Communications of the Association for Information Systems

Volume 16 Article 42

December 2005

Special Theme of Research in Information Systems Analysis and Design -II. Data Modeling or Functional Modeling - Which Comes First? An Experimental Comparison

Peretz Shoval
Ben Gurion University of the Negev, shoval@bgu.ac.il

Judith Kabeli
Ben-Gurion University of the Negev, kabeli@bgu.ac.il

Recommended Citation
Shoval, Peretz and Kabeli, Judith (2005) "Special Theme of Research in Information Systems Analysis and Design -II. Data Modeling or Functional Modeling - Which Comes First? An Experimental Comparison," *Communications of the Association for Information Systems*: Vol. 16 , Article 42.
DOI: 10.17705/1CAIS.01642
Available at: https://aisel.aisnet.org/cais/vol16/iss1/42

Communications of the Association for Information Systems

SPECIAL THEME OF RESEARCH IN INFORMATION SYSTEMS ANALYSIS AND DESIGN - II

DATA MODELING OR FUNCTIONAL MODELING - WHICH COMES FIRST? AN EXPERIMENTAL COMPARISON

Peretz Shoval
Judith Kabeli
Dept. of Information Systems Engineering
Ben-Gurion University of the Negev
shoval/kabeli@bgu.ac.il

ABSTRACT

The software analysis process consists of two main activities: data modeling and functional modeling. While traditional development methodologies usually emphasize functional modeling via dataflow diagrams (DFDs), object-oriented (OO) methodologies emphasize data modeling via class diagrams. UML includes techniques for both data and functional modeling which are used in different methodologies in different ways and orders.

This article is concerned with the ordering of modeling activities in the analysis stage. The main issue we address is whether it is better to create a functional model first and then a data model, or vice versa. We conduct a comparative experiment in which the two opposing orders are examined. We use the FOOM methodology as a platform for the experiment as it enables the creation of both a data model (a class diagram) and a functional model (hierarchical OO-DFDs), which are synchronized. The results of the experiment show that an analysis process that begins with data modeling provides better specifications than one that begins with functional modeling.

Keywords: data modeling, FOOM, functional modeling, experimentation, method evaluation and comparison, modeling methods, requirements engineering, systems analysis, systems development methodologies.

I. INTRODUCTION

Over the years software systems became larger and more complex, making their development a more complicated task. A primary strategy to deal with size and complexity is decomposition, namely breaking a problem down into manageable units. Essential software engineering paradigms define different dimensions of decomposition. The functional approach decomposes the problem into functions/processes (functional dimension), while the OO approach decomposes the problem into object classes (data dimension). These dimensions of decomposition define

different development processes, especially at the stage of system analysis. More precisely, they support different orders for analyzing and modeling the functional and the data structural (static) aspects of a system.

Traditional development methodologies emphasize functional modeling. The main tool of analysis in those methodologies is DFDs [DeMarco, 1978, Yourdon and Constantine, 1979], and the main activity in the analysis stage is functional analysis, namely identifying system functions. This task is usually performed in a top-down manner, referred to as functional-hierarchical decomposition. The main product of this process is a hierarchy of DFDs, each consisting mainly of functions connected by data flows to external (user) entities and to data-stores. In these methodologies, the data modeling aspect plays a secondary role: after completing functional decomposition, the data-stores defined in the DFDs are transformed to database relations, either

- by applying a normalization process or
- by first creating an entity-relationship diagram (ERD) and then mapping it to normalized relations [Shoval, 1988,Yourdon, 1989].

OO methodologies tend to emphasize data (structural) modeling. They center on object classes. The functionality (behavior) of the system is expressed mainly by methods encapsulated within the classes. Many OO development methodologies evolved over the years providing different techniques for modeling aspects of the system. For example, Coad and Yourdon [1990] emphasize data modeling. They define five main activities in the analysis stage:

- finding the object-classes;
- defining structures (inheritance and whole-parts relationships);
- defining subjects by grouping objects into groups of common semantic;
- defining attributes of classes and association relationships; and
- defining services (methods) for object-classes and message relationships between them.

Hence, only the last activity deals mainly with the functionality (behavior) of the system.

A similar example is Booch's methodology [Booch, 1994], which first addresses the data structure of the system by a class diagram and then its behavior via Statecharts. OMT [Rumbaugh, 1995] uses three models in the analysis phase: an object, a dynamic, and a functional model. The first activity is object modeling, namely creating a class diagram, followed by dynamic modeling using Statecharts, and finally functional modeling with DFDs. Without going into detail about how well these three modeling activities fit together, with respect to the order of activities, data modeling comes first and functionality later.

In contrast to such methodologies that prescribe creating the data model first and then the functional model, other OO methodologies suggested different order of activities. For example, Jacobson et al. [1992] suggest a "use-case driven approach". Their methodology distinguishes between two models in the analysis phase: a requirements model that describes the system bounds and specifies its behavior through use cases, and an analysis model that refines the use cases and produces a class diagram.

Use cases became a major technique in the UML repertoire, where they are mainly used for defining functional requirements. While sequence diagrams, collaboration diagrams and state charts are used in subsequent stages of development, use cases are used in the early stage of requirements definition. Numerous UML-based methodologies, e.g. [Larman, 1998, Jacobson et al., 1999, and Mathiasssen et al., 2000], suggest an iterative and incremental approach beginning with the creation of use case diagrams and their narrative descriptions. Based on that, they provide guidelines for creating a class diagram.

In the Unified Process (UP) [Rational Unified Process], the first analysis activity is creating a use case model, whilst the initial class diagram is only created in the next phase called Use Case Analysis. Larman's methodology [Larman, 2002] which is based on UP, also starts with creating use cases. The iterative process suggests starting with an initial use case model stating the

names of the use cases in the systems and describing only the most important and risky ones, and continuing with analysis-design-development iterations. In each iteration of the analysis phase, the use cases are detailed and a semantic class diagram, called Domain Model, is created. The concepts for the domain model are identified from the nouns in the use cases descriptions.

