

**Information technology – Metamodel framework for interoperability  
(MFI) Part 7: Metamodel for service registration**

**Warning**

This document is not an ISO International Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an International Standard.

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

### Copyright notice

This ISO document is a working draft or committee draft and is copyright-protected by ISO. While the reproduction of working drafts or committee drafts in any form for use by participants in the ISO standards development process is permitted without prior permission from ISO, neither this document nor any extract from it may be reproduced, stored or transmitted in any form for any other purpose without prior written permission from ISO.

Requests for permission to reproduce this document for the purpose of selling it should be addressed as shown below or to ISO's member body in the country of the requester:

[Indicate the full address, telephone number, fax number, telex number, and electronic mail address, as appropriate, of the Copyright Manger of the ISO member body responsible for the secretariat of the TC or SC within the framework of which the working document has been prepared.]

Reproduction for sales purposes may be subject to royalty payments or a licensing agreement.

Violators may be prosecuted.

# Contents

Page

Foreword .....	v
Introduction.....	vi
<b>1 Scope .....</b>	<b>1</b>
<b>2 Conformance .....</b>	<b>1</b>
<b>2.1 General .....</b>	<b>1</b>
<b>2.2 Degree of conformance .....</b>	<b>2</b>
<b>2.2.1 General .....</b>	<b>2</b>
<b>2.2.2 Strictly conforming implementation .....</b>	<b>2</b>
<b>2.2.3 Conforming implementation.....</b>	<b>2</b>
<b>2.3 Implementation Conformance Statement (ICS).....</b>	<b>2</b>
<b>3 Normative references .....</b>	<b>3</b>
<b>4 Terms, definitions and abbreviated terms .....</b>	<b>3</b>
<b>4.1 Terms and definitions .....</b>	<b>3</b>
<b>4.2 Broad terms .....</b>	<b>3</b>
<b>4.3 Abbreviated terms .....</b>	<b>3</b>
<b>5 Structure of MFI Service registration .....</b>	<b>4</b>
<b>5.1 Overview of MFI Service registration .....</b>	<b>4</b>
<b>5.2 Relationship between MFI service registration and other parts in MFI .....</b>	<b>5</b>
<b>5.3 MFI Service registration.....</b>	<b>6</b>
<b>5.3.1 Service .....</b>	<b>6</b>
<b>5.3.2 Quality_Property.....</b>	<b>6</b>
<b>5.3.3 QoSAssertion.....</b>	<b>6</b>
<b>5.3.4 LogicalExpr .....</b>	<b>6</b>
<b>5.3.5 Predicate.....</b>	<b>7</b>
<b>5.3.6 Argument.....</b>	<b>7</b>
<b>5.3.7 Operation.....</b>	<b>8</b>
<b>5.3.8 Precondition.....</b>	<b>8</b>
<b>5.3.9 Postcondition.....</b>	<b>8</b>
<b>5.3.10 Input.....</b>	<b>9</b>
<b>5.3.11 Output.....</b>	<b>9</b>
<b>Annex A (informative) List of existing service description languages .....</b>	<b>10</b>
<b>Annex B (informative) Example .....</b>	<b>11</b>
<b>B.1 Example of WSMO Service Registration .....</b>	<b>11</b>
<b>B.2 Example of WADL Service Registration .....</b>	<b>13</b>

## Figures

Figure 1 — Scope of MFI service registration .....	1
Figure 2 - Metamodel of MFI service registration .....	5
Figure 3 – Relationship between MFI service registration and other parts in MFI .....	5

## Tables

Table A.1 – List of Existing Service Description Languages .....	10
Table B.1.1 - Code of “Book Ticket Web Service” (fragment) .....	11
Table B.1.2 – <i>Service</i> Registration Information .....	12
Table B.1.3 – <i>Operation00</i> Registration Information .....	12
Table B.1.4 <i>Input &amp; Output</i> Registration Information .....	12
Table B.1.5 – <i>Precondition02</i> Registration Information .....	13
Table B.2.1 – Code of “Amazon search items Web Service” .....	13
Table B.2.2 - “Amazon search items Web Service” according to MFI Service Registration .....	15

## Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 19763-7 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information Technology*, Subcommittee SC 32, *Data management and Interchange*.

