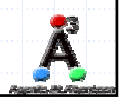
Constraint Logic Programming Applications on the Semantic Web

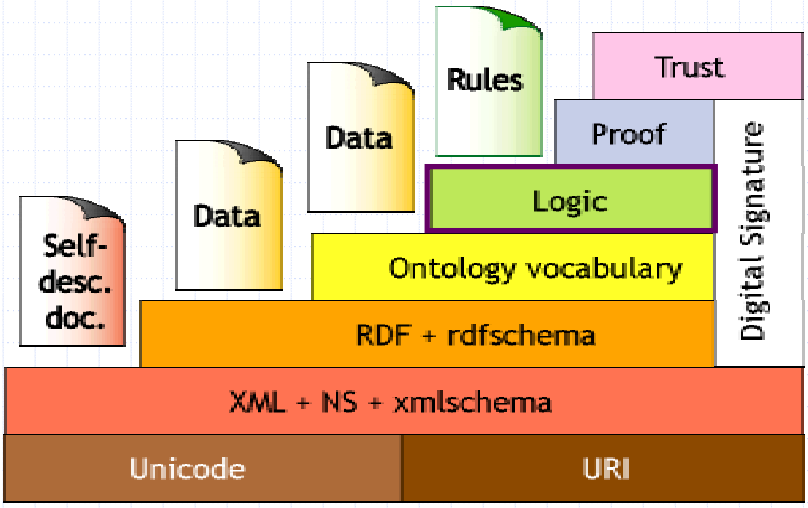
Alun Preece
Department of Computing Science
University of Aberdeen, Scotland, UK
apreece@csd.abdn.ac.uk
http://www.csd.abdn.ac.uk/~apreece

Semantic Web

- Aim: to create a network of machine-processable resources
- Existing in parallel with the current World Wide Web
- Information is marked-up against semantic data models
- Enables software agents to carry out tasks on users' behalf
- Moving from a Web of "finding things" to a Web of "doing things"
- New Semantic Web services will exploit AI techniques



Semantic Web architecture



[Semantic Web 'layer cake' slide due to Tim Berners-Lee]

Problem-solving via CLP

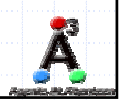
- Many problem-solving tasks can be modelled and enacted as constraint logic programs
  - tasks concerning a single user
    e.g. configuring a product to meet a set of requirement constraints
  - tasks involving multiple users
    e.g. arranging a meeting to satisfy everyone's scheduling constraints
- Such tasks are very relevant in the Semantic Web context
  - offering new services to users by exploiting semantically marked-up information
    e.g. product catalogues, people's schedules

This talk

- Issues
  - representing constraints in a Semantic Web-friendly format
  - interfacing constraint solvers to the open Web environment
- Example applications using CLP to deliver Semantic Web services
  - Planning an evening's entertainment for a visitor to Aberdeen
  - "Infotainment" service-provider coalition formation
  - Supporting design teams

Semantic Web constraint format

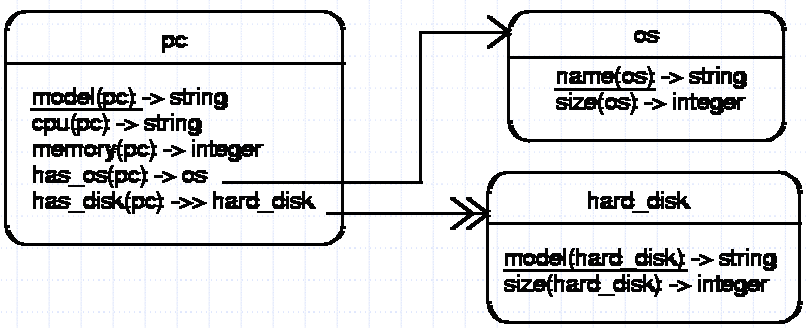
- Serialisable into XML, to make it maximally portable and open
- Constraints should be represented as resources in RDF, so statements can be made about the constraints
- No modification to the existing RDF and RDFS specs, so the CIF would be layered cleanly on top of RDF (and the layers above RDF)
- Possible for constraints to refer explicitly to terms defined in any RDF Schema



