3.ax').       %YES    CORRECT

%include('../services/sellerservice1.ax').      %YES    WRONG, handled by service ontology pre-filter
%include('../services/sellerserviceIKEA.ax').   %YES    WRONG, handled by service ontology pre-filter
%include('../services/sellerservice3.ax').      %YES    WRONG, handled by service ontology pre-filter


% Buyer Goal Instance 4
%include('../goals/goalinstances/114.ax').
% all relevant Services
%include('../services/buyerservice1.ax').       %NO     CORRECT
%include('../services/buyerservice2.ax').       %NO     CORRECT
%include('../services/buyerservice3.ax').       %YES    CORRECT

%include('../services/sellerservice1.ax').      %NO     CORRECT
%include('../services/sellerserviceIKEA.ax').   %NO     CORRECT
include('../services/sellerservice3.ax').       %NO     CORRECT

%-------------------------------------------------

% Seller Goal Instance 1
%include('../goals/goalinstances/121.ax').
% all relevant Services
%include('../services/sellerservice1.ax').          %YES    CORRECT
%include('../services/sellerservice2.ax').          %NO    CORRECT
%include('../services/sellerserviceIKEA.ax').       %YES    CORRECT
%include('../services/sellerserviceLEINER.ax').     %NO    CORRECT
%include('../services/sellerserviceKIKA.ax').       %NO    CORRECT
%include('../services/sellerservice3.ax').          %NO    CORRECT

%include('../services/buyerservice1.ax').           %YES    WRONG, handled by service ontology pre-filter

% Seller Goal Instance 1.2
%include('../goals/goalinstances/1212.ax').
% all relevant Services
%include('../services/sellerservice1.ax').          %YES    CORRECT
%include('../services/sellerservice2.ax').          %NO     CORRECT
%include('../services/sellerserviceIKEA.ax').       %NO    CORRECT
%include('../services/sellerserviceLEINER.ax').     %YES    CORRECT
%include('../services/sellerserviceKIKA.ax').       %NO    CORRECT
%include('../services/sellerservice3.ax').

%include('../services/buyerservice1.ax').

% Seller Goal Instance 1.3
%include('../goals/goalinstances/1213.ax').
% all relevant Services
%include('../services/sellerservice1.ax').
%include('../services/sellerservice2.ax').
%include('../services/sellerserviceIKEA.ax').
%include('../services/sellerserviceLEINER.ax').
%include('../services/sellerserviceKIKA.ax').
%include('../services/sellerservice3.ax').

%include('../services/buyerservice1.ax').

% Seller Goal Instance 3
%include('../goals/goalinstances/123.ax').
% all relevant Services
%include('../services/sellerservice1.ax').          %YES    CORRECT
%include('../services/sellerservice2.ax').          %NO     CORRECT
%include('../services/sellerserviceIKEA.ax').       %NO     CORRECT
%include('../services/sellerserviceLEINER.ax').     %NO     CORRECT
%include('../services/sellerserviceKIKA.ax').       %NO     CORRECT
```

```
%include('../services/sellerservice3.ax').          %YES    CORRECT

%include('../services/buyerservice1.ax').           %YES,   WRONG, but handled by service ontology pre-filter

% Seller Goal Instance 3.2
%include('../goals/goalinstances/1232.ax').
% all relevant Services
%include('../services/sellerservice1.ax').          %NO    CORRECT
%include('../services/sellerservice2.ax').          %NO    CORRECT
%include('../services/sellerserviceIKEA.ax').       %NO    CORRECT
%include('../services/sellerserviceLEINER.ax').     %NO    CORRECT
%include('../services/sellerserviceKIKA.ax').       %NO    CORRECT
%include('../services/sellerservice3.ax').          %YES    CORRECT

%include('../services/buyerservice1.ax').           %NO     CORRECT

%------------------------------------------------------------------------
% Proof Obligation GG Discoverer for PurchaseContracts Goal Instances:
% - Intersection for postconditions
% - Intersection for effects

input_formula(po, conjecture,(
? [X] : ( goalpostcondition(X) & wspostcondition(X) ) &
? [Y] : ( goaleffect(Y) & wseffect(Y) ) )).
%------------------------------------------------------------------------
```

Listing. GIS Matcher Results for Query Goals

```
%------------------------------------------------------------------------
% File     : testGGMatcherQ.p
% Domain   : SWF Use Case GG Discovery Matchmaking Test
% Problem  : tests all combinations for GG Discovery for Querying Goals
% Version  : 1.2
%------------------------------------------------------------------------

include('../ontologies/onto.ax').
include('../ontologies/kb.ax').
include('../ontologies/universe.ax').