ISO/IEC 19763 consists of the following parts, under the general title *Information technology — Metamodel framework for interoperability (MFI)*:

*Part 1: Reference model*

*Part 2: Core model*

*Part 3: Metamodel for ontology registration*

*Part 4: Metamodel for model mapping*

*Part 5: Metamodel for process model registration*

*Part 6: Registration Process*

*Part 7: Metamodel for service registration*

*Part 8: Metamodel for role and goal registration*

*Part 9: Registry of Registries*

*TR: Using RGPS for on demand model selection*

## Introduction

Due to the spread of e-Business and e-Commerce over the Internet, the effective interchange of business transactions or other related information across countries and cultures is an important concern for people in both the IT industry and other non-IT industries.

With the rapid development of SOA (Service Oriented Architecture) and SOC (Service Oriented Computing), more and more computing resources are presented in the form of Web services. Meanwhile, business integration based on Web services is becoming a popular application development method. Web service is a kind of Web based application, which encapsulates certain computing module and is designed to support interoperable machine-to-machine interaction over a network.

In Web service registration and management, UDDI is a widely applied specification, which provides basic support for publishing and discovering Web services within and across enterprises. Keyword matching is the basic service discovery method in UDDI, thus the discovery results will be inevitably inaccurate, and the discovery process will be time-consuming. When business information interchange and integration becomes increasingly frequent, major work in service discovery should be processed by machine, therefore, it is necessary to semantically describe service information including functional and non-functional information, and provide corresponding registration and management mechanism.

This part of ISO/IEC 19763 intends to provide a generic framework for registering functional and nonfunctional information of services in an explicit way.

# Information technology – Metamodel framework for interoperability (MFI) – Part 7: Metamodel for service registration

## 1 Scope

The primary purpose of the multipart standard ISO/IEC 19763 is to specify a metamodel framework for interoperability. This part of ISO/IEC 19763 specifies a metamodel for registering services that can enable users to discover appropriate services.

The metamodel that this part specifies is intended to promote interoperation between various services.

It does not specify industry categorization of services and contact information of service providers, which are specified in UDDI.

Figure 1 shows the scope of this part of ISO/IEC 19763.

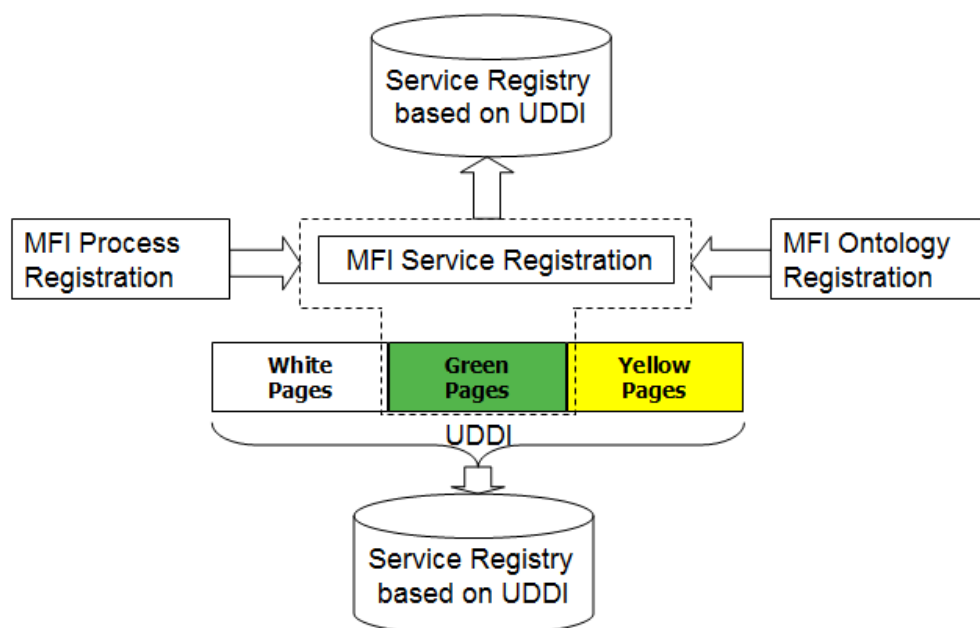


Figure 1 — Scope of MFI service registration

## 2 Conformance

### 2.1 General

An implementation claiming conformance with this part of ISO/IEC 19763 shall support the metamodel specified in 5.3, depending on a degree of conformance as described below.