XML-CIF

- Modelled using RDF Schema
- Constraints become web resources about which statements can be made (authorship, context, strength, ...)
- Constraints refer to RDFS classes (entities) & properties (relations)
- (We're using RDFS as a lightweight ontology)
- In principle, any RDFS vocabulary can use XML-CIF and our constraint solving services

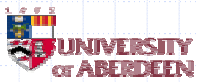
P/FDM schemas



Extended E-R semantic data model

P/FDM to RDFS mapping

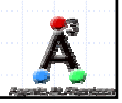
- P/FDM class c declared as c ->> entity maps to an RDF resource of type rdfs:Class
- P/FDM class c declared as c ->> s maps to an RDF resource of type rdfs:Class, with a property rdfs:subClassOf s
- P/FDM function f declared as f(c) -> r maps to an RDF resource of type rdf:Property with rdfs:domain c and rdfs:range r



Colan constraints

```
constrain each p in pc
    to have size(has_os(p))
        =< size(has_disk(p))

constrain each p in pc such that
    manufacturer(p) = "Apple"
    to have name(has_os(p))="MacOS X"
```

This is just syntactically sugared FOL, but it's aimed to be readable to domain experts.

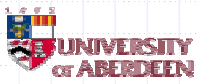


Variables in constraints

- In Colan a variable is always introduced in conjunction with a set that it ranges over.
- Terms such as (p in pc) and (e in employee) are common:

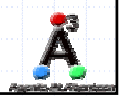
```
(p in pc) such that name (p) = "iMac"
(e in employee) such that
      salary (e) > 5000 and age (e) < 50
```

- Represented by Setmem metaclass
- Variables are described by the Variable class



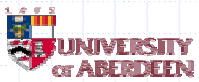
RDFS setmem defn

```
<rdfs:Class rdf:ID="Setmem">
  <rdfs:subClassOf rdf:resource="#Boolprim"/>
</rdfs:Class>

<rdf:Property rdf:ID="var">
  <rdfs:domain rdf:resource="#Setmem"/>
  <rdfs:range rdf:resource="#Variable"/>
</rdf:Property>

<rdf:Property rdf:ID="set">
  <rdfs:domain rdf:resource="#Setmem"/>
  <rdfs:range rdf:resource="#Setexpr"/>
</rdf:Property>
```



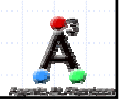
XML-CIF for "(p in pc)"

```
<cif:Setmem>
  <cif:var>
    <cif:Variable rdf:ID="#p"/>
  </cif:var>
  <cif:set>
    <cif:Entset>
      <cif:entclass rdf:resource=
      "http://www.aktors.org/domain/pc_config#pc"
      />
    </cif:Entset>
  </cif:set>
</cif:Setmem>
```



Target applications

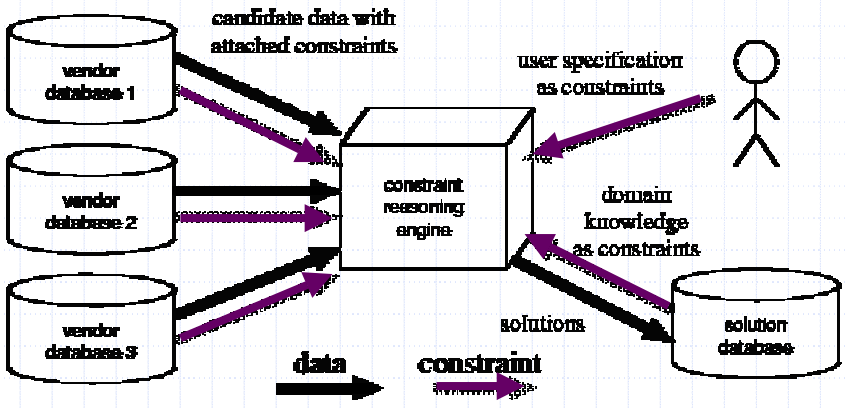
- Supports apps in which info is moved across a network with rich metalevel info on how to use it
- In B2B ecommerce, composition of package products from vendor catalogue components
  - consumer electronic equipment
  - package holidays
  - financial products
- Constraints must be aggregated and solved over available component instances