Insfran, Pastor and Wieringa [2002] prescribe a methodology to specify high-level functional requirements and then decompose them systematically into a more detailed specification that constitutes the conceptual schema of the system. They define a requirements model (RM) which captures the functional and usage aspects, and a requirements analysis process which translates those requirements into a conceptual schema specification. The RM includes a mission statement and a functional refinement tree which are used for building use case specifications. The use cases are used as basis for the conceptual modeling phase. The major activities at this stage are to define an object model, a dynamic model and a functional model. This process is based on the analysis of each use case aimed at identifying classes and allocating them responsibilities. It uses sequence diagrams which are created for each use case, where each sequence diagram shows the classes that participate in the use case.

An enhancement of UP is UMM [Christian, 2001], which is specifically intended to model business processes and support the development of electronic data interchange (EDI) systems. Unlike the previously presented methodologies, UMM starts with a Business Modeling phase in which the first step is performing domain analysis. Identifying the "top-level entities" in the problem domain is done using Business Operations Map, a tool that presents a hierarchy of business areas, process areas and business processes. Subsequently, use cases are created to describe the functional requirements. Though not using class diagrams; UMM precedes the functional analysis with a mapping of the problem domain.

ICONIX [Rosenberg and Kendall, 2001] suggests starting the analysis by creating a class diagram describing the real world entities and concepts in the problem domain, and also uses the name Domain Model for this preliminary class diagram. The general class diagram, which describes the domain and not a specific solution, is an important basis and a glossary for creating the use cases that describe the functional requirements [Rosenbrg and Kendall, 2001]. ICONIX is the only UML-based process we found that actually discusses the issue of analysis order and explains why it is better to create a domain model before detailing the functional requirements.

Maciaszek [2001], on the other hand, claims that there is no particular order for writing the use cases and class diagram, as these two activities are done simultaneously and are feeding one another. However, since the analysis should start somehow, he eventually leaves the decision in the hands of the analyst.

Another, non-UML methodology worth mentioning with respect to the ordering of analysis activities is OPM [Dori, 2002]. OPM integrates the functional and OO approaches and treats objects and processes as equal entities. It analyzes objects and processes with a single graphical tool – the Object-Process Diagram. This tool combines data-structure elements (objects and the relationships between them) with procedural elements which are the processes and the transformations they cause to object states. Hence, OPM offers a way of performing system analysis that combines both aspects.

Another methodology that integrates the functional and object approaches is FOOM [Shoval and Kabeli, 2001], which provides two main products at the analysis stage: a functional model in the form of OO-DFDs, and a data model in the form of a class diagram. These two products can be produced in any order but eventually they are synchronized to provide a coherent specification of the system (Section II).

In conclusion, the order of performing the analysis activities differs among methodologies; some begin with data modeling and continue with functional modeling; some do the opposite; while still others give them equal status. We found no research addressing this issue scientifically, i.e., research that tries to examine empirically or in another systematic way which order is preferred and if or how different orders of activity affect the quality of the analysis products. The issue in the

real-world about the performing order of the analysis activities is that different orders of activities may yield different analysis products (specifications), perhaps of different quality. It is important to find out if the order of activities matters, and if so – which order is better. This question is the main concern of this study.

To examine this issue we decided to perform a comparative experiment in which two groups of analysts perform the same tasks of modeling an information system, but analysts in each group working in different orders. Analysts in one group created a functional model first and then a data model; Analysts in the other group worked in the opposite order. We used FOOM methodology as the platform for the experiment because this methodology allows performing these analysis activities in any order and provides the same products - a functional model and a data model, both being used as input to the stage of system design[1].

The reminder of the paper is structured as follows: Section II provides a brief overview of the analysis phase of the FOOM methodology. Section III discusses the research goals and hypotheses, and describes the experiment. Section IV presents the results, and Section V discusses the results and suggests further research issues.

II. FOOM METHODOLOGY

FOOM is a methodology for analysis and design of information systems that combines two essential software engineering paradigms: the functional- and the object-oriented (OO) approaches [Shoval and Kabeli, 2001]. FOOM makes a clear distinction between the analysis and design stages of development and provides a smooth transition from one stage to the other. The analysis stage consists of two main activities: data analysis and functional analysis. The two main products of this stage are:

1. a data model, in the form of an initial class diagram, and
2. a functional model, in the form of hierarchical OO-DFDs.

The initial class diagram consists of data (entity) classes with attributes and various structural relationships, but with no methods. Methods and other classes will be added at the design stage, as based on the functional model created at the analysis phase. Figure 1 shows an example of an initial class diagram of the *IFIP Conference* case study presented in Kabeli and Shoval, [2003].[2]

The OO-DFDs are similar to traditional DFDs but they include data classes instead of data-stores. Figure 2 shows an example of one OO-DFD of the *IFIP Conference* case study. Obviously, a complete functional model of a system consists of a hierarchy of OO-DFDs, and the example shows one arbitrary OO-DFD that consists of elementary functions (namely functions which are not decomposable), external/user entities and data classes.

The activities and products of the FOOM analysis phase can be performed in two opposing orders,

1. Begin with the creation of OO-DFDs and then the initial class diagram, or
2. Begin with the creation of an initial class diagram and then the OO-DFDs.

When beginning with functional analysis, the analyst first produces a hierarchy of OO-DFDs (based on some description of user requirements). The analyst then creates an initial class diagram including classes already appearing in those OO-DFDs. This process mainly defines appropriate associations among these classes and their attributes.

---

[1] In a follow-up study we performed a similar experiment using a UML-based methodology. The results of that study, which will be reported separately, support the results of the present study.

[2] The notation of class diagram in FOOM is slightly different from that of UML. For example, in FOOM classes have reference attributes in addition to links between the classes.

