

The 2007 IEEE Conference on E-Commerce Technology (CEC 2007) and
IEEE Conference on Enterprise Computing, E-Commerce, and E-Services (EEE 2007)

1st Runner-up Team

Web Services Challenge 2007

Performance

University of Kassel

Steffen Bleul

Thomas Weise

Kurt Geihs

WSC-07 Co-chair, M. Brian Blake

CEC/EEE07 General Co-chair

Pressemitteilung – 15. August 2007

Wie bereits im letzten Jahr hat das Team Steffen Bleul und Thomas Weise, beide Doktoranden an der Universität Kassel im Fachgebiet Verteilte Systeme von Prof. Dr. Kurt Geihs, erfolgreich am internationalen Wettbewerb „Web Service Challenge“ teilgenommen. Im Vorjahr war das Kasseler Team bei seiner ersten Teilnahme gleich als Sieger hervorgegangen. In diesem Jahr wurde die Aufgabenstellung nochmals deutlich erschwert. Die Kasseler Software meisterte auch dieses Mal alle Hürden und belegte einen hervorragenden zweiten Platz, nur knapp geschlagen durch eine Mannschaft aus der Volksrepublik China. Auf dem dritten und vierten Platz folgten Teams aus Österreich und den USA. In dem zum vierten Mal stattfindenden, jährlichen Wettkampf im Rahmen der 9. IEEE Conference on E-Commerce Technology (CEC' 07) und der 4. IEEE Conference on Enterprise Computing, E-Commerce and E-Services (EEE ' 07) maßen sich Teilnehmer aus aller Welt auf dem Gebiet der automatischen Service-Komposition.

Ein Web Service ist eine über das Internet zugreifbare elektronische Dienstleistung, wie z.B. ein Routenplaner, eine Suchmaschine, ein Währungskonvertierer oder ein Bilderarchiv. Die Eigenschaften und Benutzungsvorschriften solcher Web Services werden mit einer genormten Beschreibungssprache festgelegt. Web Services sind aber nicht nur für das Internet von Bedeutung. Auch für den Informationsaustausch in und zwischen Unternehmen spielt diese Technik eine immer wichtigere Rolle.

Gegenstand des Wettbewerbs ist das Finden von passenden Dienstangeboten zu einer Dienstanfrage eines Kunden. Auf der Nachfragerseite werden komplexe Suchanfragen formuliert. Der Wettbewerb bestand darin, ein Programm zu entwickeln, das die Menge der Dienstangebote möglichst effizient und schnell durchsucht, um passende Dienste gemäß der vorgegebenen Suchanfrage ausfindig zu machen. Damit könnte z.B. ein Routenplaner selbständig den günstigsten Anbieter für Wettervorhersagen wählen und einbinden.

Mit Hilfe des Forschungsbeitrags der Kasseler Gruppe "Verteilte Systeme" können Programme benötigte Dienste automatisch mit hoher Geschwindigkeit finden und miteinander kombinieren. Die dazu verwendete Software stellt eine Weiterentwicklung des Systems dar, mit dem das Team im Vorjahr den Web Service Challenge gewann. Dieser kontinuierliche Erfolg sorgte für eine weitere erfreuliche Entwicklung für die beiden Teammitglieder: Die Berufung in das Organisationskomitee für den Web Service Challenge 2008, den sie mit ihrer Erfahrung und Kompetenz noch anspruchsvoller gestalten sollen.

Weitere Informationen erhalten Sie von Steffen Bleul, bleul@vs.uni-kassel.de.



The 2007 IEEE Conference on E-Commerce Technology (CEC 2007) and
IEEE Conference on Enterprise Computing, E-Commerce, and E-Services (EEE 2007)

1st Runner-up Team

Web Services Challenge 2007

Performance

University of Kassel

Steffen Bleul

Thomas Weise

Kurt Geihs

WSC-07 Co-chair, M. Brian Blake

CEC/EEE07 General Co-chair

INTRODUCING THE WEB SERVICE CHALLENGE

Progressing the State-of-the-Practice of Tools for Service-Oriented Computing

GENERAL

[Home](#)

[About the WS-Challenge](#)

[Contact Info](#)

[Click Here to Join News List](#)

WS-Challenge

[Participants](#)

[Repositories and Downloads](#)

[Results Archive](#)

VENUES

[EEE – 07](#)

[ICEBE – 07](#)

OUR SPONSORS

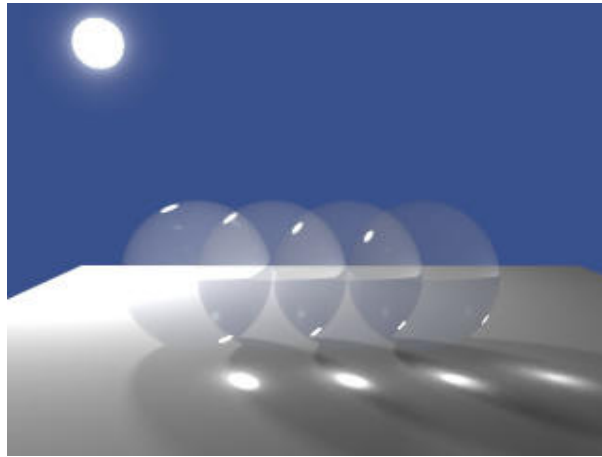


National Science Foundation
WHERE DISCOVERIES BEGIN

Award Information



Georgetown UNIVERSITY est. 1789



The Web Service Challenge (WS-Challenge)
 With growing acceptance of service-oriented computing, an emerging area of research is the investigation of technologies that will enable the discovery and composition of web services. Using the same approach as the popular [Trading Agent Competitions \(TAC\)](#), the Web Services Challenge is the original event geared towards the management of web services. The annual competition solicits industry and academic researchers that develop software components and/or intelligent agents that have the ability to discover pertinent web services and compose them to create higher-level functionality.

WS-Challenge @ EEE'07



[WSC '07 – TOKYO](#)

WS-Challenge '07 Rounds Are Complete.

See WSC '07 Webpage for More Information.

News and Information

- Check out the results page for WSC '07 results and other archives.
- WS-Challenge is funded by the [National Science Foundation](#)

<p>WS-Challenge Syntactic Competition Syntactic Repositories EEE'05 repository ICEBE'05 repository</p>	<p>WS-Challenge Software Several "initial" software packages are available for to help participants get started: Georgetown Java Software Installation Guide</p>	<p>WS-Challenge Background Brian Blake and William Cheung conceptualized the idea for the competition. The first year rules, repository, and software were developed at Georgetown University. Read more about it here.</p>
<p>Results from EEE-07 Speed Challenge Champion – Tsinghua University First Runner-Up – University of Kassel, Germany Second Runner-Up – Vitalab (University of Vienna)</p>	<p>Results from EEE-07 Architecture Challenge Champion – University of California, Irvine First Runner-Up – University of Charleston Second Runner-Up – Pennsylvania State University</p>	<p>Related Events Services Computing Conference (SCC) 2008 International Conference on Web Services (ICWS) 2008 International Conference on Service Oriented Computing (ICSOC)</p>

[CEC-07 & EEE-07](#)

**Web Services
Challenge 2007
Tokyo, Japan**

23-26 July 2007



Call for Participation

in the
2007 Web Services Challenge
in conjunction with
EEE'07 and CEC'07

23-26 July 2007, Tokyo, Japan

The Web Services Challenge (WS-Challenge) is a venue where researchers can collaborate on web service composition tools and techniques. The competition solicits industry and academic researchers that develop software components and/or intelligent agents that have the ability to discover pertinent web services and also compose them to create higher-level functionality.

The 2007 Web Services Challenge is the 4th challenge and will be co-located with the 2007 Conference on Electronic Commerce and the Conference on Enterprise Computing, E-Commerce and E-Services (CEC/EEE 2007). This fourth competition is more narrowly defined than that of the last year, which included both syntactic and semantic matching of WSDL part names. This year, the competition will focus exclusively on semantic composition of web service chains. Additionally, the challenge will incorporate use of OWL ontologies rather than XML Schema to define services and their relationships to each other. The participants will be required to determine relations between different types during the process of service composition.

The WS-Challenge invites the participation of students and researchers addressing the composition challenge stated above, with details posted on the Web Services Challenge website. The competition entails the submission of a four-page description of the approach to be submitted in early 2007. Papers should be formatted according to the [IEEE Conference Publishing Services guidelines](#), in an 8.5" x 11" two-column layout. After a peer-review process, the technical descriptions will be included in the conference proceedings of the joint conference. As a next step later in the spring of 2007, the participants will be required to provide a preliminary version of their software for a pre-competition evaluation stage. The pre-competition evaluation is required in order for teams to participate in the challenge taking place during the conference.

For submissions or inquiries, please contact:

David Cummings
Research Assistant
Department of Computer Science
Georgetown University
E-mail: wsctokyo07 [at] gmail.com

[Overview](#)

[Schedule & Deadlines](#)

[Student Travel Awards](#)

[Technical Details](#)

[Interface Package](#)

[Contact Information](#)

[Competition Results](#)

[CEC-07 & EEE-07](#)

**Web Services
Challenge 2007
Tokyo, Japan**

23-26 July 2007



Important Dates:

March 31, 2007

Submission of Technical Description (four pages, formatting conforming to IEEE CS Press Proceedings)

April 15, 2007

Notification of Acceptance of Technical Description

May 1, 2007

Final, Camera-Ready Version of Technical Description

June 4, 2007

Pre-Evaluation of Composition Software

July 23-26, 2007

Competition on Conference Site

[Overview](#)

[Schedule & Deadlines](#)

[Student Travel Awards](#)

[Technical Details](#)

[Interface Package](#)

[Contact Information](#)

[Competition Results](#)

**Web Services
Challenge 2007
Tokyo, Japan**

23-26 July 2007



Student Travel Awards

The 2007 Web Services Challenge will be awarding student scholarships in order to encourage and support participation by U.S.-based student teams. These awards will cover the expenses of travel, hotel accommodations, meals, and conference registration for those who would otherwise be unable to attend. These travel awards are sponsored by the National Science Foundation; teams from all U.S. colleges are eligible to apply.

Student teams should submit a Student Travel Award application that contains:

- o Contact Information. This should include the applicant's full name, school name and address, home and email addresses, and phone number(s) where the applicant can be reached during Spring 2007.
- o Applicants should state their interest in attending the competition.
- o Statement of Current Research Progress/Related Projects and Future Research Plans as related to the competition.

Teams should send in this application together with their technical description submission. Late submissions will be considered as funds remain available. Scholarship award recipients will be notified shortly before the pre-competition evaluation. Successful participation in the pre-competition evaluation is required for the travel funding. For more information, please contact:

M. Brian Blake
Department of Computer Science
Georgetown University
E-mail: [blakeb \[at\] cs.georgetown.edu](mailto:blakeb[at]cs.georgetown.edu)

[Overview](#)

[Schedule & Deadlines](#)

[Student Travel Awards](#)

[Technical Details](#)

[Interface Package](#)

[Contact Information](#)

[Competition Results](#)



Semantic Composition Description

NOTE: For information regarding changes since the 2006 competition, please read the "[What's New](#)" section below.

Semantic Representation

In the syntactic competition matching of services is based on the string equivalence of partnames of the input and output messages of a service. In the semantic competition we adopt the idea of so called Semantic Web Services that represent Web Services with a semantic description about the interface and its characteristics. Most approaches split the semantic description into two parts. The first part clarifies the interpretation about the concepts used in the particular application domain. The second part refers to a definition about the elements of a Web Service, such as defining that a Web Service features input parameters or provides specific preconditions. Several proposals exist already to standardise the semantic description about Web services, such as the semantically annotated WSDL approach (WSDL-S), the OWL Services ontology and the Web Service Modelling Ontology (WSMO).

For the semantic version of the challenge, we define a simplified version of Semantic Web Services. In 2007 WS-Challenge, the same set domain concepts is provided in two different formats, namely XML Schema as well as OWL. A simplified type hierarchy defined in XML Schema (with the same format as that provided in 2006 WS-Challenge) is provided as a common definition about the domain concepts. This type hierarchy will provide the type definitions about the input and output parameters of the considered Web services. The semantic description about the provided and required services will also use a representation that complies with the XML Schema. Consequently, we extend the matching of parameter names to the matching of parameter types defined by the XML Schema type hierarchy. Addition to the use of the XML Schema, the domain concepts will also be described in OWL format in case some of the teams find this format more convenient to work with. Note that the matching challenge does not cover the semantic descriptions about the e.g. the service category, pre- and postconditions as provided by OWL-S or the WSMO idea.

Semantical Discovery Sample

A Web Service may have a number of input and output parameters each referring to a type related to a type system potentially providing an inheritance hierarchy. Considering that a Web Service represents a programming interface we need to consider the contravariance of types for the interfaces, which result in the following rules for a successful type match of parameters:

- regarding one input of a service, the service requester must have specified a parameter type that is equivalent to or subsumed by the parameter type that the provider has specified. If the parameter type used by the invocation of the service requester does not provide the required or more special characteristics as specified by the provider, we cannot guarantee the successful execution of the service.
- regarding one output of a service, the direction of the subsumes-relation is opposite to the relation specified for the inputs. For the successful execution of a service, we regard the processing of outputs as irrelevant. However, the requester has defined requirements that we need to ensure. For example, the output of a service may represent the input of its subsequent service. Thus, the output type provided by the service must be equivalent to or be subsumed by the required output from the service requester.

In other words, the new matching challenge does not consider the string equivalence of parameter names but the contravariance of types to identify successful matches. Please note also that the invocation of a Web Service represents a call of a software API. Thus, we must also consider:

- that the service requester must show a successful match to all provided input-parameters with their equivalents or subsuming types. Otherwise a service call invokes an interface without setting all parameters which leads usually to a malfunction.
- that the service provider must at least match all required output-parameters with their equivalent or subsuming types. Otherwise the provided service does not fulfill the requirements of the service requester by not delivering one or more required output parameters.

The type system will be encoded using XML Schema. We consider XML Schema as a lightweight way to define complex types to establish a type hierarchy. In addition, we consider the information expressed by using XML Schema as easy to parse and process. Let us consider the following type example specified based on XML Schema as follows:

```
<complexType name="Address" >
  <sequence>
    <element name="name" type="string" minOccurs="0"/>
    <element name="street" type="string"/>
    <element name="city" type="string"/>
  </sequence>
</complexType>
```

In addition we define the following two subtypes using XML Schema:

```
<complexType name="US-Address" >
  <complexContent>
    <extension base="Address" >
      <sequence>
        <element name="state" type="US-State"/>
        <element name="zip" type="positiveInteger"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

[Overview](#)

[Schedule & Deadlines](#)

[Student Travel Awards](#)

[Technical Details](#)

[Interface Package](#)

[Contact Information](#)

[Competition Results](#)

```
</complexType>
```

and

```
<complexType name="EU-Address">
  <complexContent>
    <extension base="Address">
      <sequence>
        <element name="postcode" type="EU-Postcode"/>
      </sequence>
      <attribute name="export-code"
        type="positiveInteger" fixed="1"/>
    </extension>
  </complexContent>
</complexType>
```

The resulting subsumes relation is as such that the concept of Address subsumes the concepts of EU-Address and US-Address in the way that both subtypes show all characteristics of their supertype. For this example, we consider the following as successful match:

- o The provided service defines Address as input. Then we can identify a successful match, if the requester defines the equivalent type (Address) or subsumed types (EU-Address and US-Address) as the inputs he uses to invoke the service.
- o The provided service defines one of the both types EU-Address or US-Address as output. Then a successful match is considered, if the description of required service either defines the equivalent type, depending on what is provided either EU-Address or US-Address, or a concept that subsumes the provided output type. In this case, this would be the type Address.

Semantical Composition Sample

Along the lines of the Semantical Discovery Competition the composition competition can also be changed into a semantic version. Based on the matching rule defined above, the aim is to derive a minimal chain of services providing a match with the input and output parameter types of the query. To give an example, consider the following required service: a service requester seeks a service that returns an address for a given client name. Based on the Name type:

```
<complexType name="Name">
  <sequence>
    <element name="Lastname" type="string" minOccurs="1"/>
    <element name="Firstname" type="string" minOccurs="1"/>
  </sequence>
</complexType>
```

the following query has to be fulfilled based on the type Address introduced above:

```
<WSChallenge>
  <CompositionRoutine name="composition1-20-16-10">
    <Provided> Name </Provided>
    <Resultant> Address </Resultant>
  </CompositionRoutine>
</WSChallenge>
```

However, the repository does not contain such a service, but a service having an input parameter of type Name and an output parameter of type Clientid:

```
<simpleType name="Clientid">
  <restriction base="string"/>
</simpleType>
```

In addition, the repository contains another service having an input parameter of type Clientid and an output parameter of the desired type EU-Address. Then, the two services can be composed and provide a solution to the specified query.

Query and Solution Formats

The type definition specified in XML Schema is provided as a single file used by all WSDL files related to a particular repository. Besides of that the specification of the query and the expected result format are equivalent except that

- o the part names are replaced by complex type names defined in the XML Schema file, and
- o the type attribute in the solution format should be now Semantic Discovery Solution and respectively.

What's New

The 2007 Web Services Challenge incorporates several new aspects:

- o Entries must receive requests and return results via SOAP. See the [interface package](#) page for more information.
- o Solution chains of WSDL files will vary in length. More specifically, there will be multiple correct answers where some solutions are longer than others. Shorter chains can be perceived as being less expensive, thus will be considered the best answer. Entries will be judged on how quickly they are able to produce the "best" result. An [example query and solution](#) are available here.
- o Some scenarios in the challenge will require entries to handle branching. In 2006, all solution chains were sequential. In 2007, some scenarios may result in solutions chains that require branching and merging. Samples are forthcoming.
- o There will be scenarios created using OWL representations. Samples are forthcoming.



The WSC-07 Interface Package

Latest News

2007-06-03: small bugfix in the deployment part of the build script. Download recommended.

What is the Interface Package?

This software contains (1) a simple Web service implementation that implements the **required** interface of the query as a part of the Web Service Challenge '07 (WSC'07) conditions. Moreover, it also provides (2) a client that allows submitting queries and checking the returned results.

How it works

The software consists of two elements: one element is a simple Web service that implements the required interface. This simple Web service needs to be deployed and made ready for invocation. Its main purpose is to serve as the counterpart of the invocation by a client.

Important Note: for the competition, it is **not** mandatory that you actually use and deploy our `wsc07.aar`. The important condition is that your software works with our client test. The server implementation is just a dummy that serves as a counterpart to the client for demonstration only.

The Server Part

The server part is a simple Web service implementation that uses a deployed Axis2 as SOAP implementation in order to provide a reference implementation or the required interface for the software of all participants. The defined interface provides three operations:

```
long setWaitTime(long millis):
```

An operation to set the delay of the server used for the query operation for testing purposes. This operation is used also to check whether the Web service is ready. Therefore, the participant's software should implement this method **without** setting an actual delay for setting the query.

```
String setBootstrap(String file1, String file2):
```

An operation to set the bootstrap information. This information covers two parameters: the file path where the XSD file can be found and the file path where the directory with the WSDL files can be found. Then a software can start the bootstrapping.

