Requirements and Software Engineering for Tree-based Visualisation and Modelling - a User Driven Approach

Peter Hale, Tony Solomonides and Ian Beeson

University of the West of England, UWE

Abstract

This paper is about potential to provide an interactive visual ontology/taxonomy based modelling system. The research is part of efforts to structure, manage, and enable understanding of complex engineering, business and/or scientific information to enable those involved to collaborate using a systems approach. The aim and objectives are to provide a taxonomy management system to close the link between requirements gathering and end-user modellers. The research is into modelling of product data structures. This research could be adapted to business process modelling, and biology taxonomy visualisation/representation. The modelling system was developed to assist decision support for problems such as wing and engine design. The methodology involves modelling using tree structured ontology based modelling. It is argued that visualising this structure enables improved Maintenance, Extensibility, Ease of Use, and Sharing of Information, and so enables better and more accessible modelling. This is achieved by uniting the software taxonomy structure with the structure of the domain to be modelled and visualised, and using Semantic Web technologies to link this with ontologies and to end-users for visualisation. This research assists with management of development, use, and re-use of software in order to make this an integrated process. The research brings together related fields of Semantic Web, End-User Programming, and Modelling, to assist domain expert end users.

1 Introduction

There are opportunities to improve visualisation and interactivity capabilities of existing ontology representation and to combine this with modelling/end-user programming. The methodology, tools, and techniques to aid this are evaluated and discussed in section 2.2, and 2.3. This approach is combined with Semantic Web techniques to enable automated structuring and management of information, and make it more accessible via the web. This is the basis of the PhD work examined in this paper, and developed for the approach of User Driven Modelling/Programming (UDM/P).

Management, structuring, and visualisation information enables representation of complex hierarchical problems such as product and process modelling. This makes possible gathering and enabling representations of such problems, and a unified generic approach to this kind of modelling and linking of models via the Semantic Web. The aim is that a taxonomy management system will enable use of information, and a methodology for its representation and contextualisation in varied interactive ways, according to what is most useful for particular people and types of information. Applications are software and systems engineering, process and design/manufacturing modelling, and phylogenetic (biology trees). What is common to all these problems is their tree-based nature, suitable to taxonomies/ontologies.

Section 2 examines the problem to be solved, the role of ontologies, modelling, visualisation and interaction, and translation to aid end-user programming. The Position of software tools investigated within a table is analysed, to develop a way of combining tools for a User Driven Modelling/Programming (UDM/P) approach.

Section 3 examines objectives for enabling better more adaptable modelling, maintenance, extensibility, ease of use, and sharing of information for diagrammatic modelling. A methodology for UDM/P is developed

Section 4 reflects on implementing the methodology outlined in section 3 and shows that the main necessity is a translation to enable better modelling via improved human to computer translation. The importance of unifying the various tools and techniques through an umbrella tool suitable to end-users is discussed.

Section 5 concludes that the research showed a way of enabling domain expert modelling/programming by unifying tools and techniques to match end-users' needs. Section 5.1 examines future work.

2 Review of Tree-Based Modelling

2.1 Problem Statement

Software development is difficult for users because of time constraints, responsibilities and roles of employees that do not include the task of software development, and the need for experience of and access to programming tools. For modelling with relationship trees it is possible to construct visualisation software for non-programmers, and so improve ease of use and limit code writing by automating the requirements to model translation.

A methodology is required for creation of systems to enable collaborative end-user modelling/visualisation. This methodology could be applied to engineering process and product modelling and to allow scientists to model, visualise and debate taxonomies/phylogenies. Thus it could be proved that the methodology is generic to tree-based problems. This paper concentrates on engineering modelling.

Many computer literate people are experts in domains that require tree-based visualisation or modelling, such as engineering product structures, business process models, and biological phylogeny trees. The aim of this research is to convert requirements into a model and so enable these computer literate users to model and visualise problems by minimising code writing. This is a User Driven Modelling/Programming (UDM/P) approach. Models can include calculation via linked equations as in a spreadsheet but visualising the whole structure. If no calculations are needed just the visualisation is provided.

Research in ontology, modelling, visualisation and interaction is examined for integration into UDM/P.

2.2 Ontologies

The infrastructure of this research is an ontology that can be visualised and edited. This is step 1 of a translation process to generate a modelling system.