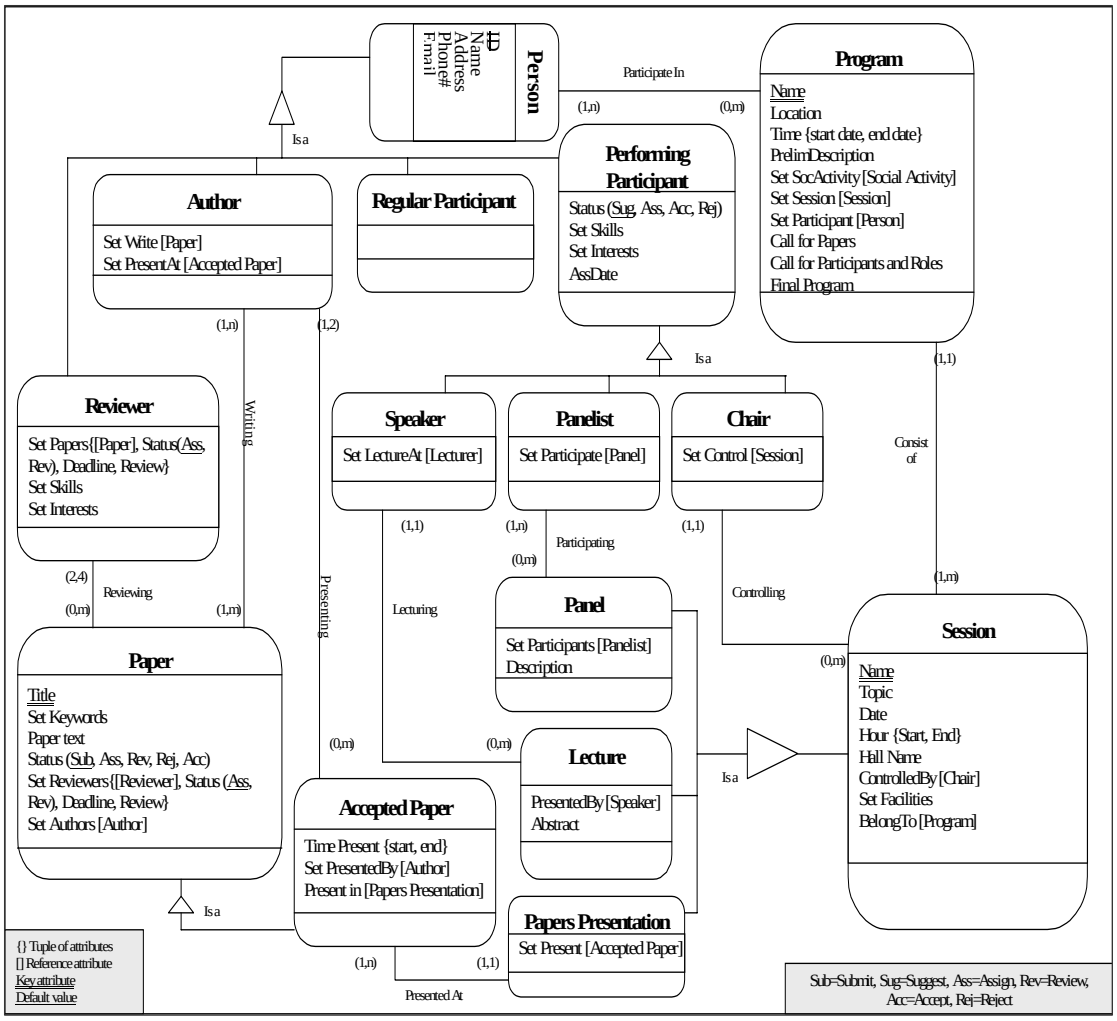

Figure 1. Initial Class Diagram of the IFIP Conference

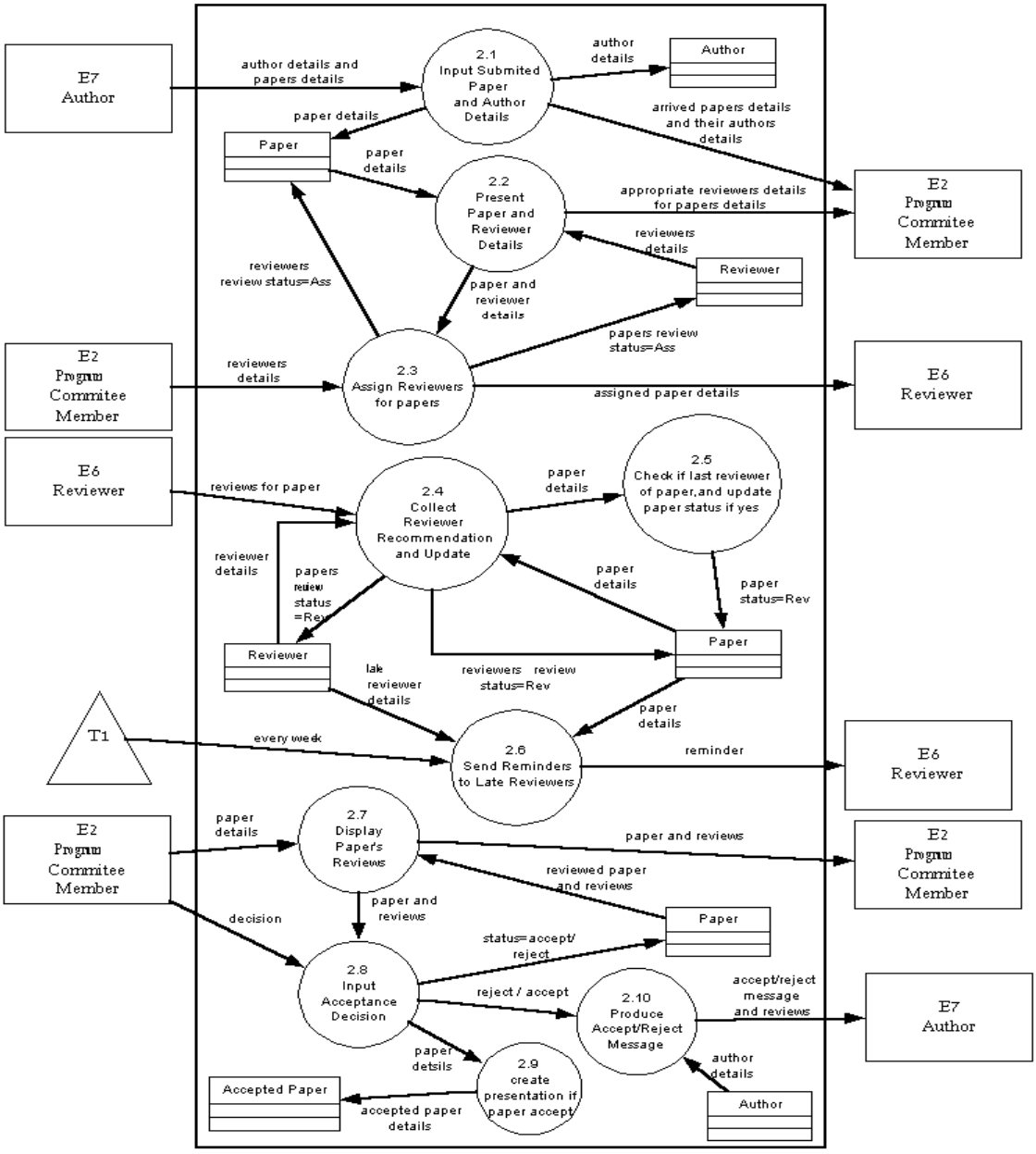

Figure 2. OO-DFD-2 of the *IFIP Conference* Systems

The activities and products of the FOOM analysis phase can be performed in two opposing orders,

1. Begin with the creation of OO-DFDs and then the initial class diagram, or
2. Begin with the creation of an initial class diagram and then the OO-DFDs.

