Incorporating Goal Analysis in Database Design: A Case Study from Biological Data Management

Lei Jiang[1], Thodoros Topaloglou[1], Alex Borgida[2], John Mylopoulos[1]
[1]*University of Toronto,* [2]*Rutgers University*
leijiang@cs.toronto.edu, thodoros@mie.utoronto.ca, borgida@cs.rutgers.edu, jm@cs.toronto.edu

Abstract

We present a case study of a real-world industrial application which produced several versions of conceptual schema design for a biological database during its evolution. We apply the techniques of early requirements analysis to produce a goal model of the problem domain. We then show that by incorporating goal analysis in the design process we can account for the original schemas by tracing them back to stakeholder goals. Moreover, the goal-oriented analysis supports the systematic examination of the space of design alternatives, and better explains what the output of the design process (a conceptual schema) really means. Our results advocate the need for an extended design methodology for databases driven by stakeholder goals.

1. Introduction

If one were to design an information system fifteen years ago, she'd want to use a requirements modelling language such as SADT or DFD for the functional part of the system, and ER diagrams to design the database. If she were brave, she might use instead object-oriented analysis techniques (which were quite new at the time) for both the data and functional parts of the system-to-be. The state-of-the-art in Requirements Engineering (RE) has changed dramatically in these fifteen years. Goal-oriented requirements analysis techniques have added an earlier phase to software requirements analysis where one starts from stakeholders and their intentions, and derives from them functional and non-functional requirements through a systematic, tool-supported process. This process explores a space of alternatives and selects one on the basis of criteria. Unfortunately, the state-of-the-art for designing the database part of an information system hasn't changed at all during this period. We are interested in an extended database design methodology in which stakeholder goals drive the design process in a systematic way.

We begin our research with a case study based on a real-world industrial application, which produced several versions of conceptual schema design for a biological database during its evolution. The case study compares two different methods for designing a database. We start with an analysis of the original conceptual schema and its evolving design. We then revisit the design process using a modified version of goal-driven requirements engineering technique. We aim to construct a goal model for the problem domain, starting from top-level goals of the application, by applying basic goal reasoning techniques. Goal analysis makes explicit and explores a space of design alternatives that lead to a set of data requirements specifications, each of which corresponds to a particular choice to fulfill the top-level goals.

The objective of this paper is to describe the case study, the methodology by which we recovered portions of the design space, and the conclusions we draw from our results. The rest of the paper is organized as follows. In Section 2, we give a brief introduction to the application domain of the biological database and describe four versions of its conceptual schema design. In Section 3, we present the modified design process for the same database, extended with a goal analysis phase. In Section 4, we compare the original design decisions with design recommendations suggested by the goal analysis, and demonstrate the benefits brought by the goal-driven approach. In Section 5, we conclude and point to our research plan towards a database design methodology driven by stakeholder goals.

2. Design of a biological database

In this section we review the design motivation and corresponding outcomes of a biological database that is part of a commercial system, for the management of

gene expression data generated using gene microarrays and for the support of gene expression analysis.

2.1. Application domain background

Genomics 101: Gene expression systems, simply put, measure the level of the activity of genes in biological samples. For the software engineer, an analog of the genome is the source code of a very large and poorly documented concurrent program, where each gene is a "function". We identify the genes by looking for matching pairs of begin and end statements. Each cell is an interpreter that has a copy of the full genome. At a given state of the execution, functions that are running represent genes that are expressed in the cell. A biological sample contains millions of cells. A gene microarray is a camera that takes a picture of the sample and counts all the pixels corresponding to each gene for all cells, and thus reports an expression value for each gene. (In fact, a gene array does not recognize all the genes of an organism, but recognizes several thousands of them that are printed on the array.) Given that we only have partial knowledge of what genes do, the significance of these pictures (experiments) is that they allow us to infer the function of the genes by correlating them with the conditions of the samples. In other words, if we know something about the samples and the identity of genes on the array, we can make sense of what these functions do. A gene expression database is an invaluable research tool for studying the biology of genes and how gene expression changes in the presence of diseases or drug treatments. For example, such a tool can handle queries like *"find genes whose expression goes up or down in samples derived from prostate biopsies of donors with PSA*[1]*>4, relative to normal tissues"* An answer to this query requires knowing about the origin of the sample, including the donor and his/her medical information. One goal of a gene expression database is to maintain comprehensive information about the experimental samples.