Fusion of constraints

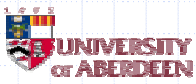
- Customer requirements
  - "I want a PC with a colour printer"
- Constraints on acceptable packages
  - "any printer must have a driver that is compatible with the PC OS"
- Constraints restricting component use
  - "this printer has drivers only for Windows OSes"

Constraint fusion services



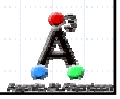
Constraint solver Web services

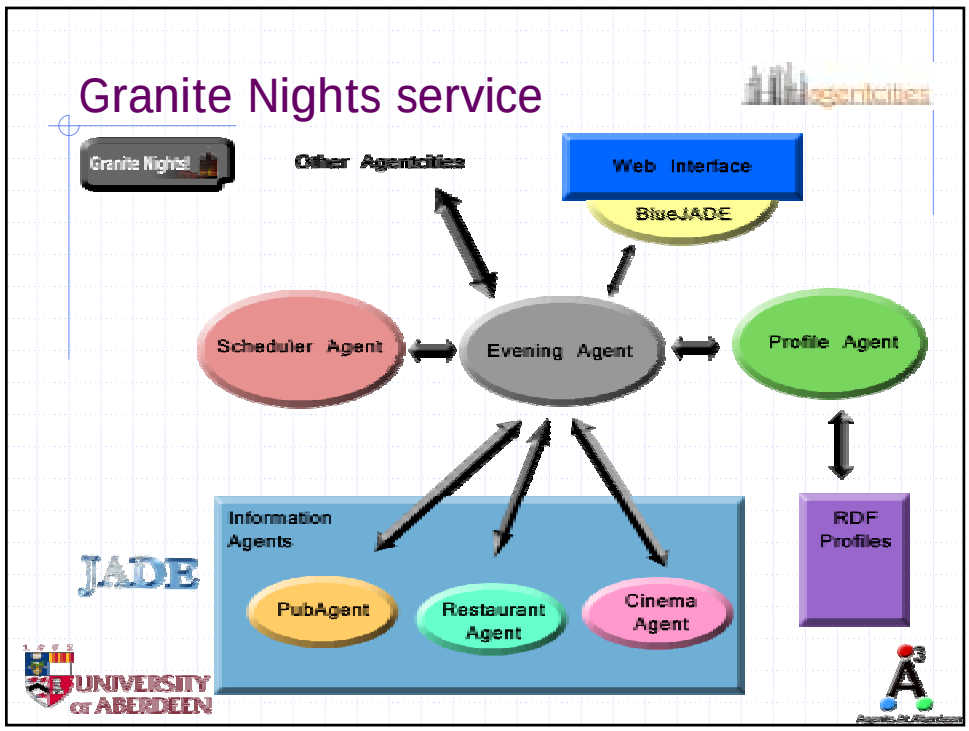
- Wrapping of Sicstus Prolog FD solver as a FIPA-compliant agent
  - Messaging is FIPA ACL over HTTP
  - Content is RDF
  - Platform is JADE+Jasper+Sicstus
- Wrapping of ECLIPSE as an XML-RPC Web service
  - Messaging is SOAP-like AKTbus
  - Content is again RDF
  - Platform is Prolog+Linda+ECLIPSE

