A LANGUAGE FOR BROKERING MATHEMATICAL SERVICES

Olga Caprotti
RISC-Linz
Olga.Caprotti@risc.uni-linz.ac.at

OpenMath TN Meeting
2-3 March 2002
A Framework for Brokering Distributed Mathematical Services


Director: Wolfgang Schreiner, Wolfgang.Schreiner@risc.uni-linz.ac.at

Researchers: Olga Caprotti, Rehabi Baraka

We propose a software framework for brokering mathematical services that are distributed among networked servers. The foundation of this framework is a language for describing the mathematical problems solved by the services. Servers register their problem solving capabilities with a “semantic broker” to which clients submit corresponding task descriptions. The broker (possibly in cooperation with a deduction system) determines the suitable services and returns them to the client for invocation. This mechanism thus hides from the client the actual implementation of mathematical services.
Service provider has registered using UDDI, (Universal Description, Discovery, and Integration) in the *yellow pages* as able to solve a certain problem.
A MathBroker Scenario: Yellow Pages

◇ Service provider has registered using UDDI, (Universal Description, Discovery, and Integration) in the *yellow pages* as able to solve a certain problem
◇ Client (*human* or *software*) has a problem to solve beyond its capabilities
A MathBroker Scenario: Yellow Pages

- Service provider has registered using UDDI, (Universal Description, Discovery, and Integration) in the *yellow pages* as able to solve a certain problem
- Client (human or software) has a problem to solve beyond its capabilities
- Client consults a broker (LB, Komet, MathWeb, ...) which, looking at the meta-data in the yellow pages, returns suitable services
A MathBroker Scenario: Yellow Pages

- Service provider has registered using UDDI, (Universal Description, Discovery, and Integration) in the *yellow pages* as able to solve a certain problem
- Client (human or software) has a problem to solve beyond its capabilities
- Client consults a broker (LB, Komet, MathWeb, ...) which, looking at the meta-data in the yellow pages, returns suitable services potentially or “nearly” suitable services
  stronger input conditions, weaker output conditions
  related problems
A MathBroker Scenario: Yellow Pages

✧ Service provider has registered using UDDI, (Universal Description, Discovery, and Integration) in the the yellow pages as able to solve a certain problem
✧ Client (human or software) has a problem to solve beyond its capabilities
✧ Client consults a broker (LB, Komet, MathWeb, ...) which, looking at the meta-data in the yellow pages, returns suitable services
  potentially or “nearly” suitable services
  stronger input conditions, weaker output conditions
  related problems

✧ Client tries these services to solve its problem
Service provider has registered using UDDI, (Universal Description, Discovery, and Integration) in the yellow pages as able to solve a certain problem.

Client (human or software) has a problem to solve beyond its capabilities.

Client consults a broker (LB, Komet, MathWeb, ...) which, looking at the meta-data in the yellow pages, returns suitable services:
- potentially or “nearly” suitable services
- stronger input conditions, weaker output conditions
- related problems

Client tries these services to solve its problem:
- input must satisfy additional service requirements
- output must satisfy additional client requirements
Mathematical resources and services

Resources are formal mathematical objects

- Axiomatized theories
- Definitions
- Theorems
- Proofs
Mathematical resources and services

Resources are formal mathematical objects

- Axiomatized theories
- Definitions
- Theorems
- Proofs

Services are mathematical problem solvers

- evaluators
- simplifiers
- provers
- solvers
Mathematical resources and services

Resources are formal mathematical objects

- Axiomatized theories
- Definitions
- Theorems
-Proofs

Services are mathematical problem solvers

- Evaluators
- Simplifiers
- Provers
- Solvers

Semantic Web of Mathematics is intended as a collection of mathematical services operating with/on formal resources
Mathematics on the Web is mainly geared to human consumption

There is almost no mathematical resource or service that is meant to or can be processed automatically:

- machine-readable (representation standards)
- machine-understandable (meta-information on properties)
State of the Art: E-Mathematics today

