A Knowledge Based Multi-Perspective Framework For Enterprise Modelling

Yun-Heh Chen-Burger
Artificial Intelligence Application Institute,
The University of Edinburgh,
80 South Bridge, Room E32, Edinburgh EH1 1HN, UK
email: jessicac@aiai.ed.ac.uk
Tel: +44 131 650 2756, Fax: + 44 131 650 6513

February 2, 2001

Abstract *Multi-perspective modelling MPM* techniques allow the presentation and analysis of complex organisational knowledge from different points of view, thus allowing the knowledge to be used for different purposes. This paper describes the multi-perspective modelling approach that has been adopted as a part of the *Air Operation Enterprise Modelling (AOEM)* project[7]. Three models have been developed: a *Business Model* to describe the concepts and processes that are used in the context of air operations, a *Role Activity and Communication Model* to identify actors involved and the operations and interactions between them, and a *Meta-Model* to provide a taxonomic structure to capture all concepts that are important to the air operation. To assist the *Multi-Perspective modelling* efforts, a framework has been proposed which uses *Meta-Model* as a light-weight ontology to provide communication of domain knowledge between models. An underlying formal method provides the basis for the automation of communication and translation and error-checking support along side human efforts. We also demonstrate how the light-weight ontology *Meta-Model* can be used as a foundation to provide partial quality assurance of multiple models. We suggest that the *MPM* approach is valuable in representing, understanding and analysing a complex domain, but that much automated support is still needed.

Key-words Multi-Perspective Modelling, Business Modelling, BSDM, Enterprise Modelling, Process Modelling, Knowledge-Based Support Tool, Case-Based Reasoning, Business Process Re-Engineering, Role Activity and Communication.

1 Introduction

Today's economy may be called a *knowledge economy* indicating that stake-holders of the right kind of knowledge may gain competitive advantages and strive in this new economy. This knowledge may be roughly divided into two types of knowledge: internal and external. The internal knowledge is the *corporate knowledge* within an organisation and the external one is the economic environment the organisation operates in.

A modern enterprise today is often a virtual entity which consists of many sub-organisations which are distributed across different geographical areas, each possessing different expertise and specialising in certain functions. This complicates the task of treating *corporate knowledge* as a whole and making effective use of it. Furthermore, the economy in which an organisation operates is very dynamic which requires an organisation to react appropriately and promptly — in adapting their goals and processes. This paper focuses on the capture of *corporate knowledge* using *Multi-Perspective Modelling (MPM)* techniques. We call the created models *Enterprise Models*.

Multi-Perspective Modelling (MPM) techniques allow one to present and analyse organisational knowledge from different points of view, which in turn allows the knowledge to be used for different purposes. In a *MPM* initiative, several different modelling languages are normally used to describe the different aspects of the same knowledge domain – in this case the business and operation domain of the enterprise. Two main motivations behind the deployment of multiple models for capturing enterprise knowledge are:

Organisational knowledge is often so complicated and of different types that normally no single modelling method can capture all of the important aspects and present them clearly and appropriately. Thus a Multi-Perspective modelling approach makes use of multiple modelling languages which compliment each other and work as a whole to describe the enterprise knowledge better.

The use of *Enterprise Models* is diverse: some may deal with only one aspect of the model, others may deal with several relevant aspects and, therefore, need to examine them together to convey a fuller view. The components of domain knowledge of an enterprise model, however, can be highly inter-dependent if they are not structured and presented appropriately. It is therefore important to have a mechanism which allows the relevant information to be gathered and presented in a clear, concise and structured way which is not overburdened with other irrelevant information. Since a multi-perspective modelling approach uses several different modelling languages, each language provides a specialised presentation and insight into specific aspects of the domain.