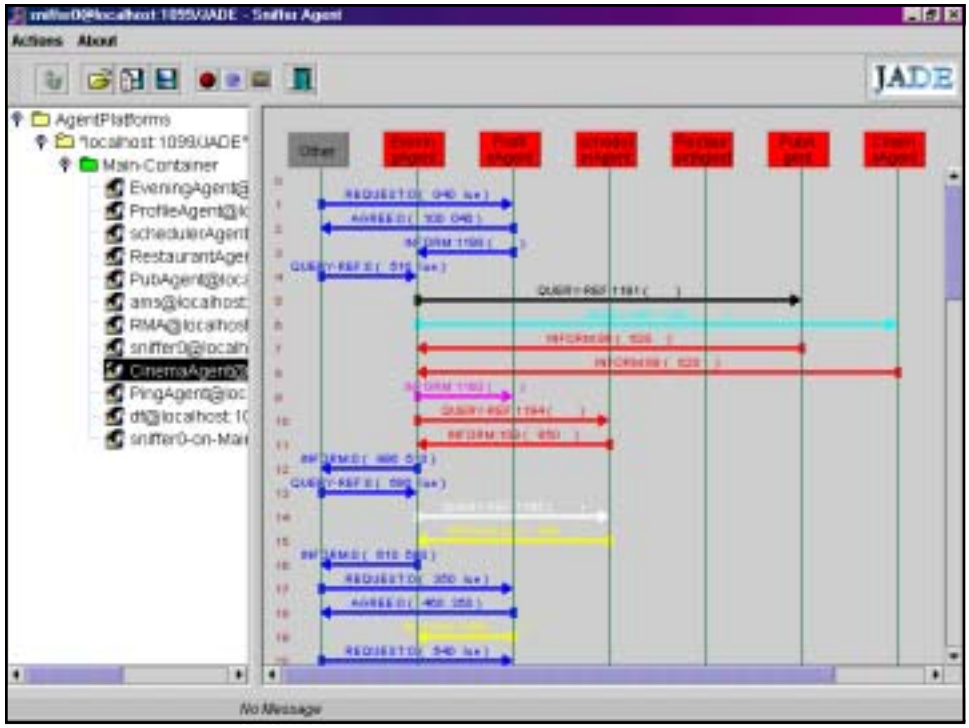


Example applications

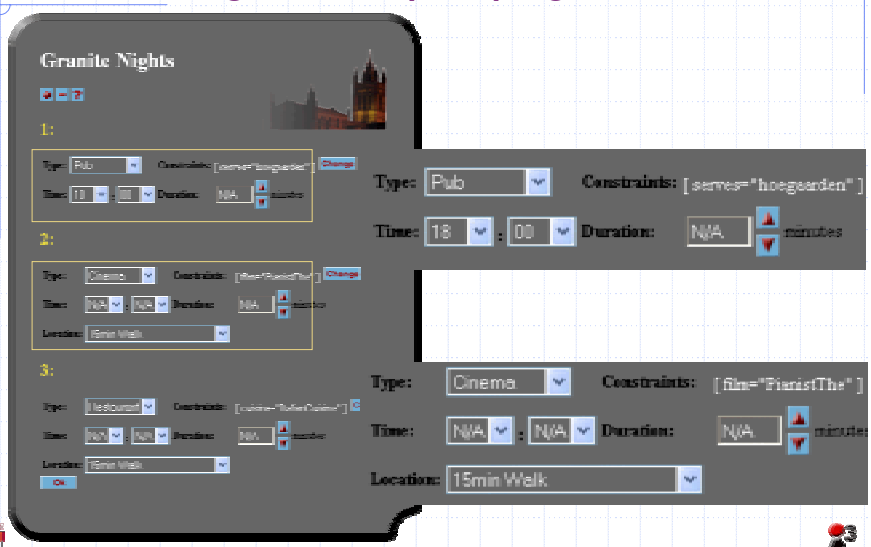
- Planning an evening's entertainment for a visitor to Aberdeen
- "Infotainment" service-provider coalition formation
- Supporting design teams



