

# Enabling Decision Support and Costing of Product Designs by using Visual Metaphors

Graduate School 2006

P Hale, R Marsh, C Bru, J Lanham

Aerospace Manufacturing Research Centre  
Faculty of Computing, Engineering & Mathematical Sciences  
University of the West of England  
Frenchay Campus  
Coldharbour Lane  
Bristol BS16 1QY

<http://www.cems.uwe.ac.uk/amrc/seeds/>

## **Abstract**

The Systems Engineering Estimation and Decision Support (SEEDS) team is part of the Aerospace Manufacturing Research Centre (AMRC) at the University of the West of England Bristol (UWE). SEEDS expertise is in working with aerospace organisations to apply techniques in managing, categorising and visualising information to support costing of products within the aerospace sector.

This paper outlines techniques used, in order to enable decision support during product development, whilst minimising dependence on specialist software and detailed programming effort. The basis of this is an Ontology that can be visualised and edited in tree form. The open standard Stanford University Ontology tool Protégé is being used for this purpose. This Ontology can be translated into a commercial Decision Support tool called DecisionPro. Software created using DecisionPro allows calculations of the cost of a design, and provides a colour-coded representation of the product tree. It is then possible to output this tree in the form of web pages, interactive diagrams and code in programming languages such as Engineous' Java based costing tool Cost Estimator. It is possible to search the information both in Protégé and on the Web as it is represented using searchable semantic web languages.

## **Introduction**

The SEEDS team has undertaken a number of projects in recent years with key regional aerospace companies. These projects created systems to facilitate management of design and cost related knowledge within those organisations, with the aim of using this knowledge to reduce the costs of manufacturing products. Arising from these projects, which have used a combination of proprietary software solutions and bespoke software developed for the projects, the SEEDS team have identified a number of new areas for work – specifically the approach of User Driven Programming (UDP). This research unites approaches of Object Orientation, the Semantic Web, and Relational Databases and event driven programming. The advantages of increasing user involvement in software development are explained by [Olsson].

When engineering organisations design and manufacture products they categorise this process into stages. From the early concept stage to the final design stage and manufacture, software is used to aid and record the process. It is common for different software to be used at different stages. If the software applications do not use open standards languages to communicate between these stages and between the various teams involved information gets lost and not re-used as the chain of information breaks. The detail and accuracy of the information that can be provided to define the product varies along this chain. The best opportunities for cost reduction are early in the product life cycle, so it is important to gather together any information that is available on that component and any similar products. It is important that users who enter information about a design concept be guided by historical values where possible and guidance information such as explanations, diagrams, and examples. The use of statistical modelling is necessary to ensure a cost can be calculated at this early stage when there is a high level of uncertainty. Validated information for products can be held in a catalogue for case based reasoning.

This paper is based on work undertaken to establish a way of representing information relating to the design and production of a wing box. The approach of developing decision support models for design and costing using a

spreadsheet is compared and contrasted with the alternative approach of using open standards ontologies and software. The paper begins with an explanation of the spreadsheet approach and explains the approach used since. The paper gives a critical evaluation of the spreadsheet the team created. It then outlines the alternative approach and justifies this as solving the modelling problem more effectively. This approach involves creating structured taxonomies that would eventually be part of an overall ontology of information relating to aerospace, and the automated generation of software to access and process this information. This approach could also be used for other types of modelling problem. Finally the paper explains how this alternative approach could be applied to a wide range of other problems.

### **Early Approach**

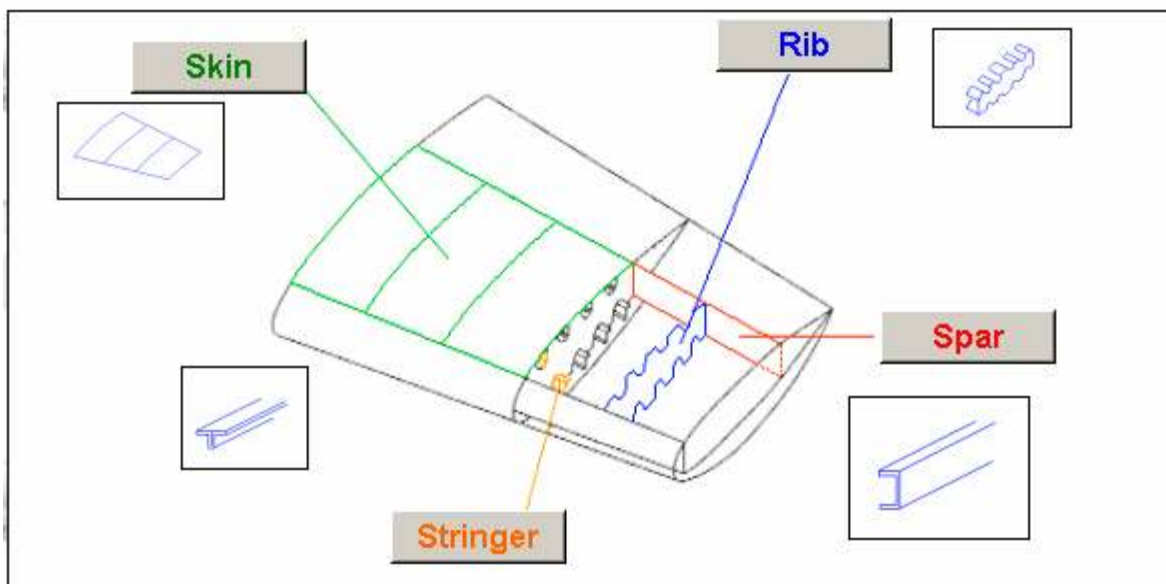
The example chosen illustrates a design that is difficult to cost because of a change in process i.e. use of composites rather than metal for the manufacture of a wing. In the early study of design options, parametric cost models are often used to statistically relate cost to factors such as weight and manufacturing process. Composites have different cost drivers than metals so this invalidates the use of parametric models based on metals in order to cost composite structures and products. The same is true for new processes such as superplastic forming and high speed machining, whose characteristics are significantly different from previously used processes. Therefore parametric models based on processes used already cannot be re-applied for the new processes, as no historical information is available.

The project was to create software for the purpose of costing a product where parametric costing was not viable. The customer specified that this should be a short project to allow the costing of manufacture of a composite wing box, and that a spreadsheet must be used for this due to the availability of this package. Costing of composites is an important area of research, as designers want to make use of the strength and weight properties of composites but need to be confident that the utilisation of such components is feasible and cost effective. This spreadsheet proved to be successful.

### **Spreadsheet Composite Wing Box Costing System**

The composite wing spreadsheet is a decision support tool for designers and manufacturers to evaluate the options for the design and manufacture of composite wing box components. It covers 4 main components, Skins, Spars, Ribs, Stringers and possible manufacturing techniques for each. The spreadsheet begins by providing the user with a page that enables the choice of component to cost. A diagram of a generic wing box and diagrams of generic components are provided to visualise the general shape and use of these components. Help pages are also provided at all stages of the costing. This start page is shown in Figure 1.

#### **Select wingbox component**



**Figure 1 - Wing box spreadsheet start page**

On choosing a component, in this case a spar, properties of the component are shown with default values. Colour coding is used to indicate to the user what values are editable. The derived values are recalculated to reflect any changes made by the user. When the user is satisfied with the definition of the component he or she can press the 'Define Processes' button to begin choosing and defining the manufacturing process(es) to be used, in a similar way to that used for choosing and defining the component. The interface is illustrated in Figure 2.

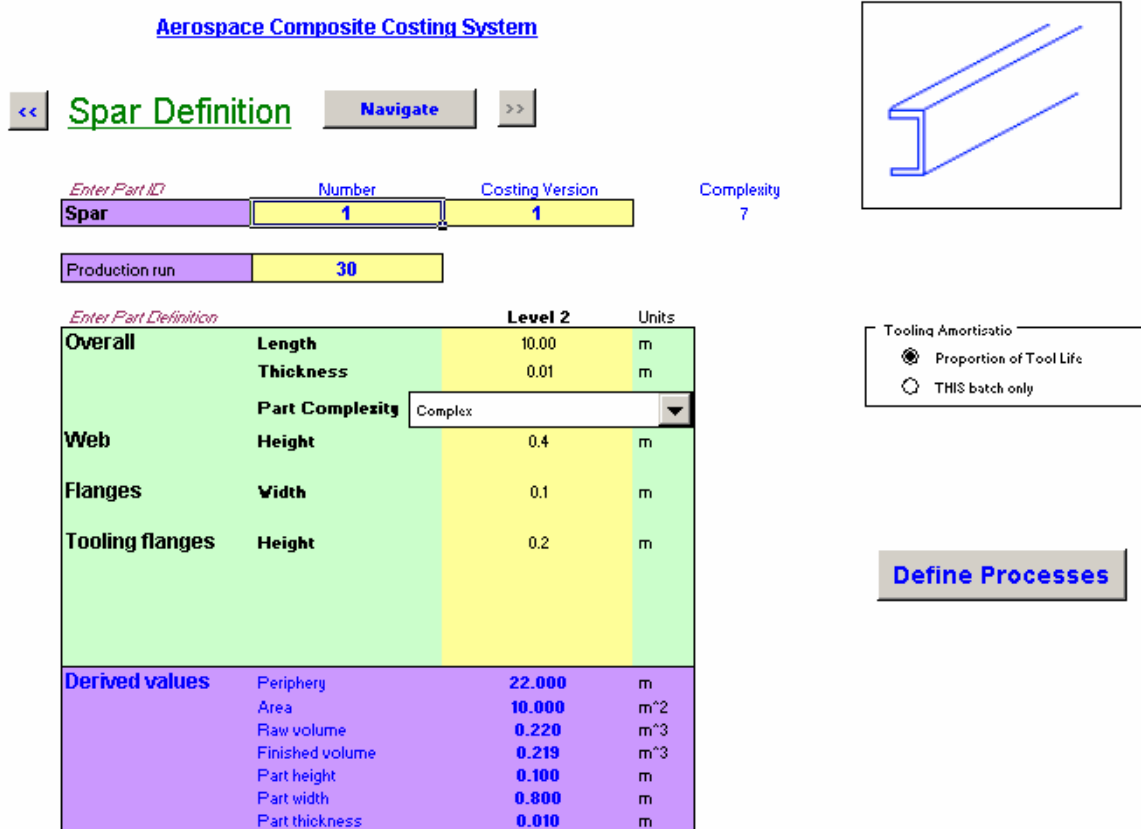


Figure 2 - Start of the costing of a Spar

Visual basic code was used in the spreadsheet to provide navigation and constrain the navigation order to ensure decisions were made in the correct order. However there were problems that this did not solve that will emerge in the longer term. These problems relate to maintenance, extensibility, ease of use, and sharing of information.

