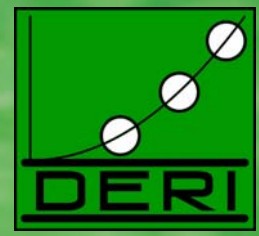


On the Semantics of Functional Descriptions of Web Services

Uwe Keller, Holger Lausen and Michael Stollberg

Uwe Keller
uwe.keller@deri.org

European Semantic Web Conference (ESWC)
Budva, Montenegro, June 11 – 14, 2006

© Copyright 2005 Digital Enterprise Research Institute. All rights reserved.

Motivation: Web Services & Semantic Description

- **Web Services (WS) & Service-oriented Architectures (SOA)**
  - WS as a main ingredient of SOA
  - SOA picked up by major players in IT industry
  - SOA will cause Web Services to be widely adopted, many WS expected
  - Automated support for dealing with a huge number of Web Services desirable: Discovery, Composition, …
- **Semantic annotation as the magic means**
  - WS as a particular type of (complex) resource in the Semantic Web
  - Semantic Descriptions can be / need to be more complex than for simple data items
  - Formal semantics allows to ensure correctness of algorithms used to deal with such descriptions in applications, e.g. matching step in the discovery processes
- **Various proposals of Semantic Description Frameworks** (submitted to W3C)
  - **WSMO / WSML**
  - OWL-S
  - SWSF
  - …

The problem to be solved …

- **What is the formal meaning of capabilities in WSMO?**
  - What is a Web Service (from the standpoint of functional descriptions)?
  - How to describe what a Web Service provides?
  - When does a Web Service provide a particular capability?

- **Design goals**
  - **Rigorous and concise model** based on mathematical concepts
  - **Focus on the elements to be described** (WS) instead of the particular description language
  - **Usable with various ontology languages** (such as síngle WSML variants)
  - Model should **extensible** to include other aspects of WS (**multi-perspective**)
  - Ideally applicable for the definition of semantics of other frameworks as well

What is a Web Service?
Web Service vs. Service

- **Service**
  - A **provision of value in some domain** (not necessarily monetary, indepent of how service provider and requestor interact)
- **Web Service**
  - **Computational entity** accessible over the Internet (using Web Service Standards & Protocols), **provides access to (concrete) services** for the clients.
- **Relation between these notions**
  - **Service** corresponds to a **concrete execution of a web service** (with given input values)
  - **Web Service** provides a **set of services** to its client; one service for each possible input value tuple

How to consider a Web Service?
Different Levels of Abstraction identified in WSMO

What? (Syntactically)

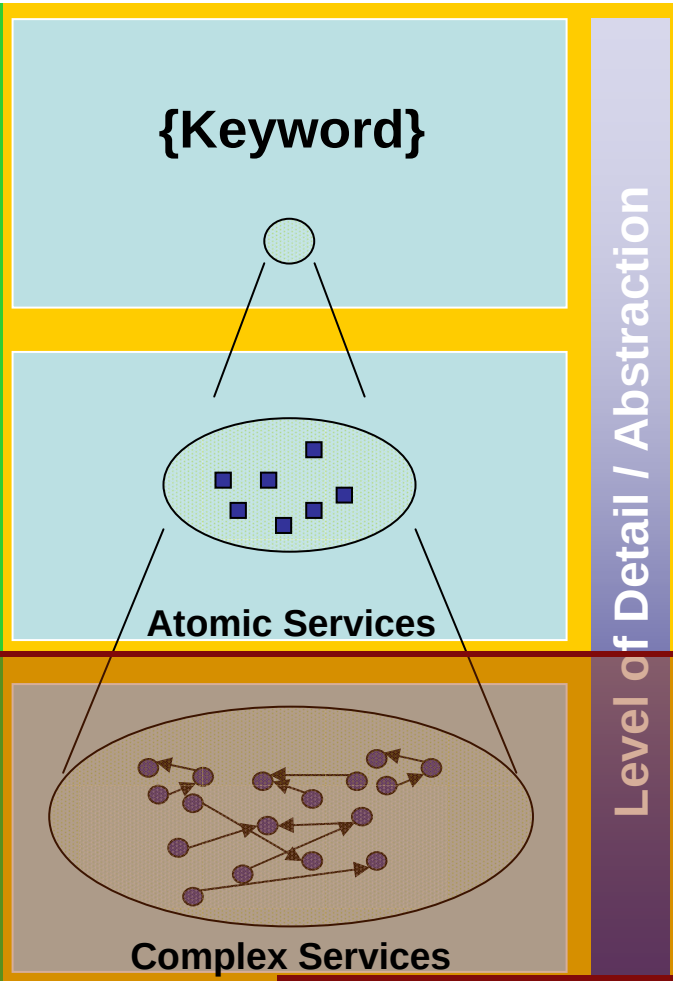
No explicit structure &
Services are not characterized as such
in machine processable form