This approach is already used in research and practice: Common KADS methodology [11] embodies several modelling languages to help understand and capture domain knowledge and to help the design of knowledge based systems; Booch, Rumbaugh and Jacobson [1] fully embrace this approach and have offered a suite of inter-supportive modelling notations as part of the *Unified Modelling Language*, for gathering requirements and development of software systems; Frank[5] advocates this approach based on which multiple notations have been used and a multi-perspective knowledge management system (MEMO) was developed; in Zachmen's[12] Framework for Enterprise Architecture, it suggests using a variety of modelling languages to capture and describe the different aspects of a domain. The importance and benefits of using multiple and complimentary modelling languages to represent a complex knowledge body is well-recognised and adopted more frequently than before.

The domain of (military) Air Operations is complex. A main source of knowledge provided for the *MPM* initiative is an *IDEF0* model[1]. It is consists of 290 functions, 307 inputs (data types which provide input information for the functions), 294 outputs (data types or results which are produced by the functions), and 45 controls (data types which provide principles, guidances and information for executing the functions). In addition, documents written in natural language, informal diagrams and tables are provided — describing different parts of the air operations. This information is aided with correspondence with domain experts. Several aspects are considered: the infrastructures used during the operations, the operations to be carried out, people involved and their actions and the interactions between them, policies that are followed, resources and information needed, and issues such as timing for cooperation during the operation.

To illustrate these aspects, three types of models are initially chosen and built: a *Meta-Model* provides a taxonomic structure to capture all the high-level and fundamental concepts that are important to the air operation, a *Business Model* to capture the infrastructure involved in the operations and the detailed concepts that are used in the context of air operations, a *Role Activity and Communication Model* to identify the type of actors who are involved in the operations, their individual operations and the interactions between them.

Section 2, 3, 4 and 5 further illustrate how models built using each modelling language can communicate with each other and how the consistency between these models is obtained based on the underlying formal method. Section 6 describes our findings. Section 7 concludes this paper.

2 Overview of Multi-Perspective Modelling Framework

To maximise the advantages of the *MPM* approach proposed in this paper, a few principles are followed. Firstly, all of the modelling languages that are chosen as a part of the enterprise model set must be suitable to describe the chosen problem domain and appropriate to achieve the modelling objectives. Secondly, the chosen modelling languages should be complimentary of each other so that all of the concerned knowledge are described among them. Thirdly, these modelling languages should be "compatible" with each other, i.e. their modelling principles are sufficiently similar to each other so that the built model can achieve a consistent and coherent view of the domain.

It is also important that all of the models are built based at the same (or at least similar) level of abstraction: if one (or more) of the modelling language allows multiple levels of abstraction, e.g. the modelling languages of *IDEF0*[9] or *IDEF3*[8], appropriate guidelines must be established to determine which level of abstraction is mapped to the other non-decomposable modelling languages. Deploying a modelling language which allows multiple levels of abstraction is possible in the *MPM* approach, although it adds some complication to the modelling exercise, e.g. to determine the right level of abstraction to be mapped to other models, and to decide if this way of extracting information is consistent within its own structure. It also adds complexities to the checking for coherence and consistency between models. The use of *Meta-Model*, in our *MPM*

[1]The IDEF0 model was developed by Larry Tonneson, ZelTech, USA.

approach, supports this mapping between model concepts described at different levels of abstraction. It provides a taxonomic structure which describes concepts at different levels of abstraction. This structure provides the mechanism for communication and knowledge transfer between models.

Figure 1 shows our *MPM* approach. As mentioned earlier, three models are used: *IBM BSDM's Business Model (BM)*[6][2], *Meta-Model (MM)*[2], and *Role Activity and Communication Diagram (RACD)*[7].[3] Each of the three circles represents the domain knowledge that is covered by each model. The overlapped area denotes the common knowledge that is covered in different models, although it is presented in different forms (i.e. using the specialised model primitives) in each model. The area that is covered by only one model denotes the specialisation of the particular modelling language that describes the kind of knowledge that cannot be captured by any other models. An example of such specialised knowledge is the type of "role" that people play in an air operation as well as its responsibilities and operations which is not covered by any other models.