### Problems with this approach

#### Maintenance

Maintenance is a major problem. Although the software is popular and is still being used, there are several pitfalls that will eventually end its' usefulness. Firstly although colour coding is used to indicate which fields should be edited, people inevitably make the mistake of editing other cells. If for example a user overrides a formula with a value, future cost calculations will be incorrect. In order to prevent this, code was written that protects the cells which are not editable, if the user switches off this protection and tries to edit such a cell, this triggers the cell protection to be switched back on, and prevent this editing. In the system as a whole there is more code to deal with such situations than there is to provide navigation and calculation of cost. This means there are much more lines of code than a future maintainer might expect. Even so it is impossible to envisage and prevent everything a user might do that could corrupt the calculations as Excel provides a wide range of menus and options. Therefore it would probably have been easier to begin developing software from scratch and add facilities than to use those provided by Excel and attempt to prevent their accidental misuse.

In order to provide all the options for costing of the components and manufacturing processes the spreadsheet needed to be very large. There are many sheets and thousands of fields and formulae. The information in the spreadsheet is structured by means of grouping it on the appropriate sheet in table form, but this is a structure that is hard to maintain. To ensure a valid calculation it is important to audit the spreadsheet to make sure all default values and formulae are reasonable, and correctly labelled. This is a major task and if this task is not undertaken often errors will creep in. If the components' design or manufacture changes, major changes would be needed to the spreadsheet in response, and these would be time consuming and difficult.

### **Extensibility**

The maintenance problems explained above also impact on the extensibility of the spreadsheet. If this spreadsheet was to be extended or re-used to cost different components or processes, it would be very difficult to find all the relevant values and formulae to change. It would probably be easier to begin development of a new piece of software.

### **Ease of Use**

Feedback indicates the system is reasonably easy to use because the interaction required is limited to editing values and pressing buttons to go forward or back. Many people are familiar with this form of navigation from using the web toolbar. However the users know they are using a spreadsheet not a web page and try to use it in generic ways that were not intended for this software. When they are prevented from exploring the alternative forms of navigation such as scrolling around a sheet or clicking on sheet tabs they may become confused.

### **Sharing of Information**

It might become necessary to export the information from the spreadsheet to a process-planning tool for example. The lack of structure in the information makes it difficult to export it to other software systems. Because the information is a flat structure exporting it in a tree based W3C [World Wide Web Consortium] standard such as XML (eXtensible Markup Language) or RDF (Resource Description Framework) would entail the development of more customised software. There are similar problems in importing information from other systems. Knowledge sharing is essential for collaboration. [Merlo and Girard] explain the necessity for collaborative information systems for designers and the need for an object-oriented approach to this.

## ***Alternative Approach***

### **Structuring the Information**

Information is scattered within organisations and often not held in such a structured way as to be easily accessed by employees or software. This problem was examined by [Lau et al] using the example of McDonnell Douglas (now part of Boeing), that demonstrated how difficult it is to gather unstructured knowledge. Therefore, it is important that research is undertaken into methods of capturing, structuring, distributing, analysing, and visualising information.

The objective is to build a catalogue and make use of it for decision support and costing systems, while demonstrating that the same approach could be used for other types of system(s). It is essential that this catalogue can query information from organisations' existing database systems. Most large organisations have key operational knowledge and information dispersed across different types of information systems, often in Relational Databases. This has the advantage of allowing the use of the standardised language Structured Query Language (SQL) to access this information. In the current research approach the focus is on combining the development of dynamic software created in response to user actions, with object oriented, rule based and semantic web techniques. The Semantic Web was defined by [Tim Berners-Lee] as 'a web of data that can be processed directly or indirectly by machines'. SEEDS research has examined ways of structuring information, processing and searching this information to provide a modelling capability. The SEEDS team have investigated research by [Aziz et al.] that examines how open standards software can assist in an organisations collaborative product development, and [Wang et al] outlines an approach for integrating distributed relational database systems. The automated production of software containing recursive SQL queries enables this. This approach now used in SEEDS research is a type of very high level meta-programming. Meta-programming and structured language is explained by [Dmitriev] and [Mens et al]. The approach proposed is intended to solve the problems of cost and time over-run, and failure to achieve objectives that are the common malaise of software development projects. The creation of a web based visual representation of the information will allow people to examine and agree on information structures.

The alternative approach explained here aims to develop an integrated decision support system where maintenance and extension of information can be undertaken by model builder(s) via visual editing of library taxonomies. To facilitate this, information is visualised clearly in a structured understandable way, and this information can be shared via web-based visualisation. This can be achieved by use of open standard languages for distribution of information over networks and between applications.

## Categories of User

It is important to distinguish between the two different types of users for the system to be developed as they would work on different parts of the overall system.

### **Model Builders**

Model builders edit the semantic representation of the model in an ontology editor in order to create models. Model builders do not need knowledge of a programming language, but do need training in how to use the ontology interface to create a model.

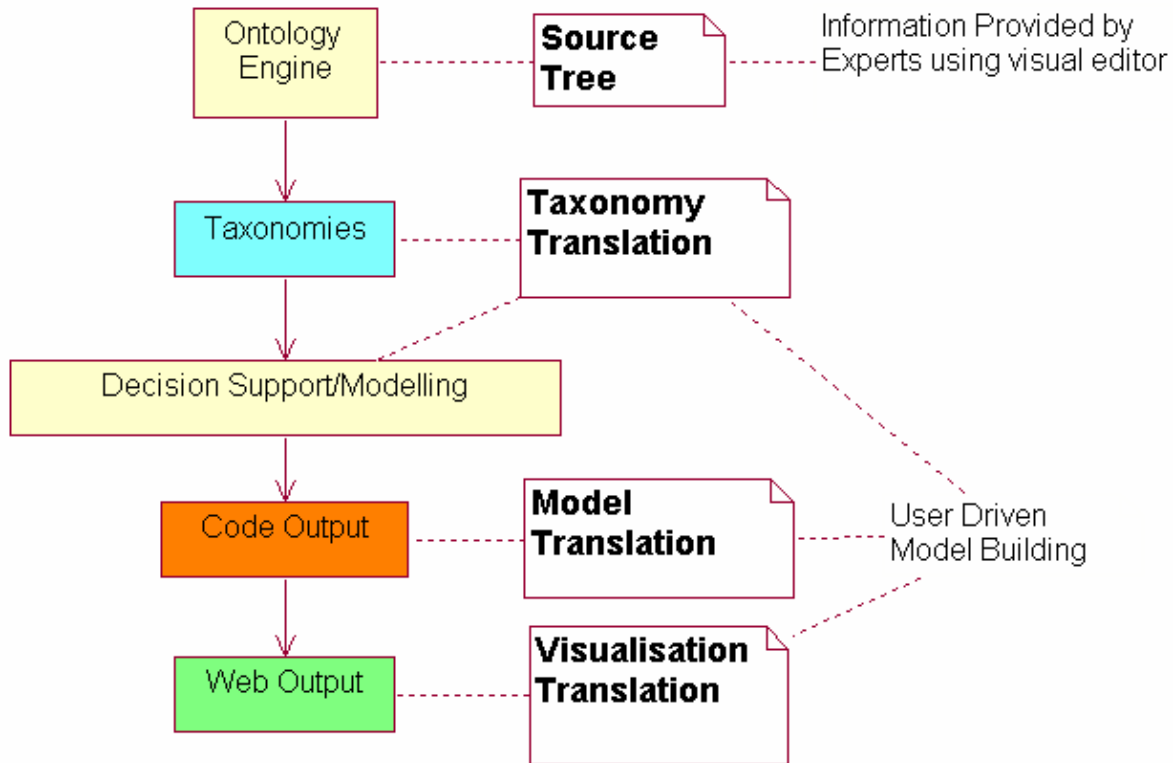
### **Model Users**

Model users make decisions based on their domain knowledge. This type of user manipulates the tree representation to obtain a result based on the input values they know, or otherwise based on default values. They will want to be able to use a model to evaluate a problem in order to help in decision making.

## Dynamic and Responsive Software

### **Translation Mechanism**

From the experience gained in this and other projects it is evident that the main challenge in software projects is to achieve a match between what is required from users of software and the functionality and capability of the software system provided. This section presents an approach involving the automation of the programming task, User Driven Programming (UDP). This provides computer literate users with a tool to enable them to drive the creation of software without the requirement that they become familiar with the syntax of a programming language. The system to be outlined is an attempt to help users who are experts at their own jobs and not computer professionals, to develop models that aid them in their work. This is a very common situation for designers and other engineers. The research and implementation is later demonstrated through the wing box example used in the spreadsheet. Figure 3 illustrates the approach to provision of a system to enable users to create and/or use their own models.



**Figure 3 - Translation Process**

This research involves adapting or creating software systems to provide the visual editor for the source tree, and model builders would create a model by editing this. By doing so they would create a generic model for a particular modelling subject. This is enabled by provision of translation software to translate the taxonomy into a decision support and modelling system. The model users could then use this decision support and modelling system to create their models. These models are a more specific subset of the generic model, and could be applied for their own analyses. Current research is on provision of a translation mechanism to convert information or models into other languages (primarily web based), and to visualise this information.

### **Research Theory influencing this Translation Mechanism**

In this work the focus is on combining the development of dynamic software created in response to user actions, with object oriented, rule based and semantic web techniques. This helps solve problems of mismatch of data between object oriented and relational database systems identified by [Ambler]. The information is highly structured. Visualisation of this structure in order to represent the relationship between things clarifies the semantics. The meaning can be seen not just by the name of each item but also by the relationship of other items to it. It is envisaged that this taxonomy will provide a design costing capability, but the taxonomy and the techniques used to put it together could be re-used for other purposes. Eventually this taxonomy could become part of an overall ontology. At first this would be a light-weight ontology and this could be evaluated for usefulness before deciding on whether it would need to be more structured. [Hunter] evaluates engineering ontologies and gives examples. Issues involved in visualisation of light weight ontologies are examined by [Fluit et al.]. An important reason for creation of an open standards central ontology is that it can be accessed by many different applications. The open standard OWL (Web Ontology Language) is explained by [Bechhofer and Carroll]. Research of others in this field have been investigated [Corcho], [Corcho and Gómez-Pérez] and [Noy].

The current approach builds on previous work undertaken for a large aerospace company to allow designers and manufacturers to visualise and share cost information. During this project one task was to automatically produce tree representations of information requested by the user. Information held in a relational database was visualised and exported in structured languages. The information was visualised in decision support software called DecisionPro [Vanguard]. This enables modelling of product design and manufacture, and provides statistical techniques for modelling uncertainty. Information was visualised in a colour-coded tree. This information could also be output as XML (eXtensible Markup Language) and SVG (Scalable Vector Graphics), and linked to stylesheets to create a web

based tree representation. Parametric cost models have also been created online (Figure 14) and a tree based menu for browsing of these or other web pages (Figure 17).