What? (Semantic „Light“)

What does WS provide,
in terms of atomic objects with properties
(not under which circumstances)

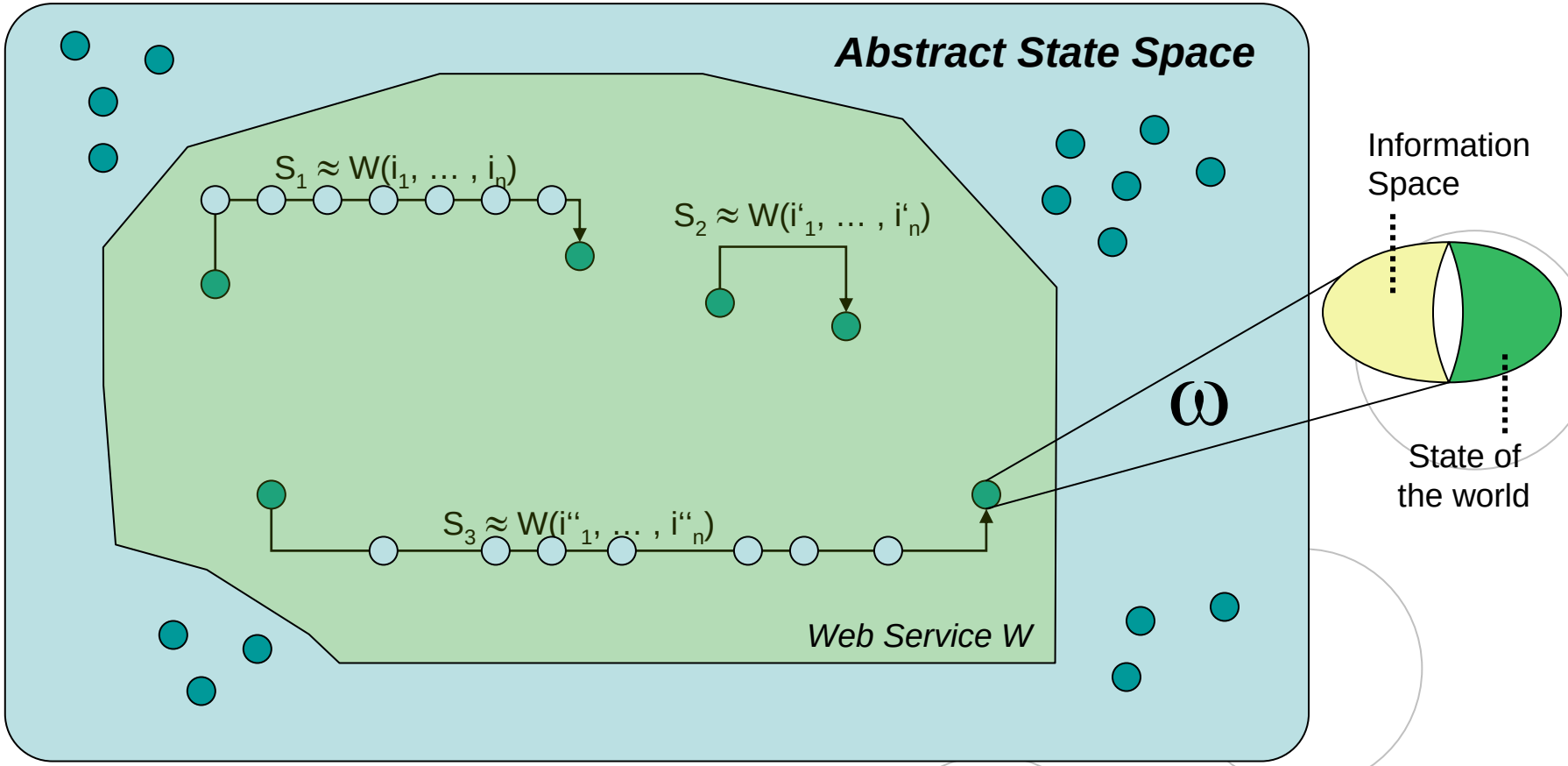
What & When? (Semantic „Detailed“)