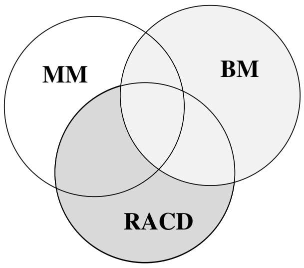

Figure 1: Overview of Multi-Perspective Modelling Approach

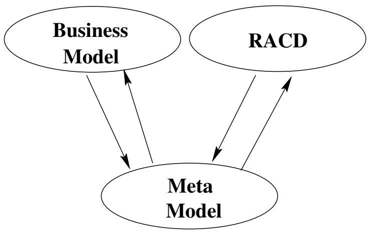

Figure 2: MPM using Meta-Model as a Backbone

Figure 2 depicts how the *Meta-Model (MM)* has been used as a backbone for the *MPM* approach. It provides a taxonomic structure to store the fundamental and important knowledge of the domain. It also stores the meta-knowledge of domain concepts.

[2] *BSDM* stands for *Business System Development Method.*

[3] *RACD* was developed by Yun-Heh Chen-Burger specifically to meet requirements for the AOEM project. It was adopted from the *Role Activity Diagram*[10] with its process notations extended with influences from *IDEF3*.

Because the information stored in the *MM* is common and sharable between different models, it forms a natural communication media for knowledge transfer and translation. Model concepts that are described in one model are mapped to *MM* which are then mapped to model concepts that are captured in another model. Given the mapping of concepts, the appropriate knowledge can be transferred and translated between models. This mapping approach is explained in more detail in Section 4. Given the mapping information, consistency checking can also be carried out across models which is illustrated in Section 5.

Based on this framework, a modelling support tool, *Knowledge-Based Support Tool for Enterprise Modelling (KBST-EM)* has been developed which is integrated with a specialised modelling tool *KBST-BM*[4][3][4] which was initially developed for *IBM BSDM's Business Modelling Method.* While providing support for fundamental model building activities, *KBST-EM* also provides some support in communication and translation of domain knowledge between models and some simple error-checking facilities. Since *KBST-BM* provides extensive support to build *BSDM's* business models, including syntactic and semantic error checking facilities, *KBST-EM* also benefits from this functionality through integration with it.

More details of the formal method and techniques used by *KBST-EM* are given in the following sections.

3 A Light-Weight Ontology: The Meta-Model

As mentioned earlier, a *Meta-Model* describes its domain knowledge in a taxonomic structure whose notation is adapted from [2]. The domain of air operations is roughly divided into 5 areas of interests: Resources, Plan and Activities, Monitored Data, Analytical Data, and Other Concepts. Resources include the physical equipment and infrastructure, the operation systems and personnel that are involved in the operations. Plan and Activities captures the different types of activities, critical assessment factors, and activity-related constraints. Monitored Data is the dynamic data about friendly forces as well as enemy forces, whereas analytical data describes an assessment based on dynamic data. "Other concepts" includes the rest of relevant information, such as weather and geographical conditions, etc.

Figure 3 shows a part of the *Meta-Model* which describes the area of resources. Two types of classes are deployed: the *Abstract* and *Concrete Class*. *Abstract Classes* provide the taxonomic structure for conceptual categories and normally describe more "general" concepts. *Concrete Classes* represent more specialised concepts and in the context of *MPM* they are often also represented in other modelling languages. For example, the concrete class, *Pilot/Mission Commander*, in the *Meta-Model* shown above is captured as a "Role" in the *RACD* model, as it is an important and specific role that a person plays in the air operation. However, the details of what a *Pilot/Mission Commander* does and how they interact with other people when in an air operation is not described in the *Meta-Model*, but in the *RACD* model (which is the speciality of *RACD*). An arrow that is drawn from class A to class B indicates an *is-a* relationship, i.e. B *is-a* (specialisation of) A.

[4] *KBST-BM* stands for *Knowledge Based Support Tool for Business Modelling*.

