The PSM Librarian: Configuring Problem-solving Applications with Protégé

Monica Crubézy

Stanford Medical Informatics

~~ July 2003 ~~
Reasoning with knowledge bases

- Knowledge bases (KBs) encode reference models and facts in a domain
  - a set of clinical guidelines for hypertension care
  - a set of components and constraints about elevators
  - an anatomy ontology

- KBs further provide a basis for performing reasoning tasks—or problem solving
  - diagnosis, therapy advising, design, classification, ...
Component-based knowledge systems

- Alternative to rule-based inference systems
- **Separate** problem-solving component(s) encode(s) the reasoning process of the system
  - Reasoning behavior is explicit and understandable
  - Maintenance of system’s behavior facilitated
- **KB only** contains domain models and facts
  - Domain knowledge is explicit, understandable and maintainable too
  - Several problem-solving components can rely on the same corpus of domain knowledge
Problem-Solving Methods (PSMs)

- Standard, explicit algorithms that address stereotypical tasks
  - Design, classification, diagnosis

- Domain-independent components that abstract the reasoning process from factual knowledge
  - Reusable for different applications and domains
  - The *Propose-and-Revise* PSM: configuring elevator designs, predicting conformations of ribosomal units

- Collected and indexed in libraries for reuse
Model of a PSM

Explicit, structured representation of problem-solving knowledge
Describing libraries of PSMs

- Formal modeling & metadata annotation of PSMs in context
- The Unified Problem-solving Method development Language (UPML)
  - Task-Method decomposition paradigm
  - PSM: pragmatics, input-output roles, pre/postconditions, knowledge assumptions, subtasks & control
  - Ontology-based modeling of knowledge components
  - Domain/Task/PSM component-adaptation approach (bridges & refiners)
The UPML ontology
The UPML ontology
The Propose-and-Revise PSM

parameters, constraints, fixes

valid design

explicit algorithm of iterative constraint-satisfaction problem solving

P&R Ontology
UPML model of Propose-and-Revise

Pragmatics
title: Propose and Revise
resource: ChronBackPnR.clp

Ontology
element: parameter, as defined by class stateVariable
element: constraint, as defined by class Constraint and its subclasses
element: fix, defined by class Fix as “A condition-expression rule associated to a constraint and a parameter"
element: consistent, defined as a logical predicate

Input-roles: parameters, constraints, fixes
Output-roles: parameter values
Subtasks: Select next parameter, Propose next set of parameter values, Verify against constraints, Revise according to fix knowledge.

Competence
preconditions: “Every fix has exactly one associated constraint." ...
postconditions: “The output parameter values are consistent regarding the constraints." ...

Operational Description
...
Configuring PSMs for an application

- A problem-solving method (PSM)
  - processes domain knowledge & data in an abstract way
  - defines an ontology of its inputs and outputs in a domain-independent way—*input–output ontology*

- Domain knowledge (defined by a domain ontology) needs to be construed in terms of the PSM’s input–output ontology
Domain: “Data Group”

- filter out invalid events
- extract & reformat source, date, location
- abstract illness category
- drop uid

PSM: “Individual Event”
Data & knowledge exchange

- **Conceptual mapping**
  - change in domain of discourse
  - difference in the level of knowledge granularity
  - split and join of concepts & attributes

- **Value transformation**
  - abstraction, reduction
  - aggregation or dispatch
  - format change (unit change)
  - custom computation (functional transformation)
Ontology-mapping approach
Mapping relations

- Domain and PSM each define an ontology of its working concepts (with their attributes)

- A set of mapping relations expresses the connections between the 2 ontologies

- Mapping relations also express rules of transformation needed to mediate data & knowledge between domain and PSM
Each instance of the target class is calculated from an instance of the source class.