## 2.2 Degree of conformance

### 2.2.1 General

The distinction between “strictly conforming” and “conforming” implementations is necessary to address the simultaneous needs for interoperability and extensions. This part of ISO/IEC 19763 describes specifications that promote interoperability. Extensions are motivated by needs of users, vendors, institutions and industries, but are not specified by this part of ISO/IEC 19763.

A strictly conforming implementation may be limited in usefulness but is maximally interoperable with respect to this part of ISO/IEC 19763. A conforming implementation may be more useful, but may be less interoperable with respect to this part of ISO/IEC 19763.

### 2.2.2 Strictly conforming implementation

A strictly conforming implementation

- a) shall support the metamodel specified in 5.1;
- b) shall not support any extensions to the metamodel specified in 5.1.

### 2.2.3 Conforming implementation

A conforming implementation

- a) shall support the metamodel specified in 5.1;
- b) may support extensions to the metamodel specified in 5.1 that are consistent with the metamodel specified in 5.1.

## 2.3 Implementation Conformance Statement (ICS)

An implementation claiming conformance with this part of ISO/IEC 19763 shall include an Implementation Conformance Statement stating

- a) whether it is a strictly conforming implementation or a conforming implementation (2.2);
- b) what extensions are supported if it is a conforming implementation.

### 3 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 19763-1, Information technology – Metamodel framework for interoperability (MFI) – Part 1: Reference model

ISO/IEC 19763-2, Information technology – Metamodel framework for interoperability (MFI) – Part 2: Core model

ISO/IEC 19763-3, Information technology – Metamodel framework for interoperability (MFI) – Part 3: Metamodel for ontology registration

ISO/IEC 19763-5, Information technology – Metamodel framework for interoperability (MFI) – Part 5: Metamodel for process model registration

ISO/IEC 19763-8, Information technology – Metamodel framework for interoperability (MFI) – Part 8: Metamodel for Role & Goal registration

## 4 Terms, definitions and abbreviated terms

### 4.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 19763-1, ISO/IEC 19763-3, ISO/IEC 19763-5 and the following apply.

### 4.2 Broad terms

#### 4.2.1

##### **Service**

a kind of Web based application, which encapsulates certain computing module and can be accessed by certain interface.

### 4.3 Abbreviated terms

#### **MFI Core**

ISO/IEC 19763-2, Information technology –Metamodel Framework for Interoperability – Part-2 : Core model

#### **MFI Service registration**

ISO/IEC 19763-7, Information technology – Metamodel framework for interoperability (MFI) – Part 7: Metamodel for service registration

#### **MFI Ontology registration**

ISO/IEC 19763-3, Information technology – Metamodel framework for interoperability (MFI) – Part 3: Metamodel for ontology registration

#### **MFI Process registration**

ISO/IEC 19763-5, Information technology –Metamodel Framework for Interoperability (MFI) – Part-5: Metamodel for process models registration

#### **UDDI**

Universal Description, Discovery, and Integration

**WSDL**

Web Service Description Language

**OWL-S**

Web Ontology Language for Services

**WSMO**

Web Service Modelling Ontology

**WADL**

Web Application Description Language

## 5 Structure of MFI Service registration

### 5.1 Overview of MFI Service registration

This part of MFI describes administrative information of services, including functional and nonfunctional description.

As Figure 2 shows, MFI service registration metamodel is provided to capture the common functional and nonfunctional information of various kinds of services. **Service** aggregates multiple **Quality\_Property** and multiple **Operation**. The nonfunctional description is depicted by **Quality\_Property**, which can be used to represent the quantitative or qualitative value of service in certain aspect such as response time, cost, reliability, and so on. An observable property of a service that can be retrieved at execution time is described by **QoSAssertion**, which may have one or multiple **LogicalExpr**. A **LogicalExpr** has one **predicate**, and one or two **argument**.

As for the functional description, since a service is an independent and modular component and it can be accessed only by interfaces, a service's functionality is embodied in multiple **Operation**. **Operation** is used to denote the execution action. An operation has **Precondition** and **Postcondition**, which are essential to invoking and executing the service. **Precondition** is used to specify the state that should be satisfied before a service is invoked, while **Postcondition** is used to specify the state that should be satisfied after a service is invoked successfully. Same as **QoSAssertion**, **Precondition** and **Postcondition** are also expressed with **LogicalExpr**. Apart from **Precondition** and **Postcondition**, **Operation** has **Input** and **Output**. **Input** is constrained by **Precondition**, and **Output** is constrained by **Postcondition**.