Gruber [1] defines and explains ontologies and examines how agreement could be achieved for ontology terms. Gruber defines an ontology as, “An ontology is an explicit specification of a conceptualization. The term is borrowed from philosophy, where an Ontology is a systematic account of existence. For AI systems, what ‘exists’ is that which can be represented” Gruber goes on to explain design criteria for ontologies. Gruber uses an engineering case study to examine usefulness of an ontology to engineers, and others who make use of equations and values with standard units. Uschold [2] states that “there is nothing inherently good about being further along the semantic continuum” ... (towards more formal ontologies) ... “In some cases there will be advantages; in other cases there will not. What is good is what works.”

Horrocks [3] explains, “An ontology typically consists of a hierarchical description of important concepts in a domain, along with descriptions of the properties of each concept.” He discusses ontology languages and their role in assisting with interoperability. Huber [4] examines issues in ensuring people transfer their knowledge and suggests organisation’s culture is important in peoples’ resistance to this. If people can see and interact with their and each others’ models this helps mitigate that problem.

Berners-Lee and Fischetti [5] sum up the advantage of the Semantic Web over other languages, and use of RDF (Resource Description Framework) Semantic Web language, “The advantage of putting the rules in RDF is that in doing so, all the reasoning is exposed, whereas a program is a black box: you don’t see what happens inside it.” The Semantic Web uses relationships to relate information and people. This relationship structure is explained as a ‘web’, and Berners-Lee and Fischetti, explain that the term ‘web’ denotes collections of nodes and links where any node can be linked to any other node. Berners-Lee and Fischetti argue for collaborative interactivity, - ‘Intercreativity’. They explain, “the world can be seen as only connections, nothing else.”

McGuinness [6] provides a useful guide on how ontologies can assist in linking distributed data. McGuinness considers the role of markup languages in defining content to be machine readable. McGuinness encourages creation of web-based visual representations of information to allow people to examine and agree on information structures. This linking and connectivity is also explained by Uschold and Gruninger [7]. McGuinness cites a diagram by Berners-Lee [8], which is further developed by Berners-Lee [9]. The concept illustrated, linked with that of ontologies contains representations of the place of each language in a layered stack alongside the purpose of the language. Each layer has an interoperable open standard interface.

McGuinness [6] outlines 7 ways ontologies are used :-

1. controlled vocabulary.
2. site organization and navigation support.
3. expectation setting.
4. “umbrella” structures from which to extend content.
5. browsing support.
6. search support.
7. sense disambiguation support.

Berners-Lee [10] explains “Despite excitement about the Semantic Web, most of the world’s data are locked in large data stores and are not published as an open Web of inter-referring resources. As a result, the reuse of information has been limited.”