Dynamic software systems such as outlined in [Huhns] have been examined. Huhns explained that current techniques are inadequate, and outlines a technique called Interaction-Oriented Software Development, concluding that there should be a direct association between users and software, so that they can create programs, in the same way as web pages are created today. [Paternò] outlines research that identifies abstraction levels for a software system. These levels are task and object model, abstract user interface, concrete user interface, and final user interface. Stages take development through to a user interface that consists of interaction objects. This approach can be used for automating the design of the user interface and the production of the underlying software. Paternò states that 'One fundamental challenge for the coming years is to develop environments that allow people without a particular background in programming to develop their own applications'. Paternò goes on to explain that 'Natural development implies that people should be able to work through familiar and immediately understandable representations that allow them to easily express relevant concepts'.

The essence of the approach now used is that a high level visual interface is used to create a meta-program, which can communicate the wishes of users. Figure 3 illustrates the theory behind this and Figure 5 the implementation. This requires the definition of taxonomies to provide the library of information to be used in the decision support modelling. A model builder could populate and maintain this without needing to know programming languages. It would then enable the possibility of designers influencing the construction of a decision support model dynamically. Once this is possible the model is responsive, instead of commissioning for software development, the designer can communicate with a visual interface, and this translates to controlling code, which writes temporary code to achieve what is required. A program is expressed in terms of a tree diagram that represents each domain of information required, and this diagram holds structured language definitions. Separation of the content of the information from any constraints of language and format enables this. This approach is similar to the way spreadsheets allow structured formula to be entered using a formula wizard. However, spreadsheets allow for construction of models that are hard to maintain and adjust as the scenario which they model changes. This is because of the difficulty that either humans or computer applications have in accessing information when the information is held in cells rather than nodes of a taxonomy. [Schrage] explains how difficult it can be to find the underlying assumptions that are represented in a spreadsheet scenario. The difficulty of tracking the information and assumptions in a spreadsheet make it harder to integrate the model with other software applications. However, the large numbers of domain experts who undertake spreadsheet development indicates their desire to be creators of software models.

The alternative approach involves creation of an elaborator that can output code, in various computer languages or a meta-programming syntax such as metaL [Lemos]. The elaborator needs only a few pieces of information. All information other than that dependant on user interaction, including the names of each node and its' relationships to other nodes, needs to be held in a standardised data structure, e.g. a database or structured text file(s). A visual interface to this ontology is required so that a model builder can maintain and extend it.

Each node (elaborator) needs to be provided with the following pieces of information -

- 1)** A trigger sent as a result of user action. This is a variable containing a list of value(s) dependant on decisions or requests made by the user the last time the user took action. Each time the user makes a request or a decision, this causes the production of a tree or branch to represent this. This trigger variable is passed around the tree or branch as it is created. The interface to enable this is connected to and reads from the ontology.
- 2)** Knowledge of the relationship between this node and its' immediate siblings e.g. parents, children, attributes. So the elaborator knows which other elaborators to send information to, or receive from.
- 3)** Ability to read equations. These would be mathematical descriptions of a calculation that contains terms that are items in the ontology. The equation would be contained within an attribute of a class, e.g. The class Material Cost would have an attribute Material Cost Calculation that holds an equation.
- 4)** Basic rules of syntax for the language of the code to be output.

The way the elaborator finds the information held in **2** and **3** is dependant on the action taken in **1**. Thus, if a suitable ontology is created the basis of the rules of construction of the code to be created are defined **4**, and the user has made choices, the user needs to take no further action and just wait for the necessary code to be output.

## Implementation

The example used to illustrate the new approach uses information taken from the composite wing box spreadsheet. An open standard semantic editor Protégé created by [Stanford University] was used to structure this information into related taxonomies. This holds the definitions of nodes representing information, and calculations to be performed. This information is saved using a generic structure based on keys that define all relationships, into a relational database. This enables storage of hierarchical data in a relational database and also allows for separation of information into tables according to category and use SQL (Structured Query Language) to query and structure the information as required. Vanguard's tree based decision support tool DecisionPro reads this information and represents it as colour-coded nodes. The decision support tool can perform calculations and so output results. Figure 4 shows how the decision support tool can represent a branch in the tree, visualise an equation and calculate a result. Red nodes represent processes, green nodes represent the part definition and magenta nodes represent resources.

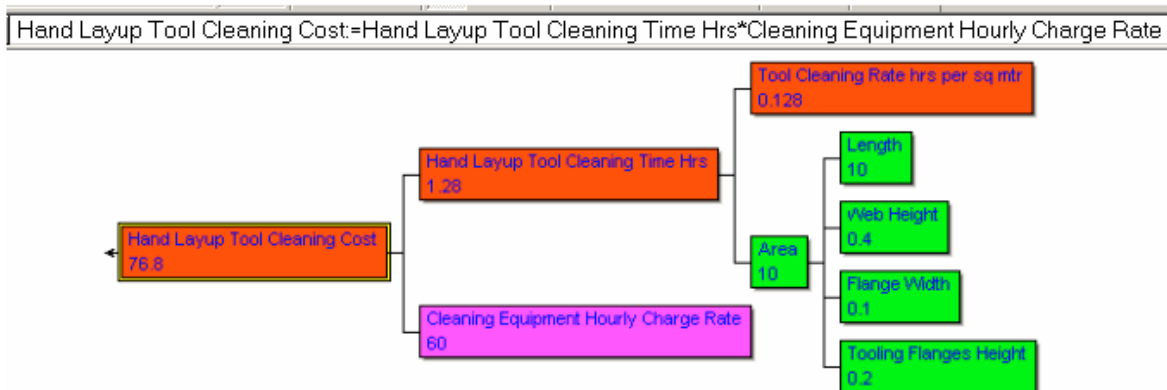
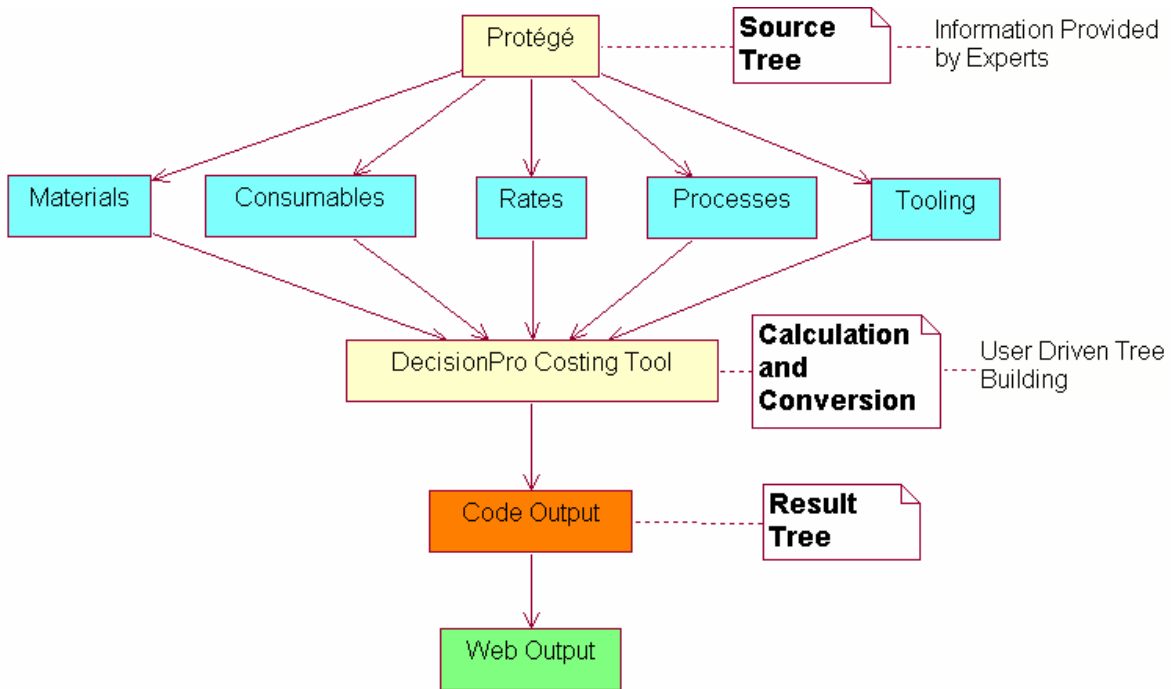


Figure 4 - DecisionPro calculation

## Translation Process

The current research uses a technique of interpreting information in order to create decision support programs automatically in response to user choices. This technique is then extended for use in the automatic creation of programs in other computer languages and systems. This can be achieved by automated translation of the DecisionPro information into other languages. The basis of this is that elaborators are nodes in the tree, which are automatically created and dynamically write objects. This allows the wing box definition to be translated to the decision support system for costing and then to other software such as web pages for further processing or visualisation. Taxonomies are created in Protégé for Parts, Materials, Consumables, Processes, Rates, and Tooling for a prototype costing system. New categories can be produced as required. Domain experts would edit the taxonomies; these experts can specify the relationships of classes and the equations to be used via a visual user interface in Protégé. These relationships are evaluated and translated to produce computer code. Figure 5 illustrates how code is produced from the semantic relationships.

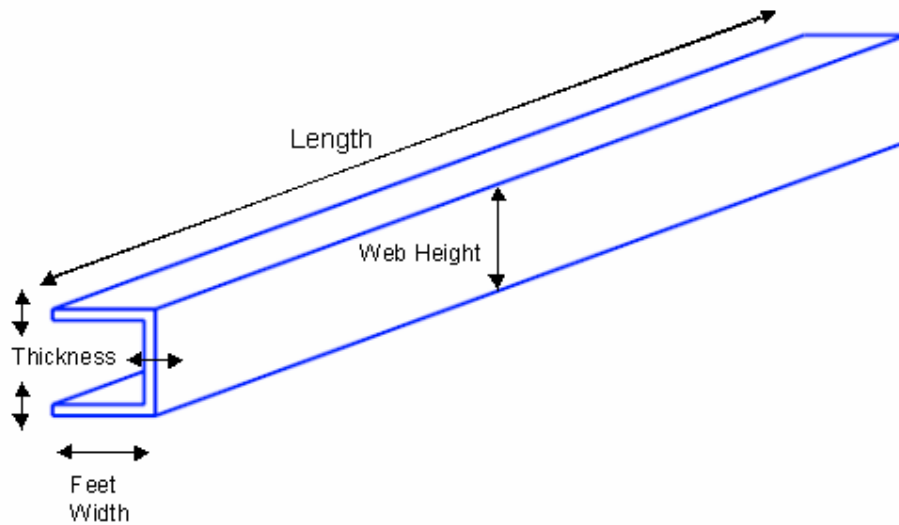


**Figure 5 - Translation Process Implementation**

For the prototype to be extended and applied for external use each taxonomy would be filled with a structured tree representation of experts' knowledge in the form of classes, values and equations. A costing tree can be automatically produced from these taxonomies. Equations created by the expert, together with choices made by the user of the decision support software, would determine how these taxonomies are linked for a particular costing. The costing tool user would then determine which costing equations are used, by answering questions on dialogue forms. These questions would be asked whenever multiple solutions were available. The benefit of this approach is that the user interface and calculations will be changed automatically to reflect any changes in the model. So if the problem to be modelled changes, only the information that defines the model needs updating, and not the user interface or calculation engine.

