Query-Driven Model Building In Enterprise-Wide Decision-Making Environments

Karl R. Lang* and Andrew B. Whinston**

*Institut für Wirtschaftsinformatik und Operations Research
Freie Universität Berlin
Berlin, Germany;

**Center for Information Systems Management
Department of Management Science and Information Systems
The University of Texas at Austin
Austin, Texas.

1. Introduction

In order to respond to new challenges in an increasingly complex and dynamic environment, modern management is using a vast amount of knowledge from various sources. Depending on the particular problem being investigated, managers switch between different perspectives and levels of detail when searching for the relevant pieces of knowledge required to provide an appropriate answer. However, lacking a centralized knowledge management facility, individual managers' access to knowledge is restricted to a relatively small subset of the collective organizational knowledge, depending on their status and function within the organization. This may inhibit the recognition of the interactions and interdependencies relevant to the problem under study.

Conceptually, we are looking for an enterprise modeling system (EMS)[1] which automatically builds and executes task-specific models as needed in response to queries posed by the user. The focus of this paper is on how we can improve model building and how we can extract the relevant parts of models to support specific analyses of enterprise-wide corporate issues. We discuss and propose

---

[1] We use the term enterprise modeling in accordance with the definition provided in Petrie (1992), p.19, where enterprise is defined as "a collection of business entities ... in functional symbiosis," and thus differs from the usage in the business re-engineering area. Business entities mean organizational (sub)units and (groups of) people and functional symbiosis refers to the interactions among a set of intraorganizational as well as interorganizational entities sharing a common goal. Hence, the scope of enterprise modeling explicitly includes external partnerships like relationships of an organization with its suppliers, subcontractors, customers, and the public. Some authors, for example Carter et al (1992) on p.4, use the term organizational decision support system to describe concepts similar to our notion of enterprise modeling.

ideas which we see as promising steps towards accomplishing this difficult endeavor. There are two, essentially disjoint, research efforts, one based in the artificial intelligence community and the other in the decision support systems community, which study model building and reasoning with multiple models. This paper draws upon both of these efforts and develops a synergistic framework for future enterprise modeling systems.

We envision a system whose reasoning about a particular organization is based upon a library of model components (or model fragments) representing significant organizational phenomena from different perspectives and at different levels of detail. Accomplishing this requires access to multiple sets of heterogeneous model fragments which differ in several dimensions, some of which might even be mutually inconsistent. We need to address the issue of model representation and model organization. That is, we need a language for expressing relationships of different kinds and for expressing underlying assumptions controlling and guiding their applicability.

Researchers in the DSS and Artificial Intelligence (AI) communities have proposed several frameworks which provide partial solutions to this formidable problem. Model management in the DSS field can be seen as a natural extension of previous work in management science and operations research. It has advanced mathematical modeling from a state where modeling was an uncoordinated task, whose success depended mainly on the technical skills and expertise of the user, to a state where systems actually know about certain types of mathematical models and appropriate solvers [Geoffrion (1987), Liang (1988), Mannino et al (1990), Krishnan and Westernberg (1991), Muhanna and Pick (1992), Basu and Blanning (1993), Dolk and Kottemann (1993), Raghunathan et al (1993)]. The emphasis of AI research has been put more on the issue of the explicit representation of modeling assumptions, and the usage and exploration of qualitative knowledge, and less on model integration and in particular on solver integration [De Kleer and Brown (1984), Kuipers (1988), Addanki et al (1991), Falkenhainer and Forbus (FF) (1991), Rickel and Porter (1992), and Weld (1992)].

2. A Framework for Query-Driven Enterprise-Wide Modeling Systems

We propose a model building strategy which builds models as needed in response to user queries. Given a query, the model formulation problem can be defined as selecting the relevant model fragments and generating a composite, task-specific model which is coherent and useful in answering it. Using different sets of assumptions and various kinds of knowledge ranging from general, qualitative knowledge to specific and precise numerical models, managers analyze organizational questions from different perspectives and at different levels of detail. Given a particular task, model building is guided by the selection of an appropriate perspective and level of detail, a modeling decision for which little

support is found in current decision support system technology. When modeling a certain organizational phenomenon, it is crucial to focus on the relevant aspects of the situation under investigation, that is, to include all the relevant objects and constraints, but also to exclude irrelevant ones and ignore unnecessary details. We suggest the software architecture depicted in figure 1 for designing such an EMS.