2.2. 3Sdb schema description

This sub-section describes the scientific information needs, the application requirements and the evolving design of the biological database in the early stages of the product. We analyze the requirements and design of four versions that emerged over a period of 18 months. These four versions of conceptual schema design (hereafter referred to as v1, v2, v3 and v4) demonstrate the evolution of the database due to improved understanding of the initial requirements and changing information needs of the application. Given the sheer size of the original database schema and specific goal of our analysis, which is linking early requirements to schema design, our discussion is limited to a small subset, concerning the sample component of the database (hereafter called 3Sdb -- small subset of sample database). Our analysis is based on design notes (from the period 1998-2000) and published product literature [1][2].

3Sdb is a repository of data on biological samples explored during gene expression analysis. It stores information on biological samples and their donors. The major decisions the database designers had to make at the conceptual modeling stage include the modeling of: (a) different categories of biological samples, (b) the relationships between samples, based on origin or other considerations, (c) experimental studies and (d) donor clinical profiles. Here we give a brief description of these decisions and how they differ between versions.

The central concept in 3Sdb is "biological sample", representing the tissue, cell or RNA material that originates from a donor of a given species. Therefore biological samples can be classified by sample type (i.e. tissue, cells, RNA) and donor species (i.e. human and animal). In v1, the concept *Sample* with a long list of attributes is used to represent all biological samples. Starting in v2, the *Sample* concept is specialized into *Tissue Sample*, *Cell Culture*, etc. Figure 1 shows the conceptual design for *Sample* introduced in v2[3]

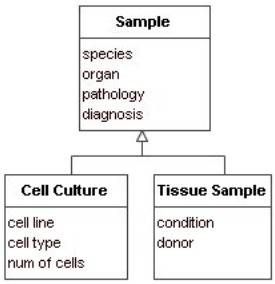

Figure 1. Design for biological sample (v2)

Biological samples are related to each other in many ways depending on the collection process and the studies they belong to. The concept *Study Group*, introduced in v1, represents a group of samples characterized according to a set of experimental

[1] PSA (Prostatic Antigen) is an early diagnostic marker for prostate cancer.

[2] Because of confidentiality and intellectual property restrictions, reference to the entire schema or complete concepts is avoided in this paper. Instead, we mention concepts and properties as they fit to the topic of discussion. We also edited the names of sensitive entities and refrain from showing complete lists of attributes.

[3] In this paper, conceptual design is represented using UML class diagrams with most of the association names omitted for brevity.

parameters. In v3, the concept *Matched Sample* is added to model samples known to be related by origin (e.g. a tumor liver and a normal liver sample from the same biopsy) or samples from the same donor without any connection to a common condition (e.g. a liver sample and a brain sample taken from the same patient many years apart). Figure 2 shows the design for these sample relationships introduced in v1 and v3.

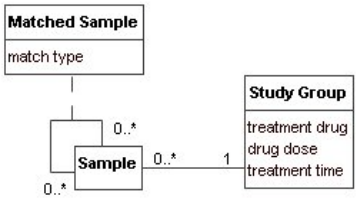

Figure 2. Design for sample relationships (v1, v3)

In addition to human samples, a significant number of samples originate from "designed experiments". Such experiments monitor, for example, the dose-response of animal subjects or cell cultures to a drug over time. Subjects are typically divided into multiple groups with different treatment conditions (e.g. drug dose), and observed at various time points. In v1, studies are modeled by concepts *Study* and *Study Group*. *Study* captures study-specific metadata (e.g. start/end date of study, number of dose groups, number of time points), while *Study Group* models information on treatments administered to the group and the associated observations (e.g. drug agent used, dose level, number of subjects per dose group at the beginning of the study). Moreover, the concept *Repeated Observation* is used to record data obtained as a result of the same type of observation repeated over time (e.g., growth, weight, temperature). In v4, the concept *Treatment* is separated from *Study Group* to allow multiple treatments within a study group. These concepts are shown in Figure 3. The evolution of the *Study* concept is due to the increased complexity of the study data encountered.