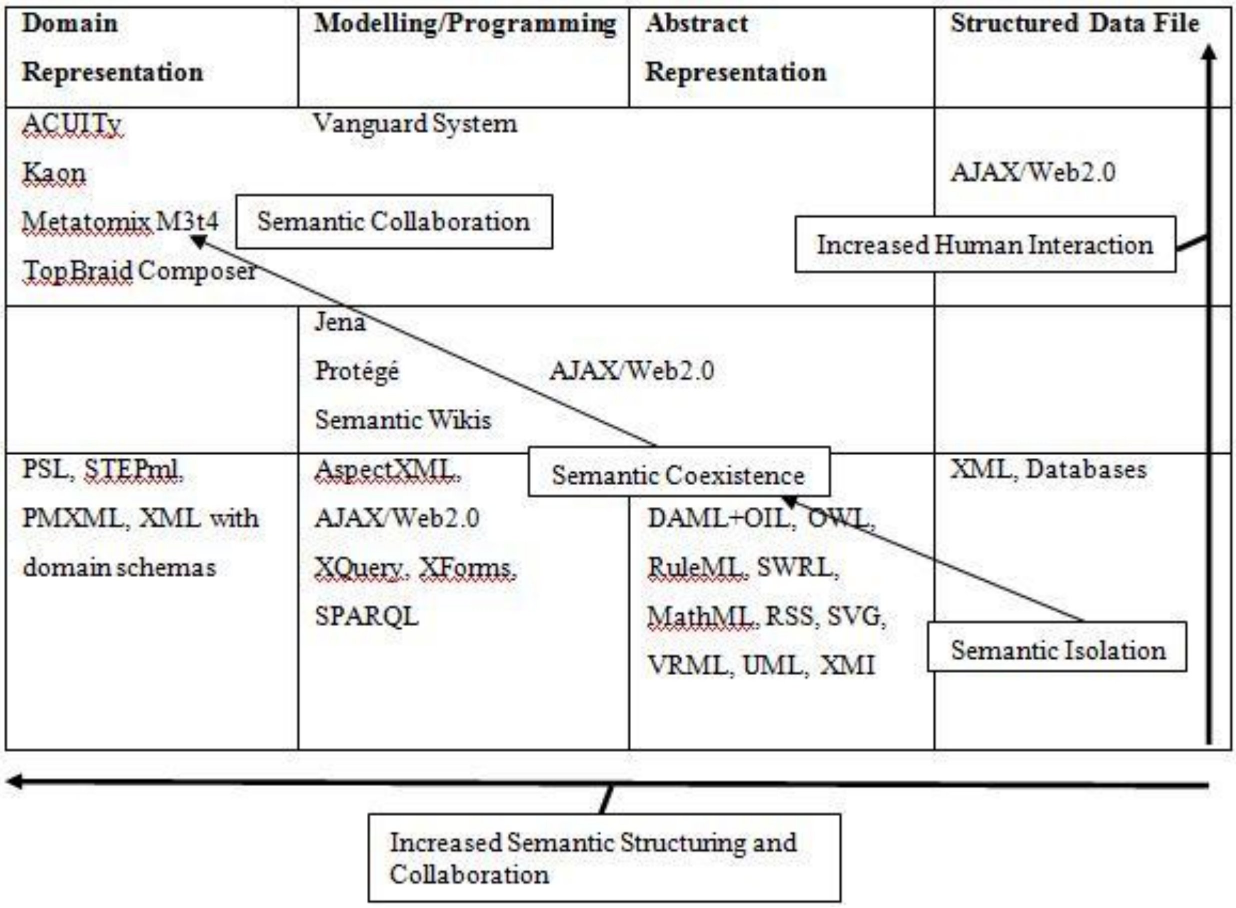

Figure 1. Language and Tool Mapping

Naeve [11] gives an example of the need for “semantic mapping” between different words with the same meaning such as ‘author’ in one ontology and ‘creator’ in another ontology in order to establish interoperability and machine readability. McGuinness [6] also investigates ontology tools/systems, and advocates their use for supporting collaboration for distributed teams. Naeve [11] describes Semantic Isolation where databases are available but hidden behind web portals, though the portals advertise their address. Semantic Coexistence is achieved by databases being structured in such a way that it is possible to search them without having to know their location. Naeve gives the example of RDF Schema RDF(S), which standardises the structuring of the information across RDF(S) databases. RDF(S) provides standardised elements for the description of ontologies, so assisting to enable Semantic mapping. Semantic mapping enables Semantic Coexistence by enabling agreement on terms. Naeve argues for semantics that are understandable to humans as well as machines; without that it is impossible for non programmer domain experts to undertake collaborative modelling.

Figure 1 outlines positioning of software to decide where each tool fits in the translation methodology to be devised. The modelling tool Vanguard System was chosen for the DATUM modelling project [12] because it handles Units and uncertainty. Advantages for the PhD were the facility to link to an ontology, collaborative and tree-based modelling capabilities, ease of use and of linking to spreadsheets and databases, facilities for web-based models, and for entering of formulae, and a high level programming language. The Protégé tree was translated into a Vanguard System tree. This fit in with the stepped translation to be developed. The open standard nature of Protégé made it possible to use it without being locked in. Tools such as TopBraid Composer provide additional higher level functionality such as an improved user interface and more facilities for user interaction and modelling by end-users. Leavers’ MSc project [13] used Jena, and there was regular contact with the developers of ACUITy [14] to examine a Jena based approach. Jena based Metatomix M3t4 was also used. So results with Jena were similar to those for Protégé and Vanguard System. Analysing the position of tools within Figure 1 to ensure the best combination for a project is the most important way of choosing tools. The ontology tools all fit in the top quarter of the table and so provide similar functionality. Such ontology tools, and all the other tools in Figure 1 are improving, so reproducing this research by modelling at high level with involvement of end-users is thus practical.