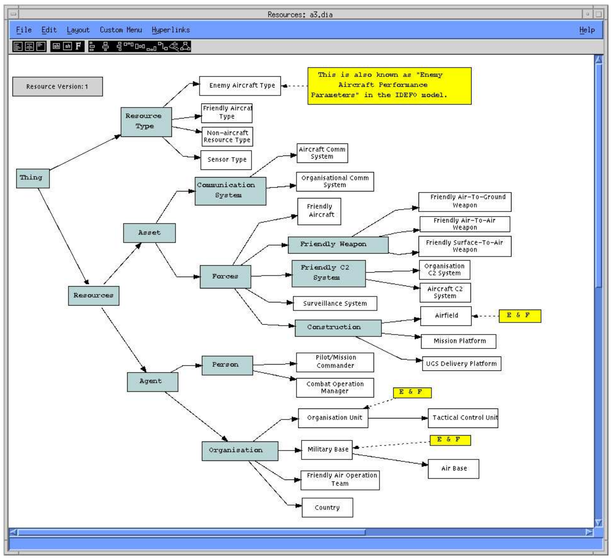

Figure 3: A Partial View of Meta-Model

The purpose of the Meta-Model is three-fold: to provide a taxonomic structure which allows the storage of meta-knowledge of the domain knowledge; to determine the core set of knowledge that is fundamental and important to the application at hand; and to provide a shared communication medium which allows information to be passed between models through it and to maintain the consistency between them. It is clear that limited complexity can be maintained if all models are mapped onto one single model, instead of mapping to every other model.

Section 5 proposes a framework to achieve *Global Consistency* across a set of Enterprise Models. The following section describes the mapping approach between different models.

4 Mapping between Models

There are four different kinds of mappings between two models: the mapping of modelling primitives, the mapping of attributes of model primitives, the mapping of model objects, and the mapping of the relationships between model primitives.[5]

The first step of mapping two models is to match the model primitives used in the two modelling languages. One pre-requisite is that the two modelling languages have to be compatible to some extent. The more similar they are, the more knowledge can be shared between them. Table 1 gives an example mapping of model primitives and their attributes between *Meta-Model (MM)*, *BSDM Business Model (BM)* and *RACD*. This mapping mainly considered the matching of model primitives from *BM* and *RACD* to *Meta-Model* but not between *BM* and *RACD*. It means information are not directly

[5]Relationships between model primitives are sometimes explicitly captured as one of the modelling primitives, but they may also be more implicitly captured, such as in a business rule or an inferred constraint between model objects.

translated between *BM* and *RACD*, but via the definition of *Meta-Model*. This enforces a standard for information transfer.

Meta-Model	Business Model	RACD
Concrete Class	Entity	Data
Concrete Class	Entity	Role
Concrete Class	Process	Process
Concrete Class Attribute	Entity Attribute	Data Attribute
Concrete Class Attribute	Entity Attribute	Role Attribute
Concrete Class Attribute	Process Attribute	Process Attribute

Table 1: Mapping of Model Primitives between MM, BM and RACD

This mapping, however, only provides the first step. The next step is to decide the mapping on the modelling object level which is to map the actual elements of a model into elements in another model. This is done by pattern matching on the name of the model object recorded in the *Meta-Model*. This approach implies a consensus and naming discipline across models that describe the same domain.

As mentioned earlier, one such mapping example is the mapping of "Concrete Class" *Pilot/Mission Commander* in the *Meta-Model* onto "Role" *Pilot/Mission Commander* in the *RACD*. Since the two model objects are the same in their semantics, if attributes of *Pilot/Mission Commander* are known in the *Meta-Model*, the corresponding matching attributes in the *RACD* are also known. This indicates a "fully equivalent" type of mapping between these two model objects. We use the notation A $\cong$ B to denote that A is (conceptually) fully equivalent to B. The above *information translation rule* can be represented below:

$$\begin{array}{l}\forall T1, M1, T2, M2, O1. \\ model\_primitive\_of(T1, M1) \cong model\_primitive\_of(T2, M2) \\ \wedge \\ object\_type((O1, T1), M1) \cong object\_type((O2, T2), M2) \\ \wedge \\ object\_attribute\_in\_model((Value, Att), (O1, T1), M1) \\ \Rightarrow \\ object\_attribute\_in\_model((Value, Att), (O2, T2), M2)\end{array}$$