Mathematics on the Web is mainly geared to human consumption

There is almost no mathematical resource or service that is *meant to* or *can be* processed automatically:

- machine-readable (representation standards)
- machine-understandable (meta-information on properties)

GOAL

make resources and *services* in particular usable as black bloxes
no human insight needed
XML, Extensible Markup Language, is a structured format for documents and data. Applications:

- **MathML**, Mathematical Markup Language
- **RDF**, Resource Description Framework
- **WSDL**, Web Services Description Language
- **CDL**, Conversation Definition Language
- **WSFL**, Web Services Flow Language
- ...
Current Situation

(+) Standards for encoding mathematical objects
- MathML
- OpenMath
- OpenMath documents
- docbook with OpenMath
Current Situation

(+) Standards for encoding mathematical objects
   ◦ MathML
   ◦ OpenMath
   ◦ OpenMath documents
   ◦ docbook with OpenMath

(+) Protocols/API for transparent access to remote services abstracting from service location and from object encoding
   ◦ IAMC
   ◦ JavaMath API
   ◦ MathWeb
Current Situation

(+) Standards for encoding mathematical objects
   ◦ MathML
   ◦ OpenMath
   ◦ OpenMath documents
   ◦ docbook with OpenMath

(+) Protocols/API for transparent access to remote services abstracting from service location and from object encoding
   ◦ IAMC
   ◦ JavaMath API
   ◦ MathWeb

(-) Abstraction from service name:
   user/client must know the service name in order to use it

Olga Caprotti, RISC-Linz
(+) Standards for encoding mathematical objects
  ◦ MathML
  ◦ OpenMath
  ◦ OpenMath documents
  ◦ docbook with OpenMath

(+) Protocols/API for transparent access to remote services abstracting from service location and from object encoding
  ◦ IAMC
  ◦ JavaMath API
  ◦ MathWeb

(-) Abstraction from service name:
  user/client must know the service name in order to use it

(-) Abstraction from service functionality:
  user/client need insight on protocol, complexity, availability . . .
Introduce Mathematical Services to Clients

- **Syntactic Interfaces**
  service endpoints, communication protocols
Introduce Mathematical Services to Clients

- Syntactic Interfaces
  service endpoints, communication protocols
- Semantic Behavior
  I/O conditions, complexity
Introduce Mathematical Services to Clients

- Syntactic Interfaces
  service endpoints, communication protocols
- Semantic Behavior
  I/O conditions, complexity
- Pragmatic Issues
  implementation limitations, machine constraints
Use a formal language to define the semantics (e.g. the meta-data) of the mathematical service:

- definitions
- domain axiomatization
- behavioral description by pre-conditions and post-conditions

Additionally consider syntactical and pragmatic issues.
Layers of formalization and description

- **Problem Class**
  - proving
  - solving
  - computing
  - simplifying
  - name
  - input
  - output
  - I/O conditions

- **Problem**
  - complexity
  - I/O conditions

- **Algorithm**
  - name
  - I/O restrictions
  - software system
  - programming language

- **Implementation**
  - name
  - host-port address
  - communication protocol

- **Service**
Layers

- divide and conquer, break down problem
- do the pragmatics first, specify the theory later
- be open to the world, incorporate available description languages
A *problem class* description defines the structure of classes of problems (parametrized by a logical system $L$ and a theory $T$ of $L$) including

- *Computing*, carrying out operations in a domain, i.e., given a term $f$ with a free variable $x$ and a value $a$ in a properly chosen domain, return the value of $f$ with $a$ substituted for $x$;
A *problem class* description defines the structure of classes of problems (parametrized by a logical system $L$ and a theory $T$ of $L$) including

◊ *Computing*, carrying out operations in a domain, i.e., given a term $f$ with a free variable $x$ and a value $a$ in a properly chosen domain, return the value of $f$ with $a$ substituted for $x$;