2.3 Modelling

Ontologies are a base for modelling tools to provide a structured system for building and editing of models. An ontology can store related information and calculations; any required calculations would then be made and translated to provide a model that can be interpreted by users. This research solves problems of translation from human to computer and vice versa. This is achieved by giving users involvement in the translation process by providing for them to interactively model the problem.

Cheung et al. [15] wrote a useful guide to visualising and editing ontologies, this and interoperability via open standards languages makes modelling practical. Linking ontologies with modelling make them useful in engineering, and science, where calculations are required.

2.4 Visualisation and Interaction

Huhns [16] and Paternò [17] explain that alternatives to current approaches of software development are required. Huhns argues that current programming techniques are inadequate, and outlines a technique called 'Interaction-Oriented Software Development', concluding that there should be direct association between users and software, so users can create programs. Translation between ontologies, models, and visualisation enables translation between levels of abstraction, and therefore from human to computer and back. This approach concentrates on visualising the entire program code to end-users as a model. This is how to allow people to program modelling solutions at the level of abstraction they are most comfortable with. Paternò [17] outlines research that identifies abstraction levels for software systems.

Crapo et al. [18] argue that spreadsheet users are considered as potential modellers, "Every one of the perhaps 30 million users of spreadsheet software can be considered a potential modeller". Crapo et al. also explain that visualisation helps modellers to maintain a hierarchy of sub models at different stages of development and to navigate effectively between them. This is the reason for breaking down the models into a tree/graph/web structure. Jackiw and Finzer [19] and Guibert et al. [20] demonstrate how a view of the problem that is visual and near to peoples' way of thinking helps modellers. Context is essential. Guibert et al. explain with an example of a numerical representation of a triangle how numbers fail to reveal the concept behind them. This representation is 'fregean' as it does not show the concept of a triangle. Beside this is a diagram of the triangle that shows the concept, this is an 'analogical' representation as it includes the context of the information. Visualisation and interaction research enables end-user programming, such as for engineers to model/program at a high level of abstraction.

2.5 End-User Programming

End-user programming development over past decades was also reviewed. Two main conclusions resulted :-

- Research that created software for end-user programmers such as children, but had limited acceptance and use in the market can be reused with new technology to assist development.
- Pragmatic research that involved creation of tools for the mass market, but which avoided more long term issues can now be extended.

Section 3 develops the knowledge gained in section 2 into a methodology for tree-based programming.

3 Development of Methodology

3.1 Objectives

An objective is to develop a process to enable decision support, minimising dependence on specialist software and detailed programming. The User Driven Modelling/Programming (UDM/P) approach and its application to systems modelling research is developed.

This research examines creation of models and modelling systems, and how this can be eased for non-programmers. It identifies ways that creation of models and modelling systems is similar to other types of programming, so the research can be applied generally. The purpose is to enable end-users to create and adjust models and so maximise maintenance, extensibility, ease of use, and sharing of information; in order to develop a systematic methodology for creation of accessible and adaptable models, applicable to a range of situations. This enables end-users to model their domain problems.

3.2 Requirements

The development process investigated is that of ontology based translation between requirements, models and visualisation Figure 2 illustrates how this is most applicable to tree-based problems and models :-

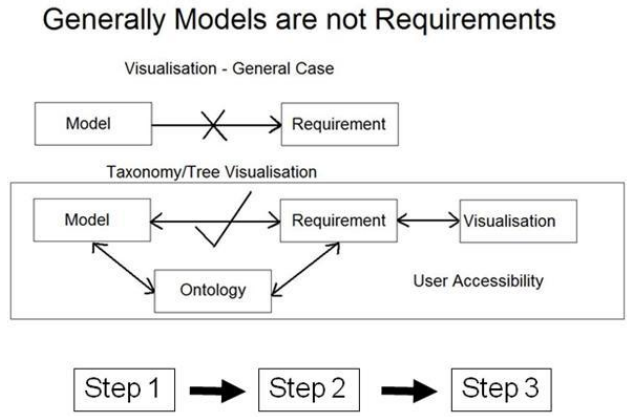

Figure 2. Visualisation and Interaction Mechanism

3.3 Methodology - Enabling Better and more Accessible Modelling

This methodology is used to enable better management of software through improved maintenance, extensibility, ease of use, and sharing of information. This makes software management an integral, consistent, continuous part of development, use and re-use.