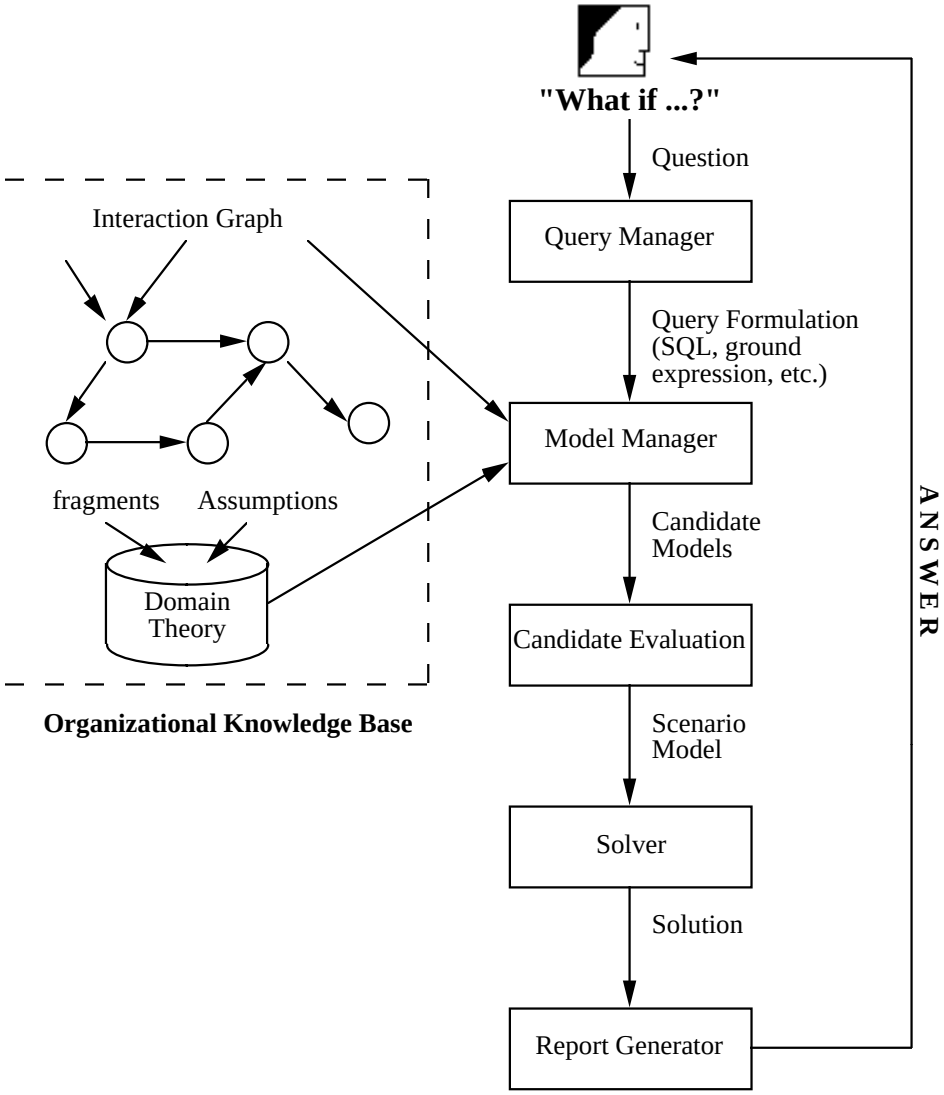

Figure 1: EMS Software Architecture.

The EMS is designed as an interactive software tool which supports decision making and problem solving when exploring various business scenarios. It comprises five functional modules: The query manager, the model manager, the candidate evaluation module, the solver, and the report generator . The query manager provides the interface between the EMS and the user, typically an organizational decision maker or a technical assistant to one. It processes user's queries such as, "How does an increase in price affect net income?" and translates them into a set of executable statements which are submitted to the model manager.

The core of the EMS is the model manager which controls access to models and data in the organizational knowledge base. The enterprise modeling framework requires first the building of a general-purpose organizational knowledge base that describes a variety of organizational objects, activities, and processes. The domain theory is represented as a library of model fragments, each describing an independent aspect from a particular viewpoint. It contains general organizational laws and rules as well as relationships that are very specific to a particular company. The organizational knowledge described in the domain theory could be obtained from research results in the organizational behavior field, which tries to formulate theories about organizations in general, that is, to find relationships that help understand the behavior of a wide variety of organizations. Since those relationships are supposed to hold for any particular organization of the class, they tend to be very qualitative in nature. Organization-specific information, on the other hand, is derived from historical data and experience accumulated within a particular company, and therefore, tends to be much more precise. This information is often encoded in a quantitative, management science/operations research (MS/OR) type of model like optimization, simulation, or forecasting models. The explicit representation of modeling assumptions in terms of abstraction level, approximation, perspective, level of detail, and granularity is another essential feature in enterprise modeling. Reasoning about those assumptions enables the EMS to identify a suitable collection of compatible model fragments and to build consistent, composite models in response to a query. Typically, there is no unique composite model and the model manager might find several feasible models, called candidate models, and passes each of them on to the next EMS module.

The candidate evaluation module then collects all candidate models and chooses the best candidate as the final scenario model. In this context, best means the simplest possible model that is coherent, comprehensive, and appropriate for the task. The solver module selects the adequate solution method and then solves or simulates the scenario model chosen by candidate evaluation. Finally, a report generator is employed as a post processor in order to translate the model solution, that is, the output of the solver, into an intelligible answer which can be presented to the user in return to the original question.

3. The Organizational Knowledge Base

In this section, we discuss the organizing principles of the organizational knowledge base (OKB) underlying our enterprise modeling framework. We view the OKB as a repository of organizational knowledge whose purpose is to provide a resource of sharable and reusable model pieces for helping to better understand, explain, and predict organizational phenomena in a variety of different situations. In order to achieve the necessary depth and versatility, the OKB needs to contain knowledge of different types: (i) relationships among organizational variables encoded as quantitative or qualitative constraints; (ii) their preconditions and associated modeling assumptions that define the presuppositions under which they hold; and (iii) knowledge about knowledge expressed as metarules which relate modeling assumptions to each other.The observation that a model consists of more than just a set of relationships, because a model always assumes a particular modeling context, leads us to a definition of an EMS model component (or model fragment) where the modeling assumptions are explicitly and separately expressed from the actual relationships. Each model fragment has two sections, one contains the specification of modeling assumptions (conditions section) and the other (relations section) contains the actual constraints and relationships that apply if the model assumptions hold. Model fragments are essentially of the form

fragment <NAME> (input port) (output port)
    {verbal description of the functionality of the model fragment}
conditions
    precondition-specifications
relations
    relationship-specifications
end

where <NAME> is an identifier of a particular model fragment instance, input port is a list of the variables whose values need to be provided, either by computing them in other model fragments or by importing them as exogenous quantities, output port is a list of the variables which are computed by this model fragment, and which can be shared with other fragments. The conditions section contains precondition-specifications, which define the modeling assumptions that an instantiation of a model fragment depends on. Lastly, the relations section contains relationship specifications, which would be constraints of a particular modeling language. We only assume that internally, that is, within a single model fragment,

the relationships are of a homogeneous type. Across model fragments, heterogeneous relationship specifications are permitted by using several modeling languages.