```
String performQuery(String queryString):
```

An operation to transfer the query as String in XML format. This method should be called asynchronously.

The Client Part

The second element is a testing client that allows the submission of queries to a Web service that implements the query interface as described by the WSC'07 conditions. The tool allows submitting a valid query. In addition, it should be capable of checking whether the result is correct or not (future work). And even more, the tool allows measuring the time of how long a request took to perform.

Currently, the tool looks like this (click to enlarge):

For the server testing, the tool provides three basic functions:

- o Checking whether the Web service is ready/available or not.
- o Submitting the bootstrapping information: the file path of Schema XSD file and the file path of the directory that contains the WSDL files.
- o Submitting the contents of a query file ("XML in a String") to the server as an asynchronous request.

Requirements

This software has been built with J2SE 5.0 as the platform. Furthermore the following software has been used:

- o [Apache Ant 1.6.2](#) as a scripting tool
- o [Apache Tomcat 5.5.20](#) as the servlet container for the Web service
- o [Apache Axis2 1.0](#) as the Web service implementation framework

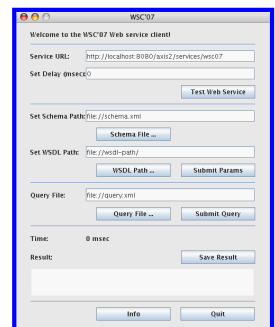
This software may also work with a 1.4.x JVM, older versions of Tomcat and older versions of Ant, but it has not been tested on such platforms.

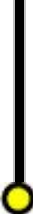
The Software

The software is provided to you as Apache Ant build project. You will need to install the mentioned prerequisites or compatible software accordingly. A description of the provided targets can be found in the project description of the build file. The software contains also the distribution of the Axis2 1.0 software including their license information for easy building.

The easiest way to deploy the Web service is to follow these steps:

- o Download the current release of the software [here](#).
- o Download and install Apache Tomcat 5.5.x



- 
- o Get the war-distribution of Apache Axis 2 1.x (it's available in the downloads section). IMPORTANT note: check to be sure that it is "Axis2", not just "Axis".
 - o Drop the `axis2.war` directly into the `webapps` directory of the Tomcat installation.
 - o Make sure that you have set `$CATALINA_HOME` correctly.
 - o Issue a nice `ant deploy` in the build project.

[CEC-07 & EEE-07](#)

**Web Services
Challenge 2007
Tokyo, Japan**

23-26 July 2007



WSC Co-Chairs:

M. Brian Blake
Department of Computer Science
Georgetown University
E-mail: blakeb [at] cs.georgetown.edu

Andreas Wombacher
Distributed Information Systems Laboratory
Ecole Polytechnique Federale de Lausanne
E-mail: andreas.wombacher [at] epfl.ch

Michael C. Jaeger
Faculty of EE and CS
Berlin University of Technology
E-mail: mcj [at] cs.tu-berlin.de

William K. Cheung
Centre of e-Transformation Research
Department of Computer Science
Hong Kong Baptist University
E-mail: william [at] comp.hkbu.edu.hk

Submissions and inquiries should be sent to:
David Cummings
Research Assistant
Department of Computer Science
Georgetown University
E-mail: wsctokyo07 [at] gmail.com

[Overview](#)

[Schedule & Deadlines](#)

[Student Travel Awards](#)

[Technical Details](#)

[Interface Package](#)

[Contact Information](#)

[Competition Results](#)



[CEC-07 & EEE-07](#)

**Web Services
Challenge 2007
Tokyo, Japan**

23-26 July 2007



Competition Results

Speed Category

- 1st Place: Tsinghua University
- 1st Runner-Up: University of Kassel
- 2nd Runner-Up: Vitalab (University of Vienna)

Architecture Category

- 1st Place: University of California-Irvine
- 1st Runner-Up: Charleston University
- 2nd Runner-Up: Pennsylvania State University

Congratulations, and thanks to all who participated in this year's competition!

[Overview](#)

[Schedule & Deadlines](#)

[Student Travel Awards](#)

[Technical Details](#)

[Interface Package](#)

[Contact Information](#)

[Competition Results](#)

[Competition Results](#)

Nicht erkannter MIME-Typ (application/zip). Klicken Sie [hier](#) für Anzeige

Nicht erkannter MIME-Typ (application/zip). Klicken Sie [hier](#) für Anzeige

INTRODUCING THE WEB SERVICE CHALLENGE

Progressing the State-of-the-Practice of Tools for Service-Oriented Computing

GENERAL

[Home](#)

[About the WS-Challenge](#)

[Contact Info](#)

[Click Here to Join News List](#)

WS-Challenge

[Participants](#)

[Repositories and Downloads](#)

[Results Archive](#)

VENUES

[EEE – 07](#)

[ICEBE – 07](#)

OUR SPONSORS



[Award Information](#)



Results Archive

EEE – 07

Speed

Champion – Tsinghua University, China

1st Runner-Up – University of Kassel, Germany

2nd Runner-Up – Vitalab, University of Vienna, Austria

Architecture

Champion – University of California, Irvine

1st Runner-Up – University of Charleston

2nd Runner-Up – Pennsylvania State University

EEE – 06 -- [Award Presentation Slide Show](#)

Semantic

Champion – Distributed System Group, University Kassel, Germany

1st Runner-Up – Electronic Commerce Competence Center, Austria

Syntactic

Champion – Tsinghua University, P.R.China

1st Runner-Up – Distributed Systems Group, TU Wien, Austria

ICEBE – 05

Champion – Fudan University, China

1st Runner-Up – Penn State University, USA

2nd Runner –Up – Electronic Commerce Competence Center, Austria

EEE – 05

Champion - Electronic Commerce Competence Center, Austria

1st Runner-Up - Industrial Informatics Group, Singapore Inst. of Manufacturing Technology

2nd Runner-Up - Whitestein Technologies AG, College of Charleston, Profactor Research

<p>WS-Challenge Syntactic Competition Syntactic Repositories EEE'05 repository ICEBE'05 repository</p>	<p>WS-Challenge Software Several "initial" software packages are available for to help participants get started: Georgetown Java Software Installation Guide</p>	<p>WS-Challenge Background Brian Blake and William Cheung conceptualized the idea for the competition. The first year rules, repository, and software were developed at Georgetown University. Read more about it here.</p>
<p>Results from EEE-07 Speed Challenge Champion – Tsinghua University First Runner-Up – University of Kassel, Germany Second Runner-Up – Vitalab (University of Vienna)</p>	<p>Results from EEE-07 Architecture Challenge Champion – University of California, Irvine First Runner-Up – University of Charleston Second Runner-Up – Pennsylvania State University</p>	<p>Related Events Services Computing Conference (SCC) 2008 International Conference on Web Services (ICWS) 2008 International Conference on Service Oriented Computing (ICSOC)</p>

INTRODUCING THE WEB SERVICE CHALLENGE

Progressing the State-of-the-Practice of Tools for Service-Oriented Computing

GENERAL

[Home](#)

[About the WS-Challenge](#)

[Contact Info](#)

[Click Here to Join News List](#)

WS-Challenge

[Participants](#)

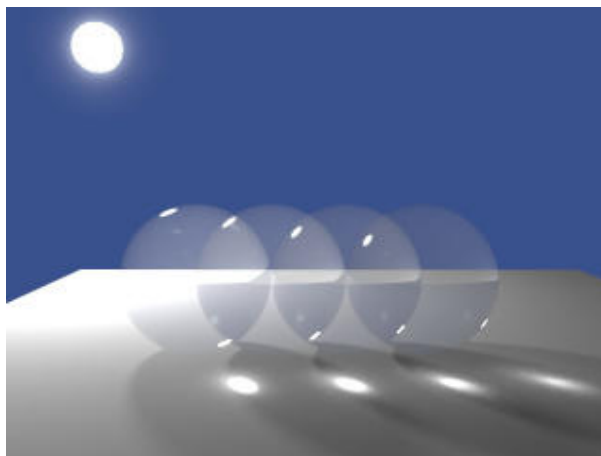
[Repositories and Downloads](#)

[Results Archive](#)

VENUES

[EEE – 07](#)

[ICEBE – 07](#)



WS-Challenge @ EEE'07



WSC '07 – TOKYO

The Web Service Challenge (WS-Challenge)

With growing acceptance of service-oriented computing, an emerging area of research is the investigation of technologies that will enable the discovery and composition of web services. Using the same approach as the popular [Trading Agent Competitions \(TAC\)](#), the Web Services Challenge is the original event geared towards the management of web services. The annual competition solicits industry and academic researchers that develop software components and/or intelligent agents that have the ability to discover pertinent web services and compose them to create higher-level functionality.

WS-Challenge '07 Rounds Are Complete.

See WSC '07 Webpage for More Information.

News and Information

- Check out the results page for WSC '07 results and other archives.
- WS-Challenge is funded by the [National Science Foundation](#)

OUR SPONSORS



Award Information



WS-Challenge Syntactic Competition

Syntactic Repositories
[EEE'05 repository](#)
[ICEBE'05 repository](#)

WS-Challenge Software

Several "initial" software packages are available for to help participants get started:

[Georgetown Java Software Installation Guide](#)

WS-Challenge Background

Brian Blake and William Cheung conceptualized the idea for the competition. The first year rules, repository, and software were developed at Georgetown University. Read more about it [here](#).

Results from EEE-07

Speed Challenge

Champion – Tsinghua University

First Runner-Up – University of Kassel, Germany

Second Runner-Up – Vitalab (University of Vienna)

Results from EEE-07

Architecture Challenge

Champion – University of California, Irvine

First Runner-Up – University of Charleston

Second Runner-Up – Pennsylvania State University

Related Events

[Services Computing Conference \(SCC\) 2008](#)

[International Conference on Web Services \(ICWS\) 2008](#)

[International Conference on Service Oriented Computing \(ICSOC\)](#)

The EEE-05 Challenge: A New Web Service Discovery and Composition Competition

M. Brian Blake
Georgetown University Washington, DC, USA
blakeb@cs.georgetown.edu

Dr. Kwok Ching Tsui
The Hong Kong and Shanghai Banking Corporation (HSBC)
kwokchingtsui@hsbc.com.hk

Andreas Wombacher
IPSI, Germany
andreas.wombacher@ipsi.fraunhofer.de

Abstract

With growing acceptance of service-oriented computing, an emerging area of research is the investigation of technologies that will enable the discovery and composition of web services. Using the same approach as the popular Trading Agent Competitions (TAC), the EEE-05 Web Services Challenge is the first event geared towards the management of web services. The competition solicits industry and academic researchers that develop software components and/or intelligent agents that have the ability to discover pertinent web services and also compose them to create higher-level functionality. This paper describes the competition details for this first year and expectations for future events.

1. Introduction

The purpose of the EEE-05 Challenge is to establish a venue where researchers can collaborate on implementations in the web service composition domain. The results from the competition can be used as performance baselines for other researchers. In addition, software design approaches can be demonstrated, enhanced, and disseminated each year as the competition evolves.

The objective of the competition, in the first year, is to encourage participants to concentrate on syntactical matching and chaining for Web Service Description Language (WSDL) documents. A successful software entry will be able to accurately and efficiently find services using the WSDL part names underlying input and output messages. Secondly, entry software is required to create chains of services by linking output part names to the subsequent input part names. The intent of the first year is for participants to create part name matching components/agents. The software created in the first year will set the foundation for

later years of this competition (i.e. each year with more technical rigor). Each year entry software should become more efficient.

2. The First Year

In this first year of competition, the web services repository will be based on WSDL 1.1. The participants were provided with two tutorials sites [2][3]. The organizers also suggested the use of the Microsoft .Net IDE as an editor for WSDL documents. The following sections describe the samples repository, discovery sample, and composition sample.

2.1 Samples Repository and Input Files

The samples repository can be found at [4]. The 2005 competition concentrates only on messages (and their underlying part names) and ports. Concrete descriptions such as bindings and services will be the focus in later years of the competition.

The service repository contains over 100 services. Many services have logical part names, however other services may be auto-generated with part names with random combination of letters. Other services are sub-sets and variations of the *correct* services.

Participants will be provided with an XML file to initiate the discovery and composition routines in the competition. A sample of the XML request is shown in Table 1. In this first year, organizers will be flexible in allowing the competitors to reformat the XML file to best meet the entry software's front-end.

Entry software is required to execute on the designated competition workstation. In the first year, a Windows-based machine will be used, and participants are required to e-mail their system requirements prior to the competition.

Table 1. Sample Competition Input File.

```
<EEE05Challenge>
<DiscoveryRoutine>
  <Provided> partname1, partname2 </Provided>
  <Resultant> partname1 </Resultant>
</DiscoveryRoutine>
<CompositionRoutine>
  <Provided> partname1, partname2 </Provided>
  <Resultant> partname1, partname2 </Resultant>
</CompositionRoutine>
</EEE05Challenge>
```

2.2 An Example Discovery Routine

Participants will be asked to find a specific service that can fulfill a certain input and output criteria. Table 2 is a sample request file.

Table 2. Sample Discovery Request.

```
<DiscoveryRoutine>
  <Provided> foodPref, custStreetAddress , custCityAddress,
    custStateAddress, custZipAddress </Provided>
  <Resultant> restaurantName, restaurantID </Resultant>
</DiscoveryRoutine>
```

The most relevant service in the repository is the *findCloseRestaurant* service as shown in Table 3. The *findCloseRestaurant* service may be considered a bit over-qualified for the

requirements, but it does fulfill them. Only one service will accurately meet the requirements in the repository. However, other services that partially meet the requirement are also included in the repository. Only accurate matches will count. The discovery portion of the competition is used to evaluate the design and speed of the software entries to execute the matching.

Table 3. Snippet of findCloseRestaurant.

```
<message name="findCloseRestaurant_Request">
  <part name="custStreetAddress" type="xs:string"/>
  <part name="custCityAddress" type="xs:string"/>
  <part name="custStateAddress" type="xs:string"/>
  <part name="custZipAddress" type="xs:string"/>
  <part name="foodPref" type="xs:string"/>
</message>

<message name="findCloseRestaurant_Response">
  <part name="restaurantName" type="xs:string"/>
  <part name="restaurantID" type="xs:string"/>
  <part name="restaurantStreetAddress" type="xs:string"/>
  <part name="restaurantCityAddress" type="xs:string"/>
  <part name="restaurantStateAddress" type="xs:string"/>
  <part name="restaurantZipAddress" type="xs:string"/>
</message>
```

2.3 An Example Composition Routine

Participants will also be posed with a composition request (finding a specific sequence of services). The routine in Table 4 can be fulfilled using the sequence, *purchaseALT.wsdl->reserveRental.wsdl->reserveRoom.wsdl->createItinerary.wsdl*, as captured in [4].

Table 4. Sample Composition Request