In order to add semantic information for service, **Input**, **Output**, **Precondition**, **Postcondition**, **Operation**, **Predicate** and **Argument** have a property of **annotation**, which are annotated by the concepts of domain ontology.

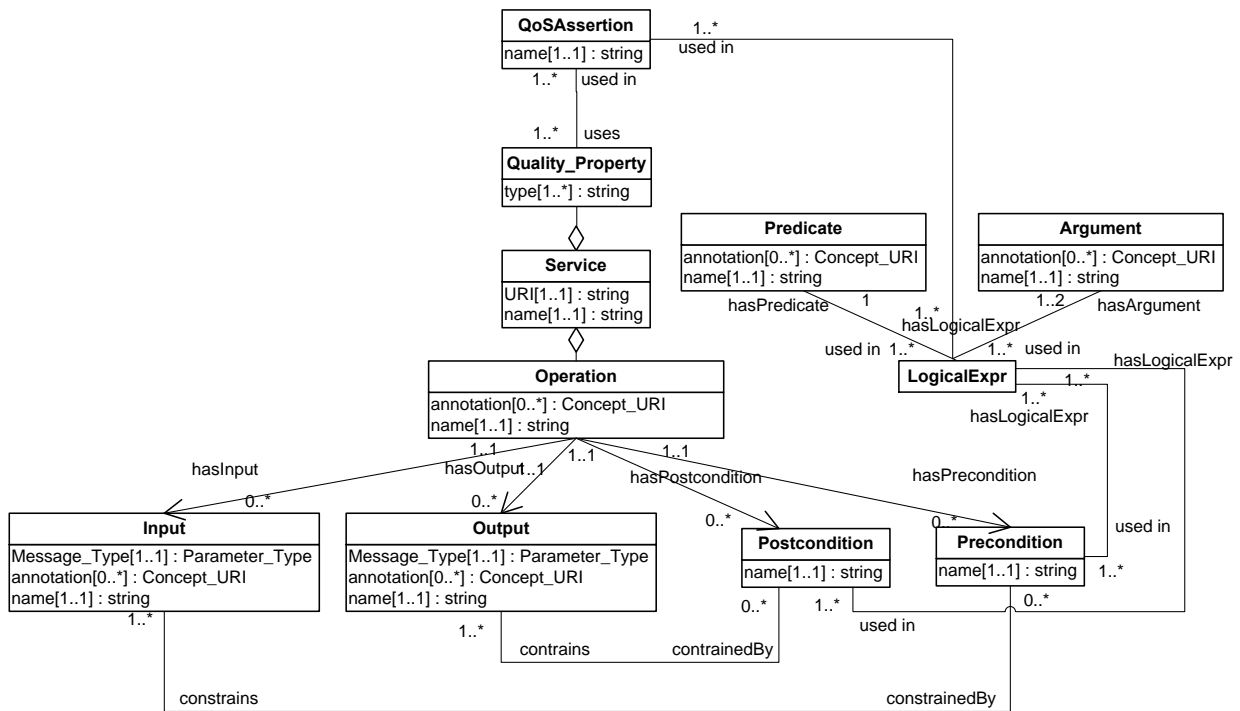


Figure 2 - Metamodel of MFI service registration

## 5.2 Relationship between MFI service registration and other parts in MFI

Figure 3 shows the relationship between MFI service registration and other parts in MFI. That is, **Service** can perform one **Process** specified in MFI process registration. **Service** can achieve one **Goal** and be invoked by one or multiple **Role** specified in MFI Role & Goal registration.

Some metaclasses will inherit `Ontology_Atomic_Construct` in MFI ontology registration such as **Operation**. See Figure 2, other metaclasses include **Input**, **Output**, **Predicate** and **Argument**.

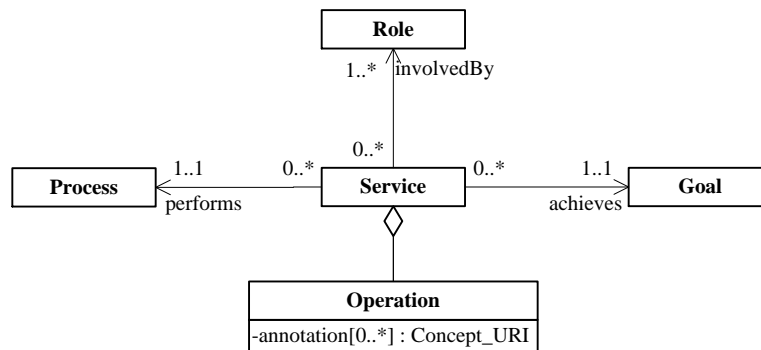


Figure 3 – Relationship between MFI service registration and other parts in MFI

### 5.3 MFI Service registration

#### 5.3.1 Service

Service is a metaclass representing an independent and modular component on the Internet.