Maintenance

Maintenance of models and modelling systems is improved by :-

- Stepped translation process, Step 1 Ontology and Taxonomy creation, Step 2 Translation and Modelling, Step 3 Translation and Visualisation.
- Use of open standards to represent information in a format available to the maximum range of maintainers without being dependent on the computer system or software used.
- Ensuring the structure of the modelling and programming system and all its related information is visualised clearly. This is ideal for point '*5. Maintainability*' Sommerville [21], this is also ideal for tree-based modelling systems.
- Minimising the amount of code necessary to create a model, and structuring the model so that all connections can be seen.

Extensibility

Extensibility is also improved by the above means; this enables understanding of models and so allows for easier re-use. A structured representation can be edited with fewer worries about unintended consequences. This is achieved by translation and visualisation to enable model builders and users to modify the ontology and model. This is the 3-step translation process developed for User Driven Modelling/Programming (UDM/P). Users make changes to whichever step is appropriate depending on the task they are performing and their interests and preferences. McGuinness [6] observes the importance of extensibility, "Extensibility. It will be impossible to anticipate all of the needs an application will have. Thus, it is important to use an environment that can adapt along with the needs of the users and the projects."

Ease of Use

- Maximising accessibility is important to ease of use and vice versa, use of open standards assists this, as does enabling models to run on software and systems familiar to users.
- Clear structuring and visualisation of information/models also assists in making a modelling system more usable.

Sharing of Information

Achievement of all the above enables collaboration. Ontologies are used as a way of representing explicit and implicit knowledge. This makes possible creation of manageable, maintainable, and flexible models. To enable sharing of information, diagrammatic ontology based representations of models are provided.

3.4 Translation

Translation capabilities are provided to enable better communication between computer systems, and between humans and computer systems. This allows visualisation of chains of equations, which are common in cost modelling, but this work is relevant to modelling in general. To model complex problems a structured approach is needed for representing explicit and implicit knowledge. A translation is provided in 3 steps :-

- Step 1 - Ontology
- Step 2 - Modelling Tool
- Step 3 - Interactive Visualisation

Step 3 visualises results and allows interaction with information to establish the meaning of results. The translation is based on Semantic Web standards to enable widespread applicability and help ensure this is a generic solution. The visualisation and interactions can be tree/graph-based, spreadsheet, and CAD style as necessary. A further alternative is translation to programming or Meta-programming languages so information can be re-used by developers who are creating systems with these languages.

In general it is likely that there will be merging between different modelling approaches and technologies. This needs organisation and management through an integrated system. UDM/P is thus an umbrella activity.

The standardisation possible in this approach allows software developers to create modelling systems for generic purposes that can be customised and developed by domain experts to model their domain. This methodology is facilitated by :-

- Modelling Tools - Building an end-user interface and extending the translation capabilities of UML and/or other modelling tools (Johnson, [22] - to be discussed in 4.3).
- Spreadsheets - Improving the structuring and collaboration capabilities of spreadsheets, and enabling customisation of spreadsheet templates for particular domains and users.
- Ontology Tools - Extending modelling capabilities and equation calculations in ontology tools and providing an end-user interface.

- Semantic Web/Web 2.0 - Extending the capabilities of Semantic Web and Web 2.0 style development tools to allow collaborative modelling.

These possible solutions are not mutually exclusive and their combination is the best way of providing usable collaborative modelling tools for computer literate end-users and domain experts. The link between these alternative ways of advancing current research is translation and User Driven Modelling/Programming (UDM/P).

Section 4 reflects on the prototyping, implications, advantages, and problems for this work.

3.5 Information management and Interaction

Figure 3 illustrates the development process. It shows how production of better, more accessible, more adaptable and applicable models was enabled by meeting objectives of enabling better Maintenance, Extensibility, Ease of Use, and Sharing of Information. These objectives were met by better structuring and visualisation; this required work on structuring using Semantic Web and Ontologies, and enabling better visualisation through end-user programming techniques. This made the models more accessible, and so easier to edit, reuse, adapt and maintain, so providing a more manageable development process.