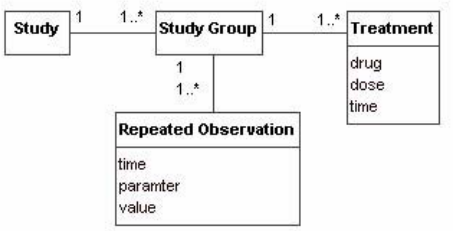

Figure 3. Design for study (v1, v4)

An important feature of 3Sdb is the association of samples to their donor medical profiles. In v1, the concept *Donor* and its sub-concepts, specialized by species (i.e., *Human Donor* and *Animal Donor*), are used to organize donor information. This information includes diagnoses, medications and family history. Here the *Donor* concept depends on the *Sample* concept, in the sense that it is used to associate a snapshot of the donor's profile to the sample. If a new sample is taken from the same donor, two donor records need to be kept in order to associate each sample to the correct donor snapshot. Figure 4 shows the design for donor profiles introduced in v1.

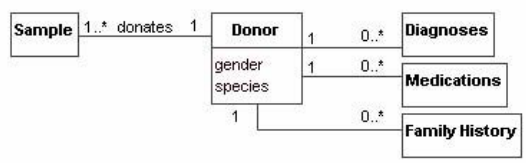

Figure 4. Design for sample donor (v1)

In v3, two new concepts *Donor Visit* and *Visit Update* are introduced. Each sample is associated with a *Donor Visit,* which may involve multiple *Visit Updates*. The concept *Visit Update* represents an event that updates the donor's information. In this design, donor is not directly associated to sample; rather a donor can participate in many visits, i.e. give many samples. The temporal dimension of donor information is controlled by *Visit Update*. Therefore the sample-specific portion of the donor information can be easily separated from the entire donor medical profile. Figure 5 shows the design for donor profiles introduced in v3.

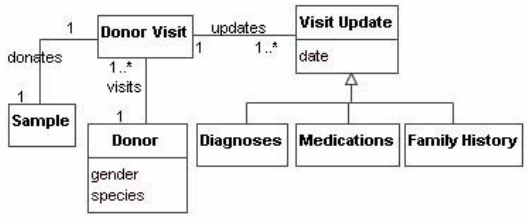

Figure 5. Design for sample donor (v3)

3. Design process revisited

We revisit the design process of 3Sdb by introducing a goal analysis step to gather data requirements before the conceptual modeling and design phase. Our aim is to start from high level goals, deriving and exploring design alternatives, much like what the Tropos methodology [4] does for software systems design. We follow the KAOS methodology [2] to identify domain concepts from the descriptions of goals and activities, and adopt the notion from the NFR framework [3] to describe stakeholder softgoals.

The goal model is constructed based on the design notes for the database. The major challenges we faced

were 1) to understand the application domain, and 2) to reserve-engineer the intentions behind the design from the design notes, which mainly provided us with the details of the design decisions that had been made. To overcome these difficulties, one of the original 3Sdb designers was consulted whenever necessary. He acted as the source of the domain knowledge, clarified ambiguous points in the design notes and provided additional information that we could not find in the documents (e.g., the initial high-level goals of the design, alternatives that had been considered).

3.1. Goal analysis

The top level goal for 3Sdb is to collect and organize data about biological samples, including clinical data about sample donors, for the purpose of *correlating sample and donor conditions with gene expression data (0)* generated from gene expression experiments using microarrays[4]. This goal is the entry point to goal analysis using basic goal reasoning techniques, such as AND/OR decomposition which refines goals into sub-goals. The result of this analysis is a goal model, which represent alternative ways to achieve the top-level goal. In the sub-problem of database design, these alternatives lead to different data requirements specifications.

Figure 6 represents a portion of the goal model[5]. To fulfill the top-level goal, one needs *to correlate gene expression data with normal organs (1.1)*, *diseases and drugs (1.2)*, and *other lifestyle factors (1.3),* such as diet, exercise and smoking habit. Goal 1.2 is further AND-decomposed into subgoals 2.1, 2,2, 2,3 and 2.4. In order *to obtain sample and donor data (2.2)*, one chooses between several experimental strategies; these include studies that involve *disease models* (*3.1)*, *human tissues (3.2)* or *both (3.3)*. Depending on the choice of strategy, the corresponding design of 3Sdb will be very different. As we will see in Section 4, the actual 3Sdb design has followed goal 3.3.

