Knowledge Based Support Throughout Business Model Development Life Cycle

Yun-Heh Chen-Burger[1], David Robertson[2]
Jussi Stader[1]

[1]AIAI, The University of Edinburgh
80 South Bridge, Room E32, Edinburgh EH1 1HN, UK
email: jessicac@aiai.ed.ac.uk,jussi.stader@aiai.ed.ac.uk

[2]Department of Artificial Intelligence
The University of Edinburgh
80 South Bridge, Room E13, Edinburgh EH1 1HN, UK
emails: dr@dai.ed.ac.uk

Abstract Enterprise model development is essentially a labour-intensive exercise. Human experts depend heavily on prior experience when they are building new models making it a natural domain to apply *Case Based Reasoning* techniques. Through the provision of model building knowledge, automatic testing and design guidance can be provided by rule-based facilities. Exploring these opportunities requires us not only to determine which forms of knowledge are generic and therefore re-usable, but also how this knowledge can be used to provide useful model building support. This paper presents our experiences in identifying and classifying the knowledge which exists in *IBM's BSDM Business Models* and applying AI techniques, *CBR* and *Rule-Based* reasoning together with a symbolic simulator, to provide more complete support throughout the enterprise model development life cycle.

Key-words Enterprise Modelling, Model Development Life Cycle, Case Based Reasoning, Business Modelling, Process Modelling, Knowledge Management, BSDM, Formal Method.

1 Introduction

The main task of *BSDM's Business Modelling* is to identify two conceptual components: entities and dependencies. Entities are things that a business needs to manage and dependencies are the relationships between these things. Certain kinds of scenarios or relationships between entities are common to many businesses. Hence, one would expect that the corresponding *BSDM Business Model* maps reflect these commonalities.

In practice, IBM provides a catalogue of such generic entity models [4]: some of them are standard and example models from the method and some of them were specifically developed for selected industries. Provided with these generic models, BSDM practitioners help clients build their business model by using this information implicitly or explicitly. For BSDM consultancy, King[5] suggested three possible ways of re-using generic/known models when addressing a new problem domain.

- *Back-Pocket Approach:* the clients are made aware of the existence of these generic models, but they are only used to support consultancy. The client will see little or none of the generic model. A consultant keeps these generic models at the back of his/her mind and tailors them to the clients' special requirements.

- *Reference Model Approach:* supply the client with a relevant and complete generic model with detailed description, together with a contractual consultancy service which provides help for the interpretation and use of the model.

- *Software System Solution:* provide developed software systems as packages which are based on generic industrial models. These software systems can then be used by the clients. The client may or may not see the generic business model which was used to develop the required software system.

The fact that similar practices are exhibited in many different businesses and business models are reusable in practice make them a perfect domain candidate for applying *CBR* techniques. *Case-Based Reasoning (CBR)*[6] was inspired by observing human reasoning when learning how to solve new problems by remembering solutions that were applied to similar problems in the past, thus becoming more competent in dealing with wider range of problems over time. In the same way, a *CBR* system solves a new problem by comparing it with old problems and their solutions, which are stored in the system's memory, a *Case Library*. In the *BSDM* context, the standard and example models from the method and the generic models built for a particular industry can be stored in the *Case Library*. The next step is to understand how one can make use of these models and provide useful automatic support for the modeller.

BSDM also provides a semi-formal step-by-step procedure for building a business model which includes modelling rules, check lists and recommendations of different strength about good modelling style. This knowledge also forms a natural source for constructing error-checking and advisory rules. However, not all model building knowledge can be formalised. For example, the rule which requires the user to examine whether all of the important concepts are included in the model can not be formalised and automatically checked by our logical rules. The initial BSDM business model is a static model with system dynamic implications. To demonstrate the dynamic aspects of the model, we have successfully extended its original notation and enabled a model execution phase in our *Business Model Simulator*. Both pieces of work are described in more detail in [3].