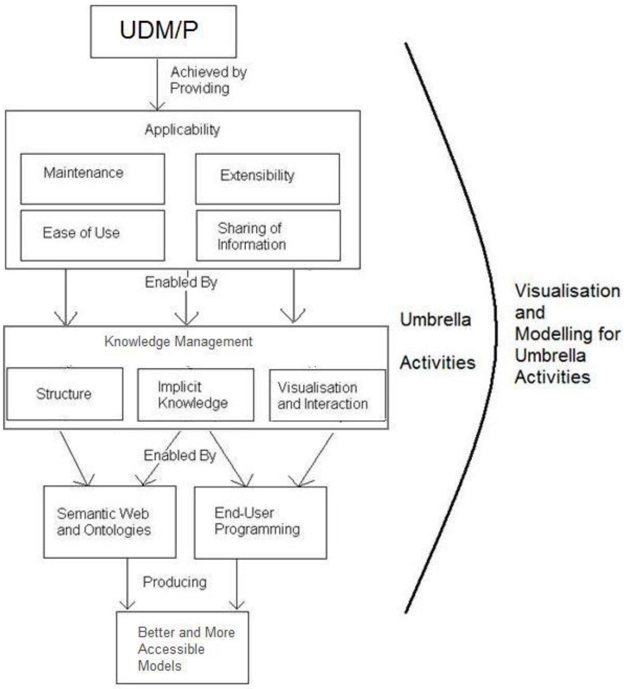

Figure 3. How Objectives and Methodology aid better modelling

Examination of this problem has indicated the need for management and co-ordination of a collaborative ontology, modelling, and visualisation process. This umbrella structure is required to manage the translation steps; in order to output accessible and better models.

3.6 Implementation of Methodology

Translation

To prototype and implement this methodology, an ontology representation was translated into a computer model. This ontology defined relationships between engineering items, the ontology was linked to Semantic Web technologies. The relationships were conveyed to a software model for evaluation. The taxonomy and CAD type view was then visualised and output to the web. The 3-step process methodology and implementation are illustrated in [23] and [24], and figure 4.

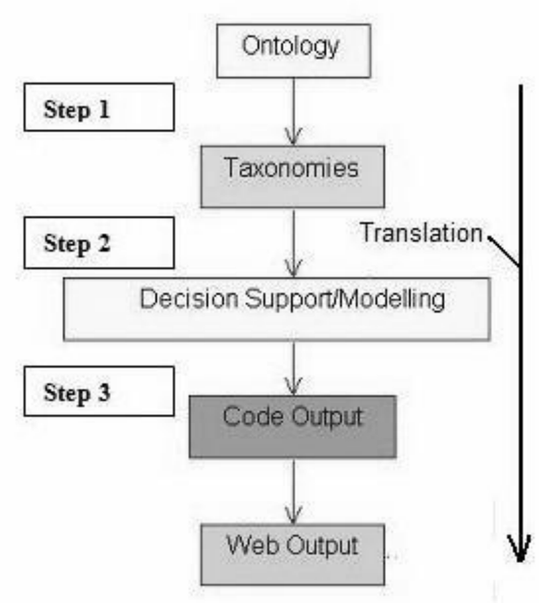

Figure 4. Translation Process Chain

4 Reflection and Discussion

This research assists with an integration of modelling, software engineering and systems engineering with a unified approach. This approach enables systems that produce systems, and models that produce models, systems that create models etc. This provides an iterative recursive translation, collaboration, and visualisation for modelling, thus improving capabilities for modelling.

This research closes the gap between developers and users, and between formal and less formal development processes. This is possible by providing an interface to model the process and the requirements and software structure in a visualised, interactive accessible way. If users drive the development process via accessible visualisation/modelling with high level modelling tools, this process then becomes agile and collaborative.

An overall modelling/visualisation structure would allow the user to establish “common ground” with the computer, an expression used by Johnson [22]. As well as translating between users and computer systems it is important to provide translations between different computer systems. Solving this would enable providing a modelling and simulation environment as a product of translation from an ontology. Miller and Baramidze [25] establish that for a “simulation study that includes model building, scenario creation, model execution, output analysis and saving/interpreting results. Ontologies can be useful during all of these phases.” Kim et al. [26] describe their approach to modelling and simulation and how a Web-based solution can be applied to distributed process planning. So a web-based ontology editor that enables modelling and visualisation is needed.