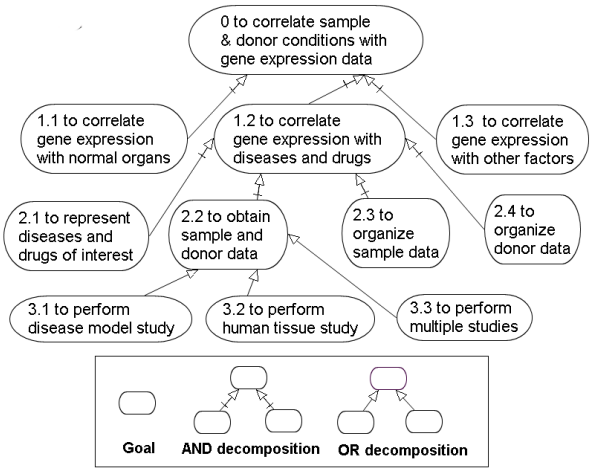

Figure 6. A partial model of high level goals

Goals can be further analyzed using another type of goal reasoning technique, means-end analysis, to reveal (alternative) activities that need to be carried out in order to achieve these goals. Activities can also be decomposed into sub-activities. For our purpose, there is no clear-cut distinction between goals and activities. Goals express desired states of affairs. Goal decomposition gradually shifts the focus from the desired states of world to the activities that can potentially help us to reach them. In this spirit, high level goals can be thought of as defining generic activities, and goal decomposition and means-end analysis as specifying these activities with more details [13]. But there are some guidelines we follow whenever possible: something is a goal, if we are exploring alternative ways to achieve it; otherwise, it is an activity, if there is an established procedure to do it.

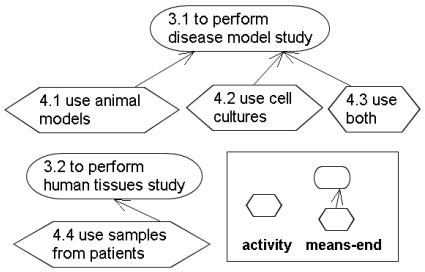

Figure 7. Means-end analysis for 3.1 and 3.2

Figure 7 shows different means to fulfill goal 3.1 and 3.2. For goal 3.1, there are three study alternatives which involve *animal models (4.1)*, *cell cultures (4.2)* or *both (4.3)*. In these studies, animal models or cell cultures are treated with drugs and then analyzed for gene expression. A further analysis of activity 4.1 reveals a sequence of steps that need to be carried out. These include: "select one or more drugs, the dosage of the drugs, and the time points of the treatment",

[4] Other component databases manage gene expression data and reference 3Sdb data through identifiers on samples.

[5] The definition of each node in the goal model consists of a short descriptive name and a detailed textual description, which we do not show here.

"collect clinical tests on animal models" and "perform gene expression measurements on animal samples".

Unlike disease model studies where treatments are performed under controlled conditions, in human tissue studies the treatments or diseases are the properties of the specimens. Therefore to fulfill goal 3.2, the investigators *use samples from patients (4.4)* to link gene expression data to its contributing factors. As a result, activity 4.4 requires different sub-activities, such as "collect human tissue samples from sources" and "obtain sample and donor clinical information". As we'll see later, study choices are sources of complexity for 3Sdb design.

3.2. Data requirements elicitation

Traditional database design techniques start with the analysis of data requirements that "dictate" the structure of data to be stored. Recent RE research suggests that instead of specifying the functionality of a system, early requirements should concentrate on modeling its environment [5]. For database design, this implies that a good requirements specification should not focus on the structure of data, but rather on the stakeholders' information needs. A goal model determines the boundary of the domain of discourse for the database-to-be. The descriptions of goals and activities reveal important domain concepts for which data need to collected and stored. These domain concepts, their associations and the queries over them form the data requirements that better capture stakeholders' information needs.

We follow the KAOS methodology [2] for identifying domain concepts from the descriptions of goals and activities. In the goal context, a domain concept represents an entity or a phenomenon the goal refers to. For example, the descriptions of goal 1.1 ~ 1.3 refer to concepts *gene expression, organ, disease, drug, diet, exercise, smoking habit, correlation of gene expression with organ*, etc. As we move along the goal hierarchy, new concepts can be introduced and existing concepts can be refined. For example, depending on which subgoal of goal 2.2 is chosen, the concept *donor* can be refined into *animal donor*, *human donor* or both.

In the activity context, a domain concept represents an entity or phenomenon that the activity has effect on or depends on. It can be identified by analyzing the input, output and control of the activity, using techniques such as UML Activity Diagram. For example, activity 4.1 is further decomposed into sub-activities, such as "order animal study", "define and execute a treatment plan", and "collect tissue samples". Concepts such as *treatment plan* and *animal tissue sample* can be identified from descriptions of these sub-activities.