where $model\_primitive\_of$*(T1, M1)* indicates that $T1$ is a model primitive (type) of model $M1$; *object_type((O1, T1), M1)* defines that the model object $O1$ is of model primitive type $T1$ in Model $M1$; *object_attribute_in_model((Value, Att), (O1, T1), M1)* stores the attribute value in $Value$ for attribute Att for model object $O1$ in model $M1$. This formula indicates that if two model primitives, $T1$ and $T2$, in models (or model languages), $M1$ and $M2$, are fully equivalent, and that the model objects, $O1$ and $O2$, of model primitive (type), $T1$ and $T2$, are also fully equivalent, then the corresponding attributes of $O1$ and $O2$ should be the same.[6]

[6]The same attribute may also be given a different name in different models. We simplify this in the formula.

The property of “fully equivalent” is transitive, i.e. if model object O1 in model M1 is fully equivalent to model object O2 in model M2, and that O2 is fully equivalent to model object O3 in model M3, then model object O1 is also fully equivalent to O3. Based on this principle, common knowledge can be transferred between models and an initial error checking across models is possible. Some such facilities are provided by *KBST-EM*.

However, not all modelling objects are fully equivalent but only similar to each other in some degree. One example is when a model object describes a more generic concept than another, i.e. it is a “generalisation” of the other, then these two model objects are similar to each other — a special type of similarity. This relationship can be naturally represented in the *Meta-Model* as super and subclasses. One other case is when two model objects share some of the common properties, e.g. having a common superclass in the *Meta-Model*.

The similarity between model objects is also a very useful piece of information. Three different levels of similarities have been identified: $A \cong B$ (A is fully equivalent to B), $A \approx B$ (A is similar to B) and $A \doteq B$ (A is related to B). The weaker the similarity the weaker the *information translation rule* that can be enforced. When two concepts are only “similar” or “related” to each other, only recommendations are made, as the translation rule is applied with a weaker inference operator, $\triangleright$ (may be true).

Given the mapping information of model primitives and model objects, the next step is to provide support for consistency checking (of some degree).

5 An Incremental Three-Tier Framework for Achieving Consistency

When the MPM approach is taken, it is essential that all models are coherent and consistent with each other. Since modelling is essentially a labour-intensive task, it will be advantageous if some form of automation can be provided alongside human efforts to help maintain the coherency and consistency of these models as a whole. We propose a three-tier framework to achieve *Local Consistency* within each model; *Pair-wise Consistency* between two models; and finally *Global Consistency* across all models.

To achieve the *Local Consistency* we must make sure that every model is coherent and consistent internally. This effort can be made possible via joint efforts of human and a method-specific modelling support tool, such as *KBST-BM*.

To achieve *Pair-wise Consistency* we must make sure that every model is coherent and consistency with the *Meta-Model*. At this stage, each model is compared and checked against the *Meta-Model* to ensure its content is consistent with the *Meta-Model*. Once an inconsistency has been found that requires updates in the *Meta-Model*, this update must also be propagated to the other models which include this concept. Consistency on this update must be reached among all models before resuming the pair-wise checking. Once all models are consistent with the *Meta-Model*, we say that an “approximate” *Pair-wise Consistency* has been achieved - it is approximate because it is a consistency with the *Meta-Model* but not with each other.

Global Consistency requires us to ensure that every model is consistent with each other. After *Pair-wise Consistency* has been achieved, any two models and the *Meta-Model* are selected. Their model objects are examined to ensure the consistency is maintained among them. Once a discrepancy is found, it should be corrected. If this discrepancy involves changes made in the *Meta-Model*, a similar procedure as the one to reach the *Pair-wise Consistency* should be followed.

If this discrepancy does not involve the *Meta-Model* but other models, then according updates should be propagated to the other models to achieve a new consistency. The user may also decide to include this knowledge in the *Meta-Model*, if this sharable knowledge is important and fundamental to the domain (however, if this is done, the pair-wise consistency should be re-gained). After the first three models have reached a local consistency among them, a new model can be added for checking and the above process is repeated until all models are included. When all models are checked and are consistent with each other, we say that the *Global Consistency* has been reached.

