Brokering of Theorem Proving Services Described in MSDL

Jürgen Zimmer

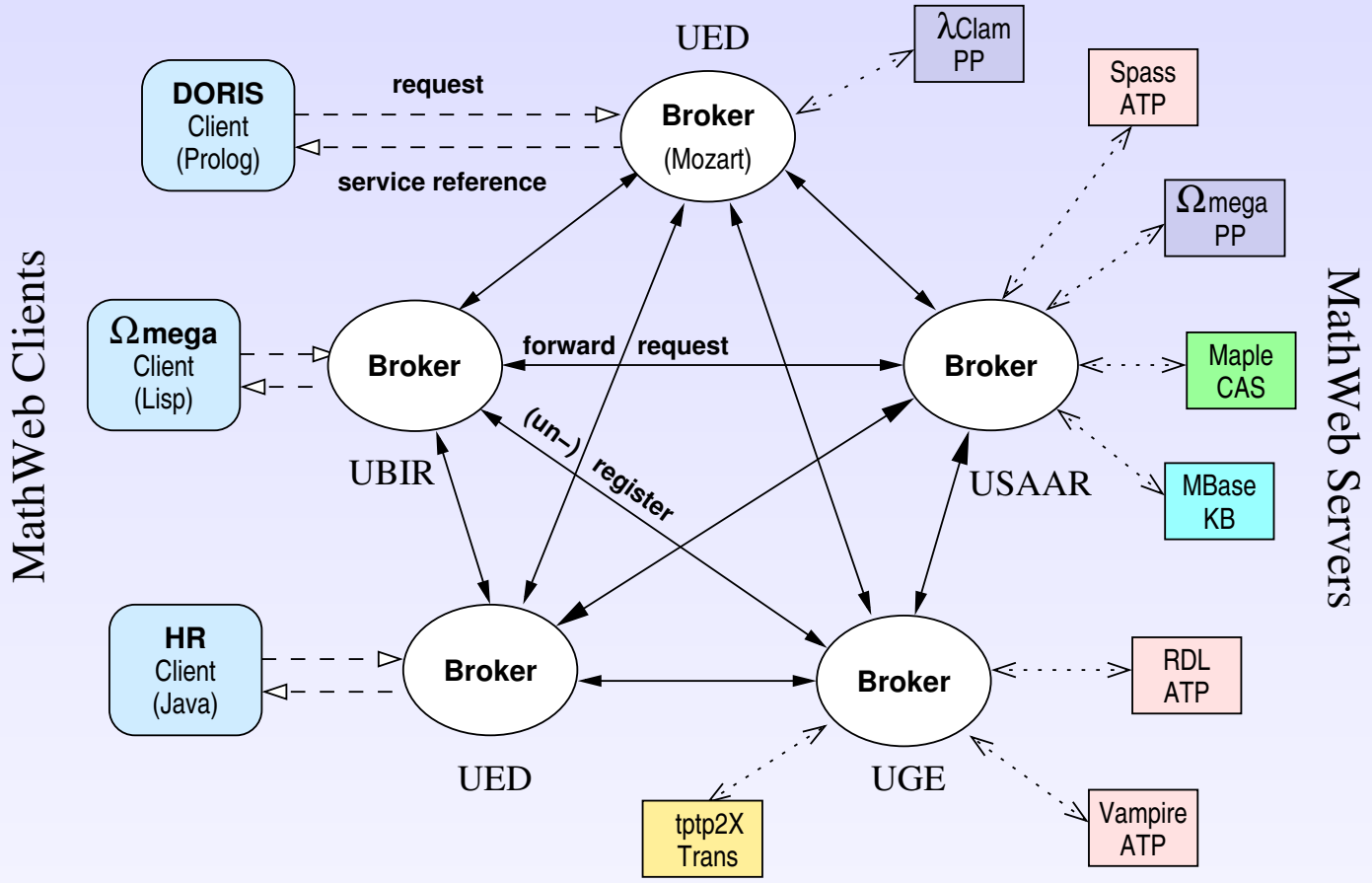
University of Edinburgh, Scotland
Universität des Saarlandes, Germany

This work is supported by the CALCULEMUS IHP Training Network
HPRN-CT-2000-00102

Overview

- Motivation
- Our Agent Framework
- An Ontology for Deductive Services
- First-order ATP Services
- Classification of Theorem Proving Problems
- Broker for FO-ATP Services
- A Little Demonstration