This paper presents how knowledge which is possessed by different stake-holders: in the business modelling method, in the built industrial models, and in individual practitioners, can be captured and formalised to provide coherent and comprehensive support throughout the *model development life cycle*. It considers two issues: is such knowledge generic and reusable, and how can this knowledge be used to provide automatic support. The paper first describes how *Case Based Reasoning* techniques can be used to provide a common platform for knowledge sharing. It then presents to which extent this knowledge can be formalised and provide assistance for model building activities.

2 Modelling Support Framework

Figure 1 shows the modelling framework which provides automatic facilities to support the *iterative plan-build-test-refine* modelling development life cycle as shown in Figure 2.

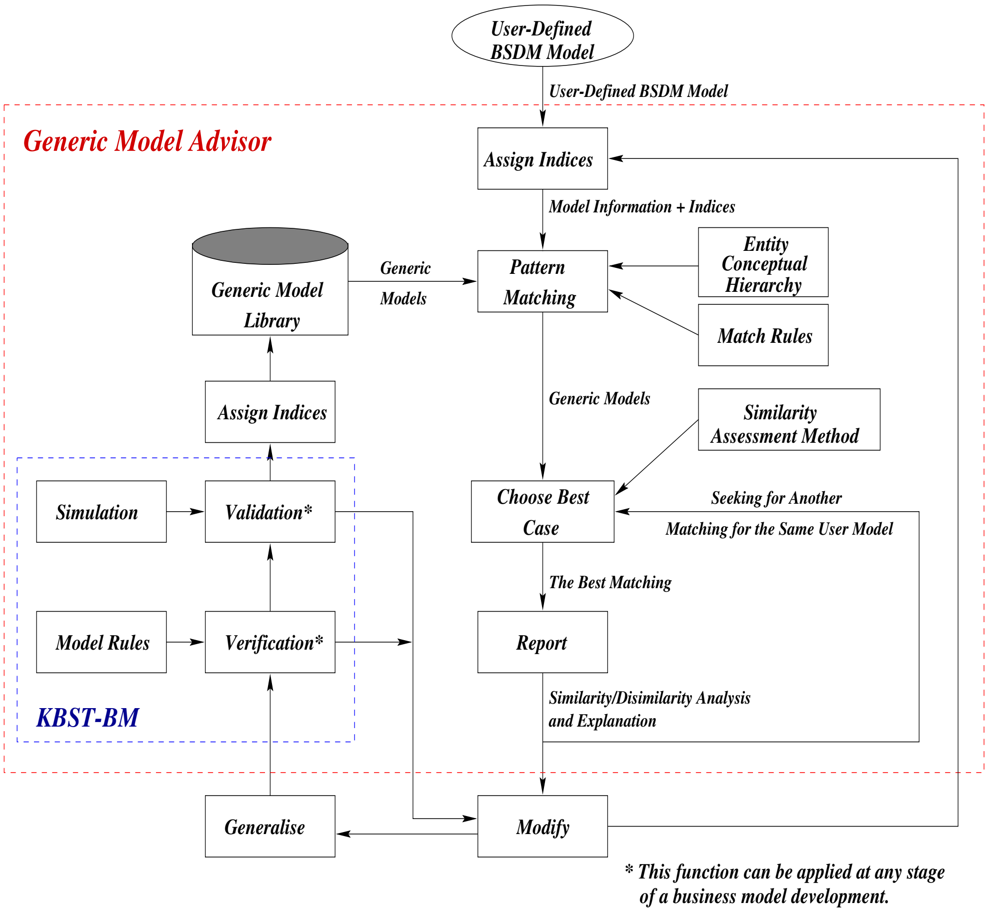

Figure 1: Architecture of Generic Model Advisor

Two integrated knowledge based support tools, *Generic Model Advisor(GMA)* and *Knowledge Based Support Tool for Business Modelling(KBST-BM)*, have been built. Since a BSDM's business model is organised and presented in *views* and diagrams, these are the "units" that *GMA* stores and retrieves. GMA identifies and assigns indices (features which characterise a model) to the problem, i.e. the user-defined BSDM model. These indices, together with the embedded domain knowledge, in our case the *Entity Conceptual Hierarchy* and *Match Rules*, are passed to the pattern matching algorithm which compares the indices of the problem and those of the generic models in the *Generic Model Library* to retrieve a set of reference models which exhibit similar characteristics to the input model.