The process of achieving *Global Consistency* is an iterative one. It sometimes requires a revisit of the model design phase as (new) information has been discovered and added to the model. Note that, since every model needs to maintain consistency with every other model, in the worst scenario, the checking and updating activities can continue indefinitely. In this case, as with other methods, human intervention is required and heuristics should be applied. However, in our experience so far, such occasions rarely occur if the modelling languages have been chosen to be compatible with each other and the models have been carefully built. Typically, when an update does trigger a few other updates it does not trigger an infinite loop.

As achieving *Local Consistency* is often helped by using a method-specific tool, it is not discussed here; instead we are more interested in the generic rules that may be used to help achieve *Pair-wise* and *Global Consistency*. To achieve *Pair-wise* and *Global Consistency*, we must maintain across models: the consistent definition of model concepts, the consistent definition of relationships between model concepts, a coherent level of abstraction on model concepts, a coherent use of information and resources and a consistent application of process/operation logic. Our method and tool support assist in this.

One example rule for maintaining the coherent level of abstraction on all model concepts is the *Concept Abstraction Consistency* rule described below:

$$
\begin{array}{l}
\forall T1, M1, T2, M2, O1. \\
model\_primitive\_of(T1, M1) \cong model\_primitive\_of(T2, M2) \\
\wedge \\
object\_type((O1, T1), M1) \cong object\_type((O2, T2), M2) \\
\wedge \\
sub\_concept(Sub1, O1, T1, M1) \\
\Rightarrow \\
\neg \exists Sub2. \\
object\_type((Sub1, T1), M1) \cong object\_type((Sub2, T2), M2) \wedge \\
sub\_concept(O2, Sub2, T2, M2)
\end{array}
$$

This indicates that if model object $O1$ in model $M1$ is fully equivalent to model object $O2$ in model $M2$, and that $Sub1$ is a sub-concept of $O1$, then it is not the case that a model object $Sub2$ can be found in model $M2$ that is fully equivalent to $Sub1$ and is the super-concept of $O2$. The rule safeguards against a contradictory abstraction of model concepts that has been captured in different models.

Other types of modelling rules regarding the consistent definition of model concepts, the coherent use of information and resources, and the coherent application of processes can also be formalised and automatic support provided in a similar fashion. Similar work has been done in [2] which provides consistency checking for an established modelling method, *BSDM's Business Modelling Method*.

It is very difficult or impossible to produce a complete set of generic consistency checking rules such as the one described above. This is largely due to the fact that not all of the knowledge needed can be formalised — it requires not only insights into the specific application domain but also expertise in each of the deployed modelling methods. Furthermore, a good enterprise model normally reflects a degree of consensus within an organisation. This consensus is required knowledge to judge the consistency of models which can not be known before the consensus is reached. Nevertheless, automatic support such as the one proposed here is valuable as a labour-saving device because it systematically eliminates a defined set of known error types.

6 Lessons Learned

We found that *Multi-Perspective Modelling* was a suitable approach to illustrate the essence of the domain of Air Operations, since it is able to capture the different aspects of the domain and allows the presentation and analysis of concerned model concepts that was required for the project.

An important *MPM* issue is to maintain the consistency and coherence among models. Since *Meta-Model* functions as a light-weight ontology, common knowledge between models can be captured and automatic error checking assistances can be provided through it. Nevertheless, the *MPM* activity still remains largely a labour-intensive task and automatic support for model building activities at the semantic level is much needed.

Although *MPM* is an excellent way to allow one to simplify a complex domain by allowing the modeller and reader to focus on only the concerned issues without overburden themselves with other irrelevant details. However, the task of understanding or analysing a full set of enterprise models, in which each model is a "simplified view" of the domain, is still fairly complicated. Support tools which provide clear illustration of the semantics and implications of the models are much needed to help understand and quality-prove these models.