Motivation


UED
λClam PP
DORIS Client (Prolog)
request
service reference
Broker (Mozart)
Spass ATP
Ωmega PP
MathWeb Clients
Ωmega Client (Lisp)
Broker
forward request
Broker
Maple CAS
(un–) register
UBIR
USAAR
MBase KB
MathWeb Servers
HR Client (Java)
Broker
Broker
RDL ATP
UED
UGE
tptp2X Trans
Vampire ATP
broker to broker communication
client to broker communication (Mozart, XMLRPC, HTTP)
server to broker communication (service offers/requests)

Motivation

Despite its success, MathWeb-SB some limitations:

- Client applications still have to know
  - which reasoning system to use, and
  - how to access the system.
- User has to coordinate different reasoning systems to solve a problem.
- The MathWeb-SB is not designed for asynchronous communication.
- Technical Problems (OS, Firewalls, Proxies)

A New Agent Framework

From MathWeb-SB to the Semantic MathWeb-SB:

- ... based on FIPA compliant agent platform (JADE)
- ... agents offering reasoning services described in service description language:

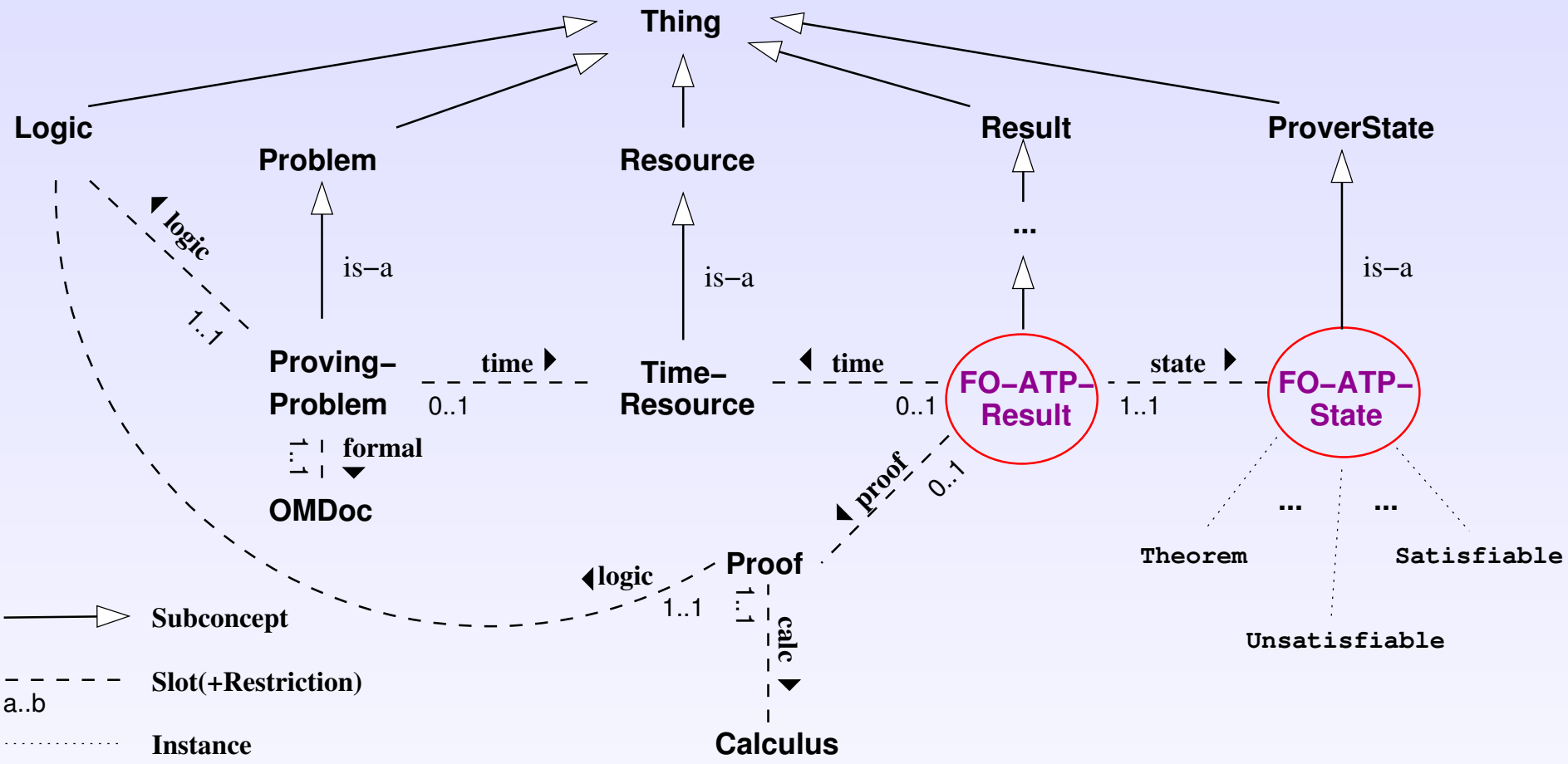
  $\Rightarrow$ currently MSDL
  (developed in MONET and MathBroker)