At this stage the retrieved similar generic models are not yet examined to determine which is a better match for the current problem. For such a comparison, GMA provides a flexible *Similarity Assessment Function* which enables the deploy-

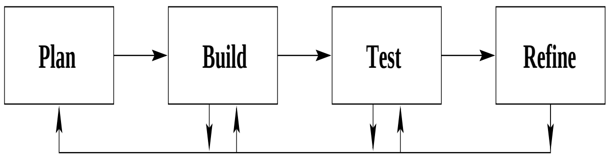

Figure 2: The Plan-Build-Test-Refine development cycle

ment of a built-in heuristic method or the users can dynamically make up their own evaluation method to explore specific matches based on the identified indices of the model.

The best matching case, according to the chosen similarity assessment method and an analysis report of similarities and differences between the user model and the retrieved reference model together with suggestions about how to eliminate the causes of the differences, are given to the user. The user can then read the report and/or ask the system to present a different matching result for another generic model. Matches are shown in the descending order of their scores in the chosen similarity assessment method.

A user-defined model may be matched with more than one generic models. The user can choose to modify his/her model and repeat the above modelling cycle as a part of an iterative process. If the user has decided to use the reference model as a basis to generate a new model, the user can export the chosen reference model from the library. At any stage of the model development, the user can choose to use the *verification* and *validation* facilities provided by *KBST-BM* to check for the completeness, soundness and appropriateness of the built model.

When the user is sufficiently satisfied with his/her model, he/she can retain this new model, i.e. write it back to GMA, by firstly generalising the new model, verifying and validating the generalised model using the integrated tool *KBST-BM*, and then storing the new generic model back to the *Generic Model Library*. The *Case Based Reasoning Cycle* is now completed, and *GMA's* knowledge can be enriched and evolved through time via the inclusion of newly acquired knowledge during operations.

The inner *KBST-BM* system box in the Figure 1 illustrates how *KBST-BM* can assist in completing the *CBR cycle*. It provides an independent *verification* and *validation (V&V)* facilities (from the user) and is included in the "Test" activity in the standard model development process shown in Figure 2. This *V&V* approach and implementation details of *KBST-BM* are given in [2].

3 Indexing, Matching and Similarity Assessment

Indices are features which can be used to distinguish models in the case memory and to find appropriate matches between a given problem and previous models. In the context of a BSDM business model, these distinguishing characteristics are embedded in the semantics of entities, the architecture of a business model, and the business area that a model describes.

Simply comparing the graphical representation of business models is not sufficient. For example, drawing an existing model upside-down does not make it a different model, the semantics of the inter-relationships (dependencies) between entities must be taken into account. Furthermore, business contextual similarities may be disguised. For instance, if a business model is a more elaborated or specialised version of another one (or vice versa), then these two models normally will not have the same architecture (e.g. one may expand parts of the model in some areas), and often they do not share the same entities (e.g. using domain specific vocabularies instead). However, because they are essentially describing the similar business operations, it will be useful to refer one to the other.

To be able to make meaningful comparisons between BSDM models, one must have an integral understanding of the business context which is described in both the architecture of a model as well as the business context that each entity represents. We capture part of this context through typing of entities via a concept hierarchy.

3.1 Entity Conceptual Hierarchy (ECH)

BSDM provides *Entity Families* which provide entities in groups according to where and how they can be used in a business model. BSDM modellers use *Entity Families* as a starting point when trying to identify entities for a new model. They also use it as a guideline to check the architecture of the model. We organise information given in the *Entity Families* in a taxonomic hierarchy, called *Entity Conceptual Hierarchy*.