◊ *Simplifying*, obtaining simpler yet equivalent objects in a domain, i.e., given a term $t$ and a set of simplification rules $R$, return a term $t' = t$ that satisfies some “simplicity” condition,
A *problem class* description defines the structure of classes of problems (parametrized by a logical system $L$ and a theory $T$ of $L$) including

- **Computing**, carrying out operations in a domain, i.e., given a term $f$ with a free variable $x$ and a value $a$ in a properly chosen domain, return the value of $f$ with $a$ substituted for $x$;
- **Simplifying**, obtaining simpler yet equivalent objects in a domain, i.e., given a term $t$ and a set of simplification rules $R$, return a term $t' = t$ that satisfies some “simplicity” condition,
- **Solving**, finding instantiations for indeterminates in a formula i.e., given a set of formulas $F[\bar{x}]$ in a particular logic, find (some or all) substitutions $\bar{a}$ that make $F[\bar{a}]$ true;
A *problem class* description defines the structure of classes of problems (parametrized by a logical system $L$ and a theory $T$ of $L$) including

- **Computing**, carrying out operations in a domain, i.e., given a term $f$ with a free variable $x$ and a value $a$ in a properly chosen domain, return the value of $f$ with $a$ substituted for $x$;
- **Simplifying**, obtaining simpler yet equivalent objects in a domain, i.e., given a term $t$ and a set of simplification rules $R$, return a term $t' = t$ that satisfies some “simplicity” condition,
- **Solving**, finding instantiations for indeterminates in a formula i.e., given a set of formulas $F[\bar{x}]$ in a particular logic, find (some or all) substitutions $\bar{a}$ that make $F[\bar{a}]$ true;
- **Proving**, showing truth of a logical statement i.e., given a closed formula $F$ in a particular logic, determine whether the formula is true in a particular domain or not. For some logic, this is equivalent to returning a proof.
A Computing Problem

<computingProblem
  xsi="http://www.mathematics.net/problem/computing/algebra/realroot">
  <name>Real root isolation</name>
  <comment encoding="xhtml">Real root isolation of a polynomial in Q[x]</comment>
  <input>
    <parameter encoding="openmath"><OMOBJ><OMV name="p"/></OMOBJ></parameter>
    <domain encoding="openmath"><OMOBJ><OMA><OMS name="PolynomialRingR" cd="polyr"/>
      <OMS cd="setname1" name="Q"/><OMV name="x"/></OMA></OMOBJ></domain>
  </input>
  <output>
    <parameter encoding="openmath"><OMOBJ><OMV name="l"/></OMOBJ></parameter>
    <domain encoding="openmath"><OMOBJ><OMA><OMS name="list" cd="list"/>
      <OMA><OMS name="isolating_interval" cd="real_roots"/>
        <OMS cd="setname1" name="Q"/></OMA></OMA></OMOBJ></domain>
  </output>
  <pre><domain="FOL" link="http://mbase.mathweb.org:8080/mbase">...p is square-free...</pre>
  <post><domain="FOL" encoding="openmath">...the elements of l are disjoint open isolating intervals for the roots of p...</post>
</computingProblem>
Problem $P$ specializes to problem $P'$ (EGCD vs GCD)

$\text{pre}(P') \rightarrow \text{pre}(P), \text{post}(P') \rightarrow \text{post}(P)$

Algorithm for $P'$ also solves $P$

Algorithm for $P'$ also solves $P$ under additional constraints

Want $P'$: try $P$

Want $P$: try $P'$ with additional checks

Want $P'$: may even try $P''$ with additional checks
Methodological Approach

The goal of this project is not to invent new conceptual models for the relationship between mathematical/logical theories but to provide a concrete software framework for the mathematical community. It shall be as far as possible based on (i.e., integrate and generalize) existing activities of other groups that have resulted in concrete models and/or software components (OpenMath, IAMC, JavaMath, MathWeb, MathBus) and make utmost use of widely accepted computing and communication standards (in particular, the XML world).