The slot values of the target instance are computed according to slot-mapping expressions that involve the source instance’s slot values.
1. Mappings ontology

- A small, generic & controlled set of possible mapping relations between classes and slots of a source (domain) ontology and of a target (PSM I/O) ontology

- For each instance required by the target component, a specific set of rules define the transformation of source instances and their attribute values
### Instance mappings

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Other Facets</th>
</tr>
</thead>
<tbody>
<tr>
<td>apply-to-subclass-instances?</td>
<td>Boolean</td>
<td>classes={target-class-description}</td>
</tr>
<tr>
<td>target-class</td>
<td>Instance</td>
<td>classes={target-class-description}</td>
</tr>
<tr>
<td>mapping-name</td>
<td>String</td>
<td>default={false}</td>
</tr>
<tr>
<td>on-demand</td>
<td>Boolean</td>
<td>default={false}</td>
</tr>
<tr>
<td>source-class-desc</td>
<td>Instance</td>
<td>classes={source-class-description}</td>
</tr>
<tr>
<td>condition</td>
<td>String</td>
<td>default={t}</td>
</tr>
<tr>
<td>reverse-mapping</td>
<td>Boolean</td>
<td>default={false}</td>
</tr>
<tr>
<td>per-instance-pre-execute-code</td>
<td>Instance</td>
<td>classes={executable-code}</td>
</tr>
<tr>
<td>per-instance-post-execute-code</td>
<td>Instance</td>
<td>classes={executable-code}</td>
</tr>
<tr>
<td>aux-source-classes-desc</td>
<td>Instance</td>
<td>classes={source-class-description}</td>
</tr>
<tr>
<td>slot-maps</td>
<td>Instance</td>
<td>classes={slot-mapping}</td>
</tr>
<tr>
<td>post-execute-code</td>
<td>Instance</td>
<td>classes={global-scope-code}</td>
</tr>
<tr>
<td>pre-execute-code</td>
<td>Instance</td>
<td>classes={global-scope-code}</td>
</tr>
</tbody>
</table>

- The class **S** of source instances
- The class **T** of target instances
- A condition to filter source instances
- A set of associated slot-level mappings
Slot mappings

- The target slot (tX)
- The slot-value computation expression, possibly involving source slots (si)
  
  local access to (sub)instance slot values: *<s1.s11>*

- Different types of slot mappings:
  - *renaming*: value(tA) = value(s1)
  - *constant*: value(tC) = constant
  - *lexical*: value(tB) = “*<s2>* / 20*<s3>”
  - *functional*: value(tC) = function()
  - *recursive*: value(tA) = instance (w/ auxiliary mapping)
Mapping Data Groups to Individual Events

<table>
<thead>
<tr>
<th>Source-class</th>
<th>V C + -</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataGroup</td>
<td></td>
</tr>
</tbody>
</table>

Apply mapping to subclasses of source-class?

<table>
<thead>
<tr>
<th>Target-class</th>
<th>V C + -</th>
</tr>
</thead>
<tbody>
<tr>
<td>IndividualEvent</td>
<td></td>
</tr>
</tbody>
</table>

Slot-maps

<table>
<thead>
<tr>
<th>Slot-map-name</th>
<th>V C + -</th>
</tr>
</thead>
<tbody>
<tr>
<td>to-validEvent</td>
<td></td>
</tr>
<tr>
<td>SF911_recordedLOINCData-to-illnessCategory</td>
<td></td>
</tr>
<tr>
<td>SF911_recordedSpatialData-to-location</td>
<td></td>
</tr>
<tr>
<td>SF911_recordedTemporalData-to-date</td>
<td></td>
</tr>
<tr>
<td>SF911_to-dataSource</td>
<td></td>
</tr>
</tbody>
</table>

Condition

```python
<LANG:Python>slotValueIsOfClass("originatingDataProvider","Emergency911CallCenter") and "+<originatingDataProvider.dataProviderNames*> "+"San Francisco 911 EMS Dispatch" and isValidSF911Record()
```

Reverse-mapping On-demand
2. Mapping interpreter

Domain "Source" Ontology

Mappings Ontology

PSM I/O "Target" Ontology

Source instances

Mapping Interpreter

Target instances
Results of mapping interpretation

Source “Data Group” instance

Resulting target “Individual Event” instance

Unique ID | Originating Data Provider
-----------|--------------------------
SMI-DATA-001 | San Francisco 911 EMS Dispatch