Before a model composition algorithm can actually search the model base and identify task-specific, relevant model fragments, it needs sufficient information to be able to evaluate the predicates in the model assumption section. This extra information needs to be either derived directly from the query or inferred from meta knowledge present in the OKB. Meta knowledge is to be specified separately from the model fragments as a set of rules. These meta rules express integrity constraints which rule out incoherent and inconsistent combinations of modeling assumptions, and also imply additional conditions as a consequence of modeling assumptions that have been already established.

Now, on the next couple of pages, we give an example that illustrates how the EMS principles discussed above apply to the development of an OKB model. For our purpose, showing just a small segment of an OKB shall be sufficient to demonstrate the essential features of an OKB. The complete enterprise description would obviously be much more elaborate. For the sake of simplicity, we have left out some details, and included some further restrictions in a separate rules section. In particular, rule R-1 limits the model base to quasi-static models, rule R-2 selects QSIM as the only solver for purely qualitative scenario models, and rule R-3 chooses RCR as the only solver for semi-qualitative models[2].

The intelligence of the EMS, however, resides mainly in the interaction graph, shown in figure 2, which represents knowledge about organizational knowledge. It is used in the OKB as a comprehensive hypermodel of the enterprise. More specifically, the interaction graph relates organizational variables, organizational relationships, modeling assumptions, and model fragments to each other. The nodes of the interaction graph represent organizational variables and arcs connecting two nodes indicate the existence of a relationship between the two corresponding variables. Arc labels identify model fragments containing such relationships. The specification of a relationship cannot be directly obtained from the interaction graph, but must be retrieved from the relations section of the containing model fragment. Likewise, modeling assumptions are to be found in the conditions section of the identified model fragment. Finally, self-loops, that is, arcs which leave from and return to the same node, indicate that the corresponding variable could be treated as being exogenous.

In the lower left corner of figure 2, we can see, for example, that model fragment f2 contains a relationship among the variables usage of information technology (IT)

---

[2] QSIM and RCR are two modeling languages which allow the user to represent purely qualitative and semi-qualitative information; see Hinkkanen et al (1995) for a detailed discussion.

and Productivity (Prd). This means that if we want to build a model which predicts or explains the value of productivity, we need to consider fragment f2 as a potential building block. The actual specification of the relationship and its associated modeling assumptions represented by the arc <IT-Prd> can be looked up in the definition of fragment f2, which is shown below. In this case, we find the monotonic relationship Productivity = M^+(IT), which holds if the four modeling assumptions

OntologyAssumption=influences, SimplifyingAssumption= qualitative, OperatingAssumption=quasi-static, and TimeScaleAssumption= medium are satisfied.

OKB CORPX

ALIASES

/Partnership, Pship/
/Product_Quality, PQual/
/Customer_Satisfaction, CSat/
/Customer_Service, CSrv/
/Marketing_Position, MPos/
/Promotional_Expenditure, PrmExp/
/Productivity, Prd/
/Information_Technology, IT/
/Revenue, Rev/
/Net Income, NInc/
/Production Cost, PCost/
/Performance, Perf/
/Goodwill, Gw/

END

ASSUMPTION CLASSES

/Ontology Assumption, OntAss/ (influences, cash flow, material flow);

/Simplifying Assumption, SimpAss/ (qual, semi-qual, quant);

/Operating Assumption, OpAss/ (static, quasi-static, dynamic);

/Time Scale Assumption, TScAss/ (short, medium, long)

END

fragment f1 (IT) (Pship)

{qualitative model describing the relationship between IT and Partnership}

conditions

OntAss=influences, SimpAss=qual, OpAss=quasi-static, TScale=medium

relations

Partnership = M^+(IT)

end

fragment f2 (IT) (Prd)

{qualitative model describing the relationship between IT and Productivity}

conditions

OntAss=influences, SimpAss=qual, OpAss=quasi-static, TScale=medium

relations

Productivity = M^+(IT)

end

fragment f3 (Prd) (Pship)

{qualitative model describing the relationship between Productivity and Partnership}

conditions

OntAss=influences, SimpAss=qual, OpAss=quasi-static, TScale=long

relations

Productivity = M^+(Partnership)

end

. . . .

. . . .

fragment f18 (Price) (Sales)

{marketing model describing the qualitative relationship between Price and Sales volume}

conditions

OntAss=influences, SimpAss=qual, OpAss=quasi-static, TScale=medium

relations

Sales = M^-(Price)

end

fragment f19 (Price) (Sales)

{marketing model describing the semi-quantitative relationship between Price and Sales volume}

conditions

OntAss=cash_flow, SimpAss=qual-quant, OpAss=dynamic, TScale=short

relations

Sales(t)=[68000,92000]+[40000,48000] *Price(t)

end

fragment f20 (Price) (Sales)

{marketing model describing the quantitative relationship between Price and Sales volume}

conditions

OntAss=cash_flow, SimpAss=quant, OpAss=quasi-static, TScale=short

relations

Sales = 80000 - 44000*Price

end

fragment f21 (Sales) (Rev)