- evaluate available e-business technologies for web services (SOAP, WSDL, UDDI)
- select two or three case studies to guide the development
- XML, Java, server-side programming
MathBroker Links

Stable server: http://poseidon.risc.uni-linz.ac.at:8080/mathbroker

Development server: http://perseus.risc.uni-linz.ac.at:8080/mathbroker

Test web services: http://perseus.risc.uni-linz.ac.at:8080/openmath
http://localhost:8080/axis/services/OMObjProcessor?wsdl
http://localhost:8080/axis/services/SymbolicEvaluator?wsdl
TOOLS

◊ **AXIS** [http://xml.apache.org/axis/](http://xml.apache.org/axis/)
  deployed services in Axis can be accessed with a standard web browser and appending "?WSDL" to the end of the URL, generates a WSDL description of the service.
  "WSDL2Java" tool builds Java proxies and skeletons for services with WSDL descriptions.
  "Java2WSDL" tool builds WSDL from Java classes.
TOOLS

◊ **AXIS** [http://xml.apache.org/axis/](http://xml.apache.org/axis/)
  deployed services in Axis can be accessed with a standard web browser and appending "?WSDL" to the end of the URL, generates a WSDL description of the service.
  "WSDL2Java" tool builds Java proxies and skeletons for services with WSDL descriptions.
  "Java2WSDL" tool builds WSDL from Java classes.

  WSDL4J and WSIL4J APIs provide an object model for WSDL documents.
  UDDI4J API for publishing and finding web services in a UDDI Registry.
  WSDLDocument API provides an way to read, write and (with WSDL4J APIs) extract information from WSDL documents.
TOOLS

◊ **AXIS** [http://xml.apache.org/axis/](http://xml.apache.org/axis/)
deployed services in Axis can be accessed with a standard web browser and appending "?WSDL" to the end of the URL, generates a WSDL description of the service.
"WSDL2Java" tool builds Java proxies and skeletons for services with WSDL descriptions.
"Java2WSDL" tool builds WSDL from Java classes.

WSDL4J and WSIL4J APIs provide an object model for WSDL documents.
UDDI4J API for publishing and finding web services in a UDDI Registry.
WSDLDocument API provides an way to read, write and (with WSDL4J APIs) extract information from WSDL documents.

◊ **JWSDP** [file:///usr/local/src/jwsp-1_0-ea1/docs/index.html](file:///usr/local/src/jwsp-1_0-ea1/docs/index.html) Java Web Services Developer Pack
TOOLS

- **AXIS** http://xml.apache.org/axis/
  deployed services in Axis can be accessed with a standard web browser and appending "?WSDL" to the end of the URL, generates a WSDL description of the service.
  "WSDL2Java" tool builds Java proxies and skeletons for services with WSDL descriptions.
  "Java2WSDL" tool builds WSDL from Java classes.

- **IBM** file:///usr/local/src/wstk-3.0/doc/api/index.html Web Services ToolKit (WSTK)
  WSDL4J and WSIL4J APIs provide an object model for WSDL documents.
  UDDI4J API for publishing and finding web services in a UDDI Registry.
  WSDLDocument API provides an way to read, write and (with WSDL4J APIs) extract information from WSDL documents.

- **JWSDP** file:///usr/local/src/jwsp-1_0-ea1/docs/index.html Java Web Services Developer Pack
It seems that WSDL is intended to model services as intended by e-business, hence a service is a collection of ports where each port is a specific aspect of a service transaction.

Future work has to establish how to use WSDL for modelling mathematical services in either way

- model mathematical services within WSDL by exploiting the extensionality bindings
  - (XSD, Relax?) schema for message syntactic types (OMOBJ vs OMOBJ-polys)
  - XSD schema for message as a triple: parameter, parameter type, conditions on the parameter
- design a new description language for mathematical services (MSDL) from which to generate WSDL