Model Management Research:

Summary of NSF IDM Workshop Breakout Session

September, 2003
Seattle, Washington

Philip Bernstein and Linda Shapiro[1]

[1]With contributions from Leonidis Fegaras, Ed Fox, Ed Robertson, Rick Snodgrass, Jianwen Su, Mike Uschold, Kang Zhang, Peter Mork, and others.

1 Background

Many important information systems problems primarily involve the manipulation of structural meta data, that is, models (e.g. schemas, interfaces, web-site maps, etc.) and mappings between models. Examples include schema evolution, XML message translation, application integration, data warehouse loading, database wrapper generation, and design tool implementation. Despite the similarity of solutions to these problems, today they are solved in an application-specific way and usually require much object-at-a-time programming.

The goal of model management is to develop a generic infrastructure that offers a major productivity improvement to builders of such model-driven applications. The main abstractions are models, metamodels, and mappings between models. The model management infrastructure treats these abstractions as bulk objects and offers operators for generating models, matching models, finding differences between models, merging models, composing mappings, and extracting sub-models. A generic implementation of these abstractions and operators would greatly reduce the effort required to build solutions for the information system applications listed in the previous paragraph. The research agenda for reaching this goal was the main subject of this breakout session.

2 Model Management Problems

Our group discussed model management in the contexts of Mathematics, Science, and Engineering, looking for the challenging problems in each.

2.1 Mathematical Problems

Mathematical problems are the formal or theoretical side of model management. We identified four major topics:

1. formalisms for structures
2. formalisms for application problems on these structures
3. reasoning about problem tractability
4. determining complexity of tractable problems

The major structures in model management are models, metamodels, and mappings between models. There are many different ways of representing these structures, depending on the application. To many database people a model is a relational schema. In the knowledge-representation community, a model might be a frame definition. For the general computer science population, a class definition, as used in object-oriented programming, is a reasonable a model. In many scientific applications, an attributed graph is the standard representation for a model. Metamodels also have many different representations. For example, a metamodel may be represented as a schema (e.g., the SQL catalog schema), a type definition (e.g., a definition for entity-relationship models expressed in UML), or as a grammer (e.g., the definition of Document Type Definitions for XML). Mappings are well-defined in mathematics, but there are many different kinds of mappings, ie. mappings between sets, mappings between groups, mappings between graphs.

Adding applications to the picture adds in the operations needed by these applications. Operations include such things as mapping generation, model integration, model translation, and change propagation. These and other common operations all must be enumerated and formally defined. Such vehicles as formal logic, formal languages, and graph grammars exist at this level. Again, a unified formalism is needed.

Once structures and operations have been formally defined, the related algorithms can be defined. However, in this stage, we have to worry about tractability. In particular, we need to know if an operator is computable at all or whether a set of operators is complete. Furthermore, we need to consider possible loss of information in certain operations. For query systems, we would like mappings between structures and queries on structures to be composable operations. Similarly, if a structure S1 is defined by its mapping to another structure S2, we would like updates on S1 to be translatable into updates on S2.

Once a set of operations is determined to be tractable, we can work on algorithm development, which brings in the need for time and space complexity analyses.

2.2 Scientific Problems

Scientific problems are those for which a hypothesis can be defined and experimentally tested. Most of the scientific studies we envision will be related to human factors. In particular, HCI specialists will want to determine people's cognitive strengths and their limitations when dealing with graphical and textual descritions of models, mappings, and query languages. A model may be very well defined in the formal sense, but it also has to work for real users.

2.3 Engineering Problems

Much of what we do in database systems and information retrieval is engineering. Developing the computer environments in which our tools will be used is a major component. Computer-aided design environments, graphical and textual languages are among the possibilities. Another engineering task is the semantic definition and implementation of model-management operators. In building a new system, we would like to be able to have some built-in models upon which to build. The development of reusable ontologies would be a good step in this direction. Once a system exists, we have to worry about version and configuration management. In all cases, we must leverage the strengths of the methodology and compensate for its weaknesses.

