**Association for Information Systems**
**AIS Electronic Library (AISeL)**

MCIS 2011 Proceedings

Mediterranean Conference on Information Systems (MCIS)

2011

# AN EMPIRICAL COMPARISON BETWEEN TWO METHODS FOR DEFINING FUNCTIONAL REQUIREMENTS: USE CASES VS. OO-DFDS

Michal Dahan
*Ben-Gurion University, Beer-Sheva, Israel*, dahanmic@bgu.ac.il

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

Arnon Sturm
*Ben-Gurion University, Beer-Sheva*, sturm@bgu.ac.il

Follow this and additional works at: http://aisel.aisnet.org/mcis2011

Recommended Citation
Dahan, Michal; Shoval, Peretz; and Sturm, Arnon, "AN EMPIRICAL COMPARISON BETWEEN TWO METHODS FOR DEFINING FUNCTIONAL REQUIREMENTS: USE CASES VS. OO-DFDS" (2011). *MCIS 2011 Proceedings*. 5.
http://aisel.aisnet.org/mcis2011/5

This material is brought to you by the Mediterranean Conference on Information Systems (MCIS) at AIS Electronic Library (AISeL). It has been accepted for inclusion in MCIS 2011 Proceedings by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.

# AN EMPIRICAL COMPARISON BETWEEN TWO METHODS FOR DEFINING FUNCTIONAL REQUIREMENTS: USE CASES VS. OO-DFDS

Dahan, Michal, Ben-Gurion University, Beer-Sheva, Israel, dahanmic@bgu.ac.il

Shoval, Peretz, Ben-Gurion University, Beer-Sheva, Israel, shoval@bgu.ac.il

Sturm, Arnon, Ben-Gurion University, Beer-Sheva, Israel, sturm@bgu.ac.il

## Abstract

*Modeling of functional requirements of an information system is an important task, and there are numerous methods for doing so. We compared two alternative modeling methods for defining functional requirements in object-oriented information system development process: one is OO-DFD transactions - part of FOOM methodology; the other is Use Cases (UC) - part of the UML. The two methods consist of diagrams followed by descriptions that add explanations to the diagrams.*

*In a controlled experiment subjects performed two tasks: a) comprehension of models; b) creation of models. In the first task, each subject was given a set of diagrams and descriptions of a certain system that has been modeled with one of the two modeling methods, and was asked questions aimed to test how well the model is understood. In the second task, each subject was given a narrative requirements document and was asked to create appropriate models using one of the modeling methods.*

*One result of the experiment was that the quality of models created with the UC method is significantly better than those created with the alternative method, though a closer examination of the results showed that the quality for the UC method is better only in one aspect of the method. On the other hand, there was no significant difference in comprehension of the two models. We concluded that while the OO-DFD transaction diagrams are more informative, their descriptions are overly detailed and structure. On the other hand, the UC diagrams are not sufficiently informative. Based on that, we propose an improved method to model functional requirements – Enhanced Use Cases – which combines features from the two explored methods.*

*Keywords: Functional analysis, Modeling method, DFD, Use case, FOOM, UML, Experimentation.*

# 1 INTRODUCTION

Functional requirements specification is a most crucial phase in information system development. In this phase the users' requirements of the sought system are modeled and conveyed to the developers who use these models to develop the system. Many methods for modeling users' requirements were introduced over the years. In the late 70's, structured systems analysis (SSA) methods emerged, which usually utilized the data flow diagrams (DFD) technique (e.g., Demarco, 1978; Gane and Sarson, 1979). One of the biggest problems with those "traditional" methods was the transformation from the analysis phase to the design phase, which involved transformation of the DFDs to Structure Charts; the transformation was not smooth and caused many problems. In order to deal with these problems, the ADISSA methodology, which enables a smooth transformation from the analysis phase and its DFDs to the design phase, was developed (Shoval, 1988). With the emergence of Object-Oriented (OO) development methodologies, Shoval adjusted ADISSA and created FOOM – Functional and Object-Oriented Methodology - that combines the functional and the OO approaches (Shoval, 2007).