{accounting model describing the qualitative relationship between Sales volume and Revenue}

conditions

OntAss=influences, SimpAss=qual, OpAss=quasi-static, TScale=medium

relations

Revenue = M^+(Sales)

end

fragment f22 (Price) (Rev)

{accounting model describing the qualitative relationship between Price and Revenue}

conditions

OntAss=influences, SimpAss=qual, OpAss=quasi-static, TScale=medium

relations

Revenue = M^+(Price)

end

fragment f23 (Price, Sales) (Rev)

{accounting model describing the quantitative relationship between Price, Sales volume, and Revenue}

conditions

OntAss=cash_flow, SimpAss=quant, OpAss=quasi-static, TScale=medium

relations

Revenue = Price*Sales

end

fragment f24 (Rev) (NInc)

{accounting model describing the qualitative relationship between Revenue and Net Income}

conditions

OntAss=influences, SimpAss=qual, OpAss=quasi-static, TScale=medium

relations

NetIncome = M^+(Revenue)

end

fragment f25 (Cost) (Price)

{financial model describing the qualitative relationship between Cost and Price}

conditions

OntAss=influences, SimpAss=qual, OpAss=quasi-static, TScale=medium

relations

Price = M^+(Cost)

end

fragment f26 (PCost) (Cost)

{financial model describing the qualitative relationship between Production Cost and Total Cost}

conditions

OntAss=influences, SimpAss=qual, OpAss=quasi-static TScale=medium

relations

Cost = M^+(ProductionCost)

end

fragment f27 (Cost) (NInc)

{financial model describing the qualitative relationship between Cost and Net Income}

conditions

OntAss=influences, SimpAss=qual, OpAss=quasi-static, TScale=medium

relations

NetIncome = M^-(Cost)

end

fragment f28 (Cost, Rev) (NInc)

{accounting model describing the quantitative relationship between Cost, Revenue, and Net Income}

conditions

OntAss=cash_flow, SimpAss=quant, OpAss=quasi-static, TScale=medium

relations

NetIncome = Revenue - Cost

end

fragment f29 (Perf) (Gw)

{marketing model describing the qualitative relationship between Performance and Goodwill}

conditions

OntAss=influences, SimpAss=qual, OpAss=quasi-static, TScale=medium

relations

Goodwill = M^{+}(Performance)

end

.

.

.

rules

R-1: OpAss(quasi-static)

R-2: SimpAss(qual) => solver(QSIM)

R-3: SimpAss(qual-quant) => solver(RCR)

R-4: ...

.

.

.

end

Hence, if we are building a qualitative model describing, among other things, the impact (or influence) of IT usage on productivity, we must consider the inclusion of fragment f2 in the composite scenario model to be built.

In general, arcs emanating from a node x indicate the variables directly influenced by variable x. Thus, usage of IT has, in our enterprise model, a direct impact on Partnership, Productivity, and Customer Service. However, besides the direct influence of IT on Productivity, there is also an indirect influence of IT on Productivity, via Partnership. Indirect influences are represented in the interaction graph as a sequence of arcs called an *interaction path.* Here, the sequence <IT-Pship>-<Pship-Prd>, or more compactly written as <IT-Pship-Prd>, expresses the indirect influence of IT on Productivity. Similarly, IT has many more indirect influences on other variables, for example, the interaction paths <It-Prd-Perf-Gw> and <IT-CSrv-CSat-Gw> represent alternative possibilities of modeling the indirect influence of IT on Goodwill. Incoming arcs of a node x represent the direct influences on variable x. Our example indicates that Productivity is directly influenced by IT usage and Partnership. However, IT has a self-loop as the only incoming arc. The arc going from node IT back to itself means that the only influence on variable IT is IT itself, in other words, IT cannot be explained within the enterprise model. IT has to be determined outside of the model, that is, IT is treated as an exogenous variable whose value needs to be imported from a separate database when IT is included in a scenario model. Exogenous variables are typically variables which are, at least to some extend, controllable. The level of IT, for example, is determined by the budget proposed and passed by the management.

An arc label actually consists of a list of fragment identifiers. Such a list may be empty, as in the case of arc <IT-IT>, indicating an exogenous variable; may contain one identifier, as in <IT-Prd> meaning that the OKB knows only about one relationship between IT and Prd; or it may contain several identifiers suggesting alternative relationships. Two examples of multiple relationships are, first, arc <Price-Rev> which lists two fragments, f22 and f23, both using a relationship between price and revenue, and, second, arc <Price-Sales> which names three alternatives, fragments f18, f19 and f20, of modeling price and sales. Relationships involving more than two variables are identified by any of the participating variables. For example, fragment f28, which specifies a relationship between three variables net income (NInc), cost (Cost) and revenue (Rev), must be instantiated when either of the two arcs <Cost-NInc> and <Rev-NInc> is considered.