Pre/Post-Cond,
takes „before-after“ relationship
& client-side requirements into account



What we consider here

What is a Web Service?
Overview of the formal Model: **Abstract State Spaces**



Formal Model: **Abstract State Spaces**

- **A changing world**
  - world as an entity that changes over time
  - each point in time, the world is in one particular state that determines how the world is perceived by agents acting in it
  - Language L needed to describe perceptions
  - State corresponds to an interpretation (in a logical sense) of L

- **Fix signature of some language L to describe states**
  - $\Sigma \supseteq$ {isAccount/1,balance/1, $\geq$, 0, 1, 2, ...}
  - Symbols with Fixed Meaning (e.g. ($\geq$, 0,))
  - Dynamic Symbols (e.g. balance(·))

- **Ontologies as background knowledge** (over this signature)
  - $\forall$?x . (isAccount(?x) $\rightarrow$ balance(?x) $\geq$ 0)

Formal Model: **Abstract State Spaces** (cont'd)

- **Example States (and respective observations ω_{rw})**
  - s_0: balance(acc_1) = 10 $\wedge$ balance(acc_2) = 100
  - s_1: balance(acc_1) = 10 $\wedge$ balance(acc_2) = 80 (e.g intermediate state)
  - s_n: balance(acc_1) = 30 $\wedge$ balance(acc_2) = 80

- **Information Space (ω_{is})**
  - Captures outputs given by the WS during execution as a set of objects
  - Example: ack(20051202,msgid23) $\wedge$ confirm($acc_1 acc_2 20$) $\in$ Is_k

- Not all sequences of abstract states are legal sequences for executions, e.g background ontology not violated or respective sequence of information spaces must be monotonic

- **Web Service** essentially defined by a mapping between pairs of initial state and input and a legal sequence of states (which represents the respective execution)

What is a Web Service?
Example

- **Abstract State Space**:
  - Set of states includes $s_{n,m}$, $s'_{n,m,a,c}$ (for all integers n,m ; $a \geq 0$) with
  - $\omega_{rw}(s_{n,m}) = \omega_{rw}(s'_{n,m,a,c})$: balance(acc_1) = N $\wedge$ balance(acc_2) = M
  - $\omega_{is}(s_{n,m})$: {}, $\omega_{is}(s'_{n,m,\ a,\ c})$: {ack-transfer($acc_1$,$acc_2$, a, c)}
- **Example Web Services** (better WS implementations)
  - W_1: Tranfers of all positive amounts in every currency
    - W_1 (a, c) := ($s_{n,m}$, $s_{n\text{-}amt,m}$, $s'_{n\text{-}amt,m+amt,\ amt,\ c}$) if c denotes a currency $a \geq 0$ and amt represents the amount a in Euros
    - W_2 (a, c) := ($s_{n,m}$) otherwise
  - W_2: Transfer of limited amounts in European currencies only
    - W_2 (a, c) := ($s_{n,m}$, $s'_{n\text{-}amt,m+amt,\ amt,\ c}$) if c is european currency, $a \geq 0$,amt is a in Euros and a (in currency c) does not exceed 10.000 Euros
    - W_2 (amt, c) := ($s_{n,m}$) otherwise

How to describe a Web Service?
WSMO Approach

- WSMO based on state-based descriptions of capabilities using Pre- and Postconditions: Precise characterization of how state changes
- General form in WSML syntax:

```
webservice wid
capability
   shared variables {?x1, ..., ?xn}
      precondition definedBy  logical-expression le1
      assumption definedBy  logical-expression le2
      postcondition definedBy logical-expression le3
      effect definedBy logical-expression le4
```

- Abstract form used here: $D = (\phi^{pre}, \phi^{post}, IF^D)$
  - ϕ^{pre}, ϕ^{post} : precondition (incl. assumptions) and postconditions (incl. effects)
  - IF^D: inputs expected by the Web Service (Capability Description Interface)

- As well present in OWL-S Service Profiles

How to describe a Web Service?
Example: Unlimited Bank Transfer

BTUnlimitedWS$_1$(?A, ?C)

pre:

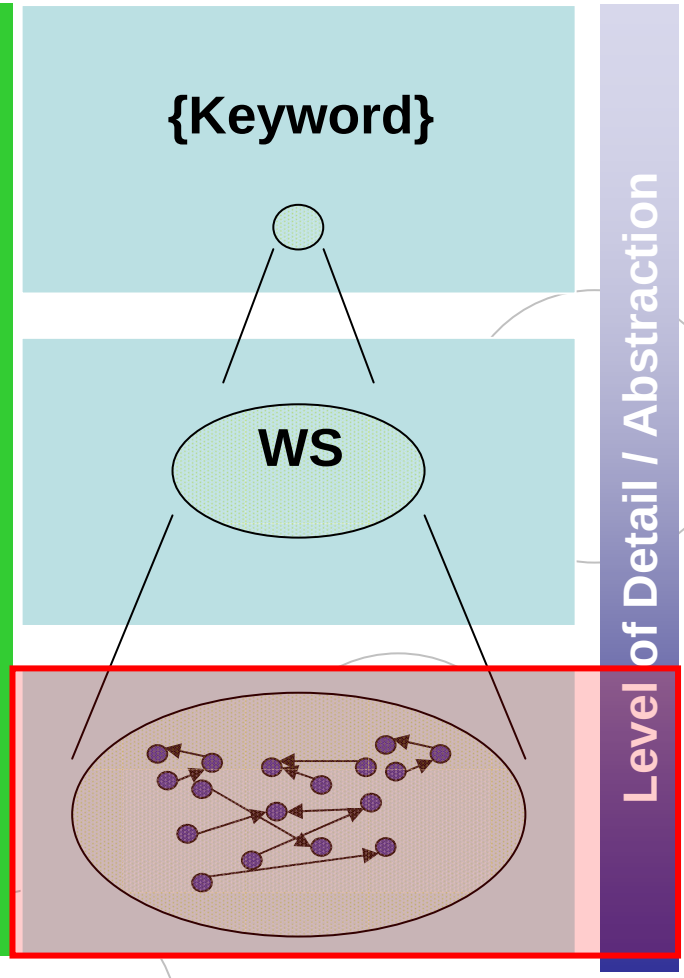
isCurrency(?C) and ?A >= 0

post:

balance(acc$_1$) = balance$_{pre}$ (acc$_1$) – convert(?A, ?C, Euro)

and

balance(acc$_2$) = balance$_{pre}$ (acc$_2$) + convert(?A, ?C, Euro)



How to describe a Web Service?
Example: Limited Bank Transfer

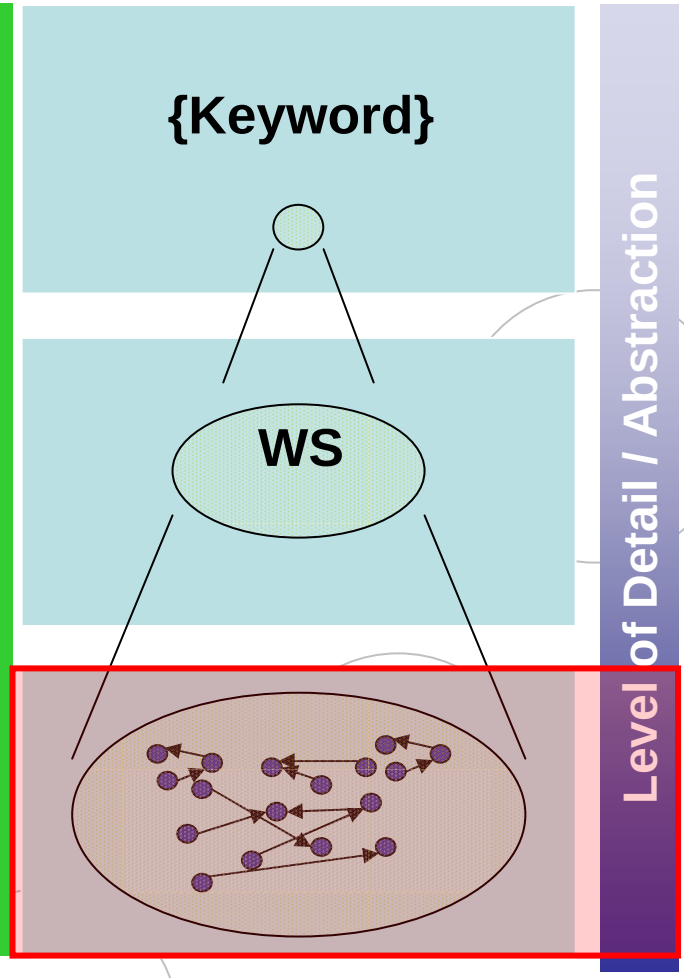
BTLimitedWS$_2$(?A, ?C)