Goal models are also sources of high level queries as well. In our case study, we did not address this problem. Instead, we encountered query requirements directly in the design notes of 3Sdb and interpret them as additional data requirements. We noticed that some queries are associated with a single node in the goal model. For example, the query *"find the animals with specified treatment conditions"* is related to the sub-activity "define a treatment plan". In other cases, interesting queries involve multiple goals or activities. For example, only when sub-activities "order animal study" and "define a treatment plan" are considered together within the context of their parent goal 1.2, will the query "*find sample tissues of animals from a specific population that are subject to specified drug treatment*" be considered.

Domain concepts and associated queries form the data requirements for the database-to-be. By selecting among the subgoals in a goal OR-decomposition, one can generate data requirements specifications that correspond to different choices to fulfill the top-level goal. For example, Table 1 shows a list of concepts and queries that are a part of a data requirements specification (DR1), assuming that activity 4.1, among others, is selected. If activity 4.2 is selected instead, in its corresponding data requirements specification (DR2), *animal model study* and *animal tissue sample* are replaced by the concept *cell culture*. Furthermore, if 3Sdb needs to support both animal model and cell culture studies, the resulting specification is the union of DR1 and DR2.

Table 1. Data requirements 1 (DR1)

Domain Concepts:
gene expression, disease, drug, correlation of disease and gene expression, correlation of drug and gene expression, animal model study, study group, treatment plan, animal tissue sample, …
Queries:
find animal tissue samples with the specified disease, find animal tissue samples subject to the specified drug treatment, …

3.3. Conceptual design

Conceptual design is the phase in the design process where a data requirements specification is gradually transformed into a formal description of data (i.e. a conceptual schema) using the constructs provided by a conceptual data model (e.g. the Entity-Relationship Model). This process involves a series of decision-making steps that are driven by stakeholder goals.

Traditional database design methodologies suggest several strategies (i.e. top-down, bottom-up, inside-out and mixed) to organize this transformation [6]. But these approaches mix the understanding of the domain with the decision-making, which is usually constrained by the data model and other design considerations (such as extensibility and performance), into a single step. We believe that separation of these two phases leads to more extensible and reusable conceptual schema design. Future research needs to show how such strategies can be extended to support the goal-driven approach to database design.

4. Why goal analysis

A comparison of the design choices for 3Sdb as originally made by its designers in successive versions and the design recommendations suggested by the goal analysis shows that the goal-driven approach results in a design space that (a) includes original schemas built through the evolution the application, and (b) suggests additional alternatives that lead to more comprehensive design. In addition, the goal-driven design (c) supports systematic evaluation of design alternatives, and (d) generates schemas with rich and explicit data semantics. Our interpretation is that goal analysis parallels the cognitive process that took place in the actual design of 3Sdb, except that it was implicit; when goals are made explicit, they lead to schema design that better responds to the purpose of the application. Bellow we discuss these points in detail.

4.1. Realistic design space for 3Sdb

An inspection of the schemas discussed in Section 2 suggests that the actual 3Sdb design supported both disease model and human tissue studies. For example, the concept *Donor* is specialized into both *Animal Donor* and *Human Donor*. This raises the question: whether the goal model can lead to all the design decisions made throughout the evolution of 3Sdb?

Figure 8 presents the portion of the goal model that refines goals 2.2, 2.3 and 2.4[6]. Shaded nodes in the model indicate leaf goals that were actually chosen by the 3Sdb designers. For example, *to perform human tissue study (3.2)*, a decision has to be made on whether to collect only *disease specific (4.13)* or *comprehensive (4.14)* donor medical profile. The actual 3Sdb design shows that the designers decided to follow goal 4.14. This led to both disease specific (e.g. diagnoses and medications) and historical (e.g. family history) medical data being associated either with the concept *Donor* (and its sub-concepts) in v1 and v2, or with *Donor Visit* in v3 and v4. In other case, the designers considered one alternative at the beginning and switched to another one later. For example, v1 ~ v3 only support animal studies where *single (5.5)* treatment is allowed on every study group. In these versions, treatment is a property of *Study Group*. The concept *Treatment* was introduced in v4 to accommodate *multiple (5.6)* treatments of the same study group over time.