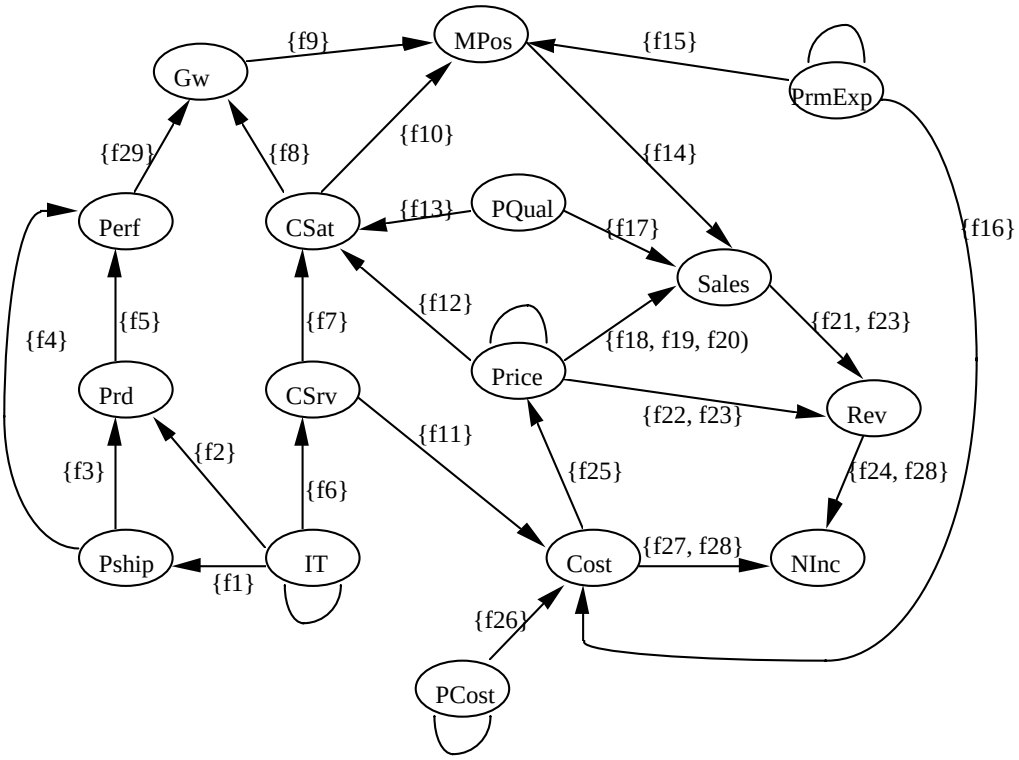

Figure 2: *Interaction Graph of the CORPX Organizational Knowledge Base.*

4. Model Composition

We refer to the real-world phenomenon under study as a scenario, and to the model representing it as a scenario model. Selecting the right model pieces to compose an appropriately integrated model for answering a given query requires modeling decisions along several dimensions. What is the best set of variables to be included in the model? What level of detail is appropriate? Which are the relevant organizational phenomena for studying the posed question? From what perspective should the problem be viewed? What kinds of approximations and abstractions should be allowed? Even the most carefully organized model fragment library won't provide enough information to find an answer to all of these questions. Therefore, we need to derive missing pieces of information from the query itself, that is, we need to look for clues provided in the query that could narrow the focus of the model composition process and reasonably constrain the set of plausible modeling assumptions.

As an example of composing a scenario model in response to a prediction question, let us suppose the user entered the query "How does an increase in price affect net income?" Conceptually based on natural language processing, a query elaboration

procedure would analyze the issued query and derive from it a set of ground expressions which would be passed on to the model manager module of the EMS for evaluation. In the absence of such a sophisticated query analyzer, we could simply devise a primitive query language which basically lists a number of ground expressions which permit the system to identify objects, quantities and relations of interest, where each of these has a referent in the organizational knowledge base. Hence, let us consider the simplified query {increase(Price), quantity(NetIncome)}, whose ground expressions increase(Price) and quantity(NetIncome) provide the input to the model manager.

The query indicates that we need a scenario model which computes net income. While the one ground expression quantity(NetIncome) of the query hints neither to a qualitative nor to a quantitative modeling approach, the other ground expression does provide a clear clue for a qualitative analysis. Since the increase operator indicates a desired direction of change without further specification, it suggests a qualitative model for investigating this effect on net income in the given scenario. Now, we could try to enumerate all possible combinations of model fragments, and to prune out those which either violate some of the modeling assumptions or prove to be irrelevant or insufficient regarding the query. However, this approach would be computationally too costly, considering the many combinations of model fragments, which would typically occur in an enterprise-wide environment. The number of modeling assumptions, on the other hand, tends to be much smaller, and therefore suggests a computationally better alternative by reasoning about combinations of modeling assumptions first, and then to select and integrate a suitable set of model fragments. We will come back to this computational issue below and discuss it in more detail.

Model composition begins with the derivation of an initial set of quantities of interest from the query. Quantities of interest correspond to variables which need to be included in the scenario model to be composed. In our example, we would take the query {increase(Price), quantity(NetIncome) and derive {Price, NetIncome} as the set of initial quantities of interest. The quantity operator in the second ground expression, {quantity(NetIncome)}, provides a hint that the value of the variable NetIncome is desired, which means that we are supposed to model a scenario wherein NetIncome is predicted. Variables to be predicted by a scenario model are called goal variables. The increase operator in the first ground expression, increase(Price), describes a manipulation to be performed on a variable. In this case we are supposed to (qualitatively) change the current value of price and then examine the effect of this change. Variables like Price which are to be changed initially in a way prescribed by manipulation operators in the query are called driving variables. Driving variables are used to drive the model building process by trying to establish a connection between them and the goal variables such that the values of the goal variables can be determined if an initial state

description is given. In the example, we would try to build a model that computes NetIncome from Price. In order to accomplish this job, the EMS employs a compositional modeling approach which searches the OKB for relevant model fragments which then can be used to construct an appropriate model.

Before invoking the model composition process, we need to define a modeling environment that selects a set of modeling commitments that are appropriate for the given query. Most importantly, this includes the selection of query-consistent modeling assumptions. In general, this is an indeterminate task. Ideally, we would choose the most suitable option along each of the modeling dimensions. However, we cannot expect that the query presented by the user conveys enough information to make indisputable decisions for all modeling assumption classes. For the sake of simplicity, let us suppose, in the example of this paper, that our query analyzer derives a uniquely determined modeling environment[3]. More specifically, we assume that

(i) the encounter of the generic increase in the query implies a purely qualitative analysis,

(ii) a qualitative analysis requires an ontological commitment to view the interactions in the enterprise as general influences between organizational variables,

(iii) the organizational processes involving price and net income work at a medium time scale, and

(iv) we restrict the analysis to quasi-static models (at least for now).