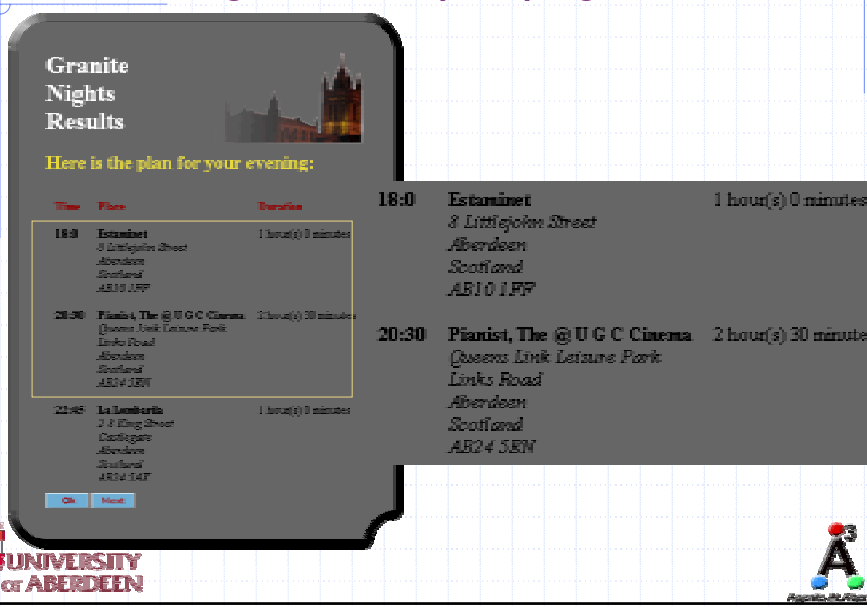

Granite Nights service
agenticities
Granite Nights!
Other Agenticities
Web Interface
BlueJADE
Scheduler Agent
Evening Agent
Profile Agent
RDF Profiles
Information Agents
PubAgent
Restaurant Agent
Cinema Agent
JADE
UNIVERSITY OF ABERDEEN


Actions About
JADE
AgentPlatforms
Main-Container
No Message

Granite Nights – input page



Granite Nights – output page



Dynamic info source (cinemas)

Cinema Agent


```
<s:Shows rdf:ID="ugc_PianistThe">
  <s:time>
    <s:ShowScheduleCollection>
      <s:consistsOf><s:ShowSchedule>
        <s:startTime>
          <c:Calendar><c:calendarDate>
           <c:Date>
            <c:dateDayOfWeek
               rdf:resource="cal#Thursday" />
            <c:year>2003</c:year>
            <c:month>1</c:month>
           </c:Date>
          </c:calendarDate><c:calendarTime>
           <c:Time>
             <c:format rdf:resource="cal#24h"/>
             <c:timeHour>20</c:timeHour>
             <c:timeMinute>20</c:timeMinute>
...
  <s:location rdf:resource="cinemas#ugc" />
  <s:description>Certificate: 15</s:description>
  <s:show>
    <s:CinemaPerformance rdf:ID="PianistThe">
    <s:title>Pianist, The</s:title>
```

Static info source (restaurants)

Restaurant Agent

```
<res:Restaurant rdf:about="#lalombarda">
  <res:name>La Lombarda</res:name>
  <res:averageMealDuration>2
    </res:averageMealDuration>
  <res:address>
    <add:Address rdf:about="rest#lombardaaddr"/>
  </res:address>
  <res:atmospheres
   rdf:resource="res#CasualAtmosphere"/>
  <res:atmospheres
   rdf:resource="res#RelaxedAtmosphere"/>
  <res:caterings rdf:resource="res#ALaCarte"/>
  <res:caterings rdf:resource="res#HomeDelivery"/>
  <res:facilities rdf:resource="res#SmokingFacility"/>
  <res:typeOfCuisine
   rdf:resource="res#ItalianCuisine"/>
</res:Restaurant>
```

RDF Query-by-Example

- Principle: query RDF using RDF
- If users can read RDF descriptions, they can write patterns that match RDF descriptions
- Example: "get all pubs serving Guinness beer"

```
<q:Query>
  <q:template>
    <p:EnglishPub>
      <p:servesBeer
          rdf:resource="beertypes#guinness"/>
    </p:EnglishPub>
  </q:template>
</q:Query>
```

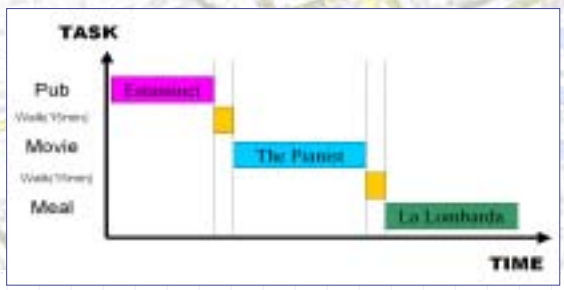
More complex queries use CIF expressions

UNIVERSITY OF ABERDEEN

Scheduling agent

Scheduler Agent

- Convert RDF to Prolog representation
- Uses Sicstus Jasper to interface Prolog and Java
- Uses Constraint Logic Programming over finite domains to schedule evening
- Coordinates within map of Aberdeen used for location constraints