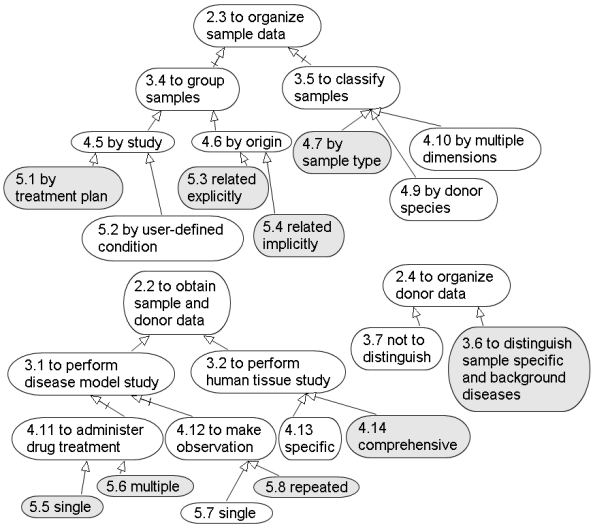

Figure 8. A partial analysis of goal 3.3

Table 2 summarizes the important concepts that were introduced in different versions of 3Sdb (shown in the horizontal heading of the table) as the result of the changing preferences to various design alternatives (shown in the vertical heading of the table), as illustrated by the goal model. For example, the concepts *Cell Culture* and *Tissue Sample* were introduced in v2 to support *classifying samples by sample type (4.7)*. Similarly, the concepts *Donor Visit* and *Visit Update* introduced in v3 were derived from goal 3.6. It solved the problem that one could *not distinguish (3.7)* a donor's condition specific to a sample (e.g. diagnosis of a cancer at sample-collecting time) from the rest of her medical profile in the background with respect to the sample (e.g. diagnosis of a cancer 1 year before sample donation).

Table 2. Summary of design decisions

[6] The subgoals of goal 3.1 apply equally no matter which activity (4.1, 4.2 and 4.3) is chosen. So these activities are not shown here.

	v1	v2	v3	v4
4.7		Cell Culture, Tissue Sample		
5.1, 5.5	Study Group			
5.3, 5.4			Matched Sample	
5.6				Treatment
5.8	Repeated Observation			
4.14	Diagnosis, Medication, Family History			
3.6			Donor Visit, Visit Update	

4.2. Additional design alternatives

The goal model not only provides an explanation why a schema was designed in a particular way, it also suggests additional alternatives that lead to more comprehensive design in terms of the coverage of stakeholder goals. Here we show two such examples. Throughout its evolution, 3Sdb gradually covers a wider range of sample grouping schemes. These include grouping of samples by *treatment plan (5.1)*, which is supported by the concept *Study Group* introduced in v1, and by sample origin in the sense that two samples originated from same donor can be either *related explicitly (5.3)* or *implicitly (5.4)*, which is supported by the concept *Matched Sample* introduced in v3. However, from the goal model, one can clearly notice that another grouping scheme was missed by the designers. Samples in the same study can also be grouped by *user specified conditions (5.2)* which are not in the pre-defined treatment plan. An advantage of this scheme is that it allows different groups to have their own ad hoc grouping conditions. If this scheme is also commonly used, the concept *Sample Set* can be introduced, as shown in Figure 9.

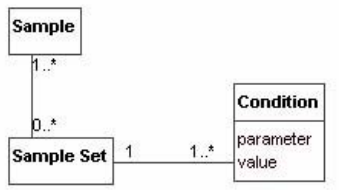

Figure 9. New design for sample relationships

As another example, 3Sdb supports classification of samples *by sample type (4.7)* through use of the concepts *Sample*, *Tissue Sample*, *Cell Culture*, and etc; other classification dimensions (e.g. donor species) are collapsed into *Sample* properties. If extensibility was of primary concern, the designers could have followed the subgoal *classify samples by multiple dimensions (4.10)*, which leads to a more comprehensive design that allows classification of samples by one or more existing or anticipated dimensions, as shown in Figure 10. In this design, we can associate dimension-specific data to each subclass of *Sample Data*.

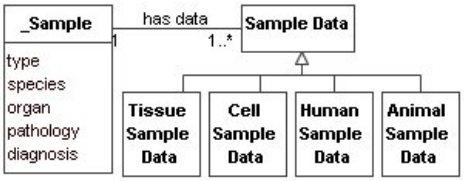

Figure 10. New design for sample

4.3. Evaluation of design alternatives

An important property of the goal model is that all the alternatives are there to be examined, regardless whether they are selected or not. Goal analysis provides a systematic way for evaluating design alternatives through use of softgoals. Softgoals are goals without clear-cut criteria for their satisfaction [7] and usually used to model non-functional requirements [3] of a software system. Design alternatives represent different information needs that may have positive or negative contributions to the fulfillment of the softgoals. For example, in Section 3.1, we discussed several alternative ways to fulfill goal 2.2: *to obtain sample and donor data*. They are repeated in table 3.