Hence, we would select

```
(OntAss=influences, SimpAss=qual, OpAss=quasi-static, TScale=medium)
```

as the current set of modeling assumptions.

Unquestionably, many queries would lead to ambiguous decisions about those modeling assumptions. On the other hand, it actually would be desirable in such cases that the EMS would generate a scenario model for all plausible modeling environments, that is, for all combinations of modeling assumptions that could

---

3 The user is supposed to have the option to define or redefine the modeling envrironment at any time, and thus can explicitly select particular modeling assumptioms or override modeling assumptions chosen by the EMS.

reasonably be derived from the query. Furthermore, in order to promote user interaction and extend user control with the EMS, the query language should allow the user to explicitly select some of the modeling assumptions anyway. And finally, one of the modeling control parameters could provide a confirmation option that forces the EMS to merely suggest important modeling decisions, and to seek user confirmation before proceeding. This would be especially useful in ambiguous situations in which the user could prevent the system from generating, and subsequently solving, unnecessary or unwanted models.

In the first step, the driving and goal variables are located in the interaction graph depicted in figure 2. Our example has only one of each, the driving variable price and the goal variable net income (NInc). Now, it needs to be checked whether initial values of the driving variables are provided. Initial values could be derived from the query, supplied by an external data base, or computed from other variables. The latter is computationally the most expensive possibility because it entails constructing a more complex scenario model, and is thus eschewed unless the former two fail. Our example query has no clue on the initial value of price. Fortunately, the second possibility applies, because the node representing price has a self-loop. This means that the variable price can be treated as an exogenous variable, that is, its current value can be obtained from an external source. Next, we try to connect the driving variables with the goal variables, that is, we search the interaction graph for interaction paths between driving and goal variables. Looking at figure 2, there are four interaction paths describing different ways of computing net income from price. Each of the four generated interaction paths suggests to make use of a different collection of fragments for building a model that predicts how an increase of price would affect the net income of the CORPX enterprise.

	Interaction Path	**# arcs**	**# nodes**	**# frags**	**# models**
1	Price-CSat-Gw-MPos-Sales-Rev-NInc	6	7	8	4
2	Price-CSat-MPos-Sales-Rev-NInc	5	6	7	4
3	Price-Sales-Rev-NInc	3	4	7	12
4	Price-Rev-NInc	2	3	4	4
	Total				24

Table 1: *Combinations of Different Interaction Paths.*

Potential scenario models, or *model candidates*, differ in their complexity measured in terms of number of variables and number of fragments involved in composing them. From table 1 we can see that the first interaction path relates

seven variables by six arcs identifying eight relationships represented in eight fragments. Since some of the arcs suggest a set of alternative relationships, we can choose from several different combinations of relationships and their associated fragments. As another example, the third interaction path in table 1, Price-Rev-NInc, consists of three arcs <Price-Sales>, <Sales-Rev>, and <Rev-NInc> which suggest the sets $\{f_{18}, f_{19}, f_{20}\}$, $\{f_{21}, f_{23}\}$, and $\{f_{24}, f_{28}\}$ as possible fragments for modeling relationships between respectively price and sales, sales and revenue, and revenue and net income. Thus, any fragment triple

$$\langle F_1, F_2, F_3 \rangle \in \{f_{18}, f_{19}, f_{20}\} \times \{f_{21}, f_{23}\} \times \{f_{24}, f_{28}\}$$

is an eligible candidate for a scenario model, resulting in 3*2*2 = 12 combinations to choose from. Likewise, we can produce yet more candidate models, and represent them as fragment n-tuples where n denotes the number of participating fragments, from the other interaction paths. Specifically, the first interaction path generates four 6-tuples, the second four 5-tuples, and the last four pairs of fragments. All together, table 2 lists a total of 24 candidates to consider when building a model of the scenario described in the given query "*How does an increase in price affect net income?*" Obviously, the number of model candidates varies with every query, and, in more intricate scenarios, can quickly reach an order of magnitude that is hard to manage.

In order to keep the model composition task tractable we would like to avoid a complete enumeration of possible model candidates. Since the final scenario model has to be internally consistent, we must eliminate those candidates from further consideration whose constituting fragment's precondition sections contain contradictory modeling assumptions because this would indicate an incompatible set of model fragments. In our example above, we have hitherto ignored the modeling assumptions which the fragments are based upon. Prior to generating candidate models, we need to check the consistency of modeling assumptions. From the current modeling environment, we obtain the active set of modeling assumptions, `(OntAss=influences, SimpAss=qual, OpAss=quasi-static, TScale=medium)` whereon the building of the scenario model rests.