Attribute	Data Type	Multiplicity	Description
name	String	1..1	Name of the corresponding service
URI	String	1..1	URI where the corresponding service exists
Reference	Class	Multiplicity	Description
Aggregation	Quality_Property	1..*	Certain quality feature of Non-functional description of the corresponding service
Aggregation	Operation	1..*	Operations that denote the execution actions of the service

#### Constraints

The value of attribute "URI" has to be unique in this metaclass.

#### 5.3.2 Quality\_Property

Quality\_Property is a metaclass that is used to represent a certain Non-functional feature for service, such as availability, response time, etc.

Attribute	Data Type	Multiplicity	Description
type	String	1..*	Type of the Non-functional feature
Reference	Class	Multiplicity	Description
used in	QoSAssertion	1..*	An atom description of a certain Non-functional property for service

#### 5.3.3 QoSAssertion

QoSAssertion is a metaclass that is used to represent an atom description of a certain Non-functional property for service.

Attribute	Data Type	Multiplicity	Description
name	String	1..1	Name of corresponding QoSAssertion
Reference	Class	Multiplicity	Description
uses	Quality_Property	1..*	A certain Non-functional feature for service, such as availability, response time, etc.
hasLogicalExpr	LogicalExpr	1..*	An assertion have one or multiple logical expressions

#### 5.3.4 LogicalExpr

LogicalExpr is a metaclass that represents logical expression within QoSAssertion, Precondition or Postcondition.

Reference	Class	Multiplicity	Description
hasPredicate	Predicate	1..1	A function that returns true or false. It contains one argument or links two arguments, which formulates a logical expression
hasArgument	Argument	1..2	Variable that is contained or linked by predicate in logical expression
used in	QoSAssertion	1..*	An atom description of a certain Non-functional property for service
used in	Precondition	1..*	State that should be satisfied before a service is invoked
used in	Postcondition	1..*	State that should be satisfied after a service is invoked successfully

### 5.3.5 Predicate

Predicate is a metaclass representing a function that returns true or false. It contains one argument or links two arguments, which formulates a logical expression.

Attribute	DataType	Multiplicity	Description
name	String	1..1	Name of corresponding predicate
annotation	Concept_URI	0..*	Annotation by the concept of domain ontology
Reference	Class	Multiplicity	Description
used in	LogicalExpr	1..*	Logical expression within QoSAssertion, Precondition or Postcondition

### 5.3.6 Argument

Argument is a metaclass representing variable that is contained or linked by predicate in logical expression.

Attribute	DataType	Multiplicity	Description
name	String	1..1	Name of corresponding argument
annotation	Concept_URI	0..*	Annotation by the concept of domain ontology
Reference	Class	Multiplicity	Description
used in	LogicalExpr	1..*	represents logical expression within QoSAssertion, Precondition or Postcondition

### 5.3.7 Operation

Operation is a metaclass that denotes the execution actions of the service.

Attribute	Data Type	Multiplicity	Description
name	String	1..1	Name of corresponding operation
annotation	Concept_URI	0..*	Annotation by the concept of domain ontology
Reference	Class	Multiplicity	Description
hasInput	Input	0..*	Information of message that the service consumes for its execution
hasOutput	Output	0..*	Information of message that the service generates after its execution
hasPrecondition	Precondition	0..*	State that should be satisfied before a service is invoked
hasPostcondition	Postcondition	0..*	State that should be satisfied after a service is invoked successfully.

### 5.3.8 Precondition

Precondition is a metaclass that specifies the state that should be satisfied before a service is invoked.

Attribute	Data Type	Multiplicity	Description
name	String	1..1	Name of corresponding Precondition
Reference	Class	Multiplicity	Description
hasLogicalExpr	LogicalExpr	1..*	logical expression within QoSAssertion, Precondition or Postcondition
constrains	Input	1..*	information of message that the service consumes for its execution

### 5.3.9 Postcondition

Postcondition is a metaclass that specifies the state that should be satisfied after a service is invoked successfully.

