User Driven Modelling and Systematic Interaction for End-User Programming

Peter Hale, Tony Solomonides, Ian Beeson UWE, Bristol
Abstract

**Problem -**
Enable translation of human problems/representation to computer models and code.

To what extent can diagrammatic representations of problems be used in order to provide modelling solutions.

**Application Area -**
Engineering Modelling, for manufacturing processes and cost, so far applied to Aerospace Composite Wing Box cost, and Aircraft Engine Design and Cost.
Introduction

Purpose -
To test this problem -
• C.S. Peirce (1906) -
• 'Prolegomena to an Apology for Pragmaticism'
• "Come on, my Reader, and let us construct a diagram to illustrate the general course of thought; I mean a system of diagrammatization by means of which any course of thought can be represented with exactitude"

To limit the Scope –
• Research restricted mainly to engineers (who often use diagrams)
• To domain of modelling (which often requires diagrams)
Introduction Continued

Benefits -

- Enables engineers to visualise problems such as representation of a product data structure in a familiar way
- Gives a visual and colour coded representation of equations
- Visualisation is easier to navigate and understand than that in spreadsheets, and more maintainable

Wider Implications -

- This research could also be used for business modelling, process modelling, and scientific taxonomy visualisation
Research Approach

• This approach involves building a systematic infrastructure and capability, and solving problems which could hamper this.

• And is based on creation of systems that can be customised to produce other systems and models, and translation from abstract diagrammatic representations to computer representations.
# Semantic Collaboration

Naeve, 2005

## Table

<table>
<thead>
<tr>
<th>Domain Representation</th>
<th>Modelling/Programming</th>
<th>Abstract Representation</th>
<th>Structured Data File</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACUIty</td>
<td>Vanguard System</td>
<td></td>
<td>AJAX/Web2.0</td>
</tr>
<tr>
<td>Kaon</td>
<td></td>
<td></td>
<td>Increased Human Interaction</td>
</tr>
<tr>
<td>Metatomix M3t4</td>
<td>Semantic Collaboration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TopBraid Composer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jena</td>
<td>Protégé</td>
<td>AJAX/Web2.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Semantic Wikis</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSL, STEPml, PMXML, XML with domain schemas</td>
<td>AspectXML, AJAX/Web2.0, XQuery, XForms, SPARQL</td>
<td>DAML+OIL, OWL, RuleML, SWRL, MathML, RSS, SVG, VRML, UML, XMI</td>
<td>XML, Databases</td>
</tr>
</tbody>
</table>

## Diagram

- Increased Semantic Structuring and Collaboration
- Semantic Collaboration
- Semantic Coexistence
- Semantic Isolation
- Increased Human Interaction
Methodology

UDDM/P

Achieved by Providing

Applicability

- Maintenance
- Extensibility
- Ease of Use
- Sharing of Information

Enabled By

Knowledge Management

- Structure
- Implicit Knowledge
- Visualisation and Interaction

Enabled By

Visualisation and Modelling for Umbrella Activities

Producing

- Better and More Accessible Models

- Semantic Web and Ontologies
- End-User Programming
Translation Process

1. **Step 1**
   - Ontology Engine
   - Taxonomies

2. **Step 2**
   - Information Provided by Experts using visual editor
   - Taxonomy Translation
   - Inferencing
   - User Driven Model Building
   - Model Translation
   - Visualisation Translation

3. **Step 3**
   - Code Output
   - Web Output
Translation Stages

Periphery = 2*(Length*Web Height + (2*Flange Width) + (2*Tooling Flanges Height))
Ontology to Modelling Translation

Step 2 – Code written within Vanguard System translates from ontology to Vanguard System.

Recursive SQL queries build model trees from ontology.

Formulae written to Vanguard System, and calculation made.

Periphery = 2 * (Length + Web Height + (2 * Flange Width) + (2 * Tooling Flanges Height))
Web Tree Representation

Rhodes et al.
2002
Summary and Findings

• Closes the gap between those producing modelling systems, and those who require them

• Makes it easier to iterate through solutions and solve problems more quickly and collaboratively

• Experienced programmers can build a modelling environment that can then be used by non programmers to create process models

• Enables collaboration, simulation and modelling by translation from a model based representation of software to the actual software

• Gives users greater involvement

• Partially automates the process of software creation via a collaborative structure that maps the problem, and user interface creation by diagrammatic and/or tree based representation
Conclusion

• This approach to modelling and end-user programming enables interoperability, and collaboration

• This assists with Maintenance, Extensibility, Ease of Use, and Sharing of Information.
References