	IP	**Model Candidate**	**Incompatible Fragments**
1	1	**(f_{12}, f_8, f_9, f_{14}, f_{21}, f_{24})**	()
2	1	(f_{12}, f_8, f_9, f_{14}, f_{21}, f_{28})	(f_{28})

3	1	(f_{12}, f_8, f_9, f_{14}, f_{23}, f_{24})	(f_{23})
4	1	(f_{12}, f_8, f_9, f_{14}, f_{23}, f_{28})	(f_{23}, f_{28})
5	2	**(f_{12}, f_{10}, f_{14}, f_{21}, f_{24})**	()
6	2	(f_{12}, f_{10}, f_{14}, f_{21}, f_{28})	(f_{28})
7	2	(f_{12}, f_{10}, f_{14}, f_{23}, f_{24})	(f_{23})
8	2	(f_{12}, f_{10}, f_{14}, f_{23}, f_{28})	(f_{23}, f_{28})
9	3	**(f_{18}, f_{21}, f_{24})**	()
10	3	(f_{18}, f_{21}, f_{28})	(f_{28})
11	3	(f_{18}, f_{23}, f_{24})	(f_{23})
12	3	(f_{18}, f_{23}, f_{28})	(f_{23}, f_{28})
13	3	(f_{19}, f_{21}, f_{24})	(f_{19})
14	3	(f_{19}, f_{21}, f_{28})	(f_{19}, f_{28})
15	3	(f_{19}, f_{23}, f_{24})	(f_{19}, f_{23})
16	3	(f_{19}, f_{23}, f_{28})	(f_{19}, f_{23}, f_{28})
17	3	(f_{20}, f_{21}, f_{24})	(f_{20})
18	3	(f_{20}, f_{21}, f_{28})	(f_{20}, f_{28})
19	3	(f_{20}, f_{23}, f_{24})	(f_{20}, f_{23})
20	3	(f_{20}, f_{23}, f_{28})	(f_{20}, f_{23}, f_{28})
21	4	**(f_{22}, f_{24})**	()
22	4	(f_{22}, f_{28})	(f_{28})
23	4	(f_{23}, f_{24})	(f_{23})
24	4	(f_{23}, f_{28})	(f_{23}, f_{28})

Table 2: Model Candidates generated from Interaction Paths (IP) using (OntAss=influences, SimpAss=qual, OpAss=quasi-static, Tscale=medium) as the currently active set of modeling assumptions.

Model Candidate	**Model Size**
(f_{12}, f_8, f_9, f_{14}, f_{21}, f_{24})	13
(f_{12}, f_{10}, f_{14}, f_{21}, f_{24})	11
(f_{18}, f_{21}, f_{24})	7
(f_{22}, f_{24})	5

Table 3: Remaining models after eliminating incompatible models. Model size measured in terms of the candidate evaluation function: eval(m)=v+r.

Table 2 shows for each model candidate those fragments that are incompatible because they violate some of the active modeling assumptions. Notice that only four (set in boldface) out of twenty-four model candidates are indeed internally consistent. Therefore, we devise a compositional modeling strategy which reasons first about the consistency of the modeling assumptions before it starts to assemble

composite model candidates. This approach reduces quickly the search space of possible scenario models from 24 to just 4 candidates (see table 3), namely (f_{12}, f_8, f_9, f_{14}, f_{21}, f_{24}), (f_{12}, f_{10}, f_{14}, f_{21}, f_{24}), (f_{18}, f_{21}, f_{24}), and (f_{22}, f_{24}). Thus, our compositional modeling method concludes, for this example

{increase(Price), quantity(NetIncome)} ==> *or* (((f_{12}, f_8, f_9, f_{14}, f_{21}, f_{24}), (f_{22}, f_{24}) (f_{12}, f_{10}, f_{14}, f_{21}, f_{24}), (f_{18}, f_{21}, f_{24})).

5. Candidate Evaluation and Model Integration

After generating a set of feasible scenario model candidates we need to evaluate and order them according to their appropriateness in order to choose the most appropriate one as the final scenario model. Unfortunately, there is no single criteria that would alone by itself describe appropriateness in a satisfying way, which makes it difficult to provide a definition of it that is not arbitrary to some extend and, at the same time, operational. Falkenhainer and Forbus (1991) define the final scenario model as the model candidate which is coherent and most useful. The former criterion requires the scenario model to be consistent with the modeling assumptions to which the EMS committed when exploring the scenario set up by a query. The latter refers to the tradeoff between information cost and significance to the query in terms of sufficiency and minimality (Balakrishnan and Whinston 1991). Sufficiency means that the answer to the query is firstly not only correct but also relevant to the question such that it provides her with the information sought, and secondly that the answer is also satisfactorily detailed and accurate. Minimality, on the other hand, calls for a parsimonious response and forbids elaborate details. In other words, we are looking for the scenario model which is minimal and (a) consistent, (b) valid, (c) relevant, (d) adequately detailed, and (e) adequately accurate.

The candidate evaluation module receives, from the model composition module as its input, a set of scenario model candidates, and produces as its output the final scenario model. First, we need to ensure that the scenario model satisfies restrictions (a) to (e). Fortunately, our compositional modeling method was designed such that it generates only feasible model candidates, that is, candidates which do satisfy the above restrictions. Consistency is accomplished through the explicit reasoning about underlying modeling assumptions during the model building process. We can assume validity of the relationships used in building model candidates because model fragments are only applied if the assumptions stated in their preconditions section hold. We ensure relevance by assuming that our query analyzer identifies quantities of interest correctly, and that only such fragments are considered which the interaction graph relates to a driving variable or a goal variable. An adequate level of detail and accuracy is achieved by assuming that the query analyzer, in connection with user interaction, is able to

derive and establish a proper set of modeling assumptions which include an appropriate description of level of detail and accuracy required in the given scenario.

Now that we are assured that all remaining candidate models are indeed feasible and appropriate in the sense that we can expect them to yield comparably satisfactory answers, we want to select the minimal or simplest one. Let us define simplicity of a model in terms of model size and define an evaluation function $eval(m)$ as a function of number of variables v and number of relationships r, for example as $eval(m) = v + r$. From table 3 we can see that candidate (f_{22}, f_{24}) is chosen as the final scenario model in our example and is passed on to the solver for model execution.