When beginning with functional analysis, the analyst first produces a hierarchy of OO-DFDs (based on some description of user requirements). The analyst then creates an initial class diagram including classes already appearing in those OO-DFDs. This process mainly defines

appropriate associations among these classes and their attributes.

When the performing order begins with data modeling, the analyst first creates an initial class diagram (based on some description of user requirements) and then creates the OO-DFDs using the already defined classes.[3]

No matter which order of activities is followed, FOOM completes the analysis phase by verifying that the two models are in sync, namely:

1. each class appearing in the class diagram appears also in the OO-DFDs, and vise versa;
2. each attribute of a class is updated by at least one function and retrieved by at least one function in the OO-DFDs.

These analysis products are used to design the components of the system.

The main products of the design phase are:

1. a complete class diagram, including (in addition to the initial data classes) a Menus class, whose objects are the various menus; a Forms class, whose objects are the various input forms/screens; and a Reports class, whose objects are the various output screens and reports;
2. detailed descriptions of the methods of all classes; these are derived from the functions in the OO-DFDs and the way they interact with each other and with the external/user entities and the classes. The methods are described by either pseudo code or message charts (which is similar to a program flowchart but it also includes classes and messages to other methods.)

The design phase and its products are discussed in more detail in Shoval and Kabeli, [2001] and in Kabeli and Shoval [2003].

III. THE EXPERIMENT

RESEARCH GOALS AND HYPOTHESES

Our major goal is to find which order of analysis activities is better: to begin the analysis by creating a data model and then a functional model, or vice versa? Obviously, system analysis is an iterative process of refinement, not a linear sequence of activities. But still, the analysis must begin with some specific activity, because an analyst cannot do everything in parallel. As we saw, different methodologies prescribe different orders of activity – usually without explaining why. We found no research which proposes a theory about the order of activities, or which provides any empirical evidence on the preferred order.

Intuitively, it appears that to begin with functional modeling (creating OO-DFDs) is more difficult than to begin with data modeling (creating an initial class diagram) because DFDs are

- more complex;
- they consist of more construct types (general functions, elementary functions, external entities, time and real-time entities, data classes and data-flows among them), and
- analysts must follow strict rules of the hierarchical decomposition of functions and diagrams.

In contrast, an initial class diagram consists only of classes, attributes and relationships. Usually only one diagram must be created. Creating OO-DFDs first also seems to be more complex because the analysts actually deals at the same time with the functional and structural (data)

[3] It is worth mentioning that instead of creating an initial class diagram directly from user requirements, it is possible to do so indirectly, by first creating an ERD and then utilizing well-formed rules to translate the ERD into an initial class diagram. Some analysts may prefer this indirect way as there is evidence that analysts make fewer errors when creating an ERD rather than when creating an equivalent class diagram [Shoval and Shiran, 1997].

aspects, while when beginning with a class diagram, the analyst deals only with the structural aspect; then, when he/she works on the second task, the classes are already defined, and therefore the task of creating the OO-DFDs (which utilize those classes) becomes easier. Hence, it seems that data modeling should precede functional modeling. But, as discussed in the Introduction, there are in fact different methodologies which advocate different orders of activities or do not refer to this issue, and there is still no scientific evidence which order is better. Therefore, we hypothesize that there is no difference in the quality of the analysis products created in the opposing orders of activities.

Two supplementary goals of this study are

- to find out if there is any difference in the time spent by analysts on the two tasks with respect to the order of analysis activities, and
- which order of activities analysts prefer.

With respect to time, since data modeling usually involves creating just one class diagram, compared to functional modeling which involves creating many OO-DFDs, we hypothesize that data modeling takes less time, no matter the order of activities. With respect to preferences of analysts: following our intuition on the better order of activities, it seems that analysts would prefer modeling the data first. But, again, as we have no a-priori evidence on this, we hypothesize that there is no difference in preference.

EXPERIMENTAL DESIGN

The conceptual research model is presented in Figure 3.

Three dependent variables are surrogates for performance:

1. Quality of model: Quality may have many facets. We mean correctness of the analysis products – the functional and data models. Correctness is measured according to a grading scheme (to be detailed later).
2. Time: The time it takes to complete each of the two analysis tasks.
3. Preference: Which order of analysis the analysts believe is better.

The independent variable is analysis order. One order begins with data modeling and continues with functional modeling (marked D-F); the other is in opposite order (marked F-D).

The control variables include the tasks and the subjects:

1. Tasks: Two tasks were given to each subject – to create an initial class diagram (the data model) and to create a set of OO-DFDs (the functional model) for the IFIP Conference case study [Mathiasssen et al., 2000]. The user requirements for the case study were described in narrative – a requirements document. In reality, the analysts elicits user requirements by performing such tasks as interviewing the users and observing their work, But in our controlled experimental setting we prepared a requirements document which attempts to mimic the reality. Such a requirements document must, of course, include both the data and the functional requirements; but these requirements may be prescribed in different orders. For example, a requirements document may first detail data-related requirements and then functional-related requirements; or vice versa. A threat to validity of the experimental results may be that a certain ordering of the requirements would bias the performance of the subjects. To avoid a possible bias because of the effect of requirements order, we created four versions of the same **r**equirements document: [4]

[4] Note that we do not treat the order of requirements as an independent variable. We only control the possible effect of different ordering of requirements by preparing the fours versions and distributing them at random among the subjects within the two groups.