```
<CompositionRoutine>
  <Provided> firstName, lastName, middleInitial, creditCardNum,
creditCardExp, creditCardSecID, departCity, departState, destCity,
destState, rentalPref , roomPref, hotelName </Provided>
  <Resultant> ItineraryURL </Resultant>
</CompositionRoutine>
```

The *reserveRental* and *reserveRoom* require the output of *purchaseALT*, while *createItinerary* requires information from *reserveRental*, *reserveRoom*, and *purchaseALT*. The competition will limit three services as predicates for a subsequent service, however, the software entries are advised to keep a running memory of available information. Participants should note that the most effective software will be combine front-to-back and back-to-front processing, perhaps simultaneously. In addition to the aforementioned more complex routine, other routines will require just two or three services with direct matchings.

3. Evaluation and Future Competitions

This first year will help to establish the most effective approach to evaluating the software. The initial evaluation approach will consist of a subjective score on the system design. Other aspects will be performance and accuracy. One idea is to allow the discovery and composition to proceed for a limited amount of time and count the accurate number of discoveries or compositions, respectively. Although a number of evaluation schemes will be tried in the first year, the competition will be run with the best intentions to keep the judging fair.

In second year, the competition will require participants to match part names that are not syntactically the

same. In subsequent years, services will have be composed with semantic languages such as OWL-S.

4. Acknowledgements

This competition has been greatly influenced by the support from and fruitful conversations with Dr. William K.W. Cheung of Hong Kong Baptist University, Dr. Eleni Stroulia of the University of Alberta, Dr. Terry Payne of the University of Southhampton, and Dr. Norman Sadeh of Carnegie Mellon University.

5. References

- [1] The Trading Agent Competition (2005): <http://www.sics.se/tac/page.php?id=1>
- [2] Microsoft Corporation: Web Service Description Language Tutorial, (2005) <http://msdn.microsoft.com/archive/default.asp?url=/archive/en-us/dnarxml/html/wsdlexplained.asp>
- [3] WWW Schools: Web Service Description Language Tutorial (2005), <http://www.w3schools.com/wsd/default.asp>
- [4] EEE05 Challenge Samples Repository (2005), http://cssun.georgetown.edu/~blakeb/EEE05/WSDL_Repos.zip

INTRODUCING THE WEB SERVICE CHALLENGE

Progressing the State-of-the-Practice of Tools for Service-Oriented Computing

GENERAL

[Home](#)

[About the WS-Challenge](#)

[Contact Info](#)

[Click Here to Join News List](#)

WS-Challenge

[Participants](#)

[Repositories and Downloads](#)

[Results Archive](#)

VENUES

[EEE – 07](#)

[ICEBE – 07](#)

OUR SPONSORS

Founders and Organizers

<p>M. Brian Blake Georgetown University 234 Reiss Science Building Washington, DC 20057 Email: blakeb@cs.georgetown.edu</p>	<p>William K. Cheung Centre of e-Transformation Research Department of Computer Science Hong Kong Baptist University Kowloon Tong Hong Kong, China Email: william@comp.hkbu.edu.hk</p>
<p>Andreas Wombacher Information Systems Group Department of Computer Science University of Twente P.O. Box 217 7500 AE Enschede, The Netherlands Email: a.wombacher@utwente.nl</p>	<p>Kwok-Ching Tsui The Hong Kong and Shanghai Banking Corporation (HSBC) Email: kwokchingtsui@hsbc.com.hk</p>

Program Committee and Advisors

<p>Fang Yan Rao IBM China Research Lab, China Email: raofy@cn.ibm.com</p>	<p>Peter Hrastnik Electronic Commerce Competence Center, Austria Email: peter.hrastnik@ec3.at</p>
--	---



Award Information



Eleni Stroulia
University of Alberta
Department of Computer Science
221 Athabasca Hall
Edmonton, AB, T6G 2E8, Canada
Email: stroulia@cs.alberta.ca

WS-Challenge Syntactic Competition

Syntactic Repositories
[EEE'05 repository](#)
[ICEBE'05 repository](#)

WS-Challenge Software

Several "initial" software packages are available for to help participants get started:

[Georgetown Java Software Installation Guide](#)

WS-Challenge Background

Brian Blake and William Cheung conceptualized the idea for the competition. The first year rules, repository, and software were developed at Georgetown University. Read more about it [here](#).

Results from EEE-07

Speed Challenge

Champion – Tsinghua University

First Runner-Up – University of Kassel, Germany

Second Runner-Up – Vitalab (University of Vienna)

Results from EEE-07

Architecture Challenge

Champion – University of California, Irvine

First Runner-Up – University of Charleston

Second Runner-Up – Pennsylvania State University

Related Events

[Services Computing Conference \(SCC\) 2008](#)

[International Conference on Web Services \(ICWS\) 2008](#)

[International Conference on Service Oriented Computing \(ICSOC\)](#)

INTRODUCING THE WEB SERVICE CHALLENGE

Progressing the State-of-the-Practice of Tools for Service-Oriented Computing

GENERAL

[Home](#)

[About the WS-Challenge](#)

[Contact Info](#)

[Click Here to Join News List](#)

WS-Challenge

[Participants](#)

[Repositories and Downloads](#)

[Results Archive](#)

VENUES

[EEE – 07](#)

[ICEBE – 07](#)

OUR SPONSORS



[Award Information](#)



WS-Challenge Participants

2007

Vitalab, University of Vienna

University of Kassel, Germany

University of Texas at Dallas, Texas, USA

Penn State University, Pennsylvania, USA

Tsinghua University, P.R.China

University of Charleston, South Carolina, USA

University of California Irvine, California, USA

2006

Distributed Systems Group, TU Wien, Austria

Distributed System Group, University of Kassel, Germany

University of Texas at Dallas, Texas, USA

Penn State University, Pennsylvania, USA

EC3 - Electronic Commerce Competence Center, Vienna, Austria

Singapore Institute of Manufacturing Technology, Singapore

Politecnico di Bari, Italy

IBM T. J. Watson Research Center

Tsinghua University, P.R.China

University of California Irvine

2005

Department of Computer Science & Engineering, Fudan University,
Shanghai, China

Penn State University

Naval Research Lab and West Virginia University

College of Charleston

Electronic Commerce Competence Center

Industrial Informatics Group, Singapore Institute of Manufacturing
Technology

WS-Challenge Syntactic Competition Syntactic Repositories EEE'05 repository ICEBE'05 repository	WS-Challenge Software Several "initial" software packages are available for to help participants get started: Georgetown Java Software Installation Guide	WS-Challenge Background Brian Blake and William Cheung conceptualized the idea for the competition. The first year rules, repository, and software were developed at Georgetown University. Read more about it here .
Results from EEE-07 Speed Challenge Champion – Tsinghua University First Runner-Up – University of Kassel, Germany Second Runner-Up – Vitalab (University of Vienna)	Results from EEE-07 Architecture Challenge Champion – University of California, Irvine First Runner-Up – University of Charleston Second Runner-Up – Pennsylvania State University	Related Events Services Computing Conference (SCC) 2008 International Conference on Web Services (ICWS) 2008 International Conference on Service Oriented Computing (ICSOC)

INTRODUCING THE WEB SERVICE CHALLENGE

Progressing the State-of-the-Practice of Tools for Service-Oriented Computing

GENERAL

[Home](#)

[About the WS-Challenge](#)

[Contact Info](#)

[Click Here to Join News List](#)

WS-Challenge

[Participants](#)

[Repositories and Downloads](#)

[Results Archive](#)

VENUES

[EEE – 07](#)

[ICEBE – 07](#)

OUR SPONSORS



[Award Information](#)



Technical Details and Downloads

Overall Technical Details

EEE – 05

[Sample Repositories](#)

[Sample Challenges](#)

[Expected Answers to Challenges \(text format\) or \(pdf format\)](#)

ICEBE – 05

[Sample Repositories, Sample Challenges, and Expected Answers to Challenges](#)

TEST DATA for ICEBE-05 (For Special Issue Submissions):

- [1. Discovery Challenge](#)
- [2. Composition \(1\) Challenge](#)
- [3. Composition \(2\) Challenge \(More Difficult\)](#)

WS-Challenge Syntactic Competition

Syntactic Repositories
[EEE'05 repository](#)
[ICEBE'05 repository](#)

WS-Challenge Software

Several "initial" software packages are available for to help participants get started:

[Georgetown Java Software Installation Guide](#)

WS-Challenge Background

Brian Blake and William Cheung conceptualized the idea for the competition. The first year rules, repository, and software were developed at Georgetown University. Read more about it [here](#).

Results from EEE-07	Results from EEE-07	Related Events
<p>Speed Challenge</p> <p>Champion – Tsinghua University</p> <p>First Runner-Up – University of Kassel, Germany</p> <p>Second Runner-Up – Vitalab (University of Vienna)</p>	<p>Architecture Challenge</p> <p>Champion – University of California, Irvine</p> <p>First Runner-Up – University of Charleston</p> <p>Second Runner-Up – Pennsylvania State University</p>	<p><u>Services Computing Conference (SCC) 2008</u></p> <p><u>International Conference on Web Services (ICWS) 2008</u></p> <p><u>International Conference on Service Oriented Computing (ICSOC)</u></p>

Copyright © 2006 Georgetown University. All Rights Reserved.
For questions about this web site, contact mfn3[AT]georgetown[DOT]edu

WEB SERVICE CHALLENGE 2006

Software Install Guide

Daniel R. Kahan
drk8@georgetown.edu

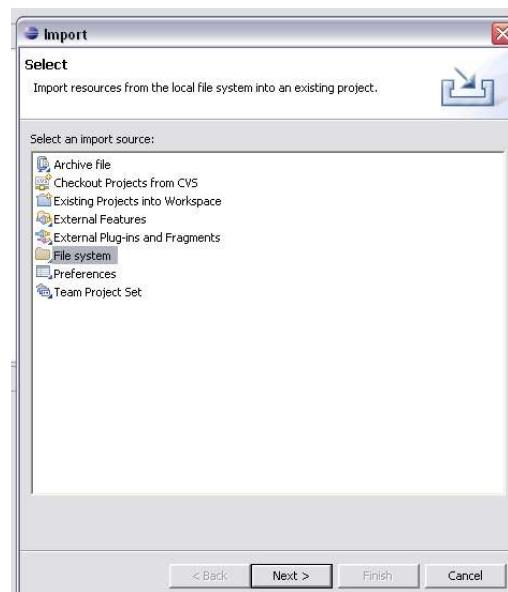
Requirements

JRE 1.4+ [<http://java.sun.com>]
Windows 2000, XP* [<http://www.microsoft.com>]
Eclipse 3.11 [<http://www.eclipse.org>]

* In theory, so long as the operating system supports Java and Eclipse, our software ought to be able to run on it. However, that theory has not been put to the test, so these directions are Windows-oriented.

IMPORTING THE PROJECT

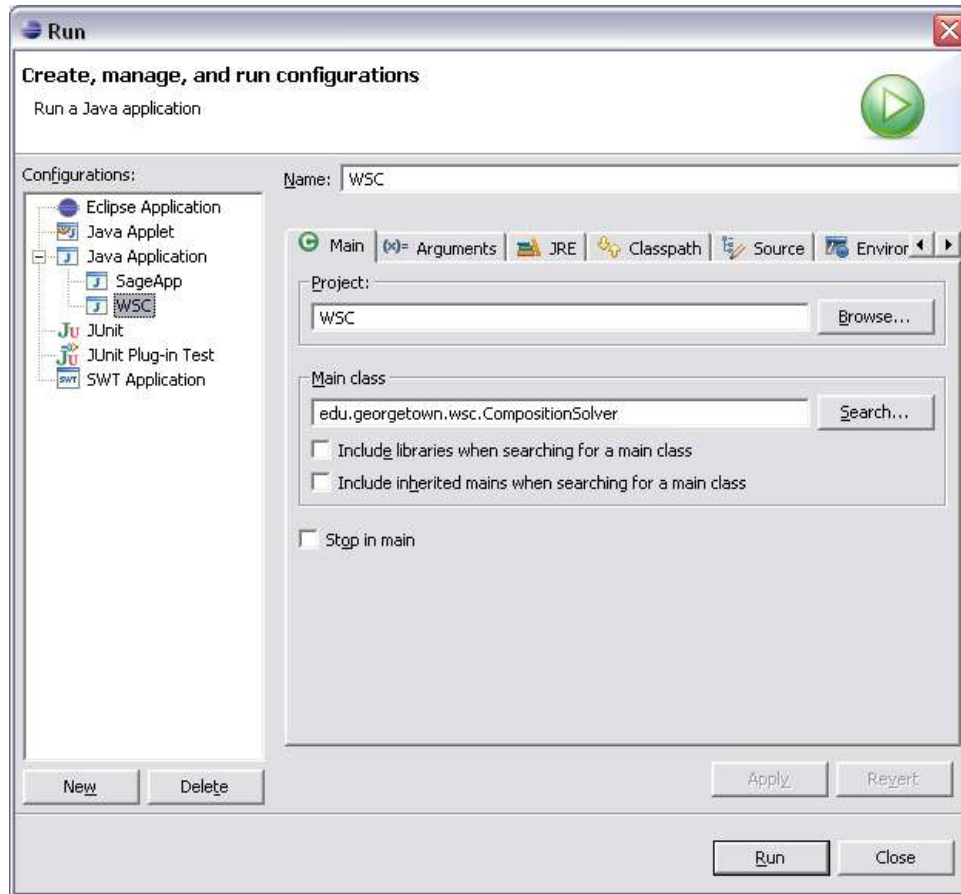
1. Unzip the project to `c:\wsc`, making sure to preserve the directory structure. Within the `wsc` directory, there should be two subdirectories, `src` and `rep`.
2. Launch Eclipse. Start a new **Java Project** in Eclipse and name it `wsc`.
3. Select **Import** from the **File** menu. Select **File System** when presented with the window below. Then click the **Next** button.



4. Once the **File System** dialog comes up, simply enter `c:\wsc\src\wsc` in the **From directory** textbox. Enter `wsc` in the **Into folder** textbox.

RUNNING THE PROJECT

1. Now that all of the files have been imported, it's time to configure Eclipse so we can run the project.
2. In Eclipse, go to the **Run** menu and select **Run...** In the right-hand pane, select **Java Application** and then click the **New** button. In the **Main** tab, select **wsc** as the project and **edu.georgetown.wsc.CompositionSolver** (not **CompositionSolverGUI**) as the **Main** class.



3. Now, the project should be ready to **Run**. Whenever you want to run the project in the future, you can simply click the green arrow (shown below) and select **wsc** as the configuration.



4. Once the project is running, you can select **CompositionRequest.xml** from the **C:\wsc** folder and **c:\wsc\rep** as the repository path. Modify the XML file to experiment with other routines. More discovery and composition routines will be available on the WS-Challenge web site in the near future [<http://ws-challenge.georgetown.edu>].

Web Services Challenge Technical Details

(Adopted from [EEE-05 Challenge](#))

1. Objectives

The objective of the competition, in this first year, is to encourage participants to concentrate on *syntactical matching* and *chaining* for Web Service Description Language (WSDL) documents. A successful software entry will be able to accurately and efficiently find services using the part names underlying their input and output messages.

Secondly, entries will have to create chains of services by linking output part names to the subsequent input part names. The intent of this competition is for participants to create part name matching components/agents that will set the foundation for later years of this competition (i.e. each year with more technical rigor). Participants should also focus on robust system design and efficient programming techniques.

2. WSDL Details

In this competition, the web services repository will be based on WSDL 1.1. Below are two tutorial sites that may be helpful for participants.

- <http://msdn.microsoft.com/archive/default.asp?url=/archive/en-us/dnarxml/html/wsdlexplained.asp>
- <http://www.w3schools.com/wSDL/default.asp>

FYI - Microsoft .Net IDE has a reasonable editor for WSDL documents.

3. Sample Repository

A sample repository can be downloaded at the following web site:

<http://www.comp.hkbu.edu.hk/~ctr/wschallenge>

Participants should notice that we are concentrating only on messages (and their underlying part names) and ports. Concrete descriptions such as *bindings* and *services* will be the focus in later years of the competition, but not considered in this competition.

The sample service repository have services with auto-generated with part names with random combination of letters. Other services will be added that are sub-sets and variations of the most correct services to use for a particular discovery of competition routines.

4. Sample Competition Routine

We will provide participants with an XML file to initiate the competition discovery and composition routines. A sample of the XML request is below.

```
<WSChallenge>
<DiscoveryRoutine>
  <Provided> partname1, partname2, partname3 </Provided>
  <Resultant> partname1 </Resultant>
```

```
</DiscoveryRoutine>
<CompositionRoutine>
  <Provided> partname1, partname2, partname3 </Provided>
  <Resultant> partname1, partname2, partname3 </Resultant>
</CompositionRoutine>
</WSChallenge>
```

All software must be able to execute on a designated competition workstation. Participants will want to send their system requirements to us via email prior to the competition. There will be a pre-competition arrangement which should be able to provide more feedback for refining their software.

5. Discovery Sample

Participants will be asked to find a specific service that can fulfill a certain input and output criteria.

The following requested routine can be fulfilled by findCloseRestaurant.wsdl:

```
<DiscoveryRoutine>
  <Provided> foodPref,custStreetAddress,custCityAddress,custStateAddress,custZipAddress </Provided>
  <Resultant> restaurantName,restaurantID </Resultant>
</DiscoveryRoutine>
```

findCloseRestaurant.wsdl (snippet of the messages)

```
<message name="findCloseRestaurant_Request">
  <part name="custStreetAddress" type="xs:string"/>
  <part name="custCityAddress" type="xs:string"/>
  <part name="custStateAddress" type="xs:string"/>
  <part name="custZipAddress" type="xs:string"/>
  <part name="foodPref" type="xs:string"/>
</message>
<message name="findCloseRestaurant_Response">
  <part name="restaurantName" type="xs:string"/>
  <part name="restaurantID" type="xs:string"/>
  <part name="restaurantStreetAddress" type="xs:string"/>
  <part name="restaturantCityAddress" type="xs:string"/>
  <part name="restaurantStateAddress" type="xs:string"/>
  <part name="restaurantZipAddress" type="xs:string"/>
</message>
```

As you can see, findCloseRestaurant is a bit over-qualified for the requirements, but it does fulfill them.

We anticipate that only one service will accurately meet the requirements in the repository. However,

other services may be in place that partially meets the requirements. Only accurate matches count.

This part of the competition will mostly judge the design and speed of the software entries to execute

the matching.

6. Composition Sample

Participants may also be posed with a composition request (finding a specific sequence of services).

The following requested routine can be fulfilled by:

purchaseALT.wsdl->reserveRental.wsdl-> reserveRoom.wsdl->createltinerary.wsdl:

```
<CompositionRoutine>
  <Provided> firstName,lastName,middleInitial,creditCardNum,creditCardExp,creditCardSecID,departCity,
    departState,destCity,destState,rentalPref,roomPref,hotelName </Provided>
  <Resultant> ItineraryURL </Resultant>
</CompositionRoutine>
```

As you can see, *reserveRental* and *reserveRoom* require the output of *purchaseALT*, while *createltinerary* requires information from *reserveRental*, *reserveRoom*, and *purchaseALT*.

7. Evaluation.

One aspect will be a subjective score on the system design. The other aspects will be performance and accuracy. More details about the judging process will later be posted at the competition web site.

8. Miscellaneous WSDL Details.

1. arrivalDateTime and departDateTme are in the same context for all services.
 - a. May decide to make unique date/time syntax for each service
 - b. Our students had some difficulty with non-unique dates and times for mapping.
2. All operations are SOAP request-response, this year.
3. All services have one binding to one port (one operation per WSDL)
4. Port name=WSDL name
5. Message parts are all of type <string>.
6. Software entries should keep a memory of part names during composition, sometimes

messages will have to be carried to services in subsequent stages of the routine.

7. No composition routine will require more than three services to build input message for a

subsequent service. (*We did this to help scope complexity.*)

8. We anticipate that participants will be provided with an XML file to do all discoveries for

the first part of the competition. The composition routine will be done on a separate run.

9. For Further Questions.

Please send emails to wschallenge@comp.hkbu.edu.hk for technical questions. It would also be advisable to contact us notifying your intention to compete.

[Updated on June 22, 2005]

Discovery

1. findCloseRestaurant
2. findISBN
3. findCloseCarDealer
4. findCloseHotel
5. scheduleDate
6. scheduleDoctor
7. scheduleSalon
8. getForecast
9. purchaseStock
10. getStockPrice
11. findThemePark
12. findCloseFlorist
13. findCloseAPT
14. purchaseLiftTicket

Composition

1. purchaseALT.wsdl->reserveRental.wsdl-> reserveRoom.wsdl->createItinerary.wsdl
2. findMostRelevantStocks->getStockPriceMany->performStockResearch->purchaseOptimalStock
3. findMostRelevantStock->getStockPrice->purchaseStock
4. getStockPrice->purchaseStock
5. purchaseFlowers
6. findCloseAttorney->scheduleAttorney
7. findCloseLibrary->findISBN->checkOutLibraryBook
8. findCloseVideoStore->rentVideo
9. findThemePark->purchaseThemeParkTicket
10. getForecast
11. findCloseFlorist->purchaseFlowers
12. findDate->scheduleDate->findDateRestaurant->reserveDateRestaurant
13. findCloseMountain->purchaseLiftTicket
14. findCloseVeterinarian->scheduleVeterinarian
15. scheduleShipment

Answer Guide Summary for the EEE-05 Challenge

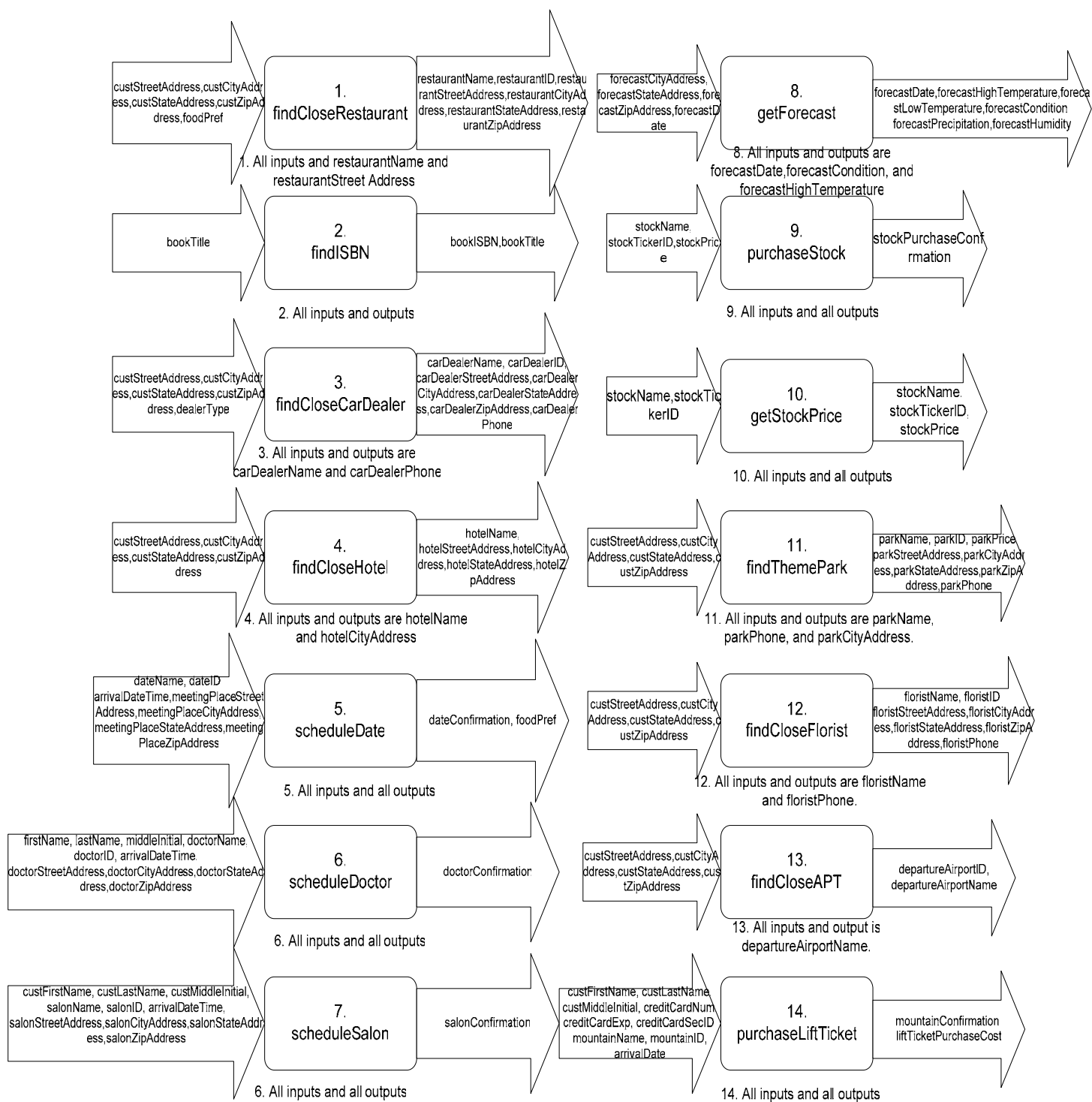
Discovery

1. findCloseRestaurant
2. findISBN
3. findCloseCarDealer
4. findCloseHotel
5. scheduleDate
6. scheduleDoctor
7. scheduleSalon
8. getForecast
9. purchaseStock
10. getStockPrice
11. findThemePark
12. findCloseFlorist
13. findCloseAPT
14. purchaseLiftTicket

Composition

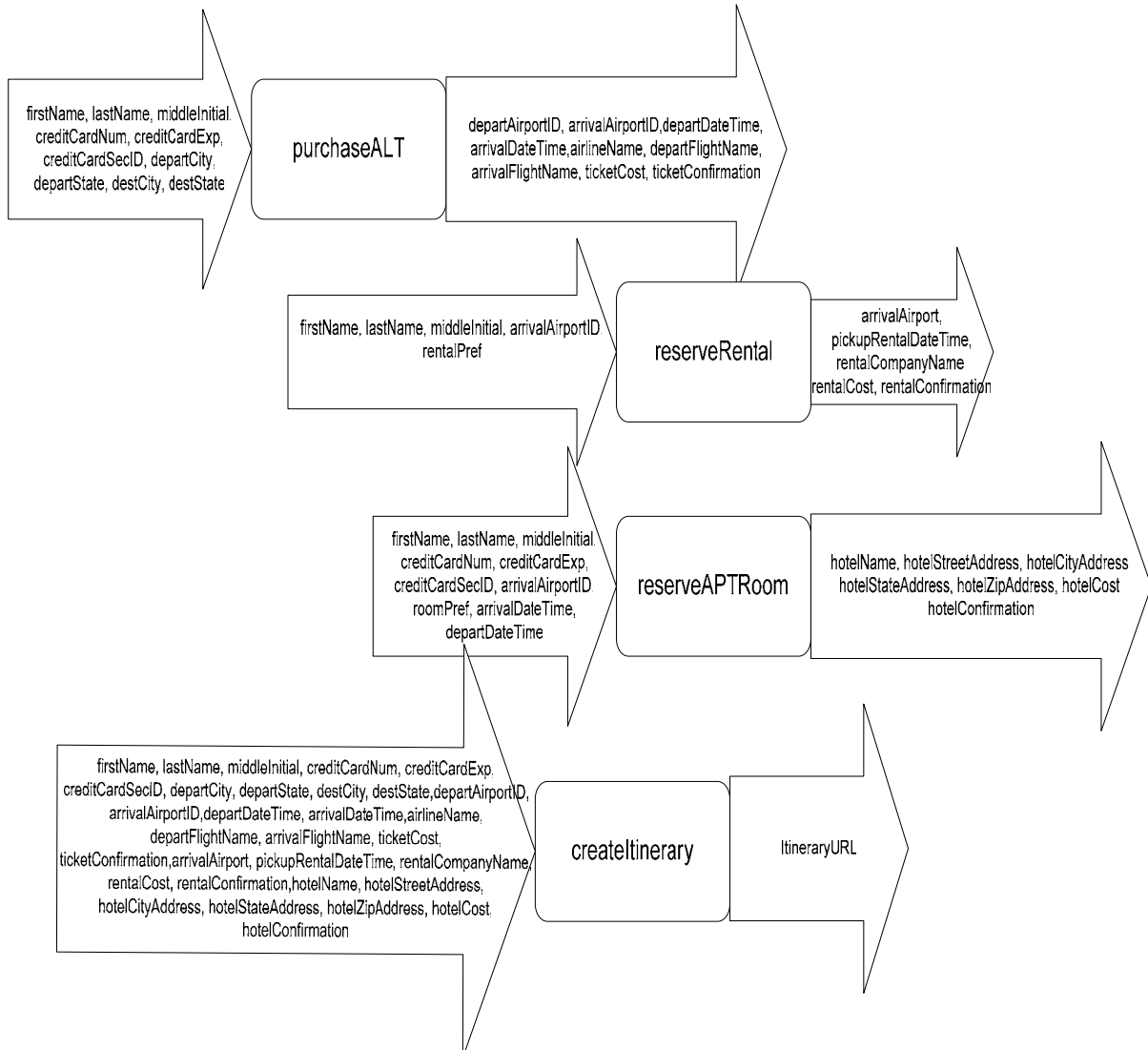
1. purchaseALT.wsdl->reserveRental.wsdl-> reserveRoom.wsdl->createItinerary.wsdl
2. findMostRelevantStocks->getStockPriceMany->performStockResearch->purchaseOptimalStock
3. findMostRelevantStock->getStockPrice->purchaseStock
4. getStockPrice->purchaseStock
5. purchaseFlowers
6. findCloseAttorney->scheduleAttorney
7. findCloseLibrary->findISBN->checkOutLibraryBook
8. findCloseVideoStore->rentVideo
9. findThemePark->purchaseThemeParkTicket
10. getForecast
11. findCloseFlorist->purchaseFlowers
12. findDate->scheduleDate->findDateRestaurant->reserveDateRestaurant
13. findCloseMountain->purchaseLiftTicket
14. findCloseVeterinarian->scheduleVeterinarian
15. scheduleShipment

Detailed Discovery Answer Guide (this corresponds to EEE05Challenge.xml)



Detailed Composition Answer Guide (this corresponds to EEE05Challenge.xml)

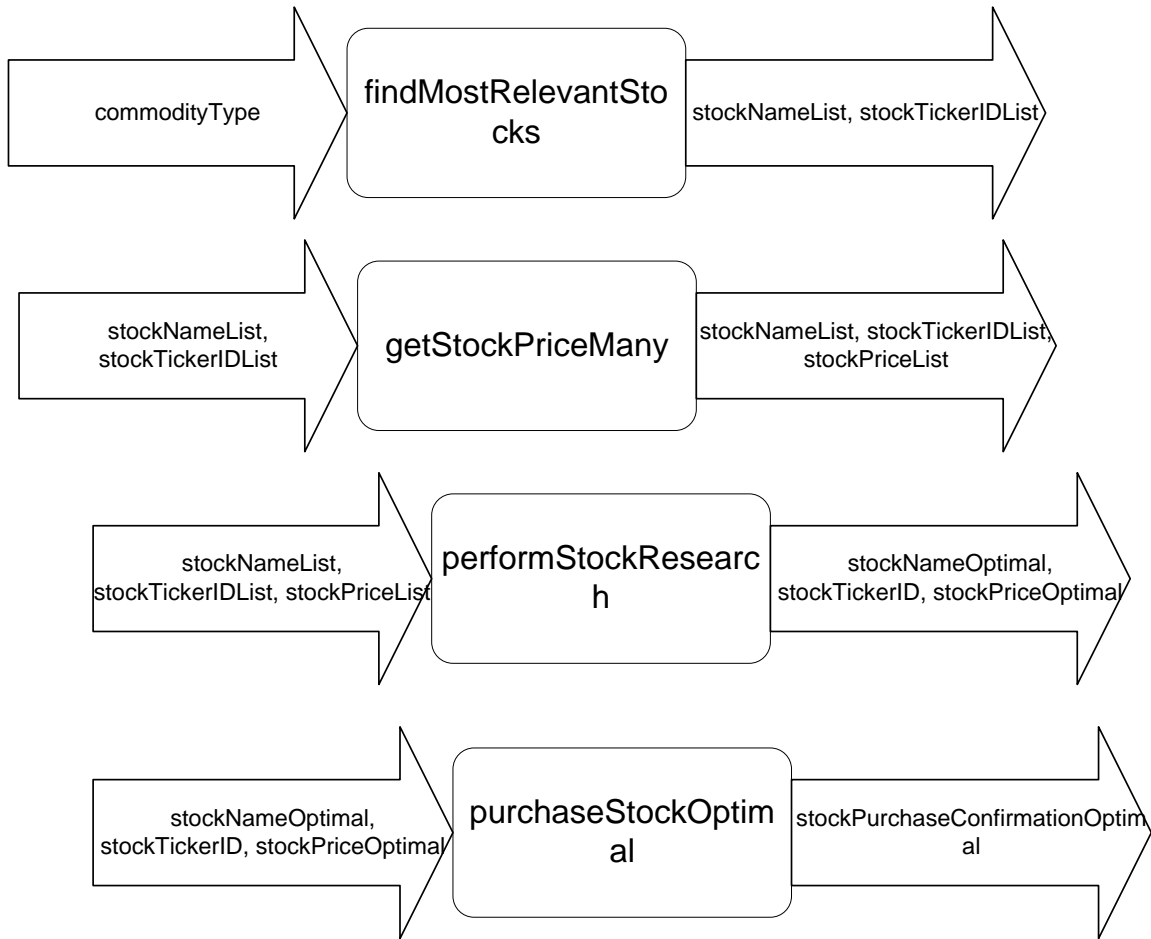
1.



1. **Inputs** = `firstName, lastName, middleInitial, creditCardNum, creditCardExp, creditCardSecID, departCity, departState, destCity, destState, rentalPref, roomPref`

Outputs = `ItineraryURL`

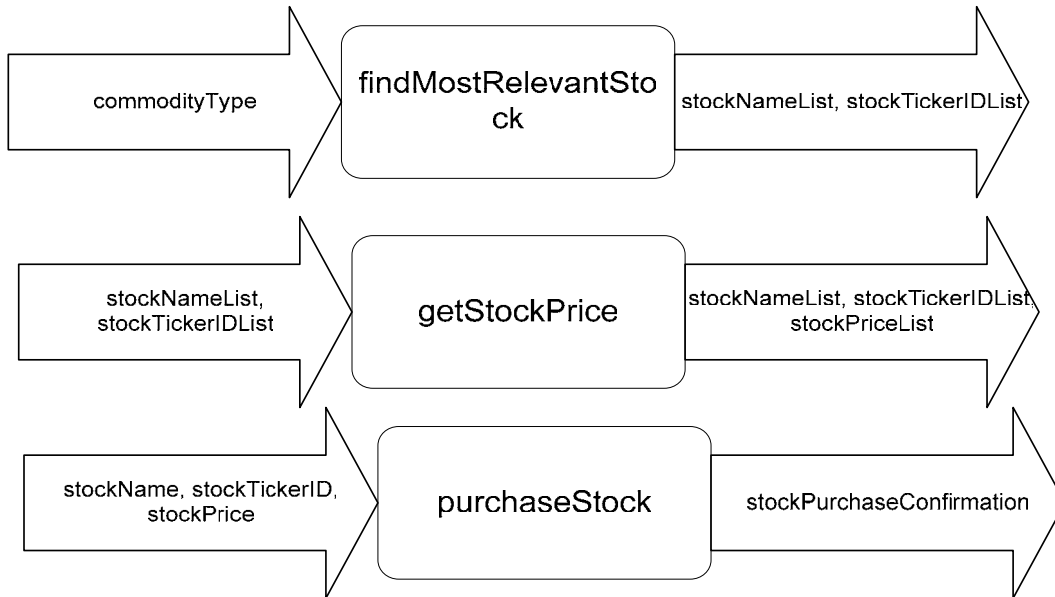
2.



2. **Inputs** = commodityType

Outputs = stockPurchaseConfirmationOptimal

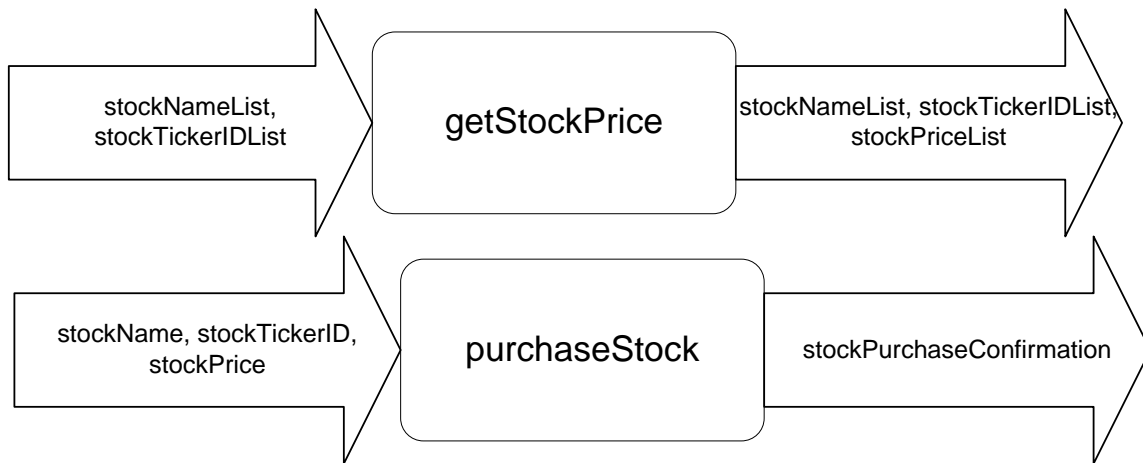
3.



3. **Inputs** = commodityType

Outputs = stockPurchaseConfirmation

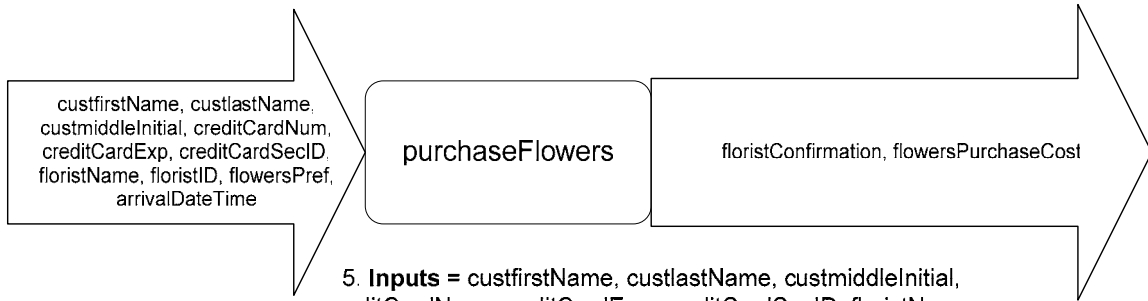
4.



4. **Inputs** = stockNameList, stockTickerIDList

Outputs = stockPurchaseConfirmation

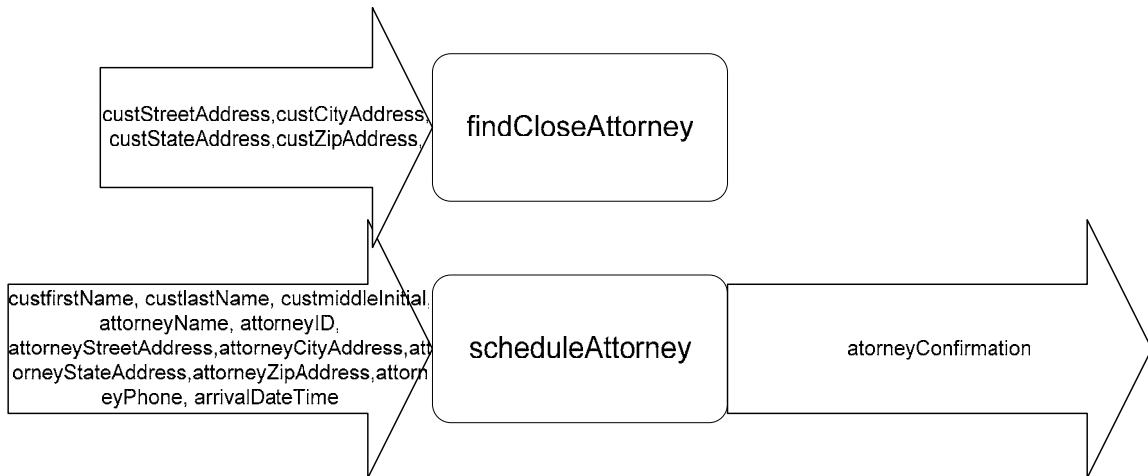
5.



5. **Inputs** = custfirstName, custlastName, custmiddleInitial, creditCardNum, creditCardExp, creditCardSecID, floristName, floristID, flowersPref, arrivalDateTime

Outputs = flowersConfirmation, flowersPurchasePrice

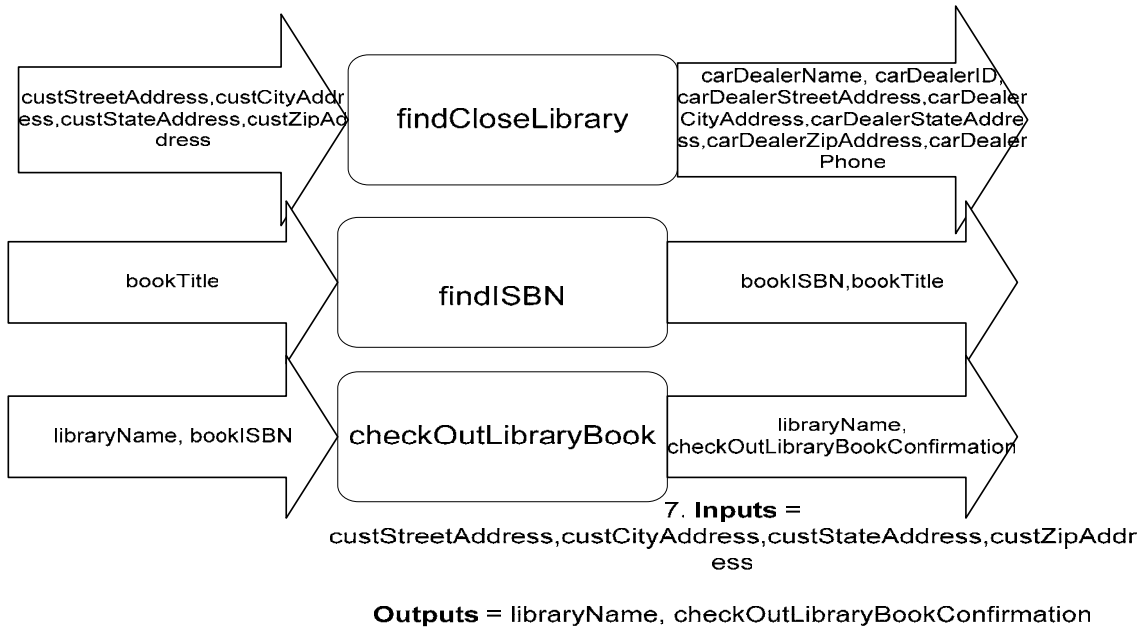
6.



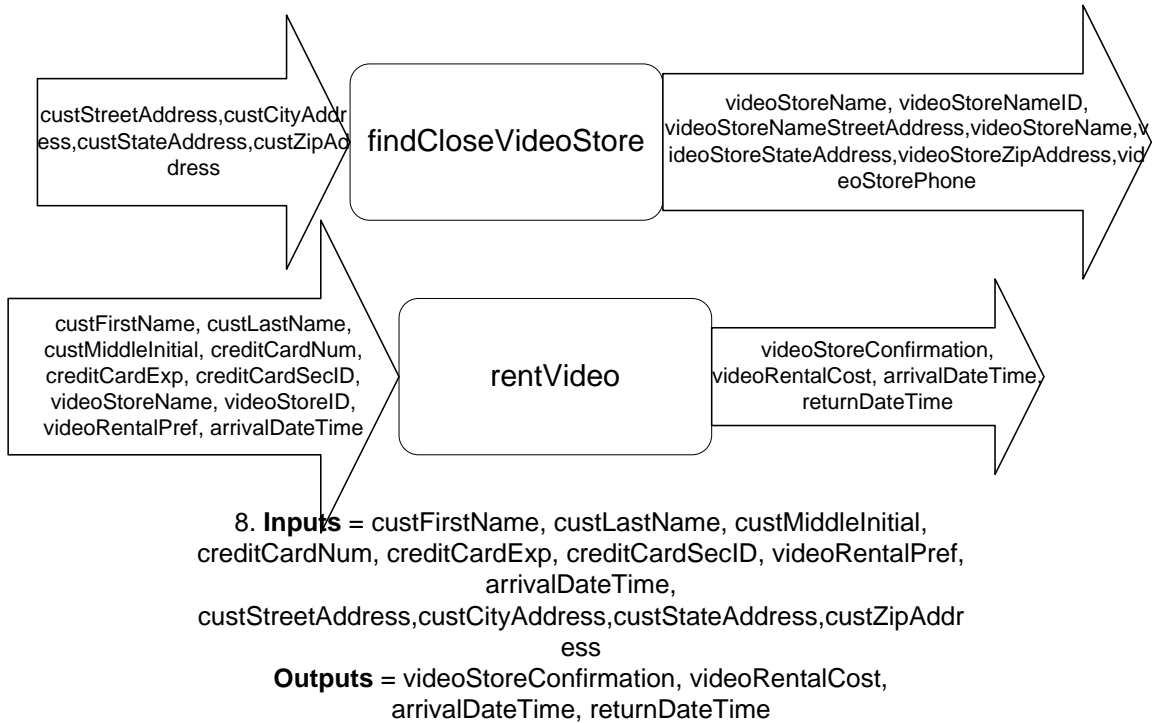
6. **Inputs** =
custStreetAddress, custCityAddress, custStateAddress, custZipAddress

Outputs = attorneyConfirmation

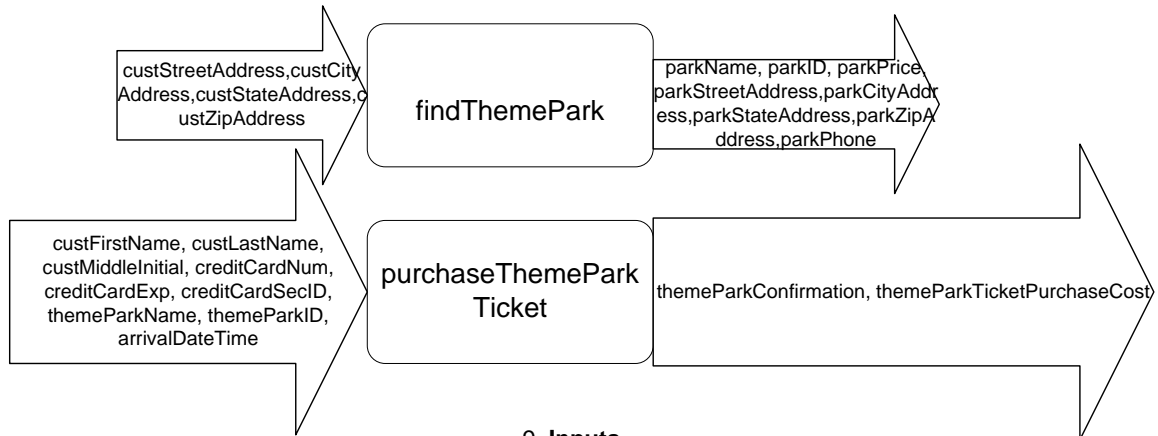
7.



8.

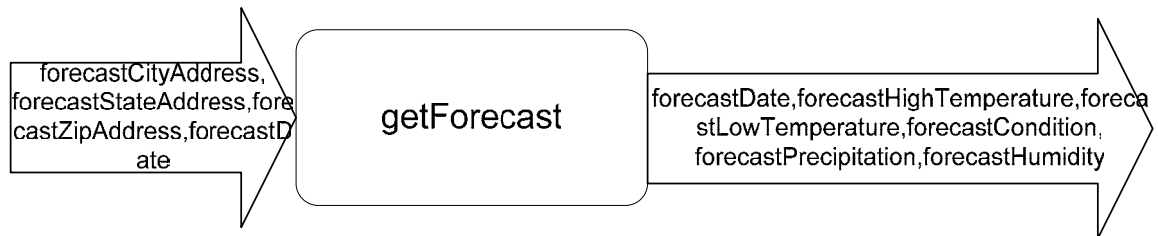


9.



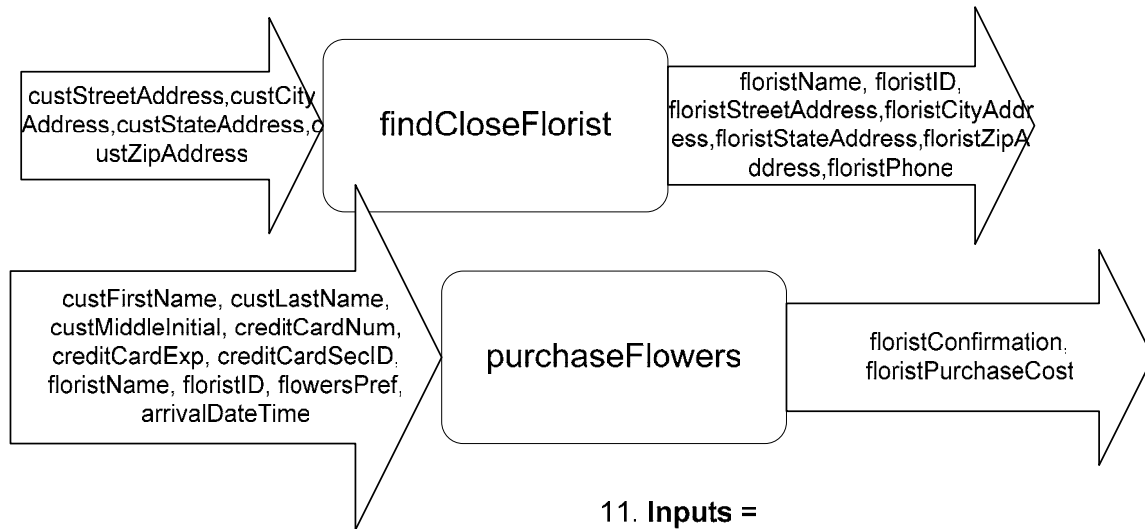
9. **Inputs** =
custStreetAddress,custCityAddress,custStateAddress,custZipAddress,custFirstName, custLastName, custMiddleInitial, creditCardNum, creditCardExp, creditCardSecID, arrivalDateTime
Outputs = themeParkConfirmation, themeParkTicketPurchaseCost

10.



10. **Inputs** = forecastCityAddress, forecastStateAddress, forecastZipAddress, forecastDate
Outputs = forecastDate, forecastHighTemperature, forecastLowTemperature, forecastCondition, forecastPrecipitation, forecastHumidity

11.

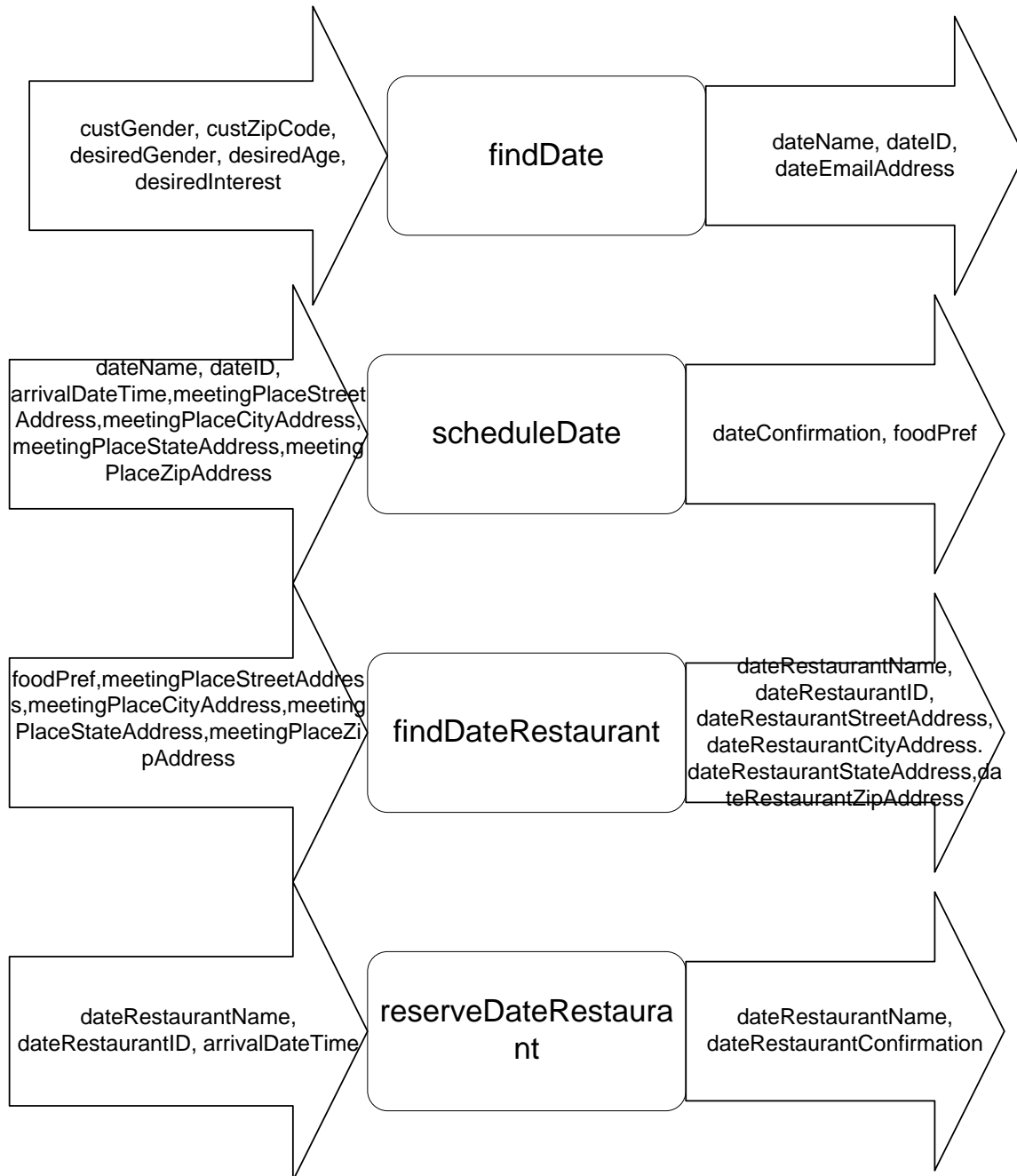


11. Inputs =

fcustStreetAddress, custCityAddress, custStateAddress, custZipAddress, custFirstName, custLastName, custMiddleInitial, creditCardNum, creditCardExp, creditCardSecID, flowersPref,

Outputs = floristConfirmation, floristPurchaseCost

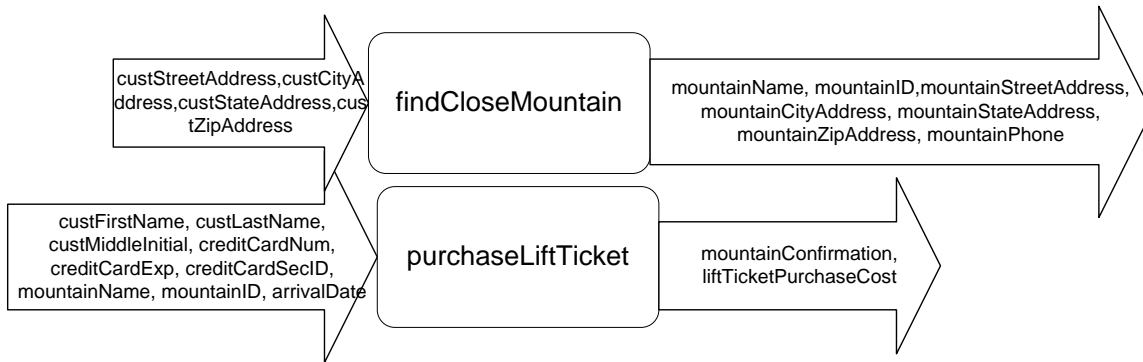
12.



12. **Inputs** = custGender, custZipCode, desiredGender, desiredAge, desiredInterest, meetingPlaceStreetAddress, meetingPlaceCityAddress, meetingPlaceStateAddress, meetingPlaceZipAddress

Outputs = dateRestaurantName, dateRestaurantConfirmation

13.

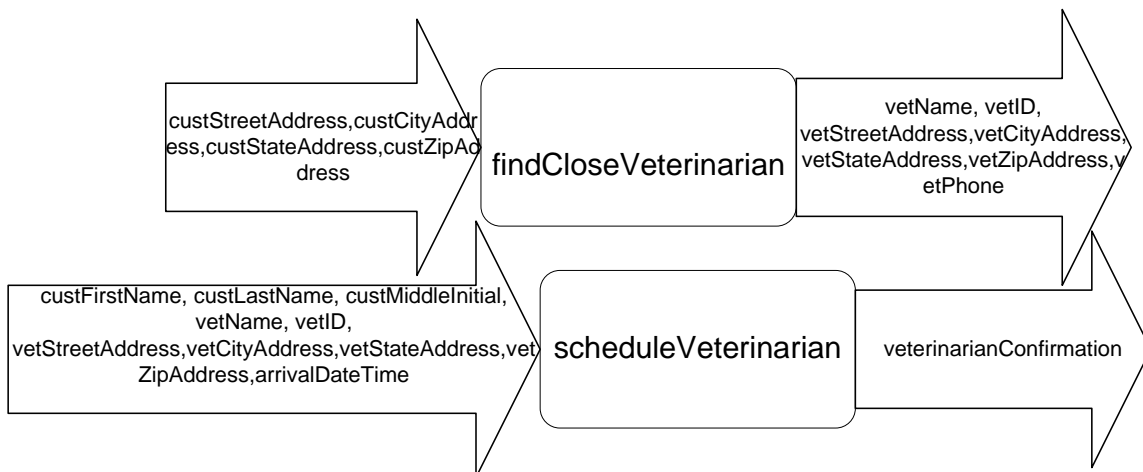


13. Inputs =

custStreetAddress, custCityAddress, custStateAddress, custZipAddress, custFirstName, custLastName, custMiddleInitial, creditCardNum, creditCardExp, creditCardSecID, arrivalDate

Outputs = mountainConfirmation, liftTicketPurchaseCost

14.

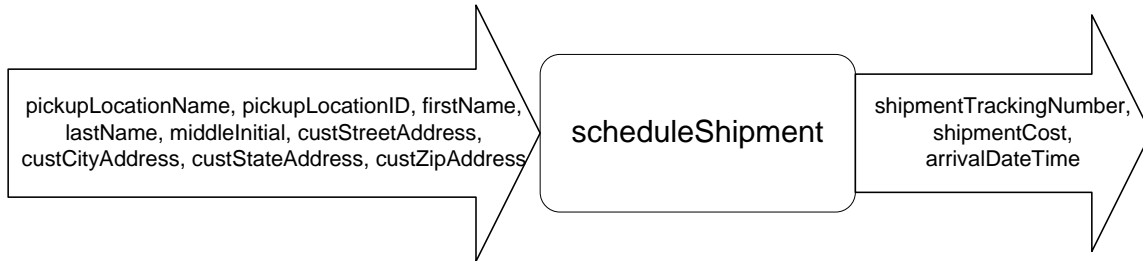


14. Inputs =

custStreetAddress, custCityAddress, custStateAddress, custZipAddress, custFirstName, custLastName, custMiddleInitial, arrivalDateTime

Outputs = veterinarianConfirmation

15.



15. **Inputs** = pickupLocationName, pickupLocationID, firstName, lastName, middleInitial, custStreetAddress, custCityAddress, custStateAddress, custZipAddress
Outputs = shipmentTrackingNumber, shipmentCost

INTRODUCING THE WEB SERVICE CHALLENGE

Progressing the State-of-the-Practice of Tools for Service-Oriented Computing

GENERAL

[Home](#)

[About the WS-Challenge](#)

[Contact Info](#)

[Click Here to Join News List](#)

WS-Challenge

[Participants](#)

[Repositories and Downloads](#)

[Results Archive](#)

VENUES

[EEE – 07](#)

[ICEBE – 07](#)

OUR SPONSORS



[Award Information](#)



WS-Challenge Participants

2007

Vitalab, University of Vienna

University of Kassel, Germany

University of Texas at Dallas, Texas, USA

Penn State University, Pennsylvania, USA

Tsinghua University, P.R.China

University of Charleston, South Carolina, USA

University of California Irvine, California, USA

2006