In the early 90's, many OO analysis and design methodologies emerged, replacing the "traditional", functional-oriented methodologies. In the late 90's, UML became a "de facto standard" for modeling systems based on the OO approach. UML consists of about 15 types of diagrammatic modeling languages; one of the most popular of them, which is used for modeling the functional requirements, is the Use Case (UC) model. The UC model consists of simple diagrams portraying the interaction between actors and UCs, and narrative description of these interactions, which provide information that is not included in the diagrams.

In this paper, we compare two techniques: the OO-DFD transactions of FOOM, and UCs of UML. In FOOM, two models are created in the analysis phase: a conceptual data model, which is an initial class diagram, and a functional model, which consists of hierarchical OO-DFDs[1]. Later on, the OO-DFDs are decomposed into transaction diagrams, each defining a certain functionality of the system. The use case (UC) model of UML consists of UC diagrams, where each UC defines certain functionality. We choose to compare these two techniques as they are quite similar; each of them enables to define and describe an independent task for a use. This similarity is also noted by Ramsin and Paige (2008).

The remainder of this paper is organized as follows. Section 2 provides a background on the compared methods. Section 3 reviews related work and discusses their limitations. Section 4 provides theoretical arguments about the pros and cons of each of the compared methods. Section 5 describes the experiment; Section 6 presents the results; and Section 7 discusses the results and proposes an enhanced modeling method. Finally, Section 8 concludes and sets the plans for future research.

# 2 BACKGROUND

## 2.1 OO-DFD Transaction Diagrams and their Descriptions

As said, an OO-DFD is a DFD adapted to the OO world: instead of data stores it includes data classes, which are already defined in the initial class diagram. The OO-DFDs are decomposed into transaction diagrams. A transaction consists of one or more elementary functions that are directly connected to each other with data flows, and of external entities and data classes that are connected to those functions. The process logic of each transaction is then described in a top-level, general description. The description is based on the components of the transaction embedded within process logic patterns. Figure 1 shows an example of a transaction diagram that has been extracted from a certain OO-DFD, while Table 1 shows its top-level description. It can be seen that the diagram includes many details, and the description actually repeats and arranges the components of the diagram in a structured way.

[1] OO-DFDs are similar to the "traditional" DFDs but instead of data stores they include data classes that are already defined in the initial class diagram.

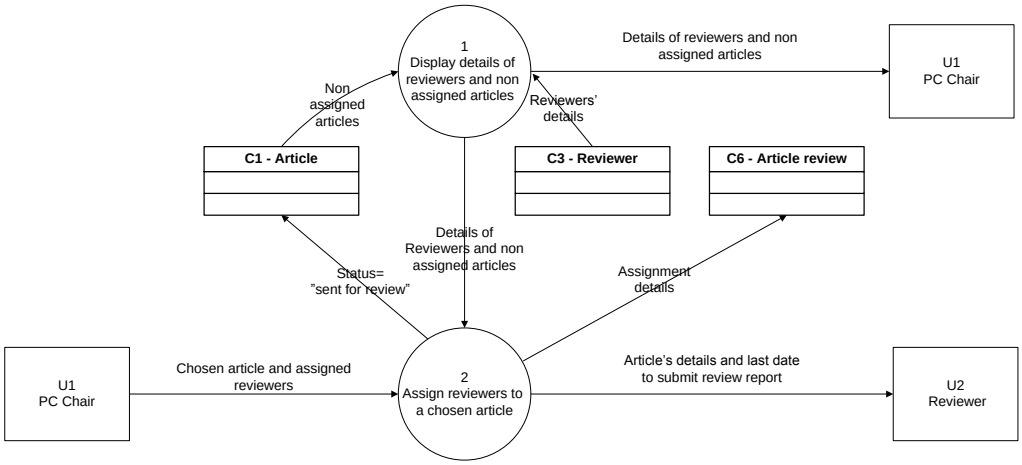


*Figure 1. A transaction diagram.*

| Component | Description |
|---|---|
| Transaction name | Assign reviewers to a submitted article |
| Transaction type | User transaction |
| Top-level description | **Begin** Transaction<br>Read from class C1 (Article): Non assigned articles<br>Read from class C3 (Reviewer): Reviewers' details<br>Execute function 1: Display details of reviewers and non assigned articles<br>Output to U1 (PC Chair): Details of reviewers and non assigned articles<br>Move to function 2: Details of reviewers and non assigned articles<br>Input from user U1 (PC Chair): Chosen article and assigned reviewers<br>Execute function 2: Assign reviewers to a chosen article<br>Write to class C1 (Article): Status = "sent for review"<br>Write to class C6 (Article review): Assignment details<br>Output to U2 (Reviewer): Article's details and last date to submit review report<br>**End** Transaction. |