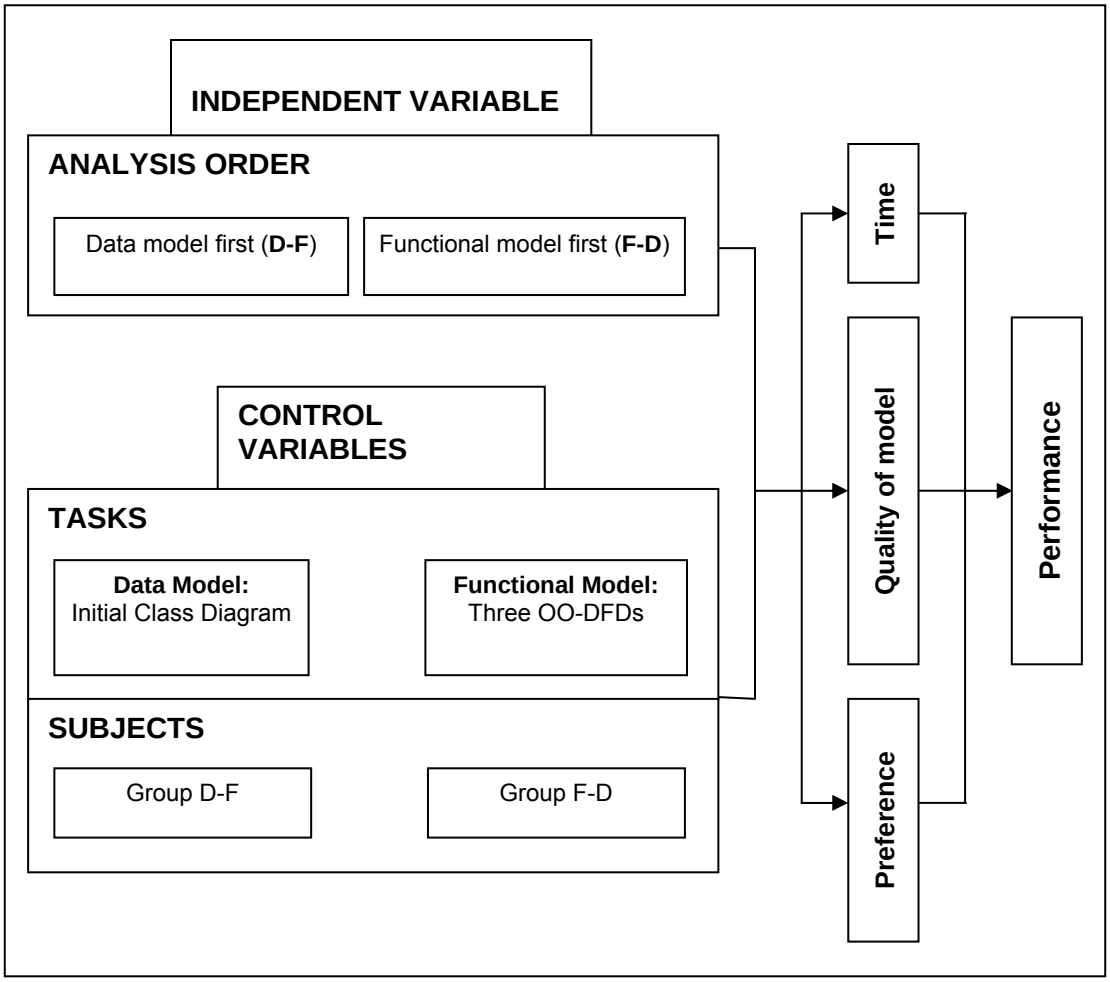

Figure 3. The Conceptual Research Model

a. all data-related requirements are presented first, followed by all functional-related requirements;
b. the requirements are grouped by main data structures, and within each its functional-related requirements
c. the requirements are grouped by main functions, and within each the data structures they use
d. all functional-related requirements are presented first, followed by all data-related requirements.

Based on the requirements document, the task of each subject was to prepare (in a predefined order) an initial class diagram and three OO-DFDs: one root (OO-DFD 0) and two sub-level OO-DFDs for two general functions. Two hours were allocated to perform the two tasks.

2. Subjects**:** The subjects were undergraduate students of Information Systems

Engineering[5] who took the same courses, including Systems Analysis & Design, and Databases.[6] They all studied the functional and OO approaches, and FOOM methodology – all in the same classes and by the same instructor. The experiment took place at the end of the semester, and the subjects were motivated by the fact that their grades on quality (correctness) of specifications would be considered as part of their course grades.[7] The subjects were divided randomly into two main groups[8]. The task of subjects in one group (D-F) was to create the initial class diagram first, and then the OO-DFDs; the task of subjects in the second group (F-D) was to do the same but in the opposite order. There were 27 subjects in Group D-F and 29 subjects in Group F-D. Putting these two independent variables together, we obtained the treatment groups shown in Table 1.

Table 1. The Treatment Groups

Group	# of subjects	Analysis Order
Group D-F	27	1. Initial class diagram 2. OO-DFDs
Group F-D	29	1. OO-DFDs 2. Initial class diagram

When the experiment began, each subject received a requirements document in one of the four versions[9], and blank solution sheets for the first task (one sheet for the class diagram; three for the OO-DFDs). The commencement time was recorded. When a user completed the first task, the completion time was recorded and then the solution sheet(s) for the second task were handed to the subject. The solution sheets of the first task were not handed in but were used in preparing the solution for the second task. At the end, the completion time was recorded and each subject was asked to complete a short questionnaire, including a question (using a 7-point scale) on the preferred order of analysis.

The quality or correctness of each analysis product (initial class diagram or OO-DFDs) was measured according to the number and type of error found, comparing them to an “expert solution” and using grading schemes to compute the number of points deducted for each error type in every model.

We created a grading scheme for each of the models, based on the idea introduced by Batra et al., [1990] and used in other studies to evaluate conceptual data models including class diagrams [Kim and March, 1995, Shoval and Shiran, 1997]. Table 2 details our grading schemes. As can be seen, for each model we identified the possible kinds of errors, according to the constructs of the model, and assigned weights of their severity (i.e., a number of points to be deducted for each

[5] It would be better to use experienced systems analysts rather than students. However, it was not feasible to do so. Students are used as surrogates for analysts in the majority of experimental research on model/method evaluation [Topi and Ramesh, 2002].