Metamodel engineering is, in itself, an interesting area. Metamodels exist in some domains, such as digital libraries and image understanding; can we use them? Given an application, how do we go about designing a metamodel into which all components of the application can fit? Once we have the metamodels and requirements for an application, how do we go about generating a full system model? Can this be done automatically? All of this implies that a formal syntax and semantics for metamodels will be needed.

3 Important Directions for Research

Given that the use of models is important in many phases of database research, the following research questions need to be answered.

- What should be in a model? Some possibilities include structure, relationships, processes, user goals, business rules, data dynamics, information flow, invariants, provenance, information quality.
- What are the right formalisms to represent models and metamodels (i.e. templates for models)? How do these choices help simplify the problem of integration?
- How are models created? How can they be reverse engineered from implemented systems?
- How do we evaluate the quality of mappings, such as whether it is easy to integrate with other models? How can we tell if one is better than the other?
- What requirements should be imposed on the structure of reusable models (a.k.a. schemas, ontologies)? Such requirements could become mandates of standards bodies. What are the best methodologies to develop reusable models and apply them to a design task?

- How does one identify submodels that are stable with respect to system changes?

For most scenarios, we need to generate and maintain mappings between models. The following are some of the relevant research questions related to mappings:

- How are mappings expressed? E.g., query languages, formal grammars, graph grammars.
- How are mappings created? How can they be reverse engineered from implemented systems?
- Under what circumstances are mappings possible? When can it be done accurately? Losslessly? When can it be generated automatically?
- What are the theoretical properties of mappings in a given formalism, such as computability, computational complexity, finite representation, invertability, and composability? How do these properties affect the design of integrated systems?
- How does consistency among models influence the design of mappings? What kinds of consistency matter and how can they be captured by mapping formalisms?
- How do we evaluate the quality of mappings? How can we tell if one is better than the other?
- How are the above issues affected by the use of different representation formalisms (i.e. meta-models) to describe the systems to be integrated?

Information dynamics affect the design and design process of integrated information systems. Models and mappings change over time, due to changing user-oriented and technical requirements. The following are some of the research questions related to change management:

- When is it appropriate to incrementally propagate a change through existing models and mappings vs. regenerating them?
- How should incremental propagation and regeneration be done?
- Given an update to a model or an instance of a model, can you check that the update propagation program is consistent with the mapping?
- How can the update propagation program be automatically generated from a given change to some models and mappings?
- How do you maintain the consistency of models, mappings, and instances in the face of such changes? What is the appropriate degree of consistency for different scenarios?

To help designers, tools need to include algorithms to generated and maintain models and mappings. Under tools, the following research problems need to be solved.

- What new or enhanced tools would make integration less expensive and more effective?
- What are people's cognitive strengths and limitations when dealing with graphical and textual descriptions of models and mappings?
- Given these strengths and limitations, what are the best ways for designers to view and manipulate models and mappings? How large do they scale in number and size of models and mappings? How do these visual metaphors translate into formal expressions.
- What technology would make it easier for groups to collaborate on designing solutions to integration problems? E.g., what version and configuration management mechanisms would help to manage these evolving designs?
- How can validated mappings be used to help simplify integration tasks? E.g. How do you create and use links between multiple standard ontologies?

Finally, information system design often involves the collaboration of designers, users, developers, and others. In the area of design, the following are research questions that must be answered.

- What processes of collective design will yield faster or clearer consensus and a more appropriate structure?
- What processes or strategies will make locally-designed models, ontologies or structures emerge with greater consistency?
- Every technology has methodologies embodied in it. What kinds of collaboration are implied by each technology? How do these kinds of collaboration work when many technologies, and hence many methodologies, are involved?
- How can community-based processes be used to solve IS integration problems? Communities may be defined by clusters of decisions to be made.

4 Recommendations

We recommend that the National Science Foundation support basic research in the area of model-management to help answer the above research questions. We also recommend funding research that applies related technology such as machine learning, data mining. symbolic reasoning, information retrieval, natural language processing, and combinatorial optimization to help solve model-management problems. We believe that there is a need for the collection of good test cases from various applications, for extensible toolkits, for university-industry collaboration, and for standards work. This is a new area of research, and our group is excited about the many possibilities.