Attribute	Data Type	Multiplicity	Description
name	String	1..1	Name of corresponding Postcondition
Reference	Class	Multiplicity	Description
hasLogicalExpr	LogicalExpr	1..*	Logical expression within QoSAssertion, Precondition or Postcondition
constrains	Output	1..*	Information of message that the service generates after its execution

### 5.3.10 Input

Input is a metaclass specifying information of message that the service consumes for its execution.

Attribute	Data Type	Multiplicity	Description
Message_Type	Parameter_Type	1..1	Data type of message that is consumed for execution of service
annotation	Concept_URI	0..*	Annotation by the concept of domain ontology
name	String	1..1	Name of corresponding input
Reference	Class	Multiplicity	Description
constrainedBy	Precondition	0..*	State that should be satisfied before a service is invoked

### 5.3.11 Output

Output is a metaclass specifying information of message that the service generates after its execution.

Attribute	Data Type	Multiplicity	Description
Message_Type	Parameter_Type	1..1	Data type of message that is generated after execution of service
annotation	Concept_URI	0..*	Annotation by the concept of domain ontology
name	String	1..1	Name of corresponding output
Reference	Class	Multiplicity	Description
constrainedBy	Postcondition	0..*	State that should be satisfied after a service is invoked successfully

**Annex A (informative) List of existing service description languages**

MFI-7 is constructed based on some existing service description languages, which is shown in Table 1:

**Table A.1 – List of Existing Service Description Languages**

Name	Description
OWL-S	A language that conforms to “OWL Web Ontology Language for Web Service”, which specifying Semantic Markup for Web Services, 2004-11-02, W3C Member Submission
WSDL	Web Services Description Language, version 1.1, 2001-03-15, W3C Member Submission
WSMO	Web Service Modelling Ontology, 2005-06-03, W3C Member Submission
SA-WSDL	Semantic Annotations for WSDL and XML Schema, 2007-08-28, W3C Recommendation
SWSO	Semantic Web Service Ontology, version 1.0, 2005-09-09, W3C Member Submission
WADL	Web Application Description Language, 2009-08-31, W3C Member Submission
SA-Rest	SA-REST: Semantic Annotation of Web Resources, 2010-04-05, W3C Member Submission
Other	



?finalBalance= (?initialBalance - ?ticketPrice) and  
 ?creditCard[po#balance hasValue ?finalBalance].

The following shows how “Book Ticket Web Service” is registered in accordance with MFI Service registration.  
 The part of **Service** Registration Information is shown in Table B.1.2:

**Table B.1.2 – Service Registration Information**

<b>service</b>	
<b>URI</b>	http://example.org/bookticketWS
<b>name</b>	Book Ticket Web Service
<b>Aggregation</b>	Operation00

The part of **Operation00** Registration Information is shown in Table B.1.3:

**Table B.1.3 – Operation00 Registration Information**

<b>Operation</b>			
<b>name</b>	reservingTicket	<b>hasPrecondition</b>	Precondition00
<b>annotation</b>	http://www.daml.org/services/owl- s/1.0/Concepts.owl#reservingTicket		Precondition01
<b>hasInput</b>	Input00		Precondition02
	Input01	Precondition03	
<b>hasOutput</b>	Output00	<b>hasPostcondition</b>	Postcondition00
	Output01		Postcondition01

The part of **Input** and **Output** Registration information is shown in Table B.1.4:

**Table B.1.4 Input & Output Registration Information**

<b>Input00</b>	
<b>Name</b>	trip
<b>Message_Type</b>	Complex Type
<b>annotation</b>	http://www.daml.org/services/owl- s/1.0/Concepts.owl#tripFromAustria

<b>Input01</b>	
<b>Name</b>	creditCard
<b>Message_Type</b>	Complex Type
<b>annotation</b>	http://www.daml.org/services/owl- s/1.0/Concepts.owl#creditCard

Output00	
<b>Name</b>	ReservationHolder
<b>Message_Type</b>	String
<b>annotation</b>	http://www.daml.org/services/owl-s/1.0/Concepts.owl#ReservationHolder

Output01	
<b>Name</b>	Ticket
<b>Message_Type</b>	Complex Data Type
<b>annotation</b>	http://www.daml.org/services/owl-s/1.0/Concepts.owl#Ticket