[6] Systems Analysis & Design is a 3-hour course; Databases is a 3.5-hours course. The two courses were given in the same semester. In Databases the students study data modeling and class diagrams; in Systems Analysis & Design they study functional modeling and FOOM methodology, including the integration of OO-DFDs and class diagrams.

[7] Since it is possible that subject assigned in one group would score less than subjects assigned in the other group, due to the possible difference in the order of analysis, we computed the average scores within each group and compensated the subjects in the "inferior" group accordingly.

[8] Random assignment of subjects eliminated bias due to possible differences among them.

[9] The documents were distributed randomly to subjects within each group, to avoid ordering bias.

error of that type).[10] Each diagram was graded separately. The grade of the functional model was computed as the average of the three OO-DFD grades. In order to verify that there is no bias in grading, twelve sets of specifications were randomly selected, copied and given to two independent examiners who used the same grading scheme. Comparison of their grades revealed no significant differences.

Table 2. The Grading Schemes

Errors in OO-DFDs	Points	Errors in Class diagrams	Points
Missing function	-4	Missing object	-3
Unnecessary function	-2	Unnecessary object	-2
Process without an input or output	-2	Incorrect use of inheritance	-2
Missing or unnecessary dataflow; missing label	-1	Missing or incorrect attribute	-1
Wrong connection (e.g. between classes)	-1	Incorrect notation	-1
Incorrect dataflow (e.g. wrong connection between functions; dataflow in wrong direction)	-1	Incorrect relationship, missing or incorrect cardinality of relationship	-1
Function not in the right diagram	-1	Missing or unnecessary relationship	-1
		Missing relationship attribute	-1/2

IV. RESULTS

QUALITY OF MODELS

Based on the grading schemes and the numbers of errors made by each subject, we computed the grade of each subject (in percent) and then the average grade of the subjects within each group/model. We performed two-tail t-tests of difference between the average grades, for each pair of models. Table 3 presents the results:

- The first row presents the results for the data models and the second row – for the functional models.
- The second column presents the two opposing ordering of analysis: D-F vs. F-D.
- The third column presents the mean grades of the subjects within each group. The fourth column is the t statistic of the independent variable;
- The fifth is the P value (at α=0.05); and
- The last column notes if the differences are significant.

Table 3. Quality of Analysis Specifications

Model	Analysis Order	Mean Grade (%)	t	P-value (α=0.05)	Significance in favor of
Data model	D-F F-D	71.48 58.50	7.23	0.00985	**D-F**
Functional model	D-F F-D	83.86 83.71	0.004	0.94740	-

[10] There might be some subjectivity in the grading schemes, in particular in the weights of severity of the error type. This limitation will be elaborated in the Discussion and Summary section. At any rate, note that we applied each grading scheme on both groups within each model, and we did not combine or compare the results across the two models.

The first row (which presents the effect of the analysis order on the mean grade of the data model, i.e. the class diagram) shows a significant difference in favor of the analysis order that begins with data modeling. In other words, when analysis begins with data modeling, the quality of the data model is much better. This result is in accord with our initial assumption.

The second row in Table 3 presents the effect caused by the independent variable on the mean grade of the functional model, i.e. the OO-DFDs. There is no significant difference, i.e., in both orders of analysis the grades are almost the same. That is, the order of analysis activities does not matter with respect to the quality of the functional mode. This result is counter to our intuition.

In both analysis orders the grades on the functional model (row 2) are significantly higher than the grades on the data model (row 1). Several explanations are possible. For example:

- In this case study the functional requirements were clearer or simpler than the data requirements;
- The subjects were better trained in functional analysis than in data analysis;
- The grading scheme (Table 2) caused the deduction of more error points on the data model.

Whether or not any of these explanations are valid, they do not affect the results, which compare only the mean grades of the same models.

Since the difference in the quality of the data models was significant and no significant difference was observed in the quality of the functional models, we conclude that, overall, it is an advantage in undertaking data modeling first.

TIME TAKEN TO COMPLETE THE TASKS

Table 4 presents the effects observed on the time taken to perform the data modeling task (D columns) and functional modeling task (F columns).

Table 4. Time to Complete the Analysis Tasks

Analysis Order	Mean Time (minutes)			t		P-value (α=0.05)		Significance in favor of	
	D	F	Total	D	F	D	F	D	F
D-F	58.09	56.29	114.38	21.19	23.09	<0.000	<0.000	F-D	D-F
F-D	38.60	74.24	112.84						

Table 4 shows that the average time taken to perform the two tasks was almost the same (and a little less than the maximal time allocated to each subject); but the allocation of time on each task depends on the order of activities: the time taken to complete the data modeling task was 150% longer when beginning with data modeling than when beginning with functional modeling (58.09 min. and 38.60 min., respectively). Similarly, the time taken to complete the functional modeling task was 132% longer when beginning with functional modeling compared to when beginning with data modeling (74.24 min. and 56.29 min., respectively). That is, subjects spent relatively more time on the first task. These results are reasonable given the set up time needed at the beginning of any task.

Those who started with data modeling (row D-F) spent almost the same amount of time on both tasks, whereas those who started with functional modeling (row F-D) spent most of the time (74.24 min.) on functional modeling, and much less time (38.60) on data modeling. As we saw in Table 3, the quality grades of these subjects on the data model were significantly lower compared to the subjects who started with data modeling. In other words, while the time factor in itself may

not be important, the results support the earlier results on quality, because they show that if a subject starts with functional modeling, he/she spends relatively more time on this task, possibly due to its compliexity, not leaving enough time for the other task, thus causing lower performance on data modeling.

To complete the analysis of time results, the overall mean times to create the data and the functional models were computed for all subjects, no matter what the analysis order is; the mean times were 48.35 min. for the data model and 65.27 min. for the functional model. Clearly, as expected, subjects spent more time on creating the functional model. Recall that whiles the task of data modeling was to create one class diagram only; the functional modeling task was to create three OO-DFDs.

PREFERENCES OF ANALYSTS

At the end of the experiment, each analyst was asked to express to what extent he/she believed that the order of analysis in which he/she worked is good, using a 7-point scale, where the score 1 (lowest) means that the order is extremely bad, while 7 means that the order is extremely good. Table 5 presents the results.