Naeve [11] argue that “combining the human semantics of UML with the machine semantics of RDF enables more efficient and user-friendly forms of human-computer interaction.” The main difficulties that need to be addressed to enable this are structural differences between Semantic Web and UML representations, and the need for improved human interaction for non programmer users. Naeve examines the strong separation between types (classes), and instances (objects) and considers this to be a weakness, which he rectifies for ULM (Unified Language Modeling) developed from UML. Johnson [22] indicates that UML tools need extending to better enable modelling of collaborative tasks. Johnson explains that successful interaction requires mapping between levels of abstraction and that translation between the levels of abstraction required by users and computers is difficult. He explains that this problem often means systems are created that make the user cope with mis-translation. Fischer [27] observes that it is the mismatches between end-users needs and software support that enable new insights. Fischer argues that software development can never be completely delegated to software professionals, because domain experts are the only people that fully understand the domain specific tasks that must be performed.

To enable computer to human common ground, an interactive, visualised ontology/modelling environment is researched. This is adapted to the way people work, with steps matched to people, skills, and roles :-

Table 1. Roles, Skills, and Translation

Step	Person Role	Skills	Tool Type
Step 1	System Creator	Programmer	Ontology
Step 2	Model Builder	End-User Programmer	Modelling Tool
Step 3	Model User	End-User	Interactive Visualisation

This stepped translation solved problems as indicated in the table below :-

Table 2. Stepped Translation and Modelling

Improvement	Achieved By
Maintenance	Structuring and Translation
Extensibility	Structuring and Visualisation
Ease of Use	Visualisation, Interaction, and Translation
Sharing of Information	Shared Ontology and Interoperability

4.1 Recommendations

- Enable people to create software visually.
- Create design abstractions familiar to domain experts e.g. diagrams for engineers.
- Ensure interoperability using open standards.
- Automate user to computer translation process.

5 Conclusions

This translation and management process was adapted to match with relevant tools, roles and skills to provide a framework for UDM/P modelling. Software tools were combined and used for end-user needs and the UDM/P approach. This enables interactive modelling and visualisation, and so widens programming participation by including computer literate non-programmers. The main ways to achieve this are through better models provided by means of improved Maintenance, Extensibility, Ease of Use, and Sharing of Information.

An issue is whether and how and by whom such an approach can be moved out of University and into industry in a practical way. This is especially difficult given the more short term pressures facing businesses/organisations. A further issue is that this approach does not suit rigid hierarchical organisations, despite being based on a hierarchical structure itself. The approach involves empowerment of users. This means it is important to enable collaboration across people, and up and down the model hierarchy. Thus this supports a democratic, decentralised structure and enables this.

5.1 Future Work

Given an analysis of Proctor et al. [28] there is a gap in research in creation and editing of web-based trees. Future work post PhD will involve improving capabilities of modelling information, such as Semantic Web technologies combined with development of increasingly interactive programmable web interfaces. This could help make possible Tim Berners-Lee’s [8][9] original vision of Web 3.0 that involves structured information linked via a stack of technologies, each providing a layer of Semantics above the layer below, to provide a computer to human translator.

References

[1] T. R. Gruber, Toward Principles for the Design of Ontologies Used for Knowledge Sharing. N. Guarino and R. Poli, ed. *Formal Ontology in conceptual Analysis and Knowledge Representation*. Kluwer Academic Publishers, 1993.

[2] M. Uschold, Ontologies Ontologies Everywhere - but Who Knows What to Think? *9th Intl. Protégé Conference, Stanford, California*, July 23-26, 2006.

[3] I. Horrocks, DAML+OIL: a Reason-able Web Ontology Language. *Proceedings of the Eighth Conference on Extending Database Technology (EDBT 2002)*, March 24-28 2002.

[4] G. P. Huber, Transfer of knowledge in knowledge management systems: unexplored issues and suggested studies. *European Journal of Information Systems*. 10:80-88, 2001.

[5] T. Berners-Lee and M. Fischetti, *Weaving the Web*. Harper San Francisco; Paperback: ISBN:006251587X, 1999.

[6] D. L. McGuinness, Ontologies Come of Age. In: D. Fensel, J. Hendler, H. Lieberman, and W. Wahlster, ed. *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential*. MIT Press, 2003.