7 Conclusion

The *Multi-Perspective Modelling* approach has been adopted to describe a complex domain. We found this approach suitable and often necessary when such a complicated domain must be captured and understood. Although *MPM* provides a clearer presentation of domain knowledge on focused issues, one important issue is the sharing of knowledge among models. This is related to the communication and translation of model concepts between models. We propose a framework whose core component is a *Meta-Model* that provides a taxonomic structure to store all of the sharable, important and fundamental concepts of the domain. Underlying the *Meta-Model* is a formal method which allows the information to be transferred and translated between models. This provides the first opportunity of checking for inconsistency existing among different models. To provide a more comprehensive checking on consistency and coherence, we propose an incremental *Three-Tier Consistency Achieving Framework* which uses logical methods to provide automatic assistance in achieving the *Local, Pair-wise* and *Global Consistency*. *MPM* is still largely a labour-intensive task. We do not propose to use machines to replace human efforts – in fact it is not possible for any model for a domain of non-trivial size – instead we provide automatic assistance in vital stages of the model development process.

8 Acknowledgement

I would like to show my appreciation to Dave Robertson[7] for his valuable comments and Albert Burger[8] for prove-reading the paper. I would also like to thank members of the AOEM team[7], John Kingston[9], Larry Tonneson[10], Mike McNeil[11], Martin Brown[11], Jean MacMillan[12] and Mike Pietrucha[13] who provide domain knowledge of Air Operations.

References

[1] Grady Booch, James Rumbaugh, and Ivar Jacobson, *The Unified Modelling Language User Guide*, Object Technology, Addison-Wesley, February 1999.

[2] Yun-Heh Chen-Burger, *Formal Support for an Informal Business Modelling Method*, Phd thesis, Artificial Intelligence, The University of Edinburgh, 2001.

[3] Yun-Heh Chen-Burger, Dave Robertson, and Jussi Stader, 'A case-based reasoning framework for enterprise model building, sharing and reusing', *Proceedings of ECAI Workshop: Knowledge Management and Organizational Memories, Berlin.*, (August 2000).

[4] Yun-Heh Chen-Burger, David Robertson, and Jussi Stader, 'Formal support for an informal business modelling method', *The International Journal of Software Engineering and Knowledge Engineering*, (February 2000). World Scientific Publishing Co.

[7] Artificial Intelligence, University of Edinburgh, UK.
[8] Computer Science, Heriot-Watt University, UK.
[9] AIAI, University of Edinburgh, UK
[10] ZelTech, USA
[11] BBN, San Diego, USA
[12] Aptima, USA
[13] SM&A, System Solutions Group, USA.

[5] Ulrich Frank, 'Multi-perspective enterprise models as a conceptual foundation for knowledge management', *Proceedings of Hawaii International Conference on System Sciences, Honolulu*, (2000).

[6] IBM, 389 Chriswick High Road, London W4 4AL, UK, *Business System Development Method, Introducing BSDM*, 2nd edn., May 1992.

[7] Defense Advanced Research Projects Agency (DARPA) Program Joint Force Air Component Commander. Air operation enterprise modelling project. Web site: www.darpa.mil/iso/jfacc/index.htm, September 1999.

[8] Richard Mayer, Christopher Menzel, Michael Painter, Paula Witte, Thomas Blinn, and Benjamin Perakath, *Information Integration for Concurrent Engineering (IICE) IDEF3 Process Description Capture Method Report*, Knowledge Based Systems Inc. (KBSI), September 1995. Available at: http://www.idef.com/overviews/idef3.htm.

[9] National Institute of Standards and Technology, *Integration Definition for Function Modelling (IDEF0)*, December 1993.

[10] Martyn A. Ould, *Business Processes: Modelling and Analysis for Re-engineering and Improvement*, John Wiley and Sons, 1995.

[11] G. Schreiber, B. Wielinga, and J. Breuker, *KADS: A Principled Approach to Knowledge-Based System Development*, Academic Press, November 1997.

[12] John A. Zachmen. The framework for enterprise architecture. http://www.zifa.com/ and http://www.barnettdata.com/fromzifa.htm. Zachman Institute for Framework Advancement.