### Implementation Example – Spar Hand Lay-Up Process

Figure 6 shows a simple diagram of a spar.



**Figure 6 - Spar Diagram**

Figure 7 shows the decision support systems tree view of the spar branch from the wing box cost model and contains information queried from the related taxonomies. The tree including all the default part definition information for the spar is produced automatically. The buttons in the tree enable choices to be made by the user about materials, consumables, rates and processes. Branches are created in response to these choices. The values in the branch nodes can then be changed as required. Figure 6 shows this.

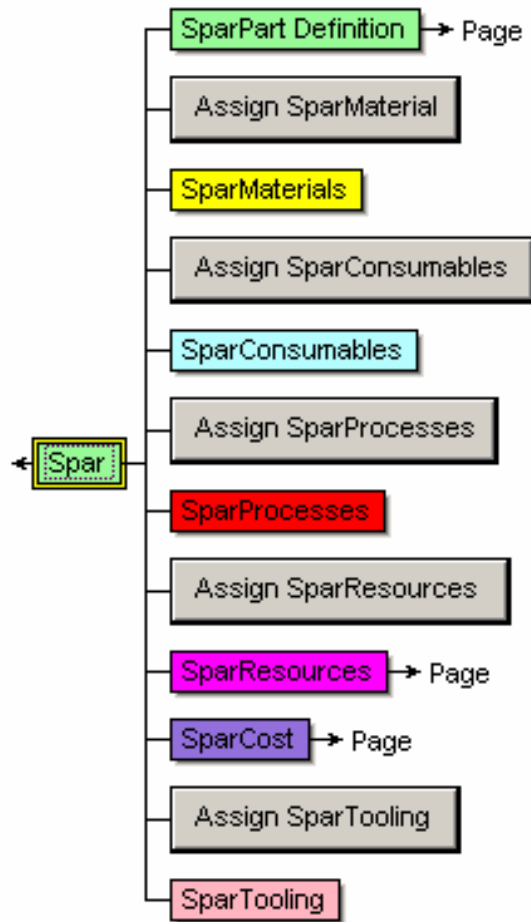
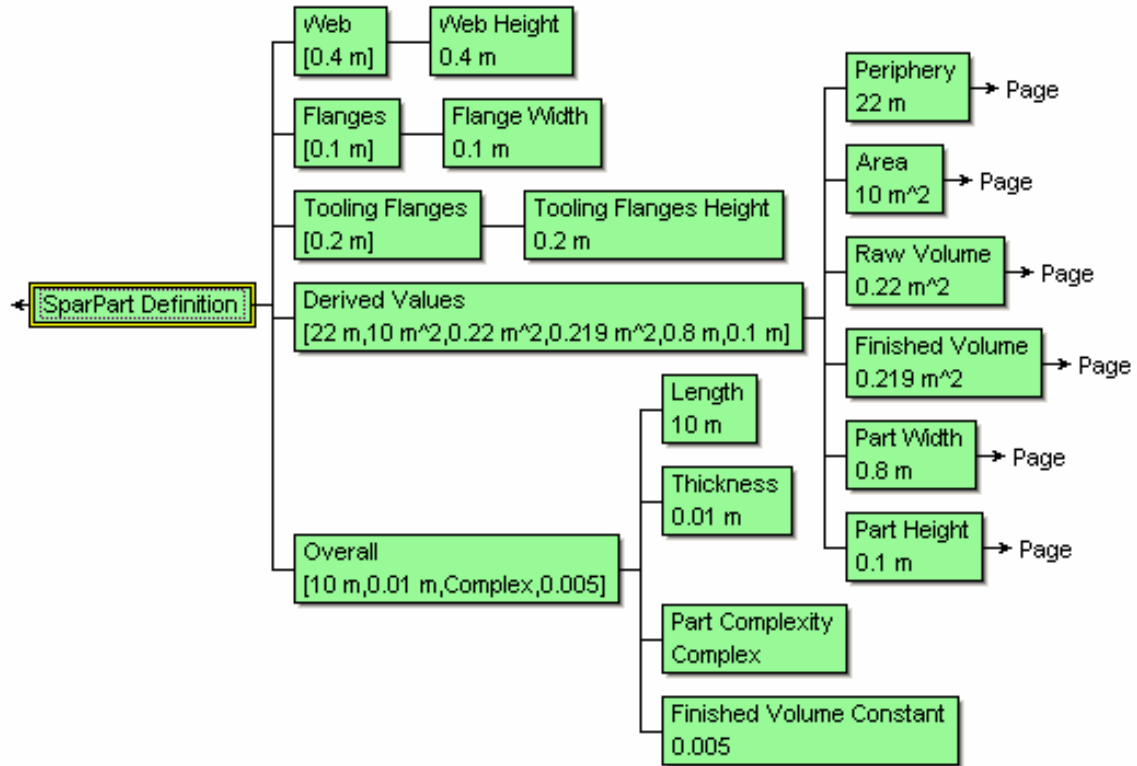


Figure 7 - Spar branch automatically created from information source

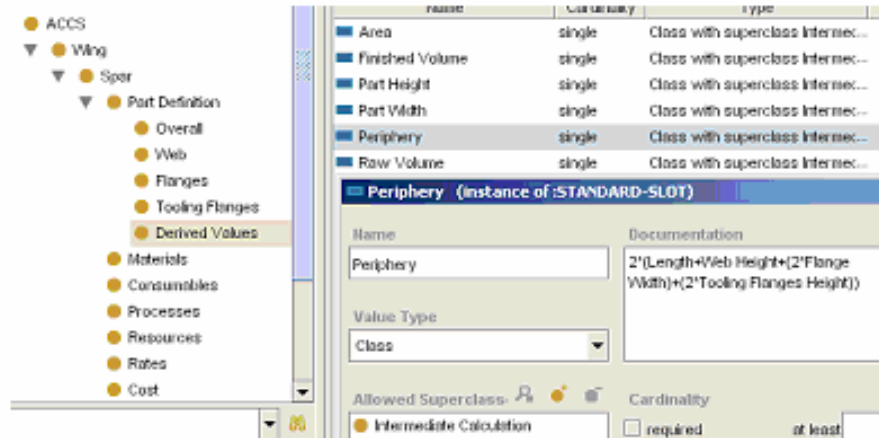
The user will make choices so the decision support tree will be a subset of the information source tree.

DecisionPro visualises large trees by breaking them into individual pages, and indicating with a right arrow where there are further pages that can be viewed. Clicking the 'Part Definition' right arrow will display the corresponding information as illustrated in Figure 8. The 'Derived Values' branch contains parameters of the spar that are calculated from the spar dimensions.



**Figure 8 - Part Definition Branch**

Figure 8 is a DecisionPro reproduction of the part definition from the Protégé taxonomy. The DecisionPro software adds an extra functionality, which is to calculate, and store the results of equations captured in the Protégé taxonomy. The equation is represented as text in the Documentation field of the Periphery attribute of Derived Values as seen in Figure 9.



$$\text{Periphery} = 2 * (\text{Length} + \text{Web Height} + (2 * \text{Flange Width}) + (2 * \text{Tooling Flanges Height}))$$

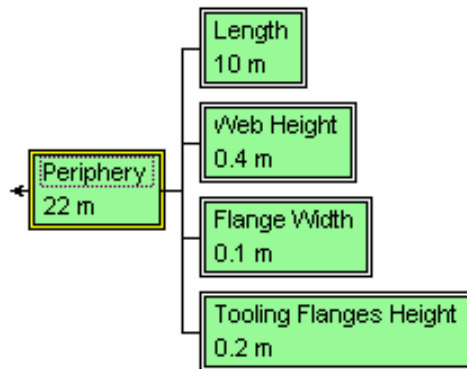


Figure 9 - Spar Periphery Calculation

Different types of information indicated by colour coding may be combined in a calculation. This is illustrated in Figure 10.

$$\text{Pre Preg Mass} = (1 + (\text{Pre Preg Waste Factor Percent} / 100)) * (\text{Pre Preg Density Kg per m}^3 * \text{Raw Volume})$$

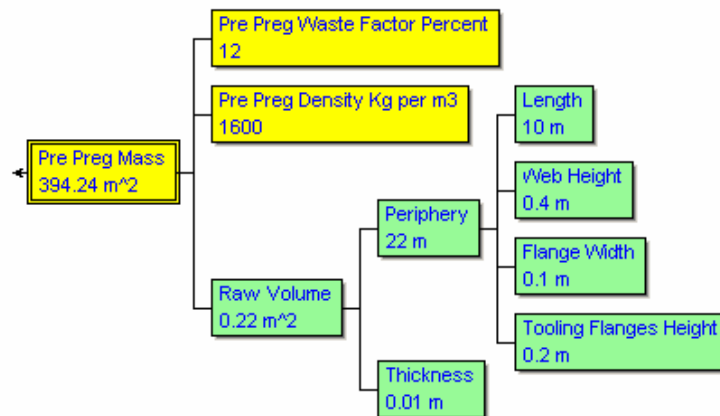
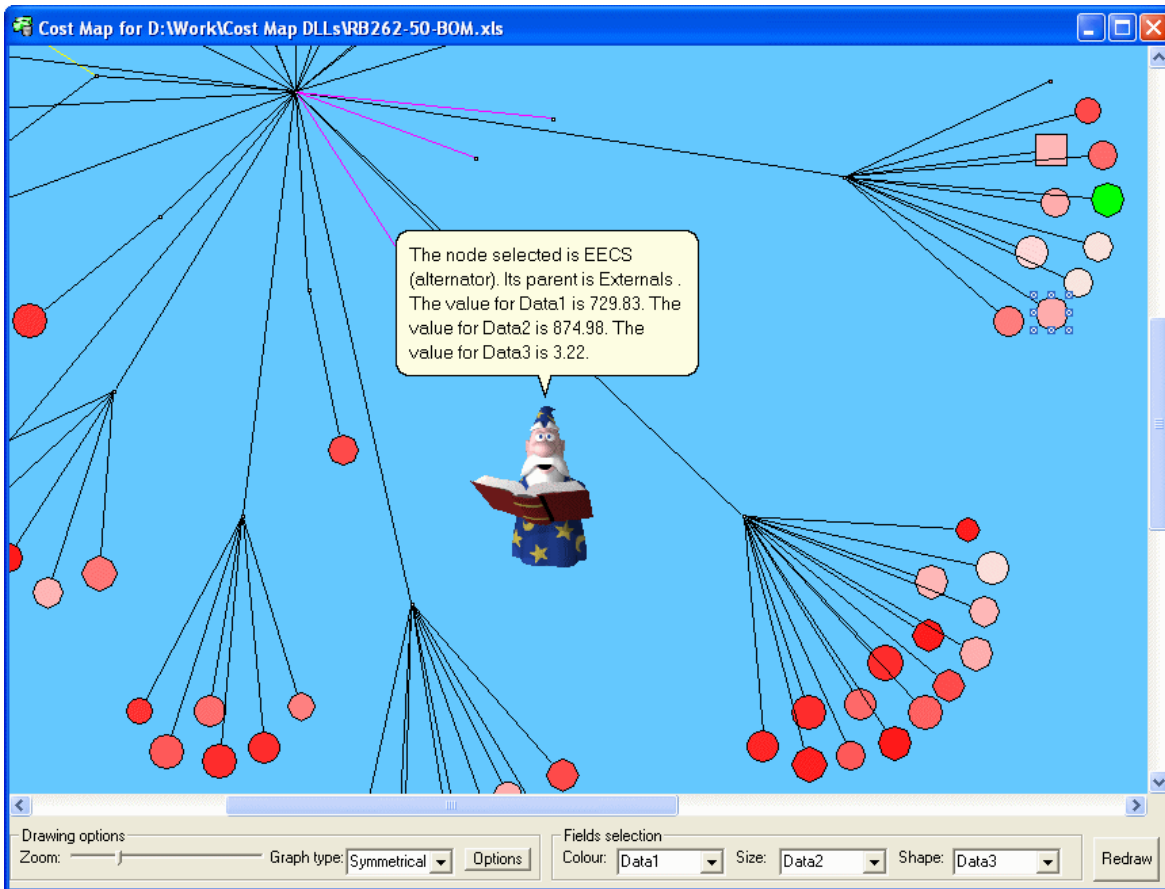