  $\Rightarrow$ in the future also OWL-S (OWL)?
- ... a brokering mechanism for reasoning services.

Ongoing Work

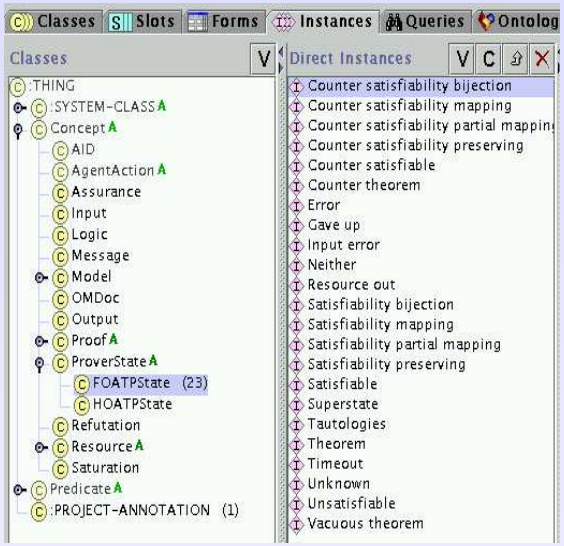
- Implementing agents and communication in JADE
- Designing an Ontology in Protégé-2000 tool (Protégé-2 supports OWL).
- Describing first-order Automated Theorem Provers (ATPs) in MSDL using our ontology.
- Mapping MSDL to Java classes.
- Implementing simple brokering:
  - analysing problems in queries.
  - find most suitable service.

An Ontology for Deduction Services



One Ontology for All

Protege Ontology:



Beangenerator

JADE Agent Ontology:

```
package org.mathweb.onto;
import jade.content.*;
import jade.util.leap.*;
import jade.core.*;

public class FOATPState extends ProverState{
private List returns = new ArrayList();
public void addReturns(Assurance o) {returns.add(o);}
public boolean removeReturns(Assurance o) {return returns.remove(o)
public void clearAllReturns() {returns.clear();}
                    ...
```

JiBX?

Ontology for Services (MSDL):

```
        ...
<input name='problem'>
  <signature>
     http://www.mathweb.org/ontology/atp#TPTPProblem
  </signature>
</input>
        ...
```

First-order Theorem Proving Services

Every first-order proving service...

- ... accepts standard problem formats TSTP and OMDoc.
- ... returns a proof object in standard format (TSTP).
- ... is specialized on a particular domain.

For an ontology for first-order ATPs use:

- Experience with the MathWeb-SB.
- Recent work with Geoff Sutcliffe and Stephan Schulz on ATP states.

First-order Theorem Proving Problems

FO proving problems classified using known features (Sutcliffe & Suttner 2001):

- Logical Class of problem:
  - $essentiallyPropositional$ (finite Herbrand Universe)
  - $realFirstOrder$
- Equality: $noEquality$, $someEquality$, $pureEquality$
- Presentation: $fofForm$ (first-order formulae)
  vs. $cnfForm$ (clause normal form)

$\Rightarrow$ 16 “Specialists Problems Classes” (SPC)

Performance data of well-known ATPs is available for every SPC.