pre:

isEuropeanCurrency(?C)
and ?A >= 0
and ?amt < 10.000

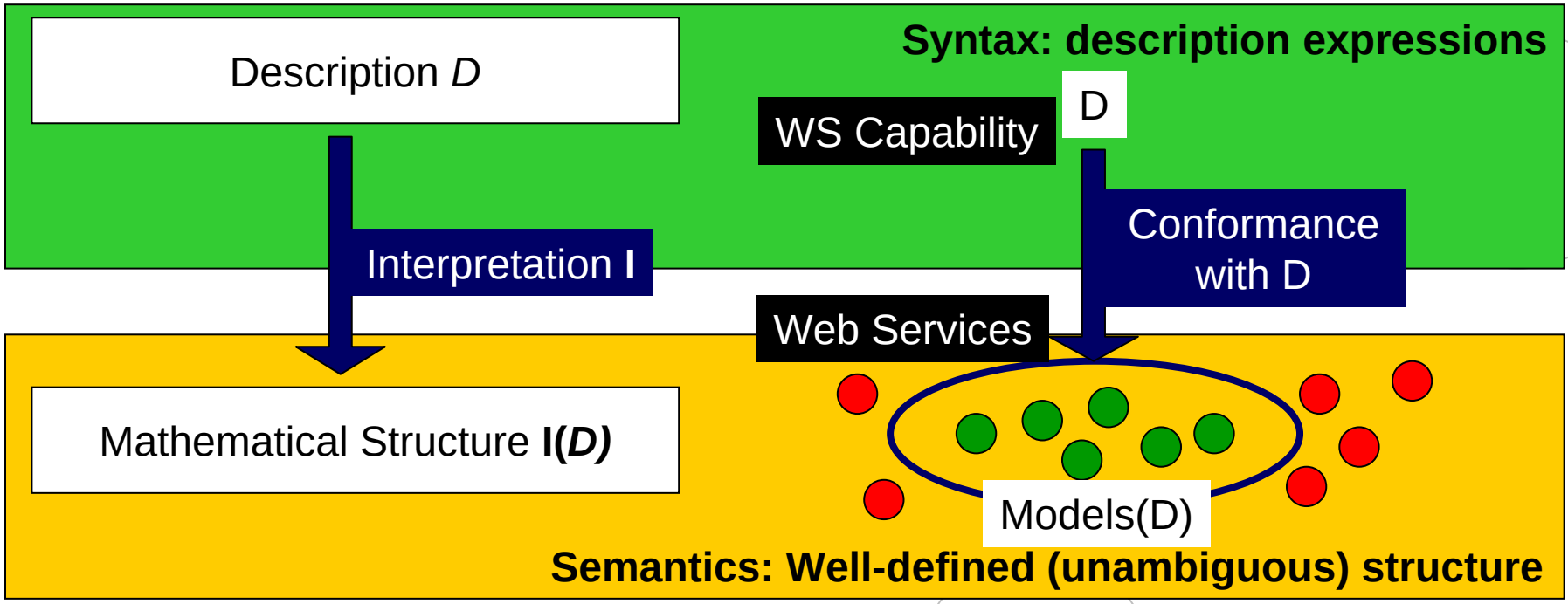
post:

balance(acc1) = balance$_{pre}$ (acc1) – convert(?A, ?C, Euro)
and
balance(acc2) = balance$_{pre}$ (acc2) + convert(?A, ?C, Euro)



When does a WS provide some specified functionality ?

- Standard method used in mathematical logic or algebra
  - Model-theoretic approach to define the semantics of formal descriptions



When does a WS provide some specified functionality ?

- Benefits of the model-theoretic approach
  - Allows to separate the model of the elements to be described from the actual description language (**Decoupling**)
  - Facilitiates reuse of the underlying model for different tasks
    - Different views on the same object (provided functionality, interaction-protocol, non-functional aspects etc. …)
    - Underlying objects can be richer than required for a single perspective
  - Allows to integrate different perspectives on the same entity

When does a WS provide some specified functionality ?

- **What is needed?**
  - Define a ,,satisfies‘‘ relation between Web Services W (our mathematical model) and Web Service Capability Descriptions D
- **Basic idea**
  - Capabilities syntactically as a dynamic layer on a static language
  - Semantic definition follows this layering: Reduction to satisfaction of state conditions in states
- **Informally**
  - Standard ,,implication‘‘-style semantics
  - **W satisfies D** iff.