Distributed Systems Group, TU Wien, Austria

Distributed System Group, University of Kassel, Germany

University of Texas at Dallas, Texas, USA

Penn State University, Pennsylvania, USA

EC3 - Electronic Commerce Competence Center, Vienna, Austria

Singapore Institute of Manufacturing Technology, Singapore

Politecnico di Bari, Italy

IBM T. J. Watson Research Center

Tsinghua University, P.R.China

University of California Irvine

2005

Department of Computer Science & Engineering, Fudan University,
Shanghai, China

Penn State University

Naval Research Lab and West Virginia University

College of Charleston

Electronic Commerce Competence Center

Industrial Informatics Group, Singapore Institute of Manufacturing
Technology

WS-Challenge Syntactic Competition Syntactic Repositories EEE'05 repository ICEBE'05 repository	WS-Challenge Software Several "initial" software packages are available for to help participants get started: Georgetown Java Software Installation Guide	WS-Challenge Background Brian Blake and William Cheung conceptualized the idea for the competition. The first year rules, repository, and software were developed at Georgetown University. Read more about it here .
Results from EEE-07 Speed Challenge Champion – Tsinghua University First Runner-Up – University of Kassel, Germany Second Runner-Up – Vitalab (University of Vienna)	Results from EEE-07 Architecture Challenge Champion – University of California, Irvine First Runner-Up – University of Charleston Second Runner-Up – Pennsylvania State University	Related Events Services Computing Conference (SCC) 2008 International Conference on Web Services (ICWS) 2008 International Conference on Service Oriented Computing (ICSOC)

INTRODUCING THE WEB SERVICE CHALLENGE

Progressing the State-of-the-Practice of Tools for Service-Oriented Computing

GENERAL

[Home](#)

[About the WS-Challenge](#)

[Contact Info](#)

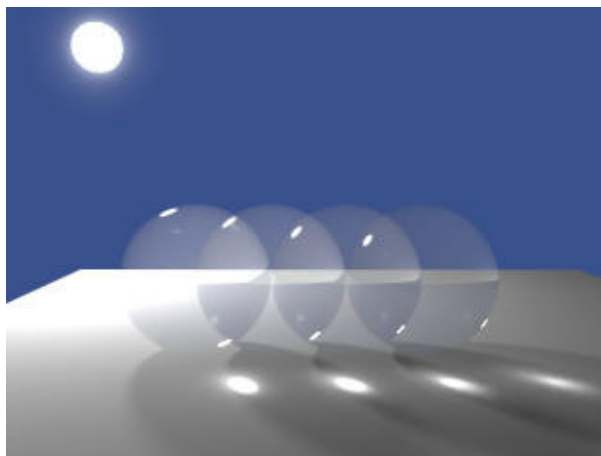
[Click Here to Join News List](#)

WS-Challenge

[Participants](#)

[Repositories and Downloads](#)

[Results Archive](#)



WS-Challenge @ EEE'07



WSC '07 – TOKYO

The Web Service Challenge (WS-Challenge)

With growing acceptance of service-oriented computing, an emerging area of research is the investigation of technologies that will enable the discovery and composition of web services. Using the same approach as the popular [Trading Agent Competitions \(TAC\)](#), the Web Services Challenge is the original event geared towards the management of web services. The annual competition solicits industry and academic researchers that develop software components and/or intelligent agents that have the ability to discover pertinent web services and compose them to create higher-level functionality.

WS-Challenge '07 Rounds Are Complete.

See WSC '07 Webpage for More Information.

VENUES

[EEE – 07](#)

[ICEBE – 07](#)

News and Information

- Check out the results page for WSC '07 results and other archives.
- WS-Challenge is funded by the [National Science Foundation](#)

OUR SPONSORS



Award Information



WS-Challenge Syntactic Competition

Syntactic Repositories
[EEE'05 repository](#)
[ICEBE'05 repository](#)

WS-Challenge Software

Several "initial" software packages are available for to help participants get started:

[Georgetown Java Software Installation Guide](#)

WS-Challenge Background

Brian Blake and William Cheung conceptualized the idea for the competition. The first year rules, repository, and software were developed at Georgetown University. Read more about it [here](#).

Results from EEE-07

Speed Challenge

Champion – Tsinghua University

First Runner-Up – University of Kassel, Germany

Second Runner-Up – Vitalab (University of Vienna)

Results from EEE-07

Architecture Challenge

Champion – University of California, Irvine

First Runner-Up – University of Charleston

Second Runner-Up – Pennsylvania State University

Related Events

[Services Computing Conference \(SCC\) 2008](#)

[International Conference on Web Services \(ICWS\) 2008](#)

[International Conference on Service Oriented Computing \(ICSOC\)](#)

The EEE-05 Challenge: A New Web Service Discovery and Composition Competition

M. Brian Blake
Georgetown University Washington, DC, USA
blakeb@cs.georgetown.edu

Dr. Kwok Ching Tsui
The Hong Kong and Shanghai Banking Corporation (HSBC)
kwokchingtsui@hsbc.com.hk

Andreas Wombacher
IPSI, Germany
andreas.wombacher@ipsi.fraunhofer.de

Abstract

With growing acceptance of service-oriented computing, an emerging area of research is the investigation of technologies that will enable the discovery and composition of web services. Using the same approach as the popular Trading Agent Competitions (TAC), the EEE-05 Web Services Challenge is the first event geared towards the management of web services. The competition solicits industry and academic researchers that develop software components and/or intelligent agents that have the ability to discover pertinent web services and also compose them to create higher-level functionality. This paper describes the competition details for this first year and expectations for future events.

1. Introduction

The purpose of the EEE-05 Challenge is to establish a venue where researchers can collaborate on implementations in the web service composition domain. The results from the competition can be used as performance baselines for other researchers. In addition, software design approaches can be demonstrated, enhanced, and disseminated each year as the competition evolves.

The objective of the competition, in the first year, is to encourage participants to concentrate on syntactical matching and chaining for Web Service Description Language (WSDL) documents. A successful software entry will be able to accurately and efficiently find services using the WSDL part names underlying input and output messages. Secondly, entry software is required to create chains of services by linking output part names to the subsequent input part names. The intent of the first year is for participants to create part name matching components/agents. The software created in the first year will set the foundation for

later years of this competition (i.e. each year with more technical rigor). Each year entry software should become more efficient.

2. The First Year

In this first year of competition, the web services repository will be based on WSDL 1.1. The participants were provided with two tutorials sites [2][3]. The organizers also suggested the use of the Microsoft .Net IDE as an editor for WSDL documents. The following sections describe the samples repository, discovery sample, and composition sample.

2.1 Samples Repository and Input Files

The samples repository can be found at [4]. The 2005 competition concentrates only on messages (and their underlying part names) and ports. Concrete descriptions such as bindings and services will be the focus in later years of the competition.

The service repository contains over 100 services. Many services have logical part names, however other services may be auto-generated with part names with random combination of letters. Other services are sub-sets and variations of the *correct* services.

Participants will be provided with an XML file to initiate the discovery and composition routines in the competition. A sample of the XML request is shown in Table 1. In this first year, organizers will be flexible in allowing the competitors to reformat the XML file to best meet the entry software's front-end.

Entry software is required to execute on the designated competition workstation. In the first year, a Windows-based machine will be used, and participants are required to e-mail their system requirements prior to the competition.

Table 1. Sample Competition Input File.