Table 3. Alternative means to fulfill Goal 2.2

4.1: (to perform disease model study) using animal models
4.2: (to perform disease model study) using cell cultures
4.4: (to perform human tissue study) using patient samples
or some combination of above

In this section, we confirmed that 3Sdb design followed the alternative (4.1)+(4.2)+(4.4). Why was this so? The 3Sdb design documents offer no clear explanation. An analysis of the high level softgoals for the gene expression system reveals the reason. Softgoals such as *greater coverage of diseases and treatment conditions (s1)*, *controlled budget of experiments (s2)* and *highest biological relevance to drug discovery (s3)* ultimately influence the ranking of these alternatives, as summarized in Table 4.

Table 4. An evaluation of study alternatives

	4.1	4.2	4.1+4.2	4.4	4.1+4.4	4.2+4.4	4.1+4.2+4.4
s1	++	+++	+++++	+	+++	++++	++++++
s2	---	--	-----	-	----	---	------
s3	++	+	+++	++++	+++++++	+++++	+++++++

A '+' is a unit of positive response of an alternative to a softgoal, while a '-' represents a unit of negative response. For example, the choice of patient samples

(4.4) burdens the budget *(s2)* by a '-', while the choice of animal models *(4.1)* is significantly more costly (represented by '---' for s2) as it yields more samples and therefore requires more experiments. With respect to biological relevance from a drug discovery perspective *(s3)*, human data *(4.4)* are more valuable than animal data *(4.1 and 4.2)*. On the other hand, cell cultures *(4.2)* give better coverage of disease and treatment conditions *(s1)* than animal models *(4.1)* or patient samples *(4.4)*, because they are easier to manipulate. From Table 5, we can infer that (4.4) is the less costly choice with the maximum return, while for the situation where the budget is less of a concern, choice (4.1)+(4.2)+(4.4) is more preferable. If the cost of database design was a softgoal, it might have tipped the balance in a different direction. Such reasoning at the requirements stage guides decisions that have an impact on the database design and the way it will be populated and used.

4.4. Better data semantics support

Another benefit of the goal-driven design process is that it helps establish explicit traces from elements in a schema to the concepts they refer to and the purpose they serve. In software development, explicit traceability management [8, 9], is critical to support decision making and reuse of design knowledge. In database design, knowledge captured at design time is essential for attaching meaning to schema elements and thus provides rich and explicit data semantics for the schema. This view is in harmony with our earlier statement that "data semantics consists of a schema, as well as the intentions behind its design" [10]. This notion of traceability is present in the example of Section 3. For example, the fulfillment of goal 3.2 "leads to" the query "*select human tissue samples with specific donor conditions*", which "is responsible" for introducing the concepts of *Donor Visit* and *Visit Update* in v3. The design rationale for modeling human sample donor conditions in this way is the softgoal that calls *to maximize biological relevance to aid drug discovery (s3)*. The links from *Donor Visit* and *Visit Update* back to goal 3.2 and softgoal s3 form a "grid" that explains the choices of the design.

5. Concluding remarks and future work

In this paper, we have considered an extended database design methodology driven by stakeholder goals. A methodology for designing general Information Systems, including a conceptual schema, which also emphasizes enterprise goals, has long been advocated by Janis Bubenko and his group [11]. A distinguishing feature of the present work is the more systematic consideration of decomposition and alternatives. A sketch of the methodology is implied in the revisited design process of 3Sdb, in Section 3. Incorporating goal analysis in database design is an endeavor that involves many challenges. Below we briefly discuss some of these open problems.

5.1. Exploration of the design space

Goal modeling is about making explicit a space of design alternatives. In the case study, we focused on the part of the design space characterized by domain stakeholders (e.g. database users) and their goals. But in the design notes that we reviewed, we found that considerations coming from development stakeholders (e.g. data modelers) are also abound. It would be interesting to see how we can combine these two sub-spaces in the design methodology. We also want to explore the notion of *generic* conceptual design, where a conceptual schema covers more than one design alternative. Such conceptual schema can be used, for example, to produce more suitable representation of the data at run-time based on the environment conditions.