*Table 1. Description of a transaction.*

## 2.2 Use Case Diagrams and their Descriptions

Use cases (UC) were first introduced by Jacobson (1987) and later on became part of UML. Nowadays, UC became one of the most popular techniques to describe the functional requirements of a system in the OO world. The technique consists of diagrams and descriptions. The main components of a UC diagram are Actor and the UC itself. Actor represents a role of an entity that interacts with the system[2]. Actors are connected to UCs with which they interact by a single association. Figure 2 shows a simple UC diagram. Many ways have been published over the years for describing UCs; a most popular way was proposed by Cockburn (2001). Table 2 shows an example of such description. As can be seen, the UC diagram is very simple but actually contains very little details; only the description of the UC can provide the missing information. Note that the UC description is less structured compared to the transaction description.

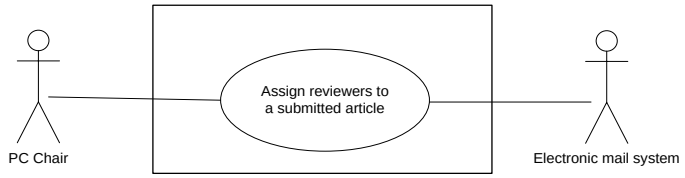


*Figure 2. A UC diagram.*

[2] Actor is not exactly equivalent to external entity in a DFD; while an actor interacts with the UC, an external entity signifies a source of input or destination of output, but not necessarily an operator of the transaction.

| Component | Description |
|---|---|
| UC name | Assign reviewers to a submitted article |
| Description | The PC Chair assigns reviewers to a submitted but not yet assigned article |
| Actors | PC Chair, Electronic mail system |
| Preconditions | There are submitted articles in the system that have not yet been assigned to reviewers |
| Postconditions | The chosen article is assigned to reviewers |
| Main success scenario | The System displays details of reviewers and non assigned articles to the PC Chair<br>The PC Chair selects and inputs to the system an article from the list and, the reviewers assigned to review it<br>The assignment details are saved in the system<br>The system sends the article's details and the last date to submit review reports to the reviewers |
| Extensions | None |

*Table 2. Description of a UC.*

# 3 RELATED STUDIES

Over the years, many studies have been published on comparisons between modeling methods and techniques. For example, Topi and Ramesh (2002) surveyed studies published between 1978 and 2001 that employed laboratory experiments to evaluate the usability of data models/methods. They found out that the most frequent independent variable in those studies was the data model. The dependent variables were mostly model correctness (i.e., the degree to which the model corresponds to a predefined solution), time used to create the model, declarative knowledge and attitude (i.e., preference to use a certain model and perceived ease of use). Surprisingly, only a few studies have been published on the comparison between DFDs and UCs.

Millet and Nelson (2007) compared DFDs and UCs using questionnaires. Their participants were students who studied these two techniques during several years in courses on system analysis. The students have been split into two groups: one got the DFD task prior to the UC task, and vice versa. The first task of the first group was to analyze a simple system using a DFD diagram and a certain CASE tool. Afterwards, the UC method was taught and the students got the second task, which was to analyze the same system but now using the UC method with another CASE tool. For the second group the tasks order was switched. After completing the two tasks, each student was asked to answer a questionnaire consisting of five claims, using a 1-7 point scale. The results were that the participants claimed that the DFD method is better in helping systems analysts communicate requirements to programmers, but there were no differences between the methods in their use, understanding and ability to help the users communicate with system analysts. This research has several drawbacks: a) the use of a questionnaire as the only research method; b) the results are based on the participants' opinions only; c) each student had to use two different CASE tools; there is a possibility that the learning process of each tool affected differently their opinions; d) the tasks were very small, including only 3 processes; e) the students had 2 days to complete each task, with no control on how they worked on the tasks.