Figure 10 - Pre Preg Mass Calculation





**Figure 13 - Cost Map Visual Basic Version with Speech Recognition**

The Visual Basic application is not only interactive via keyboard and mouse events, but also via multimedia means. Whenever information can be displayed, it can be read aloud (such is the case in Figure 13 for instance). Moreover, the package can hear spoken commands. These facilities improve the package accessibility for people having difficulty reading from the screen and/or interacting with the standard input devices (i.e. keyboard and mouse). Prior hypertextual data mining facilities, consisting of both manual and query based filters, allow the user to ignore or concentrate on particular areas of the data structure (most likely, but not limited to, a tree). Other graphical representations are available for both filtered data (in the form of graphs) and the whole dataset (in the form of matrices). Therefore, cost analysis can be carried out globally or locally; in the former case, all information is represented relatively to the whole dataset.

## Web Output

Creation of web pages for a previous aerospace customer led to the view that this could be a popular method of delivery for information and models. Web Output is important as an alternative to spreadsheets because of the problems with spreadsheets explained earlier, and because they provide a multi-user standards based freely accessible method for conveying information and models. Current research into providing web-based models is partly based on previous work completed for providing parametric models online. Figure 14 shows an example of a parametric model.

University of the West of England  
BRISTOL

Faculty of  
Computing, Engineering  
and Mathematical Sciences

Online Parametric Cost Models Examples

Model name: Generic turbofan cost model  
Model author: Christophe Bru  
Author's email: [Christophe2.Bru@uwe.ac.uk](mailto:Christophe2.Bru@uwe.ac.uk)  
Model date: 24/01/2005

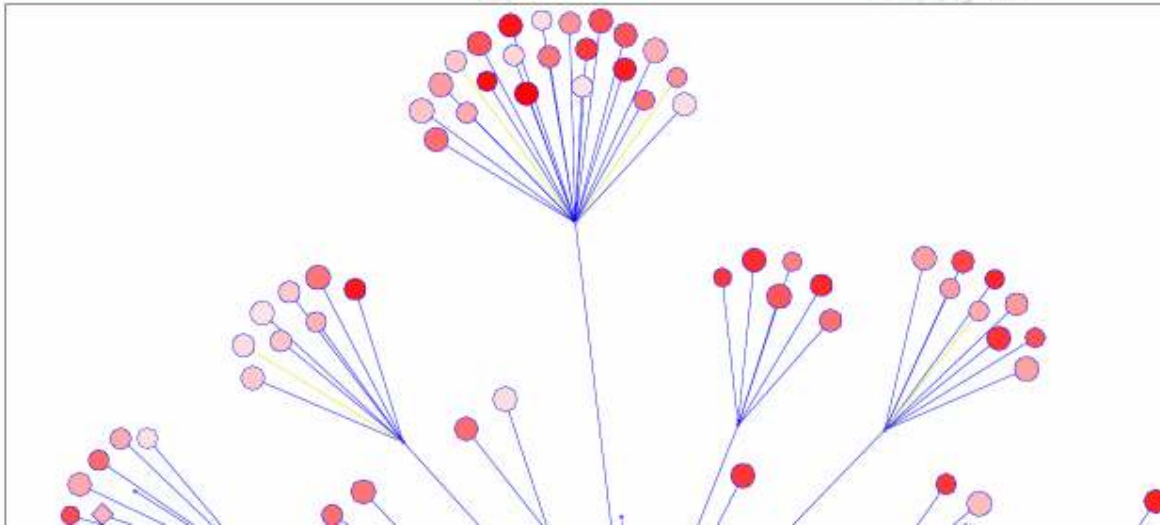
Parameter name	Estimation 1	Estimation 2	Estimation 3	Help	Range & unit	Relative Importance
Thrust	<input type="text"/>	<input type="text"/>	<input type="text"/>	Help	10000 < x < 150000 lb	50%
Weight	<input type="text"/>	<input type="text"/>	<input type="text"/>	Help	300 < x < 10000 kg	30%
TSFC	<input type="text"/>	<input type="text"/>	<input type="text"/>	Help	0.5 < x < 3 lb/hr/lb	20%
Bypass Ratio	<input type="text"/>	<input type="text"/>	<input type="text"/>	Help	6 < x < 14 n/a	20%
Price (£):	<input type="text"/>	<input type="text"/>	<input type="text"/>		Compute Print page	

**Figure 14 - Parametric Cost Estimation**

Parametric Cost Estimation samples can be found at <http://www.cems.uwe.ac.uk/~cbu/>.

Further research involved studying the methods used and success of others that had used this approach. This research is outlined in [Ciancarini et al.]; [Huang and Mak]; [Kim et al.]; [Morris et al.]; [Nidamarthi et al.]; [Reed et al.]; [Zhang et al.]. [Li] outlines how a Web-based solution can be applied to distributed process planning. The above research reinforced the view that this is a sensible research approach.

It is possible to output text files so decision trees can be translated into other programming, meta-programming or structured languages. This enables provision of a visual web interface to the models without having to be locked in to proprietary solutions, and ensures the maximum amount of access for users. Figure 15 shows the SVG version of the cost map output in this way.



**Figure 15 - SVG Cost Map**

Cost map samples can be found at <http://www.cems.uwe.ac.uk/~cbru/>.

The production of part diagrams using SVG can be automated in a similar manner to that used for the automated production of DecisionPro costing models. Figure 16 shows an example of such an interactive visualisation of a Spar. This interactive diagram was output from the part definition described in the part taxonomy.

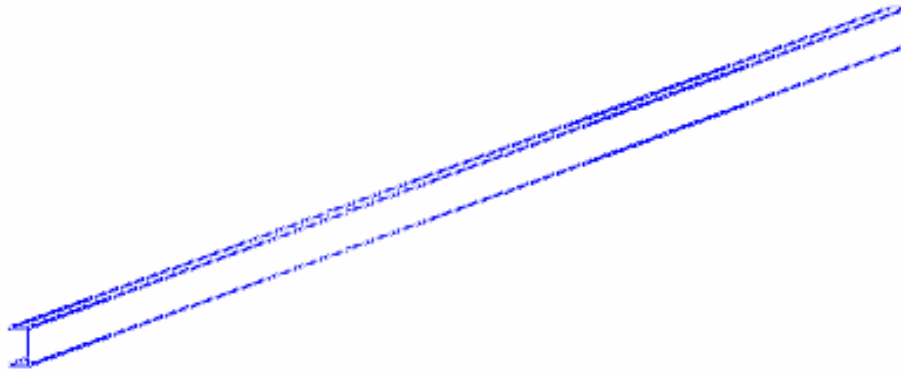


Figure 16 - Interactive Spar Diagram (SVG)

WebHeight (m)	0.4	Periphery (m)	32	$Periphery = 2 * (Length + WebHeight + (2 * FlangeWidth) + (2 * ToolingFlangeHeight))$	<input type="button" value="Zoom In"/> <input type="button" value="Zoom Out"/> <input type="button" value="Up"/> <input type="button" value="Down"/> <input type="button" value="Left"/> <input type="button" value="Right"/>
FlangeWidth (m)	0.1	Area (m <sup>2</sup> )	10	$Area = Length * (WebHeight + 2 * (FlangeWidth + ToolingFlangeHeight))$	
ToolingFlangeHeight (m)	0.2	RawVolume (m <sup>3</sup> )	0.22	$RawVolume = Periphery * Thickness$	
Length (m)	10	FinalVolume (m <sup>3</sup> )	0.218	$FinalVolume = RawVolume - (2 * Length * Thickness * FlangeVolumeConstant)$	
Thickness (m)	0.01	PartWidth (m)	0.8	$PartWidth = WebHeight + (2 * ToolingFlangeHeight)$	
PartComplexity	Complex	PartHeight (m)	0.1	$PartHeight = FlangeWidth$	
FlangeVolumeConstant	0.005				

Figure 16 - Interactive Spar Diagram (SVG)

The way this interacts with the user can be seen by viewing these examples on the Interactive SVG links page at <http://www.cems.uwe.ac.uk/~phale/InteractiveSVGExamples.htm>. These provide examples of wing box components. The Internet Explorer examples require an SVG plug in, which can be downloaded for free. The Mozilla Firefox examples are produced using a native XML implementation of SVG and so do not need a plug in. The Firefox examples are rendered in a build of the browser which integrates an SVG renderer, so SVG contents appear as an integral part of the document instead of an embedded object. Both sets of examples are produced automatically using an intermediate tree that draws the component outlines dynamically from the design parameter values. Once defined, a component or a feature could be held in a library and re-used. SVG developers should introduce this automated construction and storage of SVG diagrams, and so make this available to those who do not have access to CAD software. This could allow 3D modelling and collaboration over the web as envisaged by [Park and Fishwick].

Figure 17 shows the DecisionPro tree translated into XML and displayed as a web tree using a stylesheet. The menu uses a stylesheet created by Emmanuele De Andreis [De Andreis].