```
<EEE05Challenge>
<DiscoveryRoutine>
  <Provided> partname1, partname2 </Provided>
  <Resultant> partname1 </Resultant>
</DiscoveryRoutine>
<CompositionRoutine>
  <Provided> partname1, partname2 </Provided>
  <Resultant> partname1, partname2 </Resultant>
</CompositionRoutine>
</EEE05Challenge>
```

2.2 An Example Discovery Routine

Participants will be asked to find a specific service that can fulfill a certain input and output criteria. Table 2 is a sample request file.

Table 2. Sample Discovery Request.

```
<DiscoveryRoutine>
  <Provided> foodPref, custStreetAddress , custCityAddress,
    custStateAddress, custZipAddress </Provided>
  <Resultant> restaurantName, restaurantID </Resultant>
</DiscoveryRoutine>
```

The most relevant service in the repository is the *findCloseRestaurant* service as shown in Table 3. The *findCloseRestaurant* service may be considered a bit over-qualified for the

requirements, but it does fulfill them. Only one service will accurately meet the requirements in the repository. However, other services that partially meet the requirement are also included in the repository. Only accurate matches will count. The discovery portion of the competition is used to evaluate the design and speed of the software entries to execute the matching.

Table 3. Snippet of findCloseRestaurant.

```
<message name="findCloseRestaurant_Request">
  <part name="custStreetAddress" type="xs:string"/>
  <part name="custCityAddress" type="xs:string"/>
  <part name="custStateAddress" type="xs:string"/>
  <part name="custZipAddress" type="xs:string"/>
  <part name="foodPref" type="xs:string"/>
</message>

<message name="findCloseRestaurant_Response">
  <part name="restaurantName" type="xs:string"/>
  <part name="restaurantID" type="xs:string"/>
  <part name="restaurantStreetAddress" type="xs:string"/>
  <part name="restaturantCityAddress" type="xs:string"/>
  <part name="restaurantStateAddress" type="xs:string"/>
  <part name="restaurantZipAddress" type="xs:string"/>
</message>
```

2.3 An Example Composition Routine

Participants will also be posed with a composition request (finding a specific sequence of services). The routine in Table 4 can be fulfilled using the sequence, *purchaseALT.wsdl->reserveRental.wsdl-> reserveRoom.wsdl->createItinerary.wsdl*, as captured in [4].

Table 4. Sample Composition Request

```
<CompositionRoutine>
  <Provided> firstName, lastName, middleInitial, creditCardNum,
creditCardExp, creditCardSecID, departCity, departState, destCity,
destState, rentalPref , roomPref, hotelName </Provided>
  <Resultant> ItineraryURL </Resultant>
</CompositionRoutine>
```

The *reserveRental* and *reserveRoom* require the output of *purchaseALT*, while *createItinerary* requires information from *reserveRental*, *reserveRoom*, and *purchaseALT*. The competition will limit three services as predicates for a subsequent service, however, the software entries are advised to keep a running memory of available information. Participants should note that the most effective software will be combine front-to-back and back-to-front processing, perhaps simultaneously. In addition to the aforementioned more complex routine, other routines will require just two or three services with direct matchings.

3. Evaluation and Future Competitions

This first year will help to establish the most effective approach to evaluating the software. The initial evaluation approach will consist of a subjective score on the system design. Other aspects will be performance and accuracy. One idea is to allow the discovery and composition to proceed for a limited amount of time and count the accurate number of discoveries or compositions, respectively. Although a number of evaluation schemes will be tried in the first year, the competition will be run with the best intentions to keep the judging fair.

In second year, the competition will require participants to match part names that are not syntactically the

same. In subsequent years, services will have be composed with semantic languages such as OWL-S.

4. Acknowledgements

This competition has been greatly influenced by the support from and fruitful conversations with Dr. William K.W. Cheung of Hong Kong Baptist University, Dr. Eleni Stroulia of the University of Alberta, Dr. Terry Payne of the University of Southhampton, and Dr. Norman Sadeh of Carnegie Mellon University.

5. References

- [1] The Trading Agent Competition (2005): <http://www.sics.se/tac/page.php?id=1>
- [2] Microsoft Corporation: Web Service Description Language Tutorial, (2005) <http://msdn.microsoft.com/archive/default.asp?url=/archive/en-us/dnarxml/html/wsdlexplained.asp>
- [3] WWW Schools: Web Service Description Language Tutorial (2005), <http://www.w3schools.com/wsd/default.asp>
- [4] EEE05 Challenge Samples Repository (2005), http://cssun.georgetown.edu/~blakeb/EEE05/WSDL_Repos.zip

INTRODUCING THE WEB SERVICE CHALLENGE

Progressing the State-of-the-Practice of Tools for Service-Oriented Computing

GENERAL

[Home](#)

[About the WS-Challenge](#)

[Contact Info](#)

[Click Here to Join News List](#)

WS-Challenge

[Participants](#)

[Repositories and Downloads](#)

[Results Archive](#)

VENUES

[EEE – 07](#)

[ICEBE – 07](#)

OUR SPONSORS

Founders and Organizers

<p>M. Brian Blake Georgetown University 234 Reiss Science Building Washington, DC 20057 Email: blakeb@cs.georgetown.edu</p>	<p>William K. Cheung Centre of e-Transformation Research Department of Computer Science Hong Kong Baptist University Kowloon Tong Hong Kong, China Email: william@comp.hkbu.edu.hk</p>
<p>Andreas Wombacher Information Systems Group Department of Computer Science University of Twente P.O. Box 217 7500 AE Enschede, The Netherlands Email: a.wombacher@utwente.nl</p>	<p>Kwok-Ching Tsui The Hong Kong and Shanghai Banking Corporation (HSBC) Email: kwokchingtsui@hsbc.com.hk</p>

Program Committee and Advisors

<p>Fang Yan Rao IBM China Research Lab, China Email: raofy@cn.ibm.com</p>	<p>Peter Hrastnik Electronic Commerce Competence Center, Austria Email: peter.hrastnik@ec3.at</p>
--	---



Award Information



Eleni Stroulia
University of Alberta
Department of Computer Science
221 Athabasca Hall
Edmonton, AB, T6G 2E8, Canada
Email: stroulia@cs.alberta.ca

WS-Challenge Syntactic Competition

Syntactic Repositories
[EEE'05 repository](#)
[ICEBE'05 repository](#)

WS-Challenge Software

Several "initial" software packages are available for to help participants get started:

[Georgetown Java Software Installation Guide](#)

WS-Challenge Background

Brian Blake and William Cheung conceptualized the idea for the competition. The first year rules, repository, and software were developed at Georgetown University. Read more about it [here](#).

Results from EEE-07

Speed Challenge

Champion – Tsinghua University

First Runner-Up – University of Kassel, Germany

Second Runner-Up – Vitalab (University of Vienna)

Results from EEE-07

Architecture Challenge

Champion – University of California, Irvine

First Runner-Up – University of Charleston

Second Runner-Up – Pennsylvania State University

Related Events

[Services Computing Conference \(SCC\) 2008](#)

[International Conference on Web Services \(ICWS\) 2008](#)

[International Conference on Service Oriented Computing \(ICSOC\)](#)

INTRODUCING THE WEB SERVICE CHALLENGE

Progressing the State-of-the-Practice of Tools for Service-Oriented Computing

GENERAL

[Home](#)

[About the WS-Challenge](#)

[Contact Info](#)

[Click Here to Join News List](#)

WS-Challenge

[Participants](#)

[Repositories and Downloads](#)

[Results Archive](#)

VENUES

[EEE – 07](#)

[ICEBE – 07](#)

OUR SPONSORS



[Award Information](#)



Technical Details and Downloads

Overall Technical Details

EEE – 05

[Sample Repositories](#)

[Sample Challenges](#)

[Expected Answers to Challenges \(text format\) or \(pdf format\)](#)

ICEBE – 05

[Sample Repositories, Sample Challenges, and Expected Answers to Challenges](#)

TEST DATA for ICEBE-05 (For Special Issue Submissions):

- [1. Discovery Challenge](#)
- [2. Composition \(1\) Challenge](#)
- [3. Composition \(2\) Challenge \(More Difficult\)](#)

WS-Challenge Syntactic Competition

Syntactic Repositories
[EEE'05 repository](#)
[ICEBE'05 repository](#)

WS-Challenge Software

Several "initial" software packages are available for to help participants get started:

[Georgetown Java Software Installation Guide](#)

WS-Challenge Background

Brian Blake and William Cheung conceptualized the idea for the competition. The first year rules, repository, and software were developed at Georgetown University. Read more about it [here](#).

Results from EEE-07	Results from EEE-07	Related Events
<p>Speed Challenge</p> <p>Champion – Tsinghua University</p> <p>First Runner-Up – University of Kassel, Germany</p> <p>Second Runner-Up – Vitalab (University of Vienna)</p>	<p>Architecture Challenge</p> <p>Champion – University of California, Irvine</p> <p>First Runner-Up – University of Charleston</p> <p>Second Runner-Up – Pennsylvania State University</p>	<p><u>Services Computing Conference (SCC) 2008</u></p> <p><u>International Conference on Web Services (ICWS) 2008</u></p> <p><u>International Conference on Service Oriented Computing (ICSOC)</u></p>

Copyright © 2006 Georgetown University. All Rights Reserved.
For questions about this web site, contact mfn3[AT]georgetown[DOT]edu

INTRODUCING THE WEB SERVICE CHALLENGE

Progressing the State-of-the-Practice of Tools for Service-Oriented Computing

GENERAL

[Home](#)

[About the WS-Challenge](#)

[Contact Info](#)

[Click Here to Join News List](#)

WS-Challenge

[Participants](#)

[Repositories and Downloads](#)

[Results Archive](#)

VENUES

[EEE – 07](#)

[ICEBE – 07](#)

OUR SPONSORS



[Award Information](#)



Results Archive

EEE – 07

Speed

Champion – Tsinghua University, China

1st Runner-Up – University of Kassel, Germany

2nd Runner-Up – Vitalab, University of Vienna, Austria

Architecture

Champion – University of California, Irvine

1st Runner-Up – University of Charleston

2nd Runner-Up – Pennsylvania State University

EEE – 06 -- [Award Presentation Slide Show](#)

Semantic

Champion – Distributed System Group, University Kassel, Germany

1st Runner-Up – Electronic Commerce Competence Center, Austria

Syntactic

Champion – Tsinghua University, P.R.China

1st Runner-Up – Distributed Systems Group, TU Wien, Austria

ICEBE – 05

Champion – Fudan University, China

1st Runner-Up – Penn State University, USA

2nd Runner –Up – Electronic Commerce Competence Center, Austria

EEE – 05

Champion - Electronic Commerce Competence Center, Austria

1st Runner-Up - Industrial Informatics Group, Singapore Inst. of Manufacturing Technology

2nd Runner-Up - Whitestein Technologies AG, College of Charleston, Profactor Research

WS-Challenge Syntactic Competition Syntactic Repositories EEE'05 repository ICEBE'05 repository	WS-Challenge Software Several "initial" software packages are available for to help participants get started: Georgetown Java Software Installation Guide	WS-Challenge Background Brian Blake and William Cheung conceptualized the idea for the competition. The first year rules, repository, and software were developed at Georgetown University. Read more about it here .
Results from EEE-07 Speed Challenge Champion – Tsinghua University First Runner-Up – University of Kassel, Germany Second Runner-Up – Vitalab (University of Vienna)	Results from EEE-07 Architecture Challenge Champion – University of California, Irvine First Runner-Up – University of Charleston Second Runner-Up – Pennsylvania State University	Related Events Services Computing Conference (SCC) 2008 International Conference on Web Services (ICWS) 2008 International Conference on Service Oriented Computing (ICSOC)

WEB SERVICE CHALLENGE 2006

Software Install Guide

Daniel R. Kahan
drk8@georgetown.edu

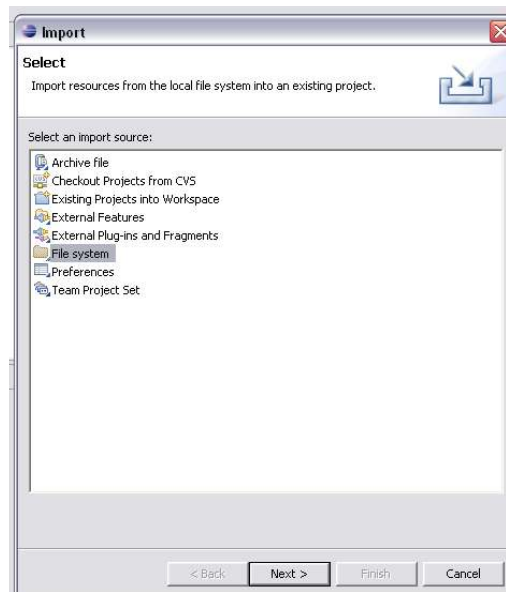
Requirements

JRE 1.4+ [<http://java.sun.com>]
Windows 2000, XP* [<http://www.microsoft.com>]
Eclipse 3.11 [<http://www.eclipse.org>]

* In theory, so long as the operating system supports Java and Eclipse, our software ought to be able to run on it. However, that theory has not been put to the test, so these directions are Windows-oriented.

IMPORTING THE PROJECT

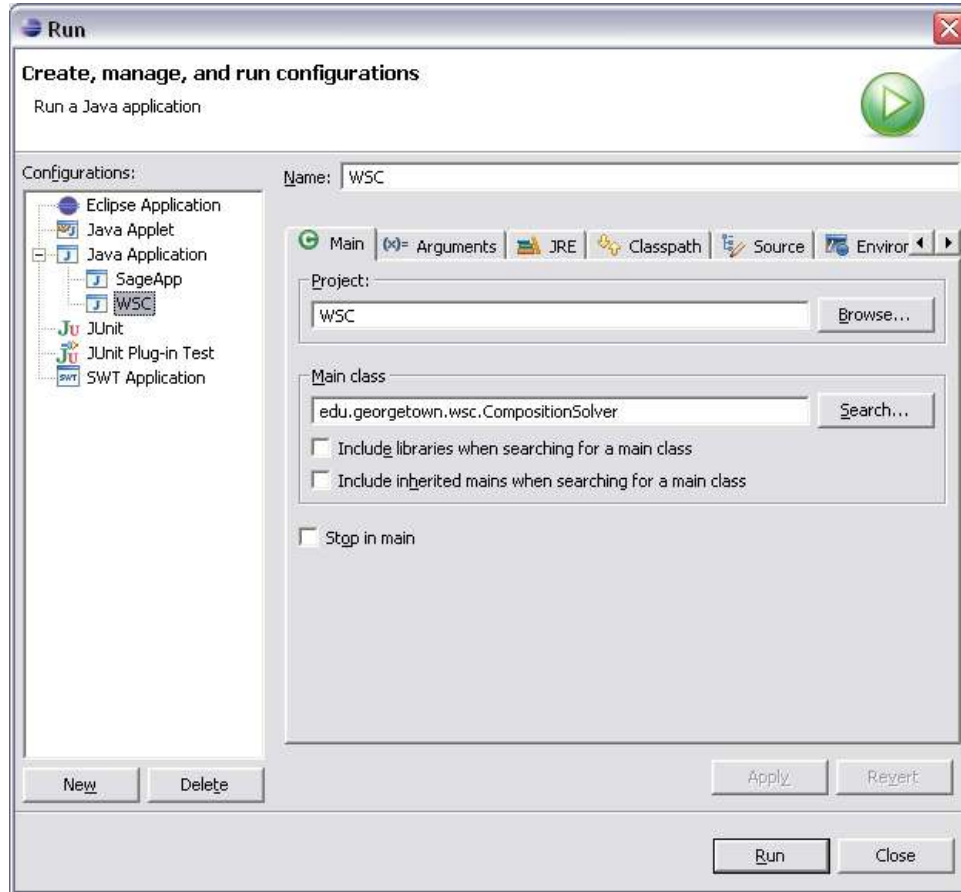
1. Unzip the project to `c:\wsc`, making sure to preserve the directory structure. Within the `wsc` directory, there should be two subdirectories, `src` and `rep`.
2. Launch Eclipse. Start a new **Java Project** in Eclipse and name it `wsc`.
3. Select **Import** from the **File** menu. Select **File System** when presented with the window below. Then click the **Next** button.



4. Once the **File System** dialog comes up, simply enter `c:\wsc\src\wsc` in the **From directory** textbox. Enter `wsc` in the **Into folder** textbox.

RUNNING THE PROJECT

1. Now that all of the files have been imported, it's time to configure Eclipse so we can run the project.
2. In Eclipse, go to the **Run** menu and select **Run...** In the right-hand pane, select **Java Application** and then click the **New** button. In the **Main** tab, select **wsc** as the project and **edu.georgetown.wsc.CompositionSolver** (not **CompositionSolverGUI**) as the **Main** class.



3. Now, the project should be ready to **Run**. Whenever you want to run the project in the future, you can simply click the green arrow (shown below) and select **wsc** as the configuration.



4. Once the project is running, you can select **CompositionRequest.xml** from the **C:\wsc** folder and **c:\wsc\rep** as the repository path. Modify the XML file to experiment with other routines. More discovery and composition routines will be available on the WS-Challenge web site in the near future [<http://ws-challenge.georgetown.edu>].

Web Services Challenge Technical Details

(Adopted from [EEE-05 Challenge](#))

1. Objectives

The objective of the competition, in this first year, is to encourage participants to concentrate on *syntactical matching* and *chaining* for Web Service Description Language (WSDL) documents. A successful software entry will be able to accurately and efficiently find services using the part names underlying their input and output messages.

Secondly, entries will have to create chains of services by linking output part names to the subsequent input part names. The intent of this competition is for participants to create part name matching components/agents that will set the foundation for later years of this competition (i.e. each year with more technical rigor). Participants should also focus on robust system design and efficient programming techniques.

2. WSDL Details

In this competition, the web services repository will be based on WSDL 1.1. Below are two tutorial sites that may be helpful for participants.

- <http://msdn.microsoft.com/archive/default.asp?url=/archive/en-us/dnarxml/html/wsdlexplained.asp>
- <http://www.w3schools.com/wSDL/default.asp>

FYI - Microsoft .Net IDE has a reasonable editor for WSDL documents.

3. Sample Repository

A sample repository can be downloaded at the following web site:

<http://www.comp.hkbu.edu.hk/~ctr/wschallenge>

Participants should notice that we are concentrating only on messages (and their underlying part names) and ports. Concrete descriptions such as *bindings* and *services* will be the focus in later years of the competition, but not considered in this competition.

The sample service repository have services with auto-generated with part names with random combination of letters. Other services will be added that are sub-sets and variations of the most correct services to use for a particular discovery of competition routines.

4. Sample Competition Routine

We will provide participants with an XML file to initiate the competition discovery and composition routines. A sample of the XML request is below.