    For any input and intial states, it must hold that the respective web service execution ends in a state satisfying the postcondition if the initial state satisfied the precondition

When does a WS provide some specified functionality ?

- Examples revisited
  - W_1 satisfies $BTUnlimitedWS_1(?A, ?C)$
  - W_1 satisfies $BTLimitedWS_2(?A, ?C)$
  - W_2 satisfies $BTLimitedWS_2(?A, ?C)$
  - W_2 does not satisfy $BTUnlimitedWS_1(?A, ?C)$
    - In case of positive amounts ?A in non-european currencies ?C W_2 actually does nothing (similar for amounts exceeding the limit)
    - No transfer is being performed althought the precondition of $BTUnlimitedWS_1(?A, ?C)$ is satisfied, post-condition is violated

Results coincide with intuitive expectation

Applying the formal Model:
Semantic Analysis of Functional Descriptions

- **We used a model-theoretic approach**
  - What happens if we consider standard notions in mathematical logic (defined in terms of models) in regard of our class of models?

- **Realizability of Functional Description**
  - Equivalent notion to **satisfiability** in logics, i.e. there is a WS which can satisfy the functional description (for all inputs)
- **Omnipotence of Functional Descriptions**
  - Equivalent to **validity** in logics, i.e. all Web Services (no matter what they do) satisfy the description
- **Functional Refinement of Functional Descriptions**
  - Equivalent to **logical entailment**, i.e. all Web Services that satisfy description D_1 satisfy description D_2 as well
  - Applicable for semantic matching during discovery process

Applying the formal Model:
Semantic Analysis of Functional Descriptions

- **Example:** Realizability of Functional Description

  Background Ontology Ω such that

  $$\Omega \models \forall\ ?x.(isAccount(?x) \rightarrow balance(?x) \geq 0)$$

  Functional Description D

  - ϕ^{pre}: isAccount(?acc) and ?amount $\geq$ 0
  - ϕ^{post}: balance(?acc) = $balance_{pre}$(?acc) - ?amount
  - IF^{D}: <?acc, ?amount>

  – **Not obvious**: Description **not realizable** with respect to Ω
    - Use an input value for ?amount that is bigger than the current balance
  – **Fix the description:** Need to extend (strengthen) precondition
    - ϕ^{pre}: isAccount(?acc) and 0 $\leq$?amount $\leq$ balance(?acc)

Applying the formal Model:
Semantic Analysis of Functional Descriptions

- **But how can we detect these properties of Functional Descriptions? How can we „reason“ with them?**
  - Option I: Design a new calculus to deal with capability descriptions
  - Options II: Try to reduce the problem from the capabilty description level to the level of the static description language

- **Option II is possible for the notions that we discussed**
  - Realizability and Omnipotence can be reduced to Functional Refinement
  - Functional refinement can be transformed to checking logical entailment within a static ontology language (depending on the expressivity of the static language)
  - Correctness of transformation can be proven based on our framework

- **Hence:** Semantic Analysis of Capabilities can be done in principle with existing ontology reasoning systems

Conclusion

- Our work provides a **mathematically precise model** of a the functionality of a WS **on a detailed level**
- **Model-theoretic approach:** Focus on the things to be described instead of a specific description language
- **Defines capabilty semantics in WSMO** in a formal way
- **Indepence of state desription language**: Reusable across various ontology languages (e.g. WSML variants, F-Logic, KIF, …)
  - Focus on the dynamics that is added by WS to a static world
  - Semantic layering similar to the syntactic layering of Capability Descriptions on top of (static) logical expressions
- **Can for other state-based frameworks as well**, especially when they strive for genericity wrt. static languages (e.g. OWL-S)
- Model designed to be modular and extensible
- Showed the use of the framework in a specific use case that is interesting in itself: **Semantic Analysis of Functional Descriptions**

Future Work

- Instantiation of our model for WSML-Family of Ontology Languages
- A more restricted style of Capability Semantics is possible: ,,and‘‘-style
  - Disallows state-sequences that start in a state where the precondition is violated
  - Can be considered as an implementers point of view in a contractual interface
  - Could be useful to make a real semantic distinction between precondition and assumptions as well as postconditions and effects in WSMO
- Extend Capability Description Elements
  - Complete / Incomplete Capability Descriptions
  - Execution Invariants
- So far only considered functional descriptions. What about further behavioural aspects of WS?
  - Integration with Choreography Descriptions, Multi-party Interactions