% Buyer Goal Instance 3
%include('../goals/goalinstances/113.ax').
% all relevant services
%include('../services/buyerservice2.ax').       %YES     CORRECT
%include('../services/sellerservice2.ax').      %YES     WRONG, handled by service ontology pre-filter
%include('../services/buyerservice1.ax').       %NO      CORRECT

% Seller Goal Instance 2
include('../goals/goalinstances/122.ax').
% all relevant services
%include('../services/sellerservice2.ax').      %YES     CORRECT
%include('../services/buyerservice2.ax').       %YES     WRONG, handled by service ontology pre-filter
%include('../services/buyerservice1.ax').       %NO      CORRECT
%include('../services/sellerservice1.ax').      %NO      CORRECT

%------------------------------------------------------------------------
% Proof Obligation GG Discoverer for PurchaseContracts Goal Instances:
% - PlugIn for postconditions
% - Exact for effects

input_formula(po, conjecture,(
! [X] : ( goalpostcondition(X) => wspostcondition(X) ) &
! [Y] : ( goaleffect(Y) <=> wseffect(Y) )
)).
%------------------------------------------------------------------------
```

4.3 WW Discoverer

For automated interaction of services as the realization of cooperative goal resolution, the Choreography Interfaces of cooperation partners have to be compatible with regard to behavioral models and messaging sequences. This is checked in WW Discovery as the last step in cooperation establishment determines, resulting in a global interaction model of the partners' services that allows automated execution of the cooperation.

Figure 4 illustrates the structure of WW Discovery for two Freds (A and B) that have been determined as potential cooperation partners, and each of them has discovered a set of possibly usable Services. The oval boxes represent the Choreography Interfaces with the external visible behavior and the related messages as defined in WSMO Choreography [WSMO Choreography]. Determining the compatibility of Choreography Interfaces consists of two aspects: first, the workflow of the Service Choreographies have to compatible (process level compatibility), and second the expected messaging sequence has to be compatible (protocol level compatibility). The former is considered to hold if either the process models are the same (sequence of activities and transitions), or if one of them subsumes the other. The latter is given if the message sequences of Service Choreography Interfaces are symmetric or inverse, in the simplest case; a more advanced technique for determining protocol level compatibility is under construction at the time of writing. In order to resolve possible mismatches, a WSMO WW Mediator can be included to establish compatibility on the process and protocol level.