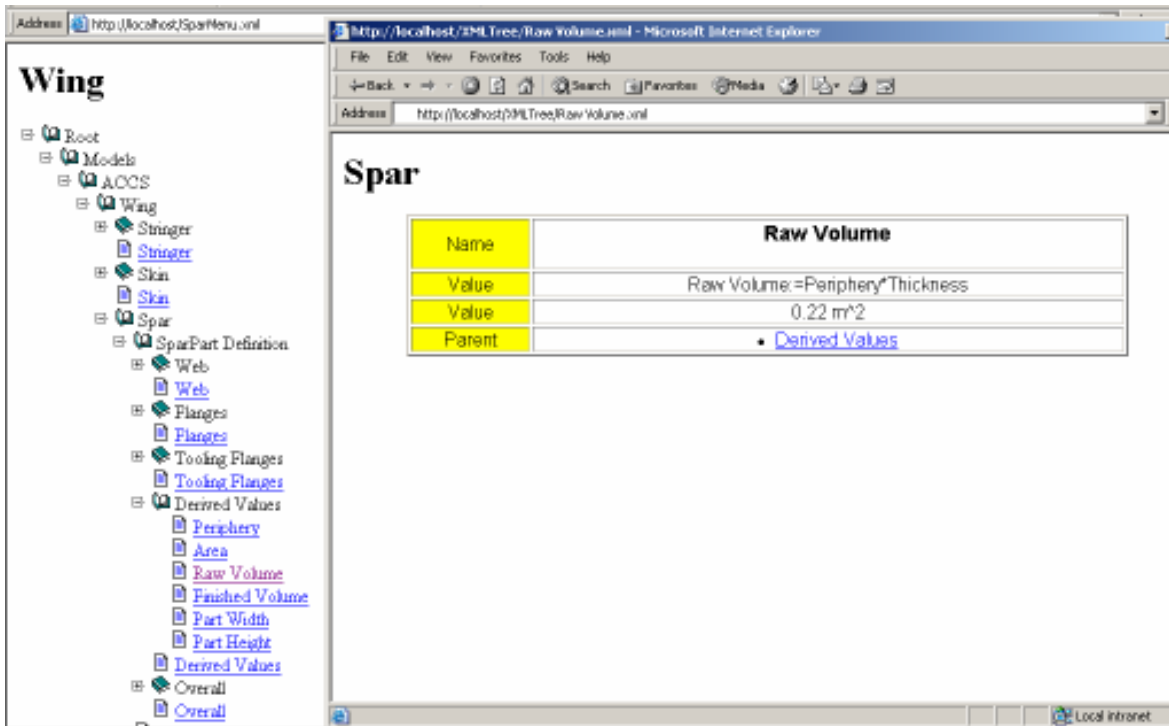


Figure 17 - XML web translation from Decision tree

This stylesheet is used for the site map on the web site at <http://www.cems.uwe.ac.uk/~phale/SiteMap.html>.

The XML can also be displayed on the web using a Flash program created by [Rhodes et al.]. This creates a tree with a three dimensional look and a use of colour, shading, and movement of the nodes that makes it an intuitive user interface that is easy to navigate. When a node is chosen, this is moved to the centre of the display and all the other nodes are moved or rotated to position themselves in relation to it. This interface is illustrated in Figure 18.

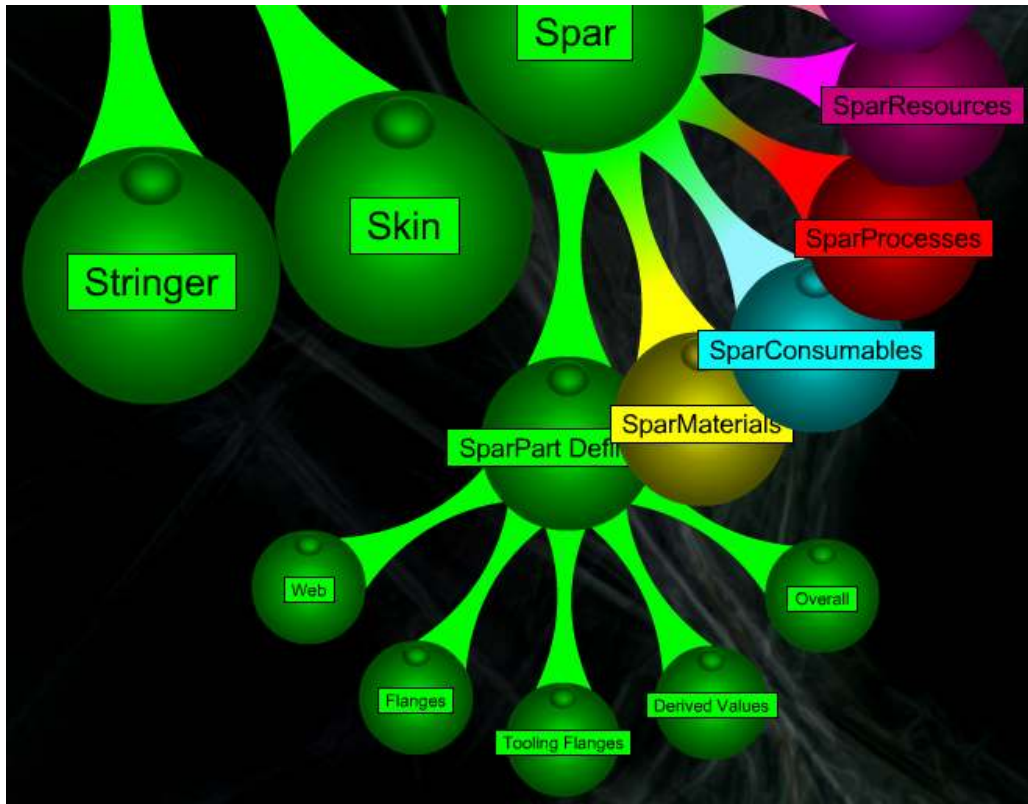


Figure 18 - Flash interface for navigating exported XML tree

Figure 19 shows the view resulting from clicking on the Spar Part Definition. This shows the parents, children, siblings, and contents of that node. It also allows navigation to any of the related nodes.

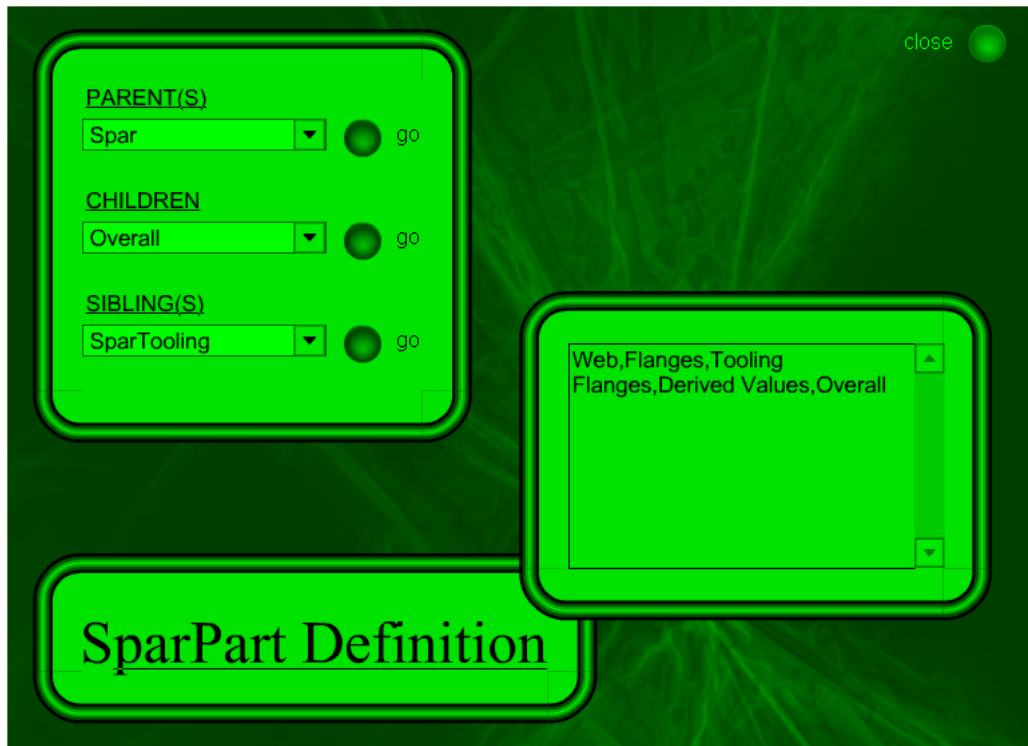
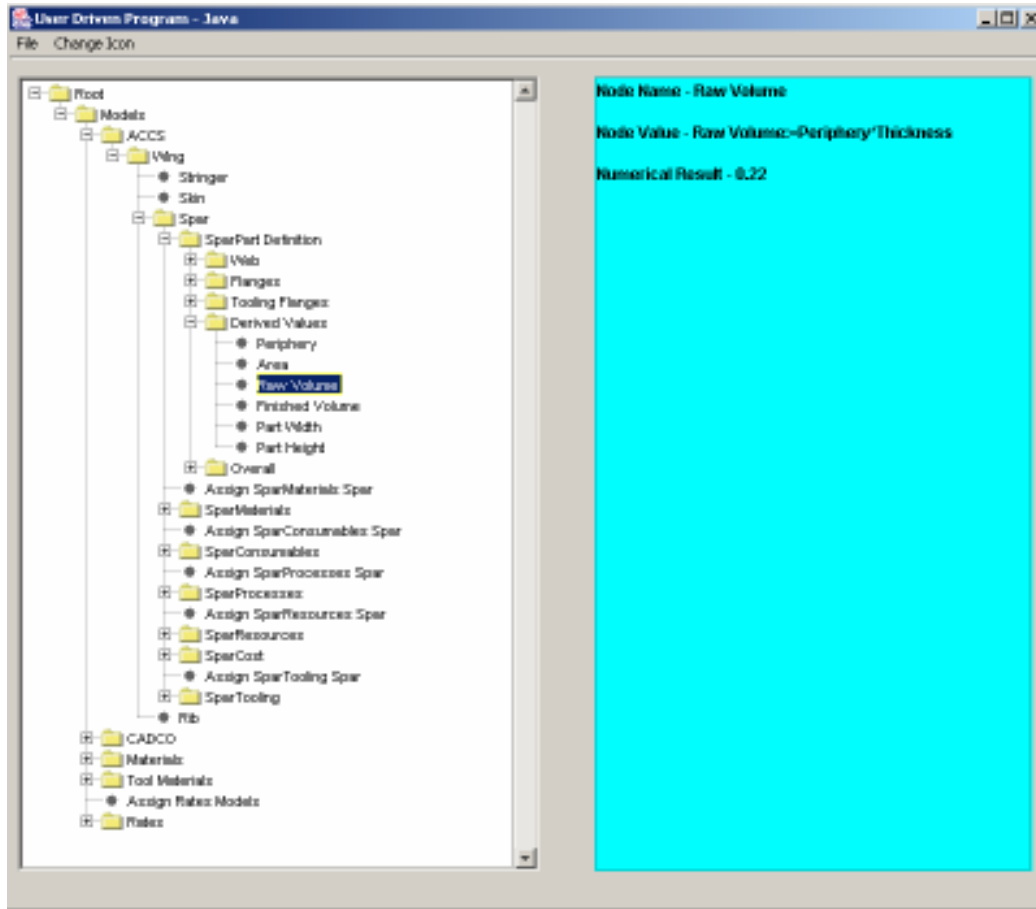


Figure 19 - Flash viewing of Spar Part Definition node

This example is on the web site at <http://www.cems.uwe.ac.uk/~phale/Flash/FlashHCI.htm>.