CONOISE

SICStus

UNIVERSITY OF ABERDEEN

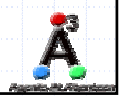
RDF to Prolog conversion

```
<ep:EveningPlan>
  <ep:events>
    <rdf:Seq>
      <rdf:_1>
          <ep:Event>
          <ep:duration>
            <c:Duration>
              <c:durationHour>2</c:durationHour>
              <c:durationMinute>0</c:durationMinute>
            </c:Duration>
          </ep:duration>
          <ep:place>
            <rdf:Alt>
              <rdf:li><pub:EnglishPub rdf:about="pubs#estaminet" /></rdf:li>
              <rdf:li><pub:EnglishPub rdf:about="pubs#wildboar" /></rdf:li>
              <rdf:li><pub:EnglishPub rdf:about="pubs#eastneuk" /></rdf:li>
            </rdf:Alt>
```

```
% data(<name>,<type>,<open>,<close>,<location>).
data('ugc_PianistThe',movie,2020,2248,8,4).
data('lighthouse_PianistThe',movie,1815,2043,7,4).

data('estaminet',pub,1000,2700,7,4).
data('wildboar',pub,1000,2400,6,4).
data('eastneuk',pub,1000,2400,7,5).
```

Example applications

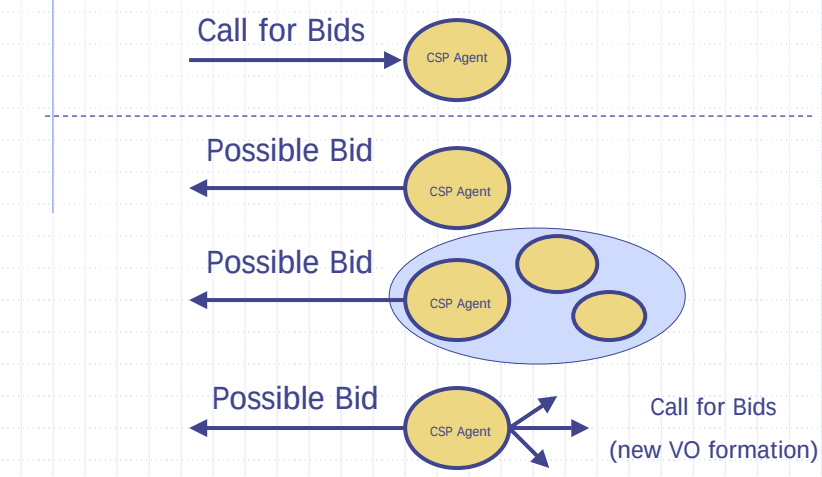
- Planning an evening's entertainment for a visitor to Aberdeen
- "Infotainment" service-provider coalition formation
- Supporting design teams



CONOISE project

- Constraint Oriented Negotiation in Open Information Services Environments
- Multi-site project: Aberdeen, Cardiff, Southampton, BT
- See www.conoise.org
- Automate & investigate Virtual Organisation life-cycle: formation, operation, and disbanding
- CONOISE@Aberdeen: using CLP to decide:
  - who to form VO with
  - when to reform VO
  - when to disband VO

CONOISE interactions

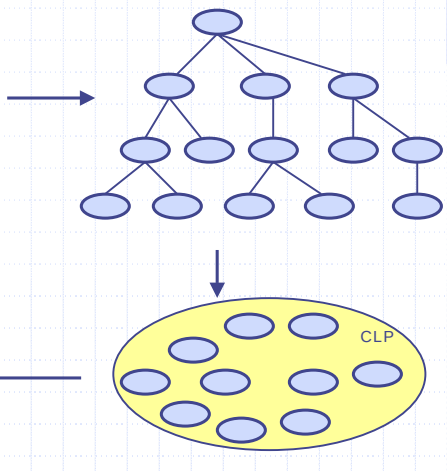


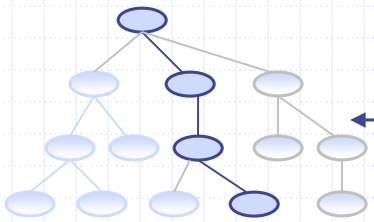
CONOISE: roles of constraints

- Types of constraint
  - user requirements / preferences (hard / soft)
    "monthly package including >= 50 text messages"
  - domain restrictions (axioms on ontology)
    "all Quicktime content requires a Quicktime player"
  - "small print" on instances
    "to get this price, must take the complete package"
  - suppliers' existing commitments
    "40% of my bandwidth is committed to customer X"
- All of these constraints must be factored-in when generating a bid
- Sample bid