Table 5. Preference of Analysis Order

Level of preference of this order	Mean preference of subjects who worked as follows:			Stan. Dev. of All	t-value of All	P-value of All	Significance in favor of
	D-F (N=27)	F-D (N=29)	All (N=58)				
D-F	4.48	5.724	5.103	1.608	3.95	< 0.000	**D-F**
F-D	4.14	2.586	3.367	1.890			

- Row D-F shows the scores of the analysis order commencing with data modeling,
- Row F-D shows the scores of the analysis order commencing with functional modeling.
- Column D-F shows the average scores given by the subjects who first performed data modeling,
- Column F-D shows the average scores given by the subjects who first performed functional analysis.
- Column “All” shows the mean scores of all subjects together. The remaining columns refer to “All” scores.

It can be seen that the subjects in the two groups believe that it is better to begin the analysis with data modeling. Furthermore, subjects who began with functional analysis (column F-D) believed even more strongly that it is better to begin the analysis with data modeling (5.724 compared to 2.586); that is, they really did not like the order in which they actually worked.

V. DISCUSSION AND SUMMARY

The main issue addressed in this study was the order of performing the two analysis activities: data modeling and functional modeling. Usually a methodology adopts a certain order of activities but does not necessarily provide a rationale for this order (Section II). Sometimes methodologies do not prescribe any order, leaving it up to the analysts. This study addressed the issue by conducting a controlled experiment comparing the alternatives. The main result of this experiment is that it is an advantage in starting the analysis by creating a data model and then a functional model. However, the advantage is only with respect to the quality of the data model. No significant quality difference was found for the functional model. When working in this order, analysts allocate the time more evenly on the two tasks, whereas when working in the opposite order they spend too much time on functional modeling. Functional modeling is more complex, leaving less time to data modeling – thus obtaining poorer results on this model. We also found that analysts believe that it is better to work in this order.

THEORETICAL BASIS

No prior research was found which addresses the specific issue raised in this study. In the Research Goals and Hypotheses subsection of Section III we discussed the pros and cons of the two alternative orders of activity. Our intuition was that it would be better to start the analysis with data modeling mainly because an initial class diagram is simpler and consists of less constructs than DFDs. Moreover, in data modeling the analysts in concerned with data structure aspects only, while in functional modeling the analyst is also concerned with data-structure aspects. Having created a data model in the first stage makes the functional modeling complex task easier. Our initial intuition, and the findings of the experiment, can be supported by the theoretical framework proposed by Wand and Weber known as the BWW (Bunge, Wand & Weber) ontological framework (e.g., [Wand and Weber, 1995; Weber, 2003]). In their framework, the world consists of things that possess properties; this fits with a data orientation, in particular with what exists in an initial class diagram. More complex notions, such as processes (functions) are built up from the foundations of things and properties; this approach fits with functional orientation, notably DFDs. If we accept that a key purpose of modeling is to represent aspects of the real world, then it follows from the framework that it is appropriate to begin the modeling process by dealing with the basic things and properties, i.e. with data modeling and then continue with the modeling of the more complex constructs, i.e. with functional modeling.[11]

Another support for our findings comes from cognitive theory, specifically from COGEVAL (Rockwell and Bajaj, 2005), a propositional framework for the evaluation of conceptual models. The COGEVAL framework tries to explain the quality and readability (comprehensibility) of models using various cognitive theories. Two propositions of the COGEVAL framework can support our results:

- Proposition 2 of the framework, which is based on theories about short-term memory, states that the greater the number of simultaneous items required to create a model, the lower is the effectiveness and efficiency of the modeling method. More specifically, when creating a model, if more aspects of reality need to be considered to create chunks or parts of the model, then some of these items will need to be stored in long-term memory or in some stable storage. This principle of parsimony implies greater possibilities of errors, as well as greater effort to produce the model. This proposition supports our result that it is better to start with data modeling, since an initial class diagram consists of fewer items than OO-DFDs.
- Proposition 4 of the COGEVAL framework, which is based on theories on levels-of-processing and long-term memory, states that the greater the amount of semantic processing required to create a model, the less the effectiveness and efficiency of the modeling method. More specifically, data modeling, that involves only structural processing, offer greater effectiveness and efficiency than functional modeling. This proposition too supports our experimental results because in data modeling the analyst is concerned at the beginning only with structural modeling, creating only an initial class diagram; then, using the already defined classes; he/she creates the functional model (OO-DFDs).

LIMITATIONS

As is common in experimental research, this research contains several limitations that should be considered when examining the validity of the results:

1. We used only one methodology, FOOM. We cannot guarantee that the results hold for

[11] This theoretical framework indeed supports our findings with respect to the quality of the data models, the allocation of time and the users' preferences. Note, however, that this framework does not provide an explanation for not obtaining significant differences between the functional models.

other methodologies. It is possible that the preferred order depends on the methodology.
2. We analyzed only one system, the IFIP Conference case study. This system is relatively simple. It can be worked out in a laboratory/classroom setting. We cannot guarantee that the results hold for large-scale systems. The problem here is that it is not feasible to conduct controlled experiments with large-scale systems.
3. The subjects who analyzed the system were students, not experienced analysts. It would be more reliable to base conclusions on experts or more experienced analysts. But finding such subjects in sufficient number is not feasible either. Most of the published experimental work in systems analysis and data modeling uses student subjects.

A potential confound of the experiment is the grading schemes we adopted. Some subjectivity may exist in the weights of severity of the error type given in Table 2. We determined the weights as based on our assessment of the importance of each error type. In doing so we followed earlier studies who too adopted subjective grading schemes to assess quality of models. For example, Batra et al. [1990] categorized errors based on severity and distinguished between minor, medium and major error types. Similarly, Kim and March [1995] distinguished between minor and major errors and assigned penalty points to each error type. They based their values on the computed overall performance of their subjects using a formula which considered the number of errors and the penalty points, similar to what we did in this study. The potential problem with such grading schemes is that the subjective weights (penalty points) assigned to error types may affect the overall results. The problem is that there are no objective weights and grading schemes for different methods or models. This issue deserves separate research.

Another potential confound is the possibility that the researchers were involved in the experiment, and may have been biased in favor of one of the alternatives. In the present case there is no reason to assume such bias because the researcher created the underlying methodology, according to which the analysis activities can be done in any order. It is the results of the experiment which led the researchers to conclude which of the two alternatives is better.