The ATP SPASS in MSDL

Service: SpassProver	
classification:	Classification with Taxonomy of services or link to Ontology ($\rightarrow$ QPQ) • $\rightarrow$ **spass problem description**
service interface:	($\rightarrow$ `fo-prover.wsdl`)
implementation details:	Information about hardware, software (calculus, etc.)

spass problem description	
input parameters:	name: $problem$, signature: FO-ATP-Problem (Ontology concept)
output parameters:	name: $result$, signature: FO-ATP-Result (Ontology concept)
pre-conditions:	$essentiallyPropositional(problem)$ (OpenMath Object) $\wedge\ noEquality(problem)$ $\wedge\ fofForm(problem)$
post-conditions:	

The ATP E in MSDL

Service: EProver	
classification:	Classification with Taxonomy of services or link to Ontology ($\rightarrow$ QPQ) • $\rightarrow$ **e problem description**
service interface:	($\rightarrow$ `fo-prover.wsdl`)
implementation details:	Information about hardware, software (calculus, etc.)

e problem description	
input parameters:	name: $problem$, signature: FO-ATP-Problem (Ontology concept)
output parameters:	name: $result$, signature: FO-ATP-Result (Ontology concept)
pre-conditions:	$realFirstOrder(problem)$ $\wedge\ pureEquality(problem)$ $\wedge\ nonUnitPureEquality(problem)$
post-conditions:	

A First Broker

Our broker ...

- ... analyses incoming proving problems
  using TSTP classifier (G. Sutcliffe).
- ... annotates problems with new preconditions
  $\Rightarrow$ puts problem in SPC.
- ... matches new problem with available services
  using tuProlog engine (DEIS, Università di Bologna).
- ... calls the first matching service.

A little Demo

- Two services: EProver & SpassProver
- Two queries:
  - **Query 1:** Did Agatha commit suicide?
    $\forall x$. Agatha hates x $\Rightarrow$ Butler hates x, ...
    classified as $essentiallyPropositional$, $noEquality$, $fofForm$
  - **Query 1:** Problem in group theory:
    $\forall x.x * e = x \wedge \forall x.x * x^{-1} = e \wedge \forall x.x * x = e \wedge \forall x.a * b = c$
    $\vdash b * a = c$
    classified as $realFirstOrder$, $pureEquality$, $nonUnitPureEquality$
- Consequently, broker selects
  SpassProver for **Query 1** and
  EProver for **Query 2**.

Problems To Solve

Among others...

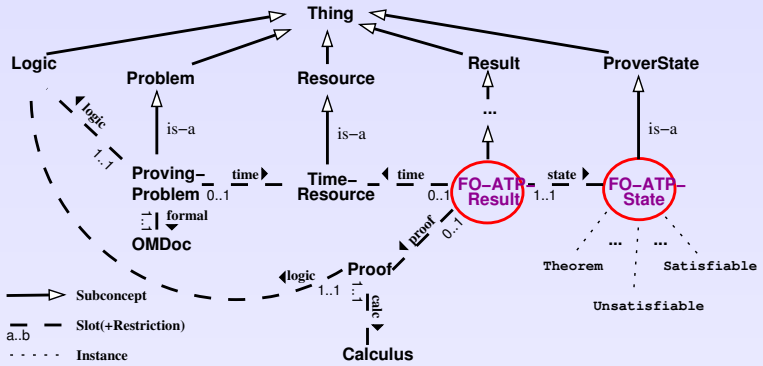
- Binding of MSDL to agent Ontology
  Currently, we use JiBX. Mathbroker uses JAXB.
- I/O parameters of MSDL abstract problems:
  ⟨OMOBJ⟩ vs. XML-serialisation of Java Objects
  (FO-ATP-Problem)
- Invocation of service through JADE agent behaviours
  (conversations).
- Combination of behaviours.

Future Work

- Knowledge retrieval from ATP users and developers.
- Case study to show benefit of brokering.
- Extend ontology to logics & calculi.
- SOAP binding for MONET and Mathbroker.
- Ontology in OWL ($\rightarrow$ Protege-2.0 & HarmonIA).
- Description of other reasoning systems
  (e.g., model generators).
- Advanced brokering (dynamic combination of services).

A Dream: MSDL Service Authoring Tool

Ontology



Repository of already specified JADE Agent Behaviours

```
<DialogicFramework id="FishDialFrw" contentLanguage
  <Ontology id="FishONTO" specificationLanguage="
  <Role id="Boss" type="InternalRole">
    <Behaviour id="Open" type="SimpleBehaviour"/>
    <Behaviour id="Discuss" type="CompositeBehavio
      <Behaviour id="InitialResolution" type="OneShot
      <Behaviour id="FinalResolution" type="OneShot
    </Behaviour>
```

revise

use

Service Editor