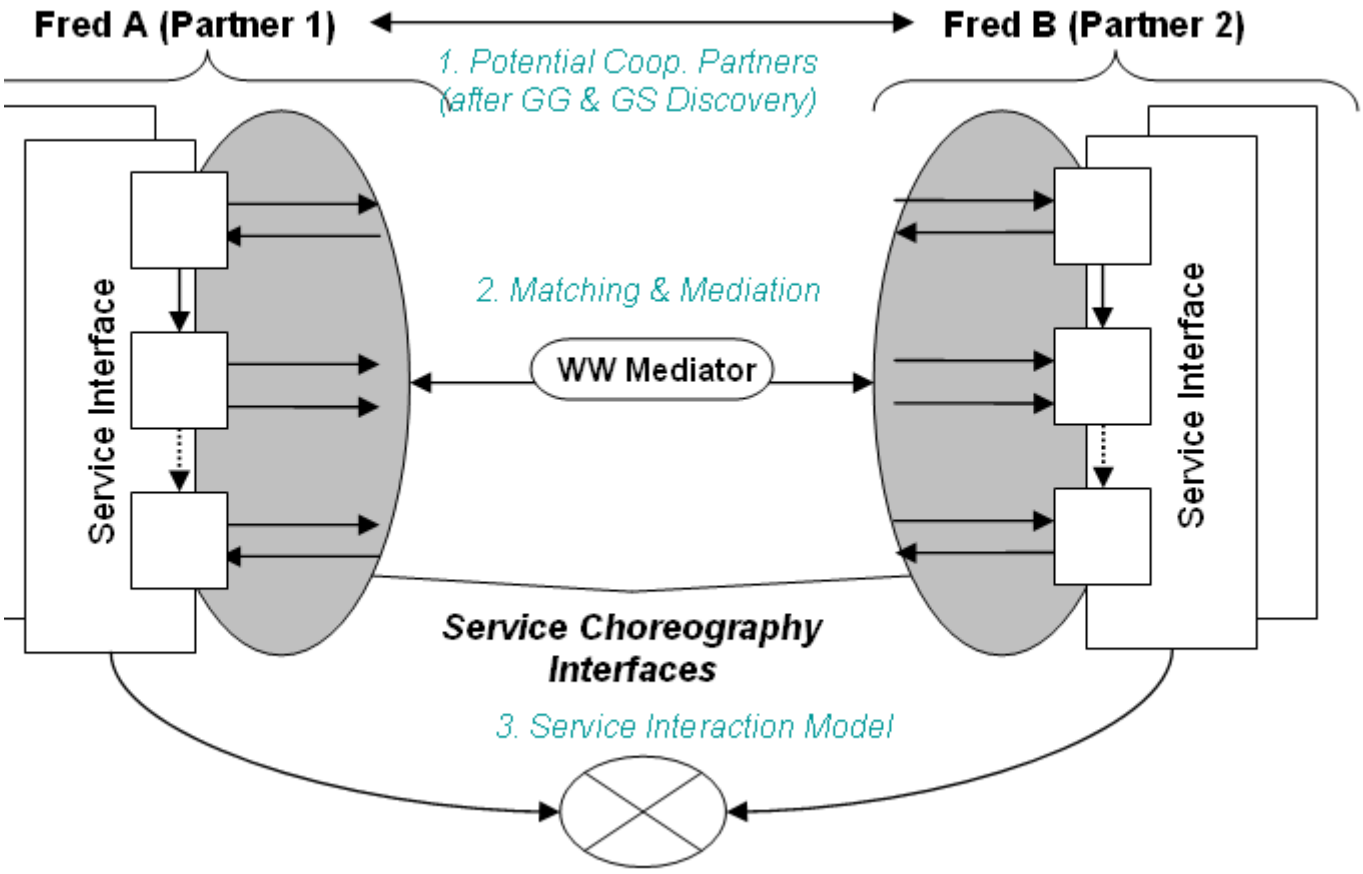

Figure 4. WW Discovery Overview

The matchmaking for the WW Discoverer is not elaborated within this version of this document (because of this reason, also the Choeographies of the services are not specified yet).

5. Conclusions and Further Work

This document specifies a complete use case for the SWF technology within the domain of purchasing furniture in a virtual, agent-driven marketplace. We have briefly summarized the SWF conceptual architecture and the SWF components, provided a conceptual overview of the use case, and the models for the needed SWF requirements in WSML.

The use case outlined in this document is intended for testing and validating as well as for showcasing the SWF technology in demonstrations. Please remark that several modeling decisions and other pay-offs have been made in order to allow implementation of the use case with ín the current development status of the SWF technology.

In future versions of this document, the SWF components descriptions will be completed, and enhanced with regard to the detailed structure of specific components.

This use case is open for re-use and adaption by the SDK-Cluster working groups around WSMO.

References

[de Bruijn, 2004] de Bruijn. J. (ed.): *WSML-Core*. WSML Working Draft D16.7, 23 August 2004; available at: http://www.wsmo.org/2004/d16/d16.7/v0.1/

[Fensel & Bussler, 2002] D. Fensel and C. Bussler: *The Web Service Modeling Framework WSMF*, Electronic Commerce Research and Applications, 1(2), 2002.

[Herzog et al., 2004] Herzog, R.; Zugmann, P.; Stollberg, M.; Roman, D. (ed.): *WSMO Registry.* WSMO Working Draft D10, 26 April 2004; available at: http://www.wsmo.org/2004/d10/v0.1/.

[Keller et al., 2004] Keller, U.; Lara;, R.; Polleres, A.; Lausen, H.; Stollberg, M., Kifer, M..: *Inferencing Support for Semantic Web Services: Proof Obligations*. WSML Deliverable D5.1, WSML Working Draft 21 October 2004. available at http://www.wsmo.org/2004/d5/d5.1/

[Riazanov and Voronkov, 2002] Riazanov, A.; Voronkov, A.: *The design and implementation of VAMPIRE*. In AI Communications 15(2), Special issue on CASC, pp. 91 -110, 2002.

[Roman et al., 2004] D. Roman, U. Keller, H. Lausen (eds.): *Web Service Modeling Ontology - Standard (WSMO - Standard)*, version 1.0 available at http://www.wsmo.org/2004/d2/v02/.

[WSMO Choreography] Roman, D., Vasiliu, L.; Stollberg, M.; Bussler, C. (ed.): *Choreography in WSMO*, WSMO Working Draft D14, available at: http://www.wsmo.org/2004/d14/.

[WSMO Use Case] Stollberg, M.; Lausen, H.; Lara, R.; Polleres, A. (ed.): *WSMO Use Case and Testing* , WSMO Working Draft D3.2, available at: http://www.wsmo.org/2004/d3/d3.2/v0.1/.