Researchers at the University of Queensland Australia have developed a hyperbolic browser to display RDF files, this is explained in [Eklund et al.].

Figure 20 shows the DecisionPro tree translated into Java and visualised.



**Figure 20 - Translation from decision tree into Java**

A translation can also be performed into the Java based Cost Estimator System [Koonce et al.] [Wujek et al.]. Figure 21 shows this.

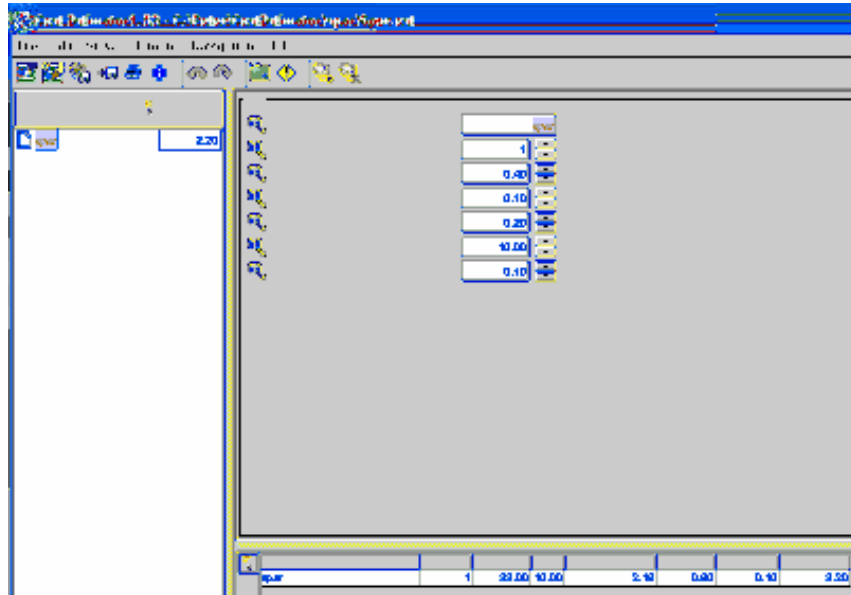


Figure 21 -Translation from decision tree to Cost Estimator

## Semantic Search

The use of open standards for representing information makes it possible to enable searches that understand the semantics of the information and so can track all of the relationships between items.

Figure 22 shows the interface for making a search. In this example the user wants to retrieve all the information related to a spar.

# Taxonomy Search Engine

Components  Query:  Exact Match:

## Components

Query - Like '\*Spar\*'

[Result Tree](#)

Figure 22 - Semantic Search interface

The result is shown as a series of trees for each item that contains the word spar. Each keyword match is the root of a tree. Each tree shows the item found and all its' children and attributes. Figure 23 shows an image of the top part of the results, this is part of the branch for the first item returned.

36 Matches

<b>Search Result</b>			
<b>1</b>			
Spar			
	SparPart		
	Definition		
		Web	
			Web Height
			0.4m
		Flanges	
			Flange Width
			0.1m
		Tooling Flanges	
			Tooling Flanges Height
			0.2m
		Derived Values	
			Periphery
			2*
			(Length+Web Height+
			(2*Flange Width)+
			(2*Tooling Flanges Height))
			Area
			Length*(Web Height+2*
			(Flange Width + Tooling Flanges Height))

**Figure 23 - Results from semantic search**

The information is held in a taxonomy so it is not HTML that is being searched but the taxonomy itself. Because the information is held in a structured way, it is much more likely that searchers will find what they are looking for because the search can follow the relationships represented in the taxonomy. One of the key aims of semantic web research and Web 2.0 is to make this kind of search possible over the web as a whole. The semantic web is a longer-term vision for managing information over the web. Web 2.0 is the shorter-term practical implementation of techniques, which can ease current information search and management problems. XML files can be queried using XQuery a W3C working draft [World Wide Web Consortium]. RDF files can be queried using SPARQL a W3C specification, for which a demonstration is available at <http://xmlarmyknife.org/api/rdf/sparql/query> and a tutorial based on this has been developed by [Dodds]. A web interface has been developed for Protégé [WebProtege]. An example of the use of this is shown in Figure 24, where a search is made for information on the cure cycle for composites manufacturing.



Figure 24 - WebProtege Searching - Cure Cycle

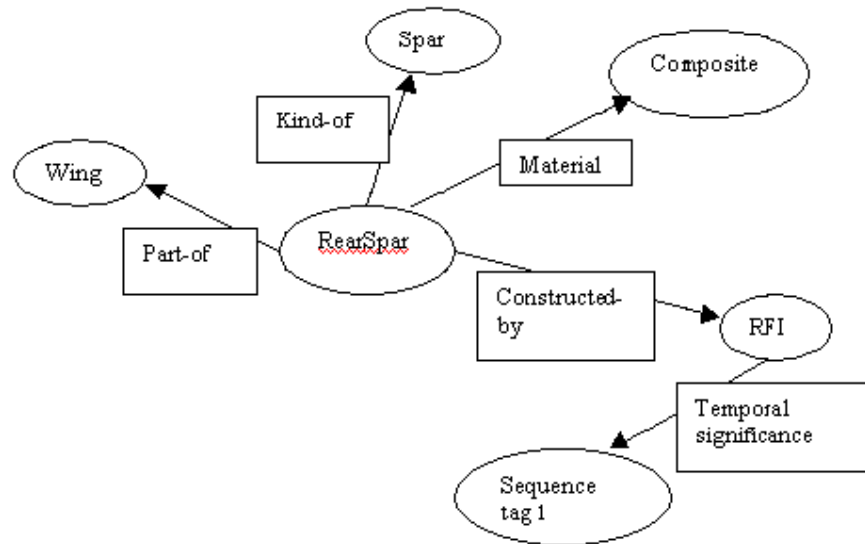
[Lee et al.] present a distributed visual reasoning system for intelligent information retrieval on the Web. The user can design a query by linking active icons, and then inputting the required parameters. The user can then see the structure of the query and obtain results from the information database.

So far the taxonomies include the traditional object oriented relationships such as child, parent, sibling, attribute, and instance. There are other types of relationship that would need to be modelled in order to maximise the capabilities of software that would use the taxonomies. Examples of these are shown in Figure 25.

Basic key relationships used within the object oriented programming domain between objects have been described. These key relationships depict families of objects that may share attributes and methods through inheritance. They also describe aggregations of objects that make (usually) some geometric sense. Semantic descriptions with more relationship types than the ones described so far allow a more expressive depiction of a problem domain, and can aid some forms of search within a model.

One of the main advantages of a semantic net description, in terms of automated model generation, is that labelling relationships between objects allows the depiction of a number of aspects of a domain in one model, and with a consistent syntax. This allows representation of, say, a product structure and its manufacturing processes together. A single node then is the only representation of that node within the model, with all its relationships depicted as arcs emanating/terminating at the node.

An example is shown in Figure 25, where a rear spar is focussed upon.



**Figure 25 -Extended Semantic information**

More expressive semantic descriptions are possible within the XML world through the use of one of the standard OWL dialects. Protégé has an OWL plug-in available that provides this functionality, together with links to reasoning tools for maintaining and analysing the logical constructs.

It is also important not to stay limited on one ontology development environment but instead explore how ontologies can be developed using a range of development tools and translated between each where necessary. For this reason Jenna and other open source tools developed by Hewlett-Packard Semantic Web Researchers [Hewlett-Packard], and KAON from the University of Karlsruhe [Volz et al.] are being investigated.

### ***Findings from using the alternative approach***

The alternative approach of using open standards taxonomies and a web interface for developing decision support models for design and costing solved the problems of the spreadsheet approach as indicated below -

#### **Maintenance**

The use of a centralised information source makes these models more reliable than the standalone spreadsheet. It is much harder to update multiple instances of the spreadsheets used by different people and ensure they all contain the same information. As a piece of information can only belong to a unique location, the problems arising from duplicate pieces of information are eliminated. The models have only the functionality that is added by the model builder so there are no other side effects to keep track of, as there are with generic functionality within spreadsheets.

#### **Extensibility**

Creating the infrastructure took much more time than it did for the spreadsheet system, but having done so it is quicker and easier to create further models. This indicates that the extra research and development time taken was worth it in the long term, most of this time was in research and this can be used in future projects. The use of open standards for information and models ensures there should be a development path, whatever changes there may be in the software market.

#### **Ease of Use**

Most people now are familiar with web pages and at least the basics of how to navigate them, and by keeping the navigation simple and standardised it is possible to make this easy. The models contain only the functionality that is added by the model builder unlike the spreadsheets which had generic functionality that was not required and led to users' confusion.

## **Sharing of Information**

The use of open standards languages for representing information makes it much easier to represent information in a way that makes it accessible both to people and software. Web browsers make it possible to share information with many users at once

## **Further Research**

A future task to be undertaken would be the inclusion of uncertainty in the automatically produced models, for situations where accurate information can not be provided for the model. This would require provision of a way of handling uncertainty for parameters within the ontology, e.g. as 3 values describing a triangular distribution rather than a unique absolute value. The decision support meta-program could be expanded to write out the code to run Monte-Carlo sampling, hence making use of the statistical uncertainty capability.

It would be interesting and useful to create an environment where people could use example models and evaluate their usability and usefulness. This could follow a similar model to that used for the development of open source software or collaborations such as [Wikipedia], and the Semantic Web Environmental directory [SWED]. This could bring together people with diverse backgrounds, interests and expertise.

Future research could involve creating an ontology to enable users to specify parameters in diagrammatic form. It could be possible to extend the semantics used in the specification of models to allow the creation of a framework for simulations. Because the ontology would use open standards, these simulations could be made broadly available on the web. It is important that the necessary infrastructure is created to allow this facility to be added. The approaches of others to this problem have been examined. [Page et al.] examine the nature of web-based simulations. [Miller et al.] explain the technology behind web-based simulations, and argues the need for demonstrating the application of web-based simulations for major projects. The authors were involved in the RUBE project that developed a system for battle simulations illustrated in [Fisher and Miller] that uses open standards and Protégé for the ontology, and outputs some code automatically. [Kuljis and Paul] evaluates progress in this field of web simulation. [Kim et al. ] explain how techniques of generating executable code from documents specified in standardised XML can be used to create simulations. [Reed et al.] examine possibilities for improving the aircraft design process with web-based modelling and simulation. Simulations could also be used for optimization and [Chen and Yücesan] investigate this.

The Grid and Semantic Web areas of research are converging, so it will be important to watch their developments for potential use. UK Universities are involved in semantic grid-computing research including Southampton and Portsmouth Universities [De Roure et al.<sub>1</sub>], [De Roure et al.<sub>2</sub>].