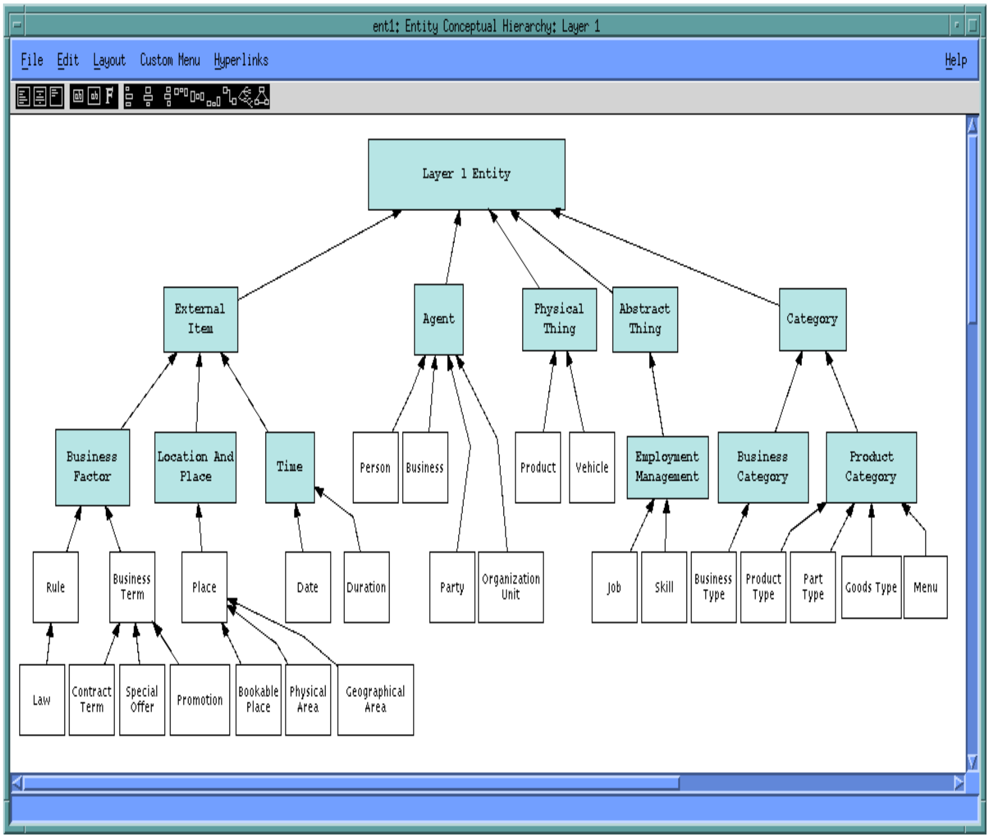

Figure 3: A Part of Entity Conceptual Hierarchy (ECH)

Figure 3 shows a screen shot from *GMA* which captures a part of the *Entity Conceptual Hierarchy* which contains the suggested entities for the top layer (layer 1) of a BSDM business model. Two types of classes have been used to describe entities: the shaded rectangular boxes represent the *Abstract Entity Types*, and the clear rectangular boxes represent the *Concrete Entity Types*. *Abstract Entity Types* provide a structure to allocate conceptual categories and normally describe more "general" concepts. *Concrete Entities* present more specialised concepts and include entities which are used in real business models (as opposed to a generalised model). An arrow from entity B to entity A indicates an *is-a* relationship from B to A, i.e. B *is-a* A.

The *Entity Conceptual Hierarchy* captures the semantics of all of the entities (in the user and reference models) as well as the relationships between them and it can be used to identify and match similar entities used in the user and reference models.