Jeyayaj and Sauter (2007) also compared between the DFD and the UC modeling techniques in an experiment. The participants were students: Business Administration students who did not learn the examined methods were considered novice users, and MIS students who studied and practiced the examined methods were considered experienced users. The experiment examined only diagram comprehension. A student registration system was modeled using a DFD and a UC diagram. Each student received the two diagrams one after the other and was asked to explain in free text what he/she understands from each diagram. The main conclusion of the researches was that experienced users believe that DFDs are better than UC diagrams for describing functional requirements. This research too has several main drawbacks: a) the participants were asked to answer only one general question: "what is your understanding of the system from the diagram presented to you?"; this is not a precise way to measure comprehension; b) the students had a week to complete each task, meaning there was no control on the process; c) this experiment dealt only with comprehension of the diagrams.

# 4 THEORETICAL ARGUMENTS ABOUT THE TWO METHODS

Each of the two modeling methods consists of diagrams and descriptions. In comparing the methods we must include both the diagrams and their descriptions, because the diagrams only are not informational equivalent (Siau, 2004; Parsons and Cole, 2005): the UC diagram includes only the UC bubble and actors connected to them; whereas a transaction diagram includes functions (usually more than one in a single transaction), data classes, external entities and directed data flows among them. So, there is no question that a transaction diagram is superior to a UC diagram regarding the amount of information it includes. The inclusion of the descriptions of diagrams makes the methods informational equivalent (and comparable), because they complement the diagrams and enable defining the users' requirements of the UC or transaction. Yet, in the case of transactions, the description is somewhat redundant; it mainly expresses the same information in a structured way, like pseudo-code. On the other hand, in the case of UCs, the description includes details that are not included in the diagram; thus there is much less redundancy. We may conclude that transaction diagrams and descriptions are semantically richer but syntactically more complex compared to a UC diagrams and descriptions. The question remains: which of the two modeling methods is better, and from what perspective.

As mentioned in the survey section, modeling methods can be compared from different perspectives. We chose to compare them from two main perspectives: a) comprehension, i.e., how easy it is to understand diagrams and descriptions of a certain modeling method given to users, and b) quality of creation, i.e., how good are diagrams and descriptions created by modellers who use a certain modeling method.

Several studies address the issues of comprehension and quality of diagrams in conceptual modeling on the basis of cognitive theories; for example, Hahn and Kim (1999), Kim et al. (2000) and Rockwell and Bajaj (2005). Rockwell and Bajaj presented the COGEVAL framework. One of their propositions is that a model which has less relationship information is less comprehensive than a model which has more relationship information. Another proposition of COGEVAL states that a model with more elements will be more complicated to understand than a model with fewer elements due to the limited capacity of the short term memory. In our case, UC diagrams certainly have less relationship information and fewer elements compared to transaction diagrams, and therefore UC diagrams are presumably easier to understand. This assumption can also be supported by Moody (2006), who provides guidelines for creating good modeling diagrams. One of his guidelines is that diagrams should not exceed perceptual and cognitive limits in the sense that it should not show too much information on a single diagram.

So, it seems that with respect to comprehension of diagrams only, there is an advantage to the UC method. But diagrams alone are not informational equivalent and we must consider also their descriptions. As said, the transaction description includes a lot of details but it is mostly redundant with the diagram, while a UC description actually carries most of the semantics of the UC. Recall also that the UC description is less structured than the transaction description – closer to natural language. So, on the one hand we have UC diagrams which are (too) simple, carrying very little information, but complemented with detailed descriptions, and on the other hand we have transaction diagrams which are more complex but include a lot of information along with redundant descriptions. We see here two conflicting "forces" that might affect comprehensibility of the models, and the dilemma is which of them is stronger: a too simple/naive diagram plus a complementing description, vs. a complex/detailed diagram plus a redundant description. Note that in the above two surveyed studies, which dealt with comprehension, the researchers concluded that DFDs are to a certain degree superior to UCs, but their results may be because they did not include descriptions - only diagrams.