## **Conclusion**

The approach of developing decision support models for design and costing using a spreadsheet was compared to the alternative approach of using open standards taxonomies and a web interface for this purpose. The conclusion is that although the use of spreadsheets for this purpose allows for the creation of models relatively quickly they are beset by problems. These relate to Maintenance, Extensibility, Ease of Use, and Sharing of Information. The alternative approach involves the development of a system, where a model builder, via visual editing of library taxonomies can undertake maintenance and extension of information. As yet a prototype only has been created for this and a great deal more work is required to finalise the structure for the holding of this information and agree standards for terminology. However dealing with this proof of concept has indicated that it is far easier to maintain, search and share information using this approach than it was using spreadsheets. Creating the infrastructure has taken much more time than it did for the spreadsheet system, but having done so it is much quicker and easier to create further models. This indicates that the extra research and development time taken though far exceeding what is required for a spreadsheet model is well worth it in the long term. Also the use of a centralised information source makes these models more reliable than the standalone spreadsheet, standalone decision support models created individually may contain out of date information. In addition, since a well constructed ontology implies that a piece of information can only belong to a unique location, the problems arising from duplicate pieces of information are eliminated. It is also much easier to create models once the infrastructure is in place. The ability to visualise, search and share the information using structured languages and web-pages, is a huge advantage for creation of dynamic image views and decision support models over the web.

## ***Acknowledgement***

Airbus UK provided support that enabled this paper to be written.

P Evans

Airbus UK  
New Filton House  
Filton  
Bristol BS99 7AR

[Peter-h.evans@baesystems.com](mailto:Peter-h.evans@baesystems.com)

## References

- Aziz, H., Gao, J., Maropoulos, P., Cheung, W. M. (2005) Open standard, open source and peer-to-peer tools and methods for collaborative product development. *Computers in Industry*, 56, 260-271.
- Bechhofer S., Carrol J. (2004) Parsing owl dl: trees or triples? *Proceedings of the 13th international conference on World Wide Web*, NY, USA pp 266 - 275.
- Bru, C., Scanlan, J., Hale, P., Dunkley, M. (2002) Visualisation of Cost Information. *Proceedings of the 9<sup>th</sup> ISPE International Conference on Concurrent Engineering Advances in Concurrent Engineering*, Cranfield, UK, pp 829-838.
- Chen C.-H., Yücesan E. (2001) Distributed Web-Based Simulation Experiments For Optimization, *Journal of Simulation Practice and Theory*, 9, 73-90.
- Ciancarini, P. & Rossi, D. & Vitali, F. (2001) Designing a document-centric coordination application over the Internet. *Interacting with Computers*, 13 677-693.
- Corcho, O., Gómez-Pérez, A. (2000) A Roadmap to Ontology Specification Languages. *Proceedings of the 12th International Conference on Knowledge Engineering and Knowledge Management*. Chicago, USA.
- Corcho, O. Fernández-López, M., Gómez-Pérez, A. (2003). Methodologies, Tools and Languages For Building Ontologies. Where is their Meeting Point?. *Data and Knowledge Engineering*, 46, 41-64.
- Fisher, P. A., Miller, J. A. (2004) Ontologies for Modeling and Simulation Issues and Approaches. *Proceedings of the 2004 Winter Simulation Conference*, Orlando, Fla, pp. 259-264.
- Fluit C., Marta S., Harmelen F. V., Staab S., Studer R. (2003) *Handbook on Ontologies in Information Systems*. Springer-Verlag.
- Huang, G. Q., Mak, K. L. (2001) Issues in the development and implementation of web applications for product design and manufacture. *Computer Integrated Manufacturing*, 14 (1), 125-135.
- Huhns, M. (2001) Interaction-Oriented Software Development. *International Journal of Software Engineering and Knowledge Engineering*, 11 259-279.
- Kim, T., Lee, T., Fishwick, P. (2002) A Two Stage Modeling and Simulation Process for Web-Based Modeling and Simulation. *ACM Transactions on Modeling and Computer Simulation*, 12 (3), 230-248.
- Kim Y., Choi Y., Bong Yoo S. (2001) Brokering and 3D collaborative viewing of mechanical part models on the Web, *Computer Integrated Manufacturing*, 14 (1), 28-41.
- Kuljis, J., Paul, R. J. (2001) An appraisal of web-based simulation: whither we wander?. *Simulation Practice and Theory*, 9, 37-54.
- Lau, H. C. W, Ning, A., Pun, K. F., Chin, K. S., Ip, W. H. (2005) A knowledge-based system to support procurement decision. *Journal of Knowledge Management*, 9 (1), 87-100.
- Lee, C. & Chen, Y. T. (2000) Distributed visual reasoning for intelligent information retrieval on the web. *Interacting with Computers*, 12, 445-467.
- Li, W. D. (2005) A Web-based service for distributed process planning optimization. *Computers in Industry*, 56, 272-288.
- Mens, K., Michiels, I., Wuyts, R. (2002) Supporting Software Development through Declaratively Codified Programming Patterns. *Expert Systems with Applications* 23, 405-413.
- Merlo C., Girard P. (2004) Information system modelling for engineering design co-ordination. *Computers in Industry*, 55, 317-334.

- Miller, J., Fishwick, P. A., Taylor, S. J. E., Benjamin, P., Szymanski, B. (2001) Research and commercial opportunities in Web-Based Simulation. *Simulation Practice and Theory*, 9, 55-72.
- Morris, S., Neilson, I., Charlton, C., Little, J. (2001) Interactivity and collaboration on the WWW - is the 'WWW shell' sufficient?. *Interacting with Computers*, 13, 717-730.
- Nidamarthi S., Allen R. H., Ram D. S. (2001) Observations from supplementing the traditional design process via Internet-based collaboration tools, *Computer Integrated Manufacturing*, 14 (1), 95-107.
- Noy N.F. (2004) Semantic Integration: A Survey Of Ontology-Based Approaches. *SIGMOD Record, Special Issue on Semantic Integration*, 33 (4).
- Olsson, E. (2004) What active users and designers contribute in the design process. *Interacting with Computers* 16, 377-401.
- Page E. H., Buss A., Fishwick P. A., Healy K. J., Nance R. E., Paul R. J. (2000) Web-Based Simulation: Revolution or Evolution? *ACM Transactions on Modeling and Computer Simulation*, 10 (1), 3-17.
- Paternò, F. (2005) Model-based tools for pervasive usability. *Interacting with Computers* 17 (3), 291-315.
- Reed, J. A., Follen, G. J., Afjeh A. A. (2000) Improving the Aircraft Design Process Using Web-Based Modeling and Simulation. *ACM Transactions on Modeling and Computer Simulation*, 10 (1), 58-83.
- Rhodes, G., Macdonald, J., Jokol, K., Prudence, P., Aylward, P., Shepherd, R., Yard, T. (2002) *Flash MX Application and Interface Design*. Friendsofed.
- Wang, C.-B., Chen, T.-Y., Chen, Y.-M., Chu, H.-C. (2005) Design of a Meta Model for integrating enterprise systems. *Computers in Industry* 56, 205-322.
- Zhang, S., Weimen, S., Hamada, G. (2004) A review of Internet-based product information sharing and visualization. *Computers in Industry*, 54, 1-15.

### **Web Links**

- Ambler, S. W. (2003) The Object-Relational Impedance Mismatch. <http://www.agiledata.org/essays/impedanceMismatch.html>.
- Berners-Lee, T. (2005) <http://www.w3.org/People/Berners-Lee/>.
- De Andreis, E. (2005) 2MXtree XML-based tree. <http://manudea.duemetri.net/xtree/>.
- De Roure, D., Baker, M. A., Jennings, N. R., Shadbolt, N. R. (2003) The Semantic Grid: A Future e-Science Infrastructure. <http://www.semanticgrid.org/documents/semgrid-journal/semgrid-journal.pdf>.
- De Roure, D., Baker, M. A., Jennings, N. R., Shadbolt, N. R. (2004) The Semantic Grid: Past, Present and Future. <http://www.semanticgrid.org/documents/semgrid2004/semgrid2004.html>.
- Dmitriev, S. (2004) Language Oriented Programming: The Next Programming Paradigm. <http://www.onboard.jetbrains.com/is1/articles/04/10/lop/>.
- Dodds, L. (2005) Introducing SPARQL: Querying the Semantic Web. <http://www.xml.com/lpt/a/2005/11/16/introducing-sparql-querying-semantic-web-tutorial.html>.
- Eklund, P., Roberts N., Green S. (2005) OntoRama: Browsing RDF Ontologies using a Hyperbolic-style Browser. <http://www.kvocentral.org/kvopapers/ontorama.pdf>.
- Fluit, C., Marta S., Harmelen F. V. (2003) Supporting User Tasks through Visualisation of Light-weight Ontologies. <http://www.cs.vu.nl/~frankh/abstracts/OntoHandbook03Viz.html>.

HP Labs Semantic Web Research (2005) <http://www.hpl.hp.com/semweb/>.

Hunter, A. (2002) Engineering Ontologies. <http://www.cs.ucl.ac.uk/staff/a.hunter/tradepress/eng.html>.

Koonce, D., Judd, R., Keyser, T., Bailey, M. A. (2000) Cost Estimation Tool Integrated into FIPER, American Institute of Aeronautics and Astronautics. <http://www.engineous.com/resources.htm>.

Lemos M. (2005) MetaL: An XML based Meta-Programming language. <http://www.meta-language.net>.

Park M., Fishwick P. A. (2005) Ontology-based Customizable 3D Modeling for Simulation. <http://www.cise.ufl.edu/~mhpark/SCS.pdf>.

Rhodes, G., Macdonald, J., Jokol, K., Prudence, P., Aylward, P., Shepherd, R., Yard, T. (2002) A Flash Family Tree, in: Flash MX Application and Interface Design. <http://www.friendsofed.com/books/1590591585/>.

Schrage M. (1991) Spreadsheets: Bulking Up On Data. <http://www.systems-thinking.org/buod/buod.htm>.

Semantic Web Environmental directory SWED (2005) Summary. <http://www.swed.org.uk/swed/about/>.

Stanford University (2005) Welcome to protégé. <http://protege.stanford.edu/>.

Vanguard (2005) <http://www.vanguardsw.com/decisionpro/>.

Volz R., Staab S., Oberle D., Motik B. (2003) KAON SERVER - A Semantic Web Management System <http://www.aifb.uni-karlsruhe.de/WBS/dob/pubs/www2003.pdf>

Wikipedia (2005) Main Page. [http://en.wikipedia.org/wiki/Main\\_Page](http://en.wikipedia.org/wiki/Main_Page).

World Wide Web Consortium - W3C (2005) <http://www.w3.org/XML/>, <http://www.w3.org/RDF/>, <http://www.w3.org/Graphics/SVG/>, <http://www.w3.org/TR/xquery/>

Wujek, B. A., Koch, P. N., McMillan, M., Chiang, W. A. (2002) Distributed, Component-Based Integration Environment For Multidisciplinary Optimal and Quality Design. *American Institute of Aeronautics and Astronautics* <http://www.engineous.com/resources.htm>.