3.2 Case Retrieving and Similarity Assessment

The *Pattern Matching Algorithm* compares the contextual and architecture information of the given user model with that of all of the reference models stored in the *Generic Model Library*. Several types of information is taken into account. Do these models describe a similar business area? Are they capturing similar concepts? Do they follow similar business rules? The contextual and architecture information is stored in the business area, *view*, links, *dependencies*, and in the *entities*.

Provided with knowledge embedded in *ECH*, one can now match views, dependencies and entities to determine if two different models are sufficiently similar. To match entities, for instance, entities which have the same name in both user and reference models produce a positive match. However, similar but variant entities (sibling relationships in the *ECH*), or "stream-line" specialisations (e.g. parent and child, or grandparent and grandchild relationships) may also produce a positive match. When deciding which is a better match between entities, the closer the relationship is between the two entities on the *ECH*, the better quality of a match it is.

We identified five features which can be used to quantify the quality of a match: the matching result of the captured business areas, the matching ratio of links (dependencies) in the selected reference model, the matching ratio of entities in the selected reference model, the matching ratio of links (dependencies) in the user model and the matching ratio of entities in the user model.

```
HEURISTIC SIMILARITY ASSESSMENT FUNCTION

Given two matches, X and Y
if match-view(X) > match-view(Y) then SELECT X
else if match-view(X) = match-view(Y)) and
        match-data-link(X) > match-data-link(Y) then SELECT X
else if match-view(X) = match-view(Y)) and
        match-data-link(X) = match-data-link(Y) and
        match-data-entity(X) > match-data-entity(Y) then SELECT X
else if match-view(X) = match-view(Y)) and
        match-data-link(X) = match-data-link(Y) and
        match-data-entity(X) = match-data-entity(Y) and
        match-case-link(X) > match-case-link(Y) then SELECT X
else if match-view(X) = match-view(Y)) and
        match-data-link(X) = match-data-link(Y) and
        match-data-entity(X) = match-data-entity(Y) and
        match-case-link(X) = match-case-link(Y) and
        match-case-entity(X) > match-case-entity(Y) then SELECT X
else SELECT Y
```

Figure 4: The Heuristic Similarity Evaluation Function

Based on these features, Figure 4 shows the heuristic evaluation method provided by *GMA*. This method produces good results using our test data (see Section 4). Alternatively, the user can dynamically design their own evaluation methods if they wish to explore specific aspects of models in the library.

4 Evaluation

For evaluation purposes, we obtained a variety of *BSDM* models form widely different domains. Part of a real industrial model which was developed by an international automobile company.[1] One generic business model for small and medium-sized restaurant was developed based on interviews of three independent family restaurant (ex-)owners to enlarge our testing base. We also captured example and standard models from *BSDM* and stored them in our *Generic Model Library*. In total, the library contains about a dozen models described in 15 different *views*, represented in 25 diagrams.

The evaluation was concerned with the following issues: (1) to which extent can the tool provide a starting point to help build a new model; (2) how capable is the tool in helping detect model errors by retrieving the appropriate reference models; (3) how well can the system help to retain new knowledge and store it for future reuse. In other words, we are interested in determining how well the tool can help to speed-start model building, encourage good modelling practice and accumulate model building knowledge.

Althoff *et al* [1] proposed a useful evaluation framework to test both the theoretical and practical aspects of a *Case-Based Reasoning* system. Adapting their method, four types of tests were designed and carried out. Firstly, by giving only very little information, a test was carried out on *GMA* to determine if it can provide any useful assistants by retrieving similar models. Secondly, to test the capability of *GMA* to cope with "noisy" models, pre-determined portions of data were deleted from the original models which were then used as input for GMA. The result was used to compare with the expected (i.e. perfect) result when the original model was used.

Thirdly, the above automobile industrial model was used as the user-defined model. Since the automobile model was developed independently by and for a real business, it would be a good testing vehicle to demonstrate if *CBR* techniques can be used to contribute to general business model building exercises. The intention was also to determine whether or not *GMA* could retrieve similar cases from the library, given sufficiently different model architecture and entity names.