```
<WSChallenge>
<DiscoveryRoutine>
  <Provided> partname1, partname2, partname3 </Provided>
  <Resultant> partname1 </Resultant>
```

```
</DiscoveryRoutine>
<CompositionRoutine>
  <Provided> partname1, partname2, partname3 </Provided>
  <Resultant> partname1, partname2, partname3 </Resultant>
</CompositionRoutine>
</WSChallenge>
```

All software must be able to execute on a designated competition workstation. Participants will want to send their system requirements to us via email prior to the competition. There will be a pre-competition arrangement which should be able to provide more feedback for refining their software.

5. Discovery Sample

Participants will be asked to find a specific service that can fulfill a certain input and output criteria.

The following requested routine can be fulfilled by findCloseRestaurant.wsdl:

```
<DiscoveryRoutine>
  <Provided> foodPref,custStreetAddress,custCityAddress,custStateAddress,custZipAddress </Provided>
  <Resultant> restaurantName,restaurantID </Resultant>
</DiscoveryRoutine>
```

findCloseRestaurant.wsdl (snippet of the messages)

```
<message name="findCloseRestaurant_Request">
  <part name="custStreetAddress" type="xs:string"/>
  <part name="custCityAddress" type="xs:string"/>
  <part name="custStateAddress" type="xs:string"/>
  <part name="custZipAddress" type="xs:string"/>
  <part name="foodPref" type="xs:string"/>
</message>
<message name="findCloseRestaurant_Response">
  <part name="restaurantName" type="xs:string"/>
  <part name="restaurantID" type="xs:string"/>
  <part name="restaurantStreetAddress" type="xs:string"/>
  <part name="restaturantCityAddress" type="xs:string"/>
  <part name="restaurantStateAddress" type="xs:string"/>
  <part name="restaurantZipAddress" type="xs:string"/>
</message>
```

As you can see, findCloseRestaurant is a bit over-qualified for the requirements, but it does fulfill them.

We anticipate that only one service will accurately meet the requirements in the repository. However,

other services may be in place that partially meets the requirements. Only accurate matches count.

This part of the competition will mostly judge the design and speed of the software entries to execute

the matching.

6. Composition Sample

Participants may also be posed with a composition request (finding a specific sequence of services).

The following requested routine can be fulfilled by:

purchaseALT.wsdl->reserveRental.wsdl-> reserveRoom.wsdl->createltinerary.wsdl:

```
<CompositionRoutine>
  <Provided> firstName,lastName,middleInitial,creditCardNum,creditCardExp,creditCardSecID,departCity,
    departState,destCity,destState,rentalPref,roomPref,hotelName </Provided>
  <Resultant> ItineraryURL </Resultant>
</CompositionRoutine>
```

As you can see, *reserveRental* and *reserveRoom* require the output of *purchaseALT*, while *createltinerary* requires information from *reserveRental*, *reserveRoom*, and *purchaseALT*.

7. Evaluation.

One aspect will be a subjective score on the system design. The other aspects will be performance and accuracy. More details about the judging process will later be posted at the competition web site.

8. Miscellaneous WSDL Details.

1. arrivalDateTime and departDateTme are in the same context for all services.
 - a. May decide to make unique date/time syntax for each service
 - b. Our students had some difficulty with non-unique dates and times for mapping.
2. All operations are SOAP request-response, this year.
3. All services have one binding to one port (one operation per WSDL)
4. Port name=WSDL name
5. Message parts are all of type <string>.
6. Software entries should keep a memory of part names during composition, sometimes

messages will have to be carried to services in subsequent stages of the routine.

7. No composition routine will require more than three services to build input message for a

subsequent service. (*We did this to help scope complexity.*)

8. We anticipate that participants will be provided with an XML file to do all discoveries for

the first part of the competition. The composition routine will be done on a separate run.

9. For Further Questions.

Please send emails to wschallenge@comp.hkbu.edu.hk for technical questions. It would also be advisable to contact us notifying your intention to compete.

[Updated on June 22, 2005]

Discovery

1. findCloseRestaurant
2. findISBN
3. findCloseCarDealer
4. findCloseHotel
5. scheduleDate
6. scheduleDoctor
7. scheduleSalon
8. getForecast
9. purchaseStock
10. getStockPrice
11. findThemePark
12. findCloseFlorist
13. findCloseAPT
14. purchaseLiftTicket

Composition

1. purchaseALT.wsdl->reserveRental.wsdl-> reserveRoom.wsdl->createItinerary.wsdl
2. findMostRelevantStocks->getStockPriceMany->performStockResearch->purchaseOptimalStock
3. findMostRelevantStock->getStockPrice->purchaseStock
4. getStockPrice->purchaseStock
5. purchaseFlowers
6. findCloseAttorney->scheduleAttorney
7. findCloseLibrary->findISBN->checkOutLibraryBook
8. findCloseVideoStore->rentVideo
9. findThemePark->purchaseThemeParkTicket
10. getForecast
11. findCloseFlorist->purchaseFlowers
12. findDate->scheduleDate->findDateRestaurant->reserveDateRestaurant
13. findCloseMountain->purchaseLiftTicket
14. findCloseVeterinarian->scheduleVeterinarian
15. scheduleShipment

Answer Guide Summary for the EEE-05 Challenge

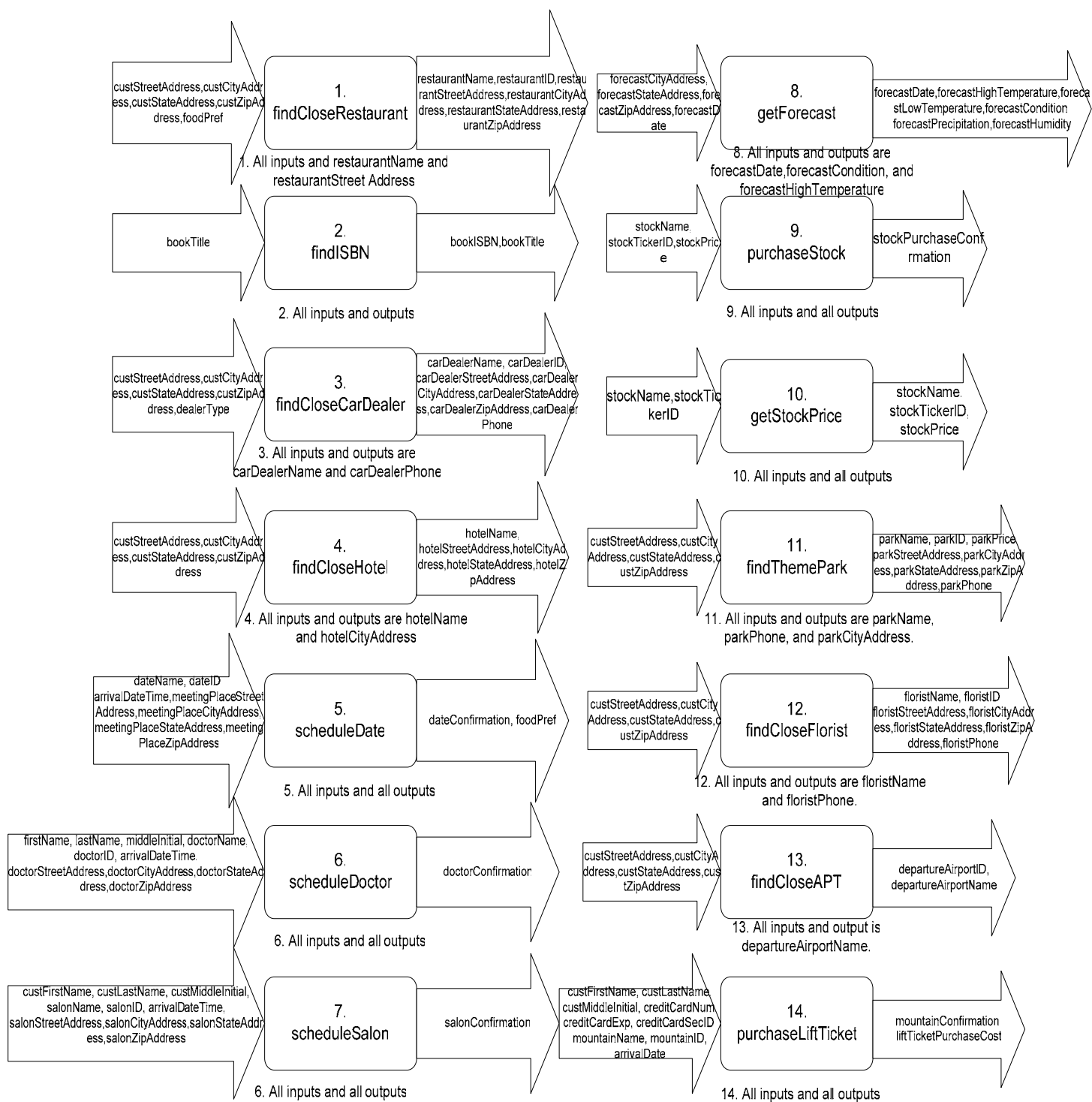
Discovery

1. findCloseRestaurant
2. findISBN
3. findCloseCarDealer
4. findCloseHotel
5. scheduleDate
6. scheduleDoctor
7. scheduleSalon
8. getForecast
9. purchaseStock
10. getStockPrice
11. findThemePark
12. findCloseFlorist
13. findCloseAPT
14. purchaseLiftTicket

Composition

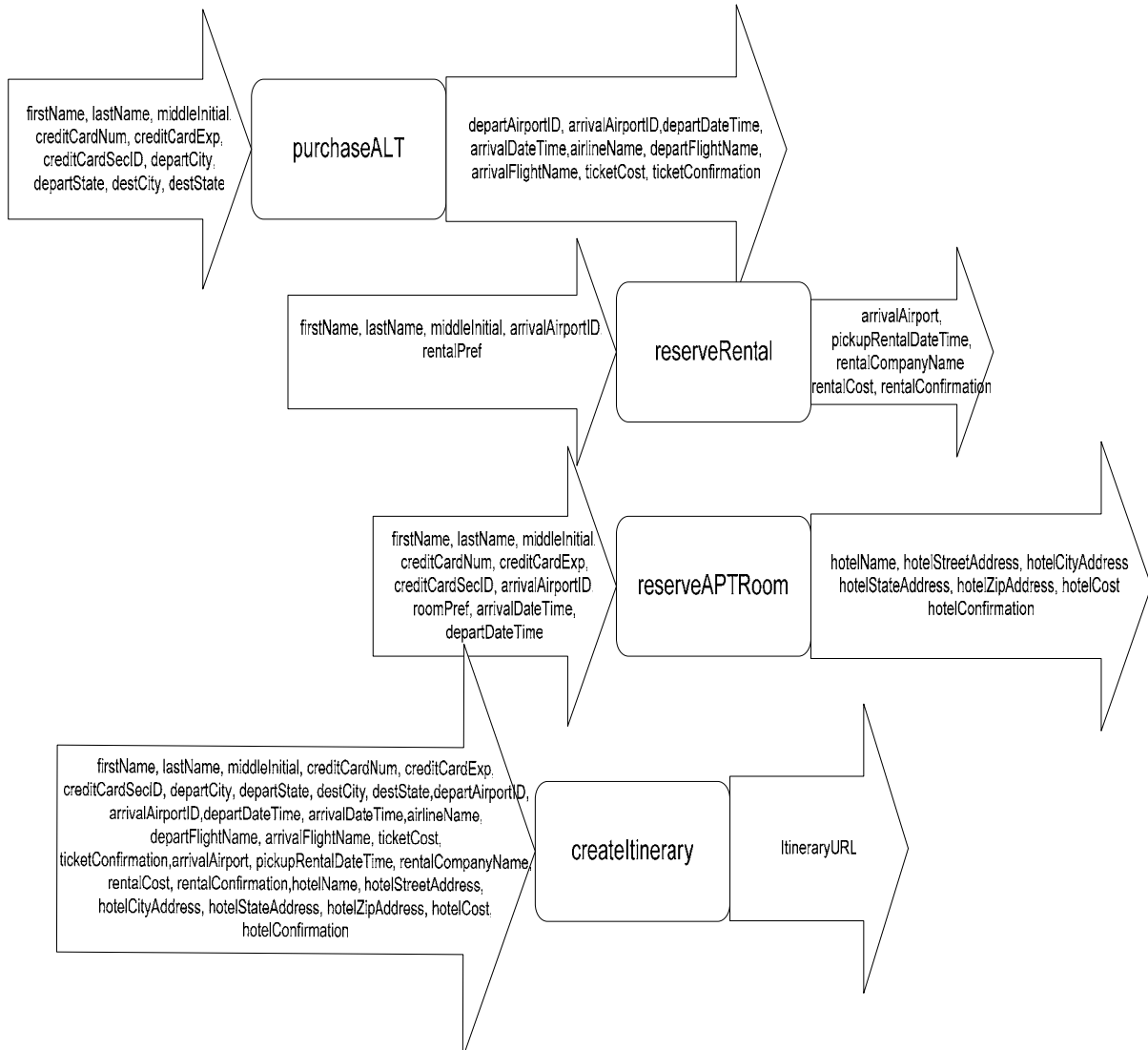
1. purchaseALT.wsdl->reserveRental.wsdl-> reserveRoom.wsdl->createItinerary.wsdl
2. findMostRelevantStocks->getStockPriceMany->performStockResearch->purchaseOptimalStock
3. findMostRelevantStock->getStockPrice->purchaseStock
4. getStockPrice->purchaseStock
5. purchaseFlowers
6. findCloseAttorney->scheduleAttorney
7. findCloseLibrary->findISBN->checkoutLibraryBook
8. findCloseVideoStore->rentVideo
9. findThemePark->purchaseThemeParkTicket
10. getForecast
11. findCloseFlorist->purchaseFlowers
12. findDate->scheduleDate->findDateRestaurant->reserveDateRestaurant
13. findCloseMountain->purchaseLiftTicket
14. findCloseVeterinarian->scheduleVeterinarian
15. scheduleShipment

Detailed Discovery Answer Guide (this corresponds to EEE05Challenge.xml)



Detailed Composition Answer Guide (this corresponds to EEE05Challenge.xml)

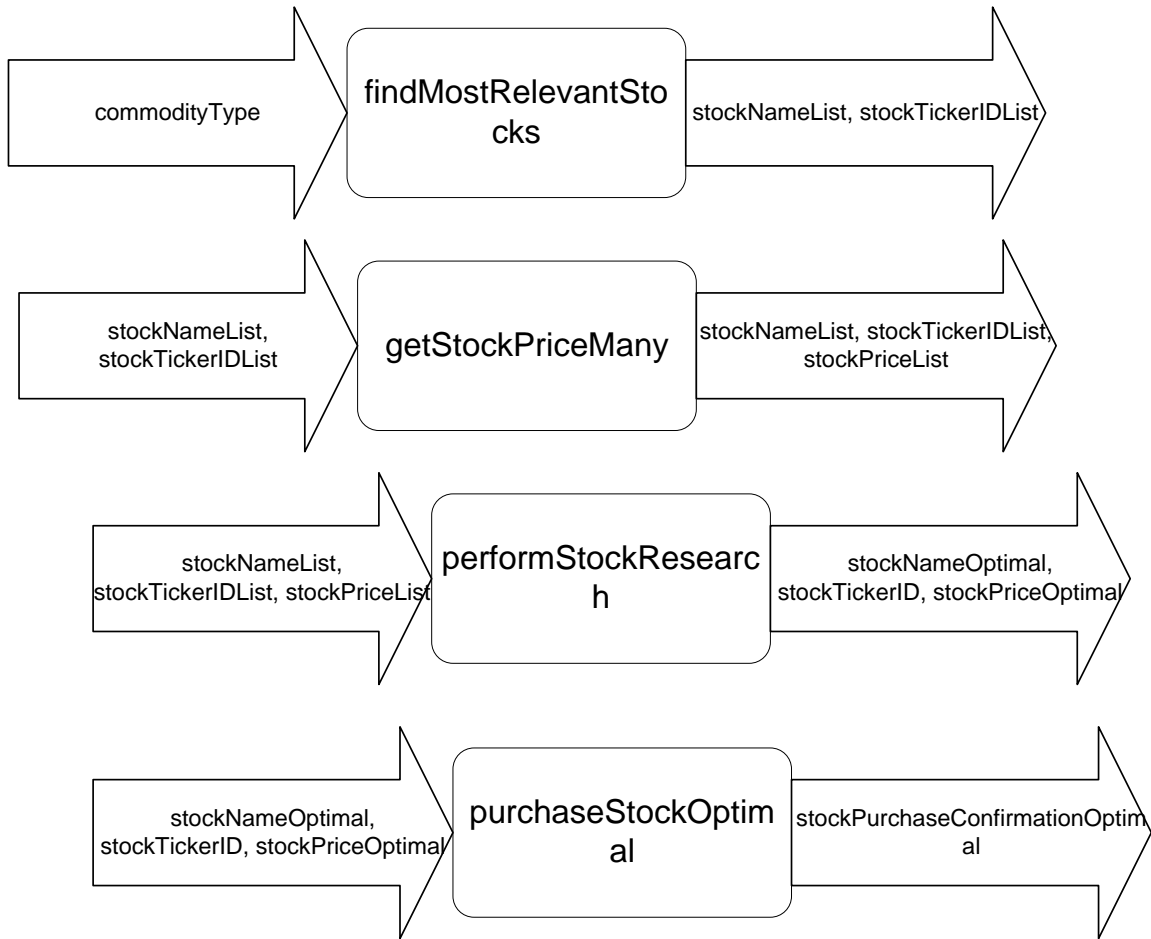
1.



1. **Inputs** = `firstName, lastName, middleInitial, creditCardNum, creditCardExp, creditCardSecID, departCity, departState, destCity, destState, rentalPref, roomPref`

Outputs = `ItineraryURL`

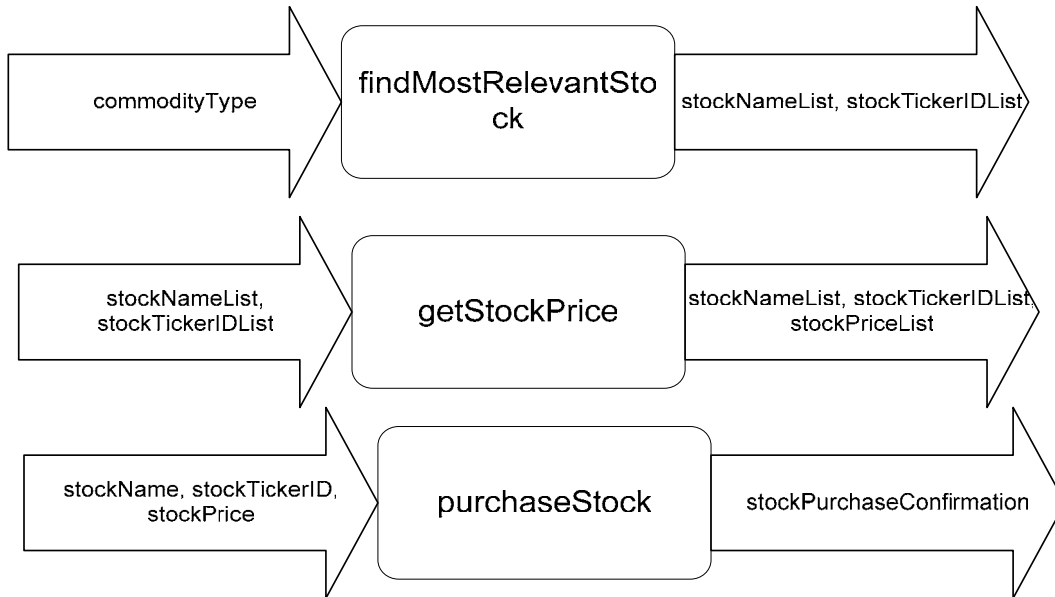
2.



2. **Inputs** = commodityType

Outputs = stockPurchaseConfirmationOptimal

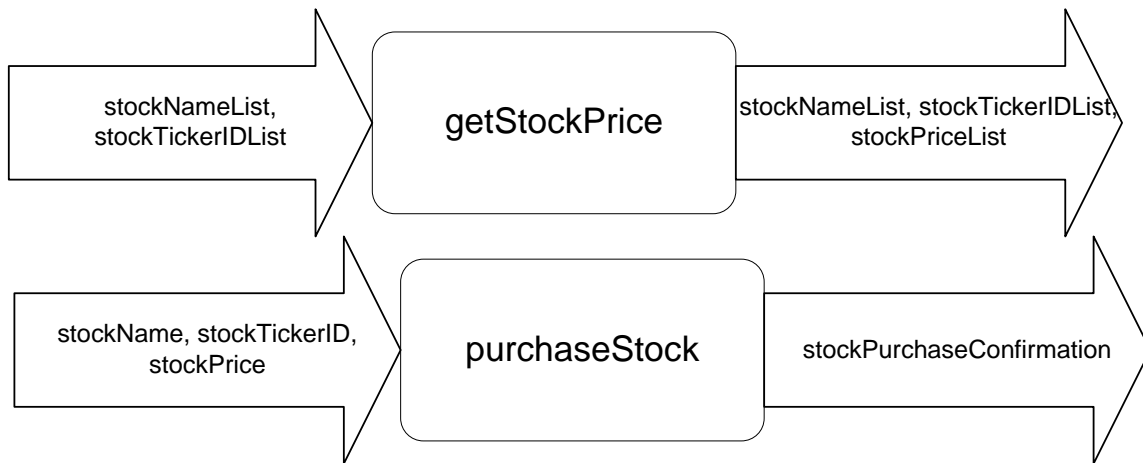
3.



3. **Inputs** = commodityType

Outputs = stockPurchaseConfirmation

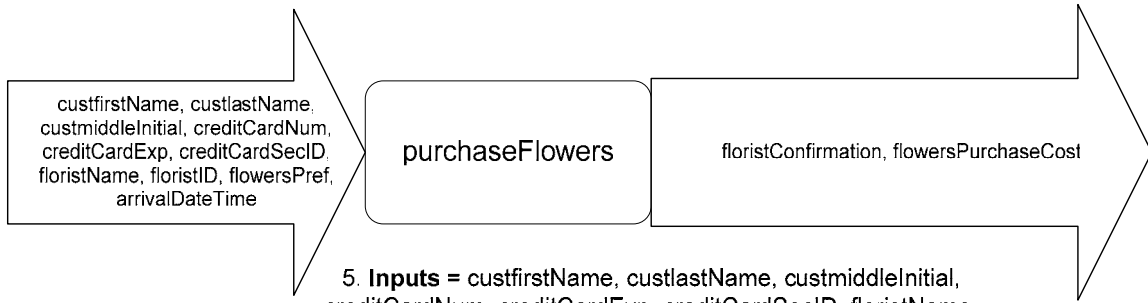
4.



4. **Inputs** = stockNameList, stockTickerIDList

Outputs = stockPurchaseConfirmation

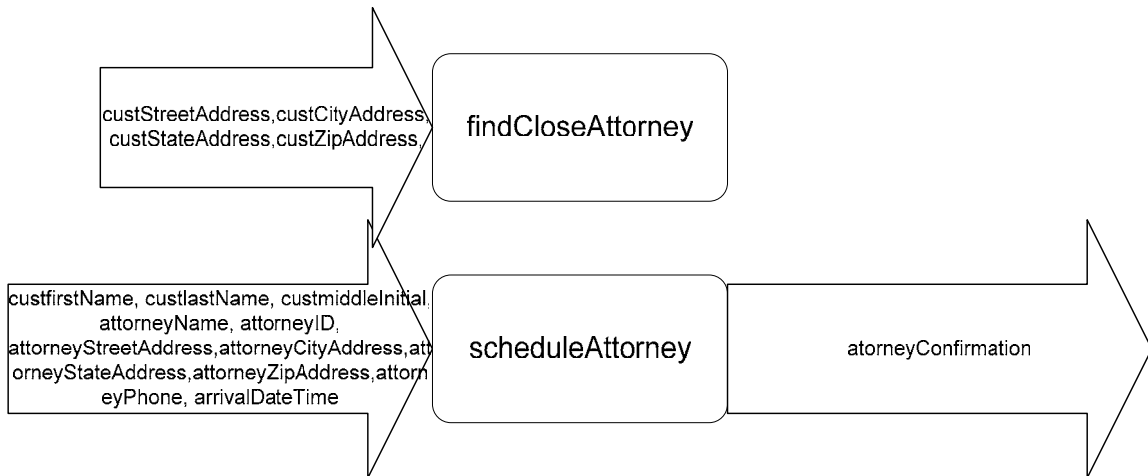
5.



5. **Inputs** = `custfirstName, custlastName, custmiddleInitial, creditCardNum, creditCardExp, creditCardSecID, floristName, floristID, flowersPref, arrivalDateTime`

Outputs = `flowersConfirmation, flowersPurchasePrice`

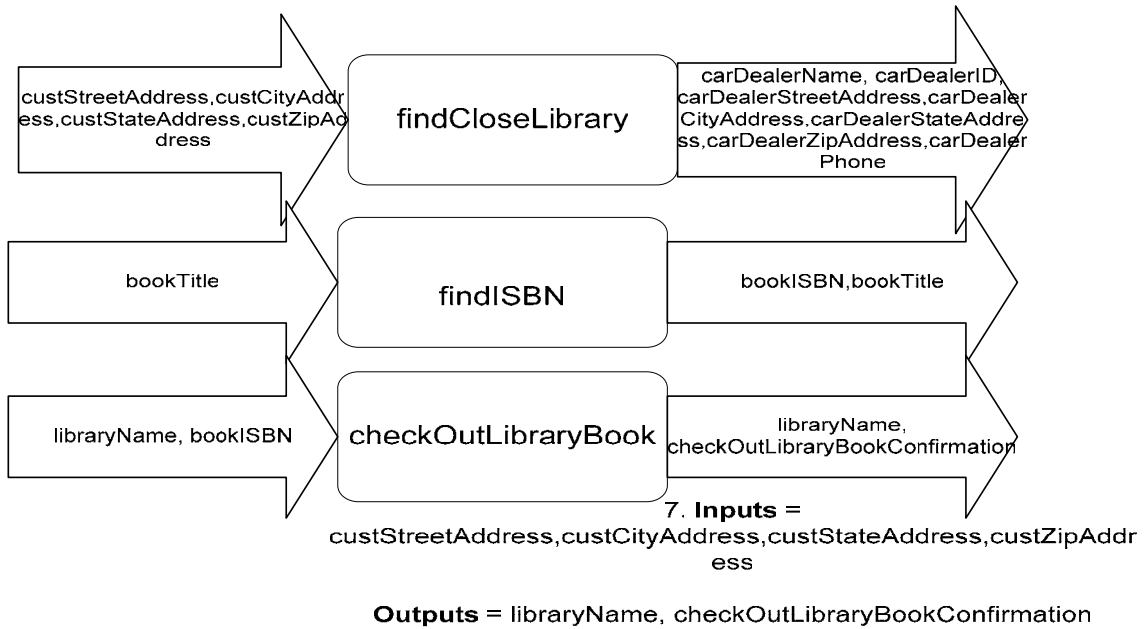
6.



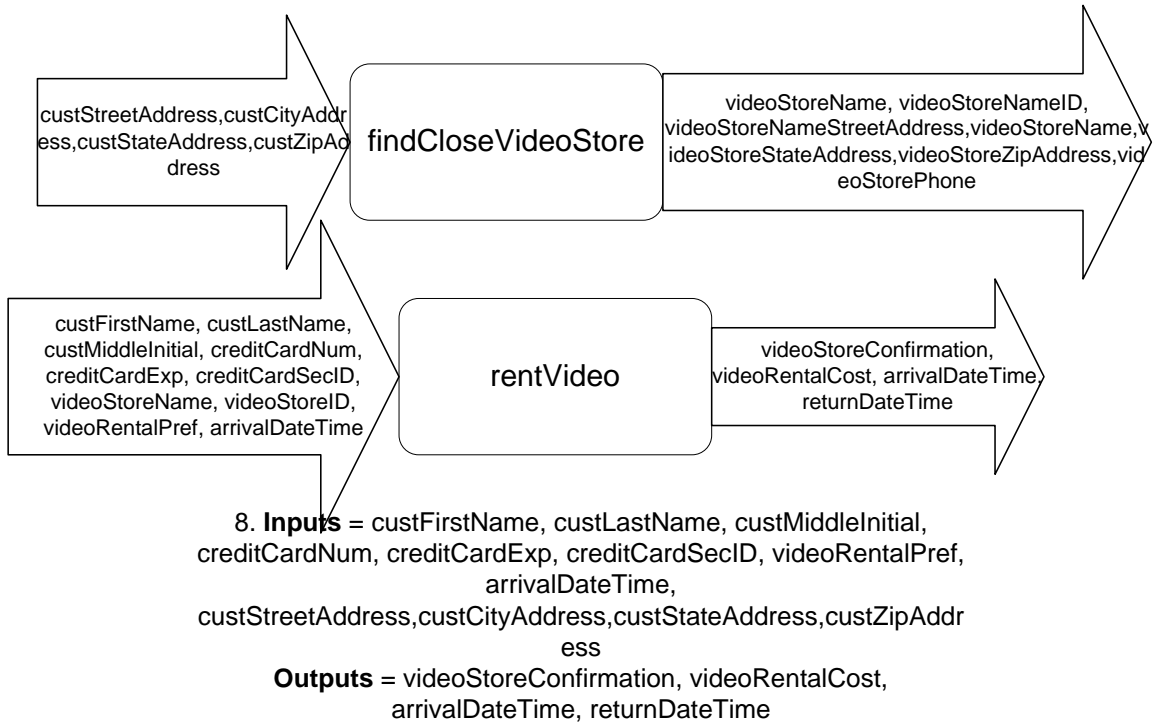
6. **Inputs** =
`custStreetAddress, custCityAddress, custStateAddress, custZipAddress`

Outputs = `attorneyConfirmation`

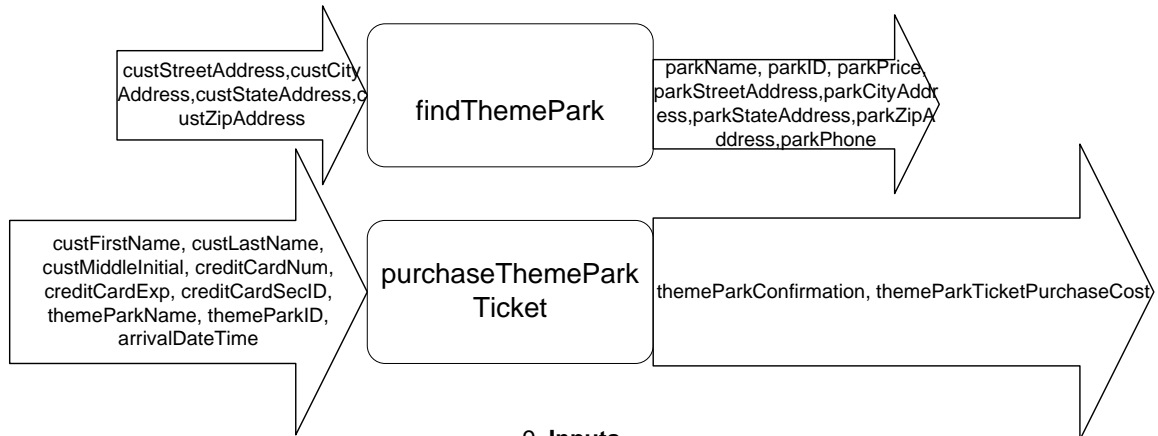
7.



8.

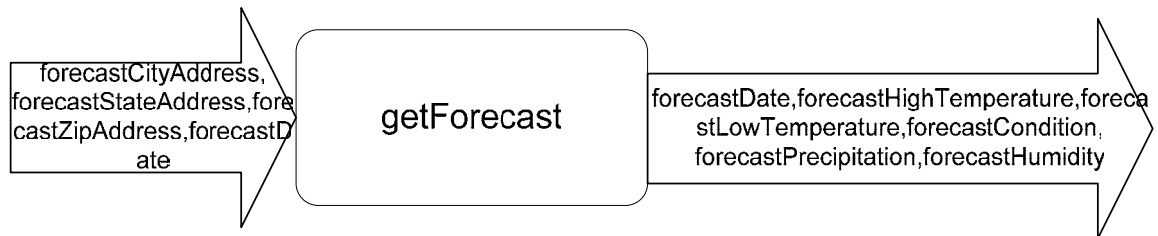


9.



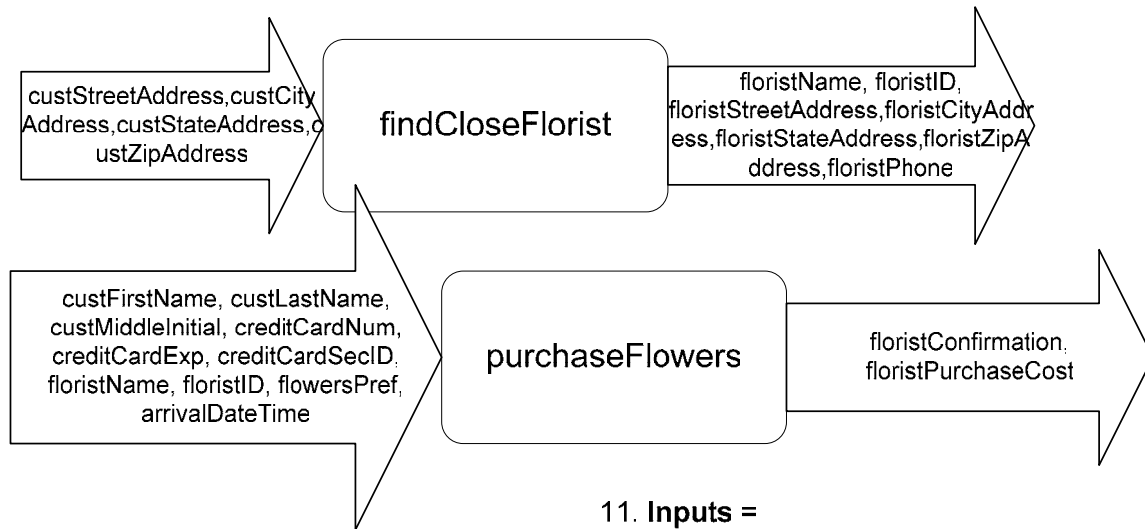
9. **Inputs** =
custStreetAddress,custCityAddress,custStateAddress,custZipAddress,custFirstName, custLastName, custMiddleInitial, creditCardNum, creditCardExp, creditCardSecID, arrivalDateTime
Outputs = themeParkConfirmation, themeParkTicketPurchaseCost

10.



10. **Inputs** = forecastCityAddress, forecastStateAddress, forecastZipAddress, forecastDate
Outputs = forecastDate, forecastHighTemperature, forecastLowTemperature, forecastCondition, forecastPrecipitation, forecastHumidity

11.

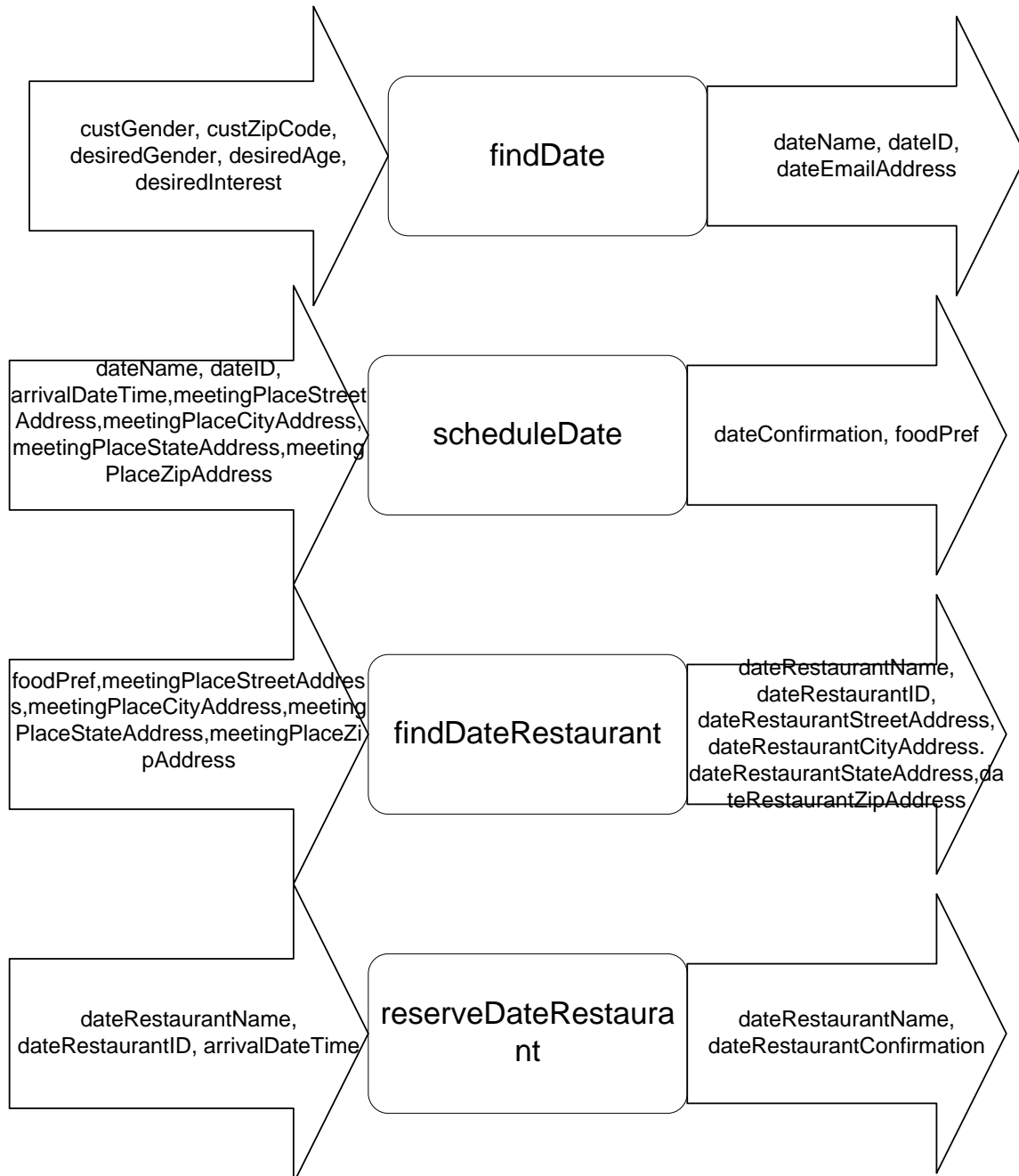


11. Inputs =

fcustStreetAddress, custCityAddress, custStateAddress, custZipAddress, custFirstName, custLastName, custMiddleInitial, creditCardNum, creditCardExp, creditCardSecID, flowersPref,

Outputs = floristConfirmation, floristPurchaseCost

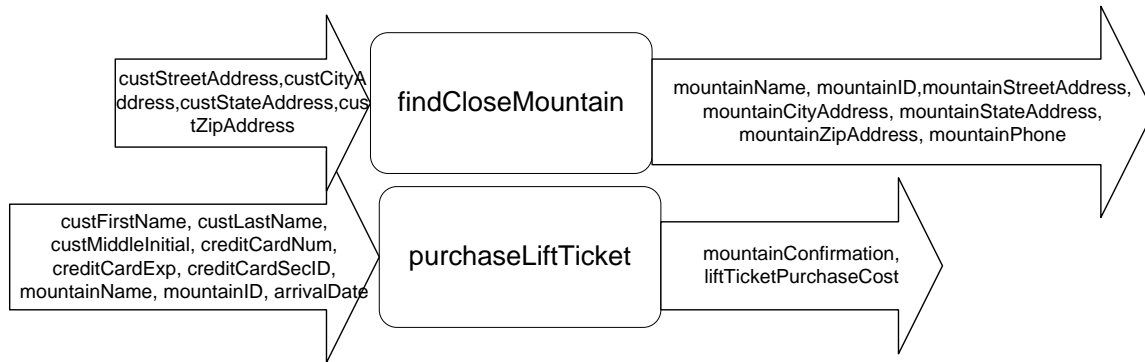
12.



12. **Inputs** = custGender, custZipCode, desiredGender, desiredAge, desiredInterest, meetingPlaceStreetAddress, meetingPlaceCityAddress, meetingPlaceStateAddress, meetingPlaceZipAddress

Outputs = dateRestaurantName, dateRestaurantConfirmation

13.

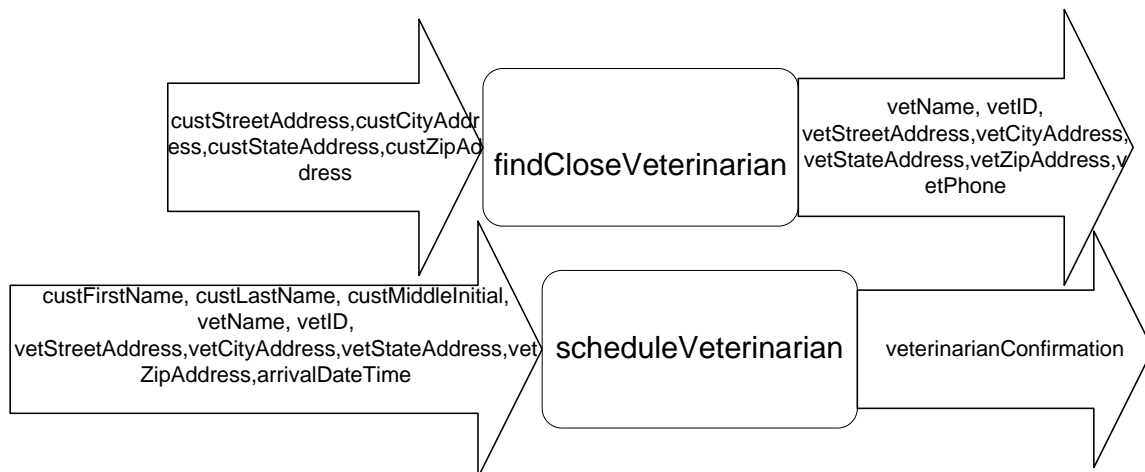


13. Inputs =

custStreetAddress, custCityAddress, custStateAddress, custZipAddress, custFirstName, custLastName, custMiddleInitial, creditCardNum, creditCardExp, creditCardSecID, arrivalDate

Outputs = mountainConfirmation, liftTicketPurchaseCost

14.

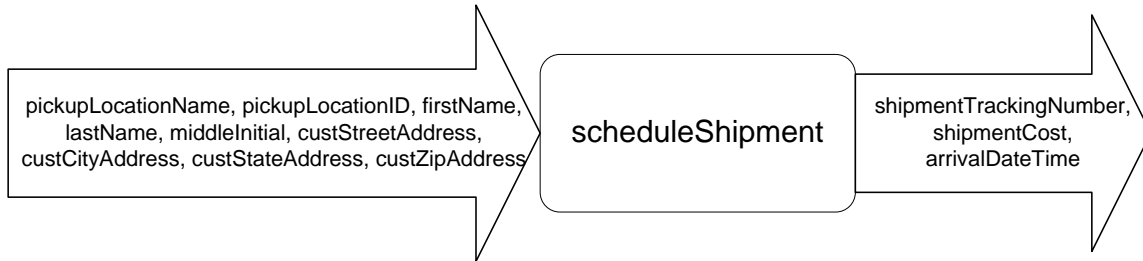


14. Inputs =

custStreetAddress, custCityAddress, custStateAddress, custZipAddress, custFirstName, custLastName, custMiddleInitial, arrivalDateTime

Outputs = veterinarianConfirmation

15.



15. **Inputs** = pickupLocationName, pickupLocationID, firstName, lastName, middleInitial, custStreetAddress, custCityAddress, custStateAddress, custZipAddress
Outputs = shipmentTrackingNumber, shipmentCost