[7] M. Uschold, M. Gruninger, Ontologies and Semantics for Seamless Connectivity. *Association for Computer Machinery - Special Interest Group on Management of Data, SIGMOD Record* December, 33(4), 2004.

[8] T. Berners-Lee, *Semantic Web on XML - Slide 10*, http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html, 2000.

[9] T. Berners-Lee. *Semantic Web "Layer Cake"*, http://www.w3.org/2004/Talks/0412-RDF-functions/slide4-0.html, 2004.

[10] T. Berners-Lee, W. Hall, J. Hendler, N. Shadbolt, D. J. Weitzner, Creating a Science of the Web. *Science 11*, 313(5788):769- 771, 2006.

[11] A. Naeve. The Human Semantic Web - Shifting from Knowledge Push to Knowledge Pull. *International Journal of Semantic Web and Information Systems (IJSWIS)* 1(3):1-30, 2005.

[12] J. Scanlan, A. Rao, C. Bru, P. Hale, R. Marsh, DATUM Project: Cost Estimating Environment for Support of Aerospace Design Decision Making. *Journal of Aircraft*, 43(4), 2006.

[13] N. Leaver, Using RDF as an Enabling Technology. MSc. Dissertation, University of the West of England, Bristol, 2008.

[14] A. Aragones, J. Bruno, A. Crapo, M. Garbiras,. An Ontology-Based Architecture for Adaptive Work-Centered User Interface Technology. *Jena User Conference*, 2006, Bristol, UK.

[15] W. M. Cheung, P. G. Maropoulos, J. X. Gao, H. Aziz, Ontological Approach for Organisational Knowledge Re-use in Product Developing Environments. *11th International Conference on Concurrent Enterprising - ICE*, 2005.

[16] M. Huhns, Interaction-Oriented Software Development. *International Journal of Software Engineering and Knowledge Engineering*, 11:259-279, 2001.

[17] F. Paternò, Model-based tools for pervasive usability. *Interacting with Computers*, 17(3):291-315, 2005.

[18] A. W. Crapo, L. B. Waisel, W. A. Wallace, T. R. Willemain, Visualization and Modelling for Intelligent Systems. C. T. Leondes, ed. *Intelligent Systems: Technology and Applications, Volume I Implementation Techniques*, 3:53-85, 2002.

[19] R. N. Jackiw, W. F. Finzer, The Geometer's Sketchpad:Programming by Geometry. A. Cypher, ed. *Watch What I Do: Programming by Demonstration. MIT Press*, Chapter 1, 1993.

[20] N. Guibert, P. Girard, L. Guittet, Example-based Programming: a pertinent visual approach for learning to program. *Proceedings of the working conference on Advanced visual interfaces*. 358-361, 2004.

[21] I. Sommerville, Software Engineering, *Eighth Edition. Addison-Wesley*, 242-243, 2007.

[22] P. Johnson, Interactions, collaborations and breakdowns. *ACM International Conference Proceeding Series; Proceedings of the 3rd annual conference on Task models and diagrams*, 86 Prague, Czech Republic, 2004.

[23] P. Hale, A. Solomonides, I. Beeson. User-Driven Modelling: Visualisation and Systematic Interaction for end-user programming, *Journal of Visual Languages and Computing*, To be published.

[24] P. Hale, User Driven Modelling: Visualisation and Systematic Interaction for End-User Programming. *PhD*, UWE, To be published 2011.

[25] J. A. Miller, G. Baramidze, Simulation and the Semantic Web. *Proceedings of the Winter Simulation Conference*, 2005.

[26] T. Kim, T Lee, P. Fishwick, A Two Stage Modeling and Simulation Process for Web-Based Modeling and Simulation. *ACM Transactions on Modeling and Computer Simulation*, 12(3):230-248. 2002.

[27] G. Fischer, Meta-Design: A Conceptual Framework for End-User Software Engineering. *End-User Software Engineering Dagstuhl Seminar*, 2007.

[28] J. B. Proctor, J. Thompson, I. Letunic, C. Creevy, F. Jossinet, G. J. Barton, Visualizing of multiple alignments, phylogenies and gene family evolution, *Supplement on visualizing biological data*, March 7(3), 2010.