One vital step for a *Case Based Reasoner* is in its ability to retain and reuse new knowledge. Therefore, the final test was to use *GMA* as a modelling tool to develop, generalise, verify and validate (with the help of *KBST-BM*) and retain a business model, and then export it from the *Generic Model Library* as a new model.

The results obtained demonstrated that even when provided with only partial and noisy models, the system was still able to retrieve all relevant reference cases where they existed in the library. We also observed that the matching result was largely influenced by the matching of the view name of the data model. However, in the absence of a matching view name, *GMA* still retrieved good matching cases from the library. In fact, out of the 10 different tests and 29 different sets of data, all of the tests successfully retrieved the best and good matching cases.

Although the above tests are encouraging, it is possible to produce scenarios where the system may not produce similarly successful results, i.e. instead of using a correct partial model, it gives an erroneous model containing vital mistakes. For example, when a business model uses an entirely wrong view name or a business model which is grossly misrepresented. When the input model is given in such a way, it will misguide the system to believe that it is more similar to another reference model, hence the retrieval case will probably be incorrect. We, however, believe that the modellers normally have sufficient judgement not to make such vital mistakes.

[1] The company wishes to keep its identity confidential.

During the third test, i.e. given an input model with significant different architecture and entity names, *GMA* was also able to retrieve all of the similar reference models for it, and present them in a reasonable order of preference. The testing result showed that although some of our cases in the library are much less complicated and smaller in scale and most of them indeed describe a different domain of business, useful similarities (in the same business areas across sectors) are still being identified using *GMA*. This also demonstrated the fact that at this level of abstraction common practices are exhibited in different business environments and can be reused.

KBST-BM integrates with *GMA* together provide a more complete framework for *CBR*, i.e. automatic indexing input data, retrieving relevant cases from library, comparing and analysing input with selected cases, revising cases for current problem, verifying and validating input, and retaining the new inputs for future reference. This allows us to use the larger *KBST-BM BSDM* modelling environment in the adaptation phase of the *CBR* cycle. We tested this route using the automobile and restaurant models.

5 Conclusion

Successful business model development requires both methodological and application domain knowledge and experience. Unfortunately, few people possess all of these capabilities. Our studies of applying CBR and Rule-Based techniques which are based on a coherent underlying formal method shows how model building knowledge can be obtained, reused and used to provide automatic verification and validation facilities. We believe that with this support we are able to enhance the level of knowledge sharing, and ability of problem solving. More importantly, it adds to our understanding of how this sort of seemingly informal method can fit into parts of the design lifecycle which require formal models.

References

[1] Klaus-Dieter Althoff, Eric Auriol, Ralph Barletta, and Michel Manago. *An AI Perspectives Report: A Review of Industrial Case-Based Reasoning Tools.* An AI Perspective Report. AI Intelligence, P.O.Box 95, Oxford OX2 7XL, 1995.

[2] Yun-Heh Chen-Burger, Dave Robertson, and Jussi Stader. *Knowledge-Based Automatic Verification and Validation for Business Models. DARPA-JFACC Symposium on Advances in Enterprise Control*, November 1999.

[3] Yun-Heh Chen-Burger, David Robertson, and Jussi Stader. *Formal Support for an Informal Business Modelling Method. Special Issue for The International Journal of Software Engineering and Knowledge Engineering*, 1999.

[4] IBM United Kingdom Limited, 389 Chriswick High Road, London W4 4AL, England. *Business System Development Method: Business Mapping Part1: Entities*, 2nd edition, May 1992.

[5] Martin King. *Knowledge Reuse in Business Domains Experience with IBM BSDM.* Technical report, Artificial Intelligence Application Institute, 1995.

[6] Janet Kolodner. *Case-Based Reasoning.* Morgan Kaufmann Publishers, Inc., 2929 Campus Drive, suite260, SanMateo, CA, USA, 1993.