Using candidate model (f_{22}, f_{24}) we obtain

scenario model f22-f24 (Price) (NInc)
conditions
OntAss=influences, SimpAss=qual, OpAss=quasi-static, TScale=medium
relations
Revenue = M^+(Price)
NetIncome = M^+(Revenue)
end

as the final model, the scenario model. This is not yet a completely specified model, one which could be solved as it is. However, it does show the complete specification of the model's constraints, the core part of the final scenario model. Although it contains only valid relationships, a price increase leads to higher revenues and highter revenues have a positive effect on net income, it may not be accurate enough for the user's current analysis. Model (f_{22}, f_{24}) ignores, for example, the influence of price changes on sales which in turn also influence the goal variable net income. Hence, after inspecting the suggested scenario model, the user may opt for rejecting it, and thus force the EMS to search for an alternative scenario model. In our example, the EMS would suggest model candidate (f_{18},f_{21}, f_{24}) as its next scenario model.

scenario model f18-f21-f24 (Price) (NInc)
conditions
OntAss=influences, SimpAss=qual, OpAss=quasi-static, TScale=medium
relations
Sales = M^-(Price)

Revenue = M^+(Sales)
NetIncome = M^+(Revenue)
end

The new, more complex model does represent the indirect effect of price on net income by including sales as an additional variable and by adding another relationship. Depending on the user's intents, it may be beneficial to trade off some model cost (in terms of model complexity) for more accuracy.

7. Conclusion

To conclude, we have presented a novel, comprehensive framework for future enterprise-wide modeling systems in this paper. Enterprise computing is a support tool to achieve organizational goals. We believe that future research in model building and model management for decision support in organizational environments requires more attention to organizational knowledge representation and to related model building and model reasoning research in artificial intelligence. One purpose of this paper is to bring to bear some of the stimulating results obtained from the AI community, and to indicate how they can be incorporated into the DSS research on model building. Among the new features we have proposed, we want to highlight those which, in our mind, map out the most promising future research directions. First, the possibility of both qualitative and quantitative model formulations, which introduces a new level of versatility to organizational model building, and which should widen the scope of computer supported decision tools considerably. Second, the explicit representation of modeling assumptions. And finally, the application of a compositional modeling strategy to automatically build task-specific scenario models, which liberates users from having to specify special modules for controlling the modeling integration process.

References

Addanki, S., R. Cremonini, and J.S. Penberthy (1991) "Graphs of Models." Artificial Intelligence. **51**: 145-178.

Balakrishnan, A., and A.B. Whinston (1991) "Information Issues in Model Specification." Information Systems Research. **2**(4): 263-286.

Basu, A. and R. Blanning (1994) "Model Integration Using Metagraphs." Information Systems Research. **5**(3): 195-218.

Bhargava, H.K. and R. Krishnan (1991) "Reasoning with Assumptions, Defeasibility, in Model Formulation." working paper, The H. John Heinz III School of Public Policy and Management, Carnegie Mellon University.

Bonczek, R.H., C.W. Holsapple, and A.B. Whinston (1981) Foundations of Decision Support Systems, Academic Press.

Carter, G.M., M.P. Murray, R.G. Walker, and W.E. Walker (1992), Building Organizational Decision Support Systems, Academic Press, San Diego, CA.

De Kleer, J., and J.S. Brown (1984) "A Qualitative Physics Based on Confluences." Artificial Intelligence. **24**: 7-83.

Dolk, D. R., and J.E. Kottemann (1993) "Model Integration and a Theory of Models." Decision Support Systems. **9**(1): 51-63.

Falkenhainer, B., and K.D. Forbus (1991) "Compositional Modeling: Finding the Right Model for the Job." Artificial Intelligence. **51**: 95-144.

Geoffrion, A.M. (1987) "An Introduction to Structured Modeling." Management Science. **33**(5): 547-588.

Hinkkanen, A., K.R. Lang, and A.B. Whinston (1995) "On the Usage of Qualitative Reasoning as Approach Towards Enterprise Modeling." Annals of Operations Research., in press.

Krishnan, R., P. Piela, A. Westernberg (1991) "On Supporting Reuse in Modeling Environments." working paper, School of Urban and Public Affairs and Engineering Design, Carnegie Mellon University.

Kuipers, B. (1988) "Qualitative simulation using time-scale abstraction." Artificial Intelligence in Engineering. **3**(4): 185-191.

Liang, T. (1988) "Development of a Knowledge-Based Model Management System." Operations Research. **36**(6): 849-863.

Mannino, M., B. S. Greenberg, and S. N. Hong (1990) "Model Libraires: Knowledge Representation and Reasoning." ORSA Journal on Computing. **2**(3): 288-301.

Muhanna, W.A., and R.A. Pick (1992) "Meta-Modeling Concepts and Tools for Model Management: A Systems Approach." forthcoming in Management Science.

Petrie, C. (Ed) (1992) Enterprise Integration Modeling: Proceedings of the First International Conference. MIT Press.

Raghunathan, S., R. Krishnan, and J. H. May (1993) "MODFORM: a Knowledge-Based Tool to Support the Modeling Process." Information Systems Research. **4**(4): 331-358.

Rickel, J. and B. Porter (1992) "Automated Modeling for Answering Prediction Questions: Exploiting Interaction Paths." In Proceedings of the Sixth

International Wrokshop on Qualitative Reasoning. p. 82-95. Edingburgh, Scotland.

Weld, D.S. (1992) "Reasoning About Model Accuracy." Artificial Intelligence. **56**: 255-300.