FUTURE WORK

We plan to conduct additional experiments on this issue, which will include systems of different sizes and from different application domains. Moreover, we plan to use other methodologies as well as FOOM. to find out whether or not the order of activities depends on the development methodology. We already conducted one follow-up experiment in which we compared the order of analysis activities using UML tools, namely use-cases and class diagrams. The results of the new experiment are consistent with the results reported here, and will be reported separately.

Editor's Note: This paper is one in a series of articles in the Research in Information Systems Analysis and Design series, guest edited by Juhani Iivari, and Jeffrey Parsons. Alan Hevner served as the CAIS departmental editor for the series. Some of the papers in this series are being published in JAIS and some in CAIS; the choice depending on the topic and approach of the paper. This paper was received on March 31, 2005. It was with the author for 2 revisions and was published on December 8, 2005.

VI. REFERENCES

Batra, D., Hoffer, J. and Bostrom, R. (1990). "Comparing Representations with the Relational and Extended Entity Relationship Model", *Comm. of the ACM*, 33 (2), pp. 126-139.

Booch, G. (1994). *Object-Oriented Design with Applications*, 2nd edition, Redwood City, CA: Benjamin/Cummings.

Christian Huemer (2001). "Defining Electronic Data Interchange Transactions with UML", *Proceedings of the 34th Hawaii International Conference on System Sciences* (HICSS-34), Volume 7**,** Washington, DC, IEEE Computer Society, pp. 1-10.

Coad, P. and Yourdon, E. (1990). *Object-Oriented Analysis*. Englewood Cliffs, NJ: Prentice Hall,

DeMarco, T. (1978). *Structured Analysis and System Specification*. New York: Yourdon Press.

Dori, D. (2002). *Object-Process Methodology - A Holistic Systems Paradigm*. Berlin: Springer Verlag.

Insfran, E., Pastor, O. and Wieringa, R. (2002). "Requirements Engineering Based Conceptual Modeling". *Requirements Engineering*, 7, pp. 61-72.

Jacobson, I., Booch, G. and Rumbaugh, J. (1999). *The Unified Software Development Process*. Reading, MA: Addison-Wesley,

Jacobson, I., Christerson, M., Johnsson, P. and Overgaard, G. (1992). *Object-Oriented Software Engineering: A Use-Case Driven Approach.* Englewood Cliffs, NJ: Prentice-Hall,

Kabeli, J and Shoval, P. (2003). "The Application of FOOM Methodology to IFIP Conference Case Study". In: Peckham, J. and Lloyd, S. (editors): *Practicing Software Engineering in the 21st Century*. Hershey, PA: IRM Press, pp. 82-95.

Kim, Y. and March, S. (1995). "Comparing Data Modeling Formalisms", *Communications of the ACM*, 38 (6), pp. 103-115.

Larman, C. (1998). *Applying UML and Patterns - an Introduction to Object-Oriented Analysis and Design*. Upper Saddle River, NJ:Prentice Hall.

Larman, C. (2002). *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design, and the Unified Process (2nd Edition)*. _ Upper Saddle River, NJ:Prentice Hall.

Maciaszek, L.A. (2001). *Requirements Analysis and System Design: Developing Information Systems with UML.* Reding, MA: Addison Wesley.

Mathiasssen, L., Munk-Madsen, A., Nielsen, P. and Stage, J. (2000). *Object Oriented Analysis and Design*. Aalborg, Denmark: Marko Publishing.

Rational Unified Process (RUP). http://encyclopedia.thefreedictionary.com/

Rockwell, S. and Bajaj, A. (2005). COGEVAL: Applying Cognitive Theories to Evaluate Conceptual Models". In: Siau, K. (editor): *Advanced Topics in Databases Research*, Hershey, PA: Idea Group, pp. 255-282.

Rosenberg, D. and Kendall, S. (2001). *Applied Use Case-Driven Object Modeling.* Reading, MA: Addison Wesley.

Rumbaugh, J. (1995). "OMT: the Dynamic Model, the Functional Model, the Object Model". *Journal of Object-Oriented Programming*, 7(9): 6-12, 8(1): 10- 14, 7(8): 21-27.

Shoval, P. (1988). "ADISSA: Architectural Design of Information Systems Based on Structured Analysis". *Information System*, 13 (2), pp. 193-210.

Shoval, P. and Kabeli, J. (2001). "FOOM: Functional- and Object-Oriented Analysis and Design of Information Systems - An Integrated Methodology". *Journal of Database Management*, 12 (1), pp. 15-25.

Shoval, P. and Shiran, S. (1997). "Entity-Relationship and Object-Oriented Data Modeling - An Experimental Comparison of Design Quality". *Data & Knowledge Engineering*, 21, pp. 297-315.

Topi, H. and Ramesh, V. (2002). "Human Factors Research on Data Modeling: A Review of Prior Research, an Extended Framework and Future Research Directions". *Journal of Database management*, 13 (2), pp. 3-19.

Wand, Y. and Weber, R. (1995). "On the Deep Structure of Information Systems". *Journal of Information Systems*, 5, pp. 203-223.

Weber, R. (2003) "Conceptual Modeling and Ontology: Possibilities and Pitfalls". *Journal of Database management*, 14 (3), pp. 1-20.

Yourdon, E. (1989). *Modern Structured Analysis*. Englewood Cliffs, NJ: Prentice Hall,

Yourdon, E. and Constantine, L. (1979). *Structured Design.* Englewood Cliffs, NJ: Prentice Hall

ABOUT THE AUTHORS

Peretz Shoval is Professor of Information Systems in the Department of Information Systems Engineering of Ben-Gurion University. He earned his Ph.D. in Information Systems (1981) from the University of Pittsburgh, where he specialized in expert systems for information retrieval. His research interests include information systems analysis and design methods, data modeling, and information retrieval and filtering systems.

Judith Kabeli is Lecturer of Information Systems in the Department of Industrial Engineering & Management of the Shenkar Academic College of Engineering & Design. She earned her Ph.D. in Information Systems (2004) from Ben-Gurion University, where she specialized in IS development methodologies. Her research interests include information systems analysis and design, and intelligent web-based user interfaces.

Communications of the Association for Information Systems

ISSN: 1529-3181