5.2. From requirements to schemas

A specification of data requirements represents the understanding of the domain of discourse and contains the high level constraints on the organization of the data to be stored. A conceptual schema accumulates a series of detailed design decisions based on the understanding and constraints, and ultimately stakeholder goals behind the design. We are interested in a systematic approach to this decision-making process.

For example, concepts derived from goal analysis include a temporal dimension, while a conceptual schema is a static, time-invariant. Decisions have to be made on how to approximate the (infinitely detailed) temporal information using finite structures. Temporal database modeling techniques [12] focus on representation of the temporal components of data, once the choices have been made. We are interested in rational ways to make such decisions based on the analysis of stakeholder intentions. As another example, our analysis does not explore the use of any domain knowledge in a formal representation (e.g., domain ontologies) to aid the design process. We believe, and the case study supports the belief, that there are "blocks" of conceptual understanding of a problem domain that can help refine the data requirements and transform requirements to schemas.

5.3. Support for explicit data semantics

As we have mentioned in Section 4.4, the "byproduct" of the goal-driven database design process is the direct trace from intentions to requirements to schemas. The knowledge captured during design can be used to attach explicit meaning to the elements in the schema and propagate to the data organized by the schema. We distinguish two classes of semantics: intentional and non-intentional. Intentional semantics conveys the purpose behind the collection, store and use of the data, while non-intentional semantics corresponds to the real-world entities or phenomena for which the data is kept. Support for explicit data semantics requires representation of these types of knowledge with conceptual schemas and calls for an extension to existing conceptual data models.

Our long-term research objective is incorporation of goal analysis in database design. The case study reported in this paper has provided evidence for the relevance of this objective, and helped us to understand the issues and outline a research plan.

6. References

[1] V. M. Markowitz and T. Topaloglou, "Applying Data Warehousing Concepts to Gene Expression Data Management", *Proc. of the 2nd IEEE International Symposium on Bioinformatics & Bioengineering (BIBE'01)*, 2001

[2] A. Dardenne, A. van Lamsweerde and S. Fickas, "Goal-Directed Requirements Acquisition", *Science of Computer Programming*, North Holland, 1993, 20:3-50

[3] L. K. Chung, B. A. Nixon, E. Yu and J. Mylopoulos, *Non-Functional Requirements in Software Engineering*, Kluwer Publishing, 2000

[4] P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos, and A. Perini, "TROPOS: An Agent-Oriented Software Development Methodology", *Journal of Autonomous Agents and Multi-Agent Systems 8(3)*, Kluwer Academic Publishers, 2004, 203-236

[5] B. A. Nuseibeh and S. M. Easterbrook, "Requirements Engineering: A Roadmap", In A. C. W. Finkelstein (ed.), *The Future of Software Engineering*, Computer Society Press, 2000

[6] C. Batini, S. Ceri, S. B. Navathe, *Conceptual database design: an entity-relationship approach*, Benjamin/ Cummings Pub. Co., Redwood City, Calif., 1992

[7] J. Mylopoulos, L. Chung, and E. Yu, "From Object-Oriented to Goal-Oriented Requirements Analysis", *Communications of the ACM*, ACM Press, 1999, 42(1):31-37

[8] B. Ramesh and M. Jarke "Toward reference models for requirements traceability", *IEEE Transactions on Software Engineering*, 2001, 27(1):58 -93

[9] O. Gotel and A. Finkelstein, "An Analysis of the Requirements Traceability Problem", *Proc. of the IEEE International Conference on Requirements Engineering*, 1994

[10] A. Borgida, J. Mylopoulos, "Data Semantics Revisited", *VLDB Workshop on the Semantic Web and Databases (SWDB'04)*, Toronto, 2004, Springer-Verlag LNCS, 9-26

[11] R. Gustas, J. Bubenko jr., B. Wangler, "Goal Driven Enterprise Modelling: Bridging Pragmatic and Semantic Descriptions of Information Systems", *European - Japanese Seminar on Information Modelling and Knowledge Bases*, Sapporo, Japan, 1995

[12] H. Gregersen and C. S. Jensen, "Temporal Entity-Relationship Models—A Survey", *IEEE Transactions on Knowledge and Data Engineering*, 1999, 11:3

[13] S. Liaskos, A. Lapouchnian, Y. Yu, E. Yu, J. Mylopoulos, "On Goal-based Variability Acquisition and Analysis", In *Proceedings of 14th IEEE International Requirements Engineering Conference*, 2006