One of precondition for “Book Ticket Web Service” is the credit card to be used must be valid. The registration information for the precondition in accordance with MFI Service Registration is shown in Table B.1.5:

**Table B.1.5 – Precondition02 Registration Information**

Precondition02	
<b>hasLogicalExpr</b>	LogicalExpr02

LogicalExpr02	
<b>hasPredicate</b>	Predicate02
<b>hasArgument</b>	Argument02

Predicate02	
<b>name</b>	isValid
<b>annotation</b>	http://www.daml.org/services/owl-s/1.0/Concepts.owl#isValid

Argument02	
<b>name</b>	creditCard
<b>annotation</b>	http://www.daml.org/services/owl-s/1.0/Concepts.owl#creditCard

## B.2 Example of WADL Service Registration

This service is described in Web Application Description Language (WADL). Presently, WADL only provides functional semantics about service. This service provided by Amazon is to search items in Amazon Internet shop. Brower will return item information after user inputs keywords or index of item. The detailed code is shown in Table B.2.1:

**Table B.2.1 – Code of “Amazon search items Web Service” (fragment)**

```
<resources base="http://webservices.amazon.com/onca/">
  <resource path="xml">
    <method href="#ItemSearch"/>
  </resource>
</resources>
```

```

<method name="GET" id="ItemSearch">
  <request>
    <param name="SubscriptionId" style="query" type="xsd:string" required="true"/>
    <param name="SearchIndex" style="query"
      type="aws:SearchIndexType" required="true">
      <option value="Books"/>
      <option value="DVD"/>
      <option value="Music"/>
    </param>
    <param name="Keywords" style="query" type="aws:KeywordList" required="true"/>
    <param name="ResponseGroup" style="query"
      type="aws:ResponseGroupType" repeating="true">
      <option value="Small"/>
      <option value="Medium"/>
      <option value="Large"/>
      <option value="Images"/>
    </param>
  </request>
  <response>
    <representation mediaType="text/xml" element="aws:ItemSearchResponse"/>
    <wadl:representation_variable name="productName"
      type="xsd:string" path="/aws:ItemSearchResponse /aws:productName"/>
    < wadl:representation_variable name="description"
      type="xsd:string" path="/aws:ItemSearchResponse /aws:description"/>
    < wadl:representation_variable name="price"
      type="xsd:decimal" path="/aws:ItemSearchResponse /aws:price"/>
  </response>
</method>

```

The registration information in accordance with MFI Service Registration is shown in Table B.2.2:

Table B.2.2 - “Amazon search items Web Service” according to MFI Service Registration

<b>operation</b>	
<b>name</b>	ItemSearch
<b>annotation</b>	http://webservices.amazon.com/onca/#ItemSearch
<b>hasInput</b>	Input00
	Input01
	Input02
	Input03
<b>hasOutput</b>	Output00
	Output01
	Output02

<b>Input00</b>	
<b>Name</b>	SubscriptionId
<b>Message_Type</b>	String
<b>annotation</b>	http://www.daml.org/services/owl-s/1.0/Concepts2.owl#SubscriptionId

<b>Input01</b>	
<b>Name</b>	SearchIndex
<b>Message_Type</b>	String
<b>annotation</b>	http://www.daml.org/services/owl-s/1.0/Concepts2.owl#SearchIndex

<b>Input02</b>	
<b>Name</b>	Keywords
<b>Message_Type</b>	String
<b>annotation</b>	http://www.daml.org/services/owl-s/1.0/Concepts2.owl#Keywords

<b>Input03</b>	
<b>Name</b>	ResponseGroup
<b>Message_Type</b>	String
<b>annotation</b>	http://www.daml.org/services/owl-s/1.0/Concepts2.owl#ResponseGroup

<b>Output00</b>	
<b>Name</b>	productName
<b>Message_Type</b>	String
<b>annotation</b>	http://www.daml.org/services/owl-s/1.0/Concepts2.owl#productName

<b>Output01</b>	
<b>Name</b>	description
<b>Message_Type</b>	String
<b>annotation</b>	http://www.daml.org/services/owl-s/1.0/Concepts2.owl#description

<b>Output02</b>	
-----------------	--

<b>Name</b>	price
<b>Message_Type</b>	decimal
<b>annotation</b>	<a href="http://www.daml.org/services/owl-s/1.0/Concepts2.owl#price">http://www.daml.org/services/owl-s/1.0/Concepts2.owl#price</a>