[FRED Whitepaper] Stollberg, M.; Lausen, H.; Arroyo, S.; Herzog, R.; Smolle, P.; Fensel, D.: *FRED Whitepaper*. DERI Technical Report DERI-TR-2004-01-09; available at: http://www.deri.at/publications/techpapers/documents/DERI-TR-2004-01-09.pdf.

[SWF Deliverable D4] Stollberg, M.; Keller, U.; Keimel, B.; Zugmann P.; Herzog, R.: *SWF Architecture, Tools, and Mechanisms*. SWF Deliverable D4, available at: (under contruction).

Acknowledgements

The Semantic Web Fred project work is funded by the Austrain government under the CoOperate program. The editors would like to thank to all the members of the WSMO working group for their advice and input into this document, especially Holger Lausen and Michael Felderer for support in modeling and technical issues.

Appendix A: Change Tracking

To facilitate retracing of changes inbetween different version of this deliverable, the following lists the essentail changes done in comparison to the preceding version.

The change tracking starts with the version of 08 September 2004.

Version: 19 October 2004
http://www.deri.at/research/projects/swf/usecase/20041019/

- corrected definition of use case resources (making them compliant to the use case KB, changing objects of interest in Goal Instances)
- revise modeling of logical expressions in goal / service postconditions and effects; now they are 1:1 compatible with the TPTP syntax (FOL-syntax used by VAMPIRE)
- added serviceontology: defines classes of services for distinguishing buyer- and seller services
- completed SWF Use Case Knowledge Base

- corrected identifier: general URL is http://swf.quarto.at/repository/, then directories for specific resources
- added section 4: overview of match making in discoverers

Version: 01 October 2004
http://www.deri.at/research/projects/swf/usecase/20041001/

- intermediate version, contains some corrections and additions
- default namespace: http://swf.quarto.at/repository/
- ontologies: corrections furnishing, swfmo, deleted date and time as not needed for use case
- changed names for WSML files, so that the naming convention with the Fred system is coherent
- added additional Goal Instances (i.e. Active WSMO Goals)
- added SWF Use Case Knowledge Base

Version: 20 September 2004
http://www.deri.at/research/projects/swf/usecase/20040920/

- first stable version
- valid WSML models for all Use Case resources in section 3
- SWF Component Description Language added in section 1.2

Version: 08 September 2004
http://www.deri.at/research/projects/swf/usecase/20040908/

- initial setup
- section 1 and section 2 completed
- section 3 (models): only initial versions

webmaster