UNIVERSITY OF ABERDEEN

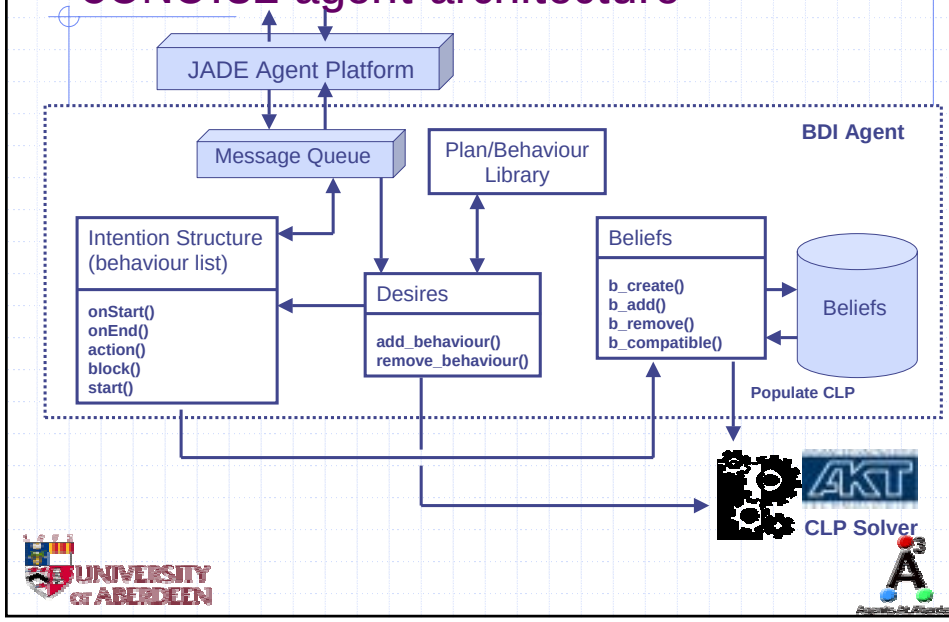
CONOISE: data model & CSPs





UNIVERSITY OF ABERDEEN

CONOISE agent architecture



Example applications

- Planning an evening's entertainment for a visitor to Aberdeen
- "Infotainment" service-provider coalition formation
- Supporting design teams

I-X/KRAFT service

- Allows people to collaborate on shared tasks
- "Technology integration experiment" (TIE):
  - Edinburgh's I-X provides process panels allowing users to identify & delegate tasks
  - Aberdeen's KRAFT offers constraint solving
- Scenario:
  - Edinburgh user identifies a technical issue, delegates it to Aberdeen
  - Aberdeen resolves issue through constraint solving
  - Example: configuring a PC to user's requirements

Interface (mock-up)

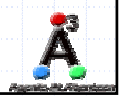
[Demo screencam]

Summary of contributions

- Issues
  - representing constraints in a Semantic Web-friendly format
  - interfacing constraint solvers to the open Web environment
- Example applications using CLP to deliver Semantic Web services
  - Planning an evening's entertainment for a visitor to Aberdeen
  - "Infotainment" service-provider coalition formation
  - Supporting design teams

Future work

- Technology
  - clean-up & simplify CIF
  - extend CIF to OWL
  - make solver public on Aberdeen Agentcity node
- Applications
  - CONOISE: VO reformation
  - E-science service composition
  - Constraints as laws; policing open service networks
  - AKT: team-to-team interactions



Credits & Questions?

- Work done at Aberdeen in collaboration with
  - Agentcities: Gunnar Grimnes, Pete Edwards
  - CONOISE: Stuart Chalmers, Tim Norman, Peter Gray
  - AKT: Kit Hui, Peter Gray, Derek Sleeman

Questions?