Data Group Specification

Recorded LOINC Data

- Type: StringDatum
  - Specification: Match Status
  - Contents: T

- Type: IntegerDatum
  - Specification: Match Score
  - Contents: 90

- Type: StringDatum
  - Specification: Incident Number
  - Contents: F990000002

- Type: StringDatum
  - Specification: Call Urgency
  - Contents: M

- Type: IntegerDatum
  - Specification: Main Call Type
  - Contents: 20

- Type: IntegerDatum
  - Specification: Call Type Modifier
  - Contents: 10

- Type: StringDatum
  - Specification: Call Disposition
  - Contents: HEA

Recorded Spatial Data

- Type: ZIPCodeArea
  - Specification: 911 Call Location (ZIP)

- Type: CensusBlockGroups
  - Specification: 911 Call Location (Block Group)

Recorded Temporal Data

- Type: Datetime
  - Specification: Date of 911 Call
The propose-and-revise example

Elevator domain

Ribosome domain

parameters, constraints, fixes valid design

P&R
Mapping ribosome range constraints to PnR constraints

<table>
<thead>
<tr>
<th>Mapping-name</th>
<th>Target-class</th>
</tr>
</thead>
<tbody>
<tr>
<td>constraint-lower</td>
<td>fix-constraint</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Slot-maps (4 values)</th>
<th>V</th>
<th>C</th>
<th>+</th>
<th>-</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Target-slot</th>
<th>V</th>
<th>C</th>
<th>+</th>
<th>-</th>
</tr>
</thead>
<tbody>
<tr>
<td>expression</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source-slot-composition</th>
</tr>
</thead>
<tbody>
<tr>
<td>(&gt; <em>&lt;lower-bound&gt;</em> (ribo-dist <em>&lt;obj1-xyz&gt;</em> ?<em>&lt;obj1-name.the-name&gt;</em>.location <em>&lt;obj2-xyz&gt;</em> ?<em>&lt;obj2-name.the-name&gt;</em>.location))</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source-slot</th>
</tr>
</thead>
<tbody>
<tr>
<td>V</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source-class</th>
</tr>
</thead>
<tbody>
<tr>
<td>constraint</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Aux-source-classes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>On-demand</th>
<th>Reverse-mapping</th>
</tr>
</thead>
</table>
Mapping ribosome range constraints to PnR constraints

<table>
<thead>
<tr>
<th>Label</th>
<th>Obj1-xyz</th>
<th>Obj2-xyz</th>
</tr>
</thead>
<tbody>
<tr>
<td>577-571</td>
<td>6.287</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-4.722</td>
<td>-2.607</td>
</tr>
<tr>
<td>Lower-bound</td>
<td>8.5</td>
<td>-8.876</td>
</tr>
<tr>
<td>Upper-bound</td>
<td>34.0</td>
<td>-27.439</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Obj1-name</th>
<th>Obj2-name</th>
</tr>
</thead>
<tbody>
<tr>
<td>H5</td>
<td>H8</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Violation-fix</th>
<th>Expression</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>The-name</th>
</tr>
</thead>
<tbody>
<tr>
<td>constraint-lower-577-571</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Condition</th>
<th>FixesList</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>fix-H5-for-H8^x2^</td>
</tr>
<tr>
<td></td>
<td>fix-H8-for-H5^x2^</td>
</tr>
</tbody>
</table>
The PSM Librarian tab

- **PSM selection support**
  - browsing & searching of UPML libraries
  - access to all elements of a PSM’s model

- **PSM configuration support**
  - integrated browsing of domain, PSM I/O & mappings ontologies
  - authoring of mapping relations
  - execution of the mapping interpreter
  - inspection of resulting instances (i.e., PSM inputs)
PSM selection support

Selected PSM: Propose & Revise
PSM input-output ontology
PSM configuration support
PSM configuration support (2)
Concluding remarks

• Benefits of explicit, structured mappings
  - Maximize independence & reuse of components
  - Minimize adaptation of components to work together
  - Isolate connection & transformation knowledge
  - More general than domain-to-PSM mapping paradigm
  - When both conceptual and syntactic transformation are needed

• Mapping is still hard
  - Generality and reuse of mappings defined by a target component as templates for other applications?
  - Complementarity with ontology-merging approaches? (PROMPT)
• PSM Librarian tab now online
• Mapping tools soon available standalone

http://protege.stanford.edu/plugins/crubezy@smi.stanford.edu