With respect to the quality of created models, the issue is which models created by analysts (based on a document expressing the users' requirements) will be of better quality. Quality can be measured in many aspects, e.g., completeness and correctness of the created models compared to the requirements. The quality of a model created by an analyst may be affected by various factors; syntax is certainly an important one. It is obvious that a method that includes, for example, many visual notations, symbols

and rules, is amenable to more errors than a syntactically simple method (Moody, 2009). In our case, it seems that an analyst is likely to make more errors when creating (complex/detailed) transaction diagrams compared to creating (simple/naïve) UC diagrams. On the other hand, creation of a transaction description seems to be easier because it involves mainly translating the diagram into a structured description, while a UC modeller has to spend most of the effort in creating a proper description. But here again comes the syntax issue in play: since a transaction description is more structured than a UC description, there are more chances to commit syntactic errors.

The COGEVAL framework (Rockwell and Bajaj, 2005) too supports the assumption that models created with the UC method may be of better quality that the alternative. One of their propositions states that when a model requires a greater number of simultaneous items to create a diagram, the diagram is of less quality than a diagram of a model which requires fewer numbers of items. As already explained, a transaction diagram includes more types of components compared to a UC diagram. Moreover, a UC description is simpler to create because it is closer to natural language, while a transaction description must include all the details/components of the diagram, and the process logic is expressed in pseudo-code. All this implies that with UCs there are less "chances" to commit errors while writing the descriptions.

If we consider the time aspect with respect to comprehension and quality of created models, we again can speculate that there will be advantage to the UC model. With respect to comprehension, since most of the information is given in the description, the user who reviews a UC can concentrate on the description where all the information he is looking for exists, while the user who reviews a transaction probably views and reviews both the diagram and the description – more time consuming. With respect to quality of model creation, it seems that it will take more time to create a transactions model, firsts because of the need to create semantically and syntactically correct diagrams, and second, because of the need to create (again) a structured description of almost the same thing.

# 5 THE CONTROLLED EXPERIMENT

## 5.1 Experiment Goal and Method

We compared between the two modeling methods from several aspects:

1. Comprehension of models: are there differences in the understanding of diagrams and descriptions by users?
2. Quality of created models: are there differences in the quality of the diagrams and descriptions created by analysts?
3. Time: are there differences in the time it takes to understand diagrams and description, and the time it takes to create diagrams and descriptions?
4. Perceived quality: are there differences in the quality of the methods as perceived by the users or analysts?

Based on these goals, we prepared two experimental tasks: (1) understanding diagrams and descriptions; (2) creating diagrams and descriptions. The participants were divided into two homogeneous groups. Each participant had to perform the two consecutive tasks using one of the two methods.

## 5.2 Experiment Models and Hypotheses

The experiment model for the comprehension task is presented in Figure 3.

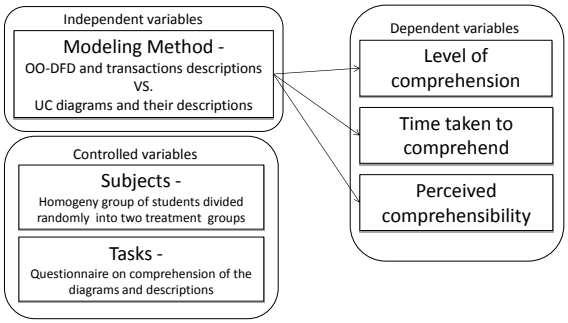


*Figure 3. The comprehension experiment model*

In spite of our speculations and the theoretical background about the possible outcomes of the comparison, we present the null hypotheses for the experiment in a "neutral" way, namely:

$H_1^o$: There is no difference between the two methods regarding the understanding of the diagrams and their descriptions.

$H_2^o$: There is no difference between the two methods regarding the time it takes to understand the diagrams and their descriptions.

$H_3^o$: There is no difference between the two methods regarding the comprehensibility of the model as perceived by the users.

The experiment model for the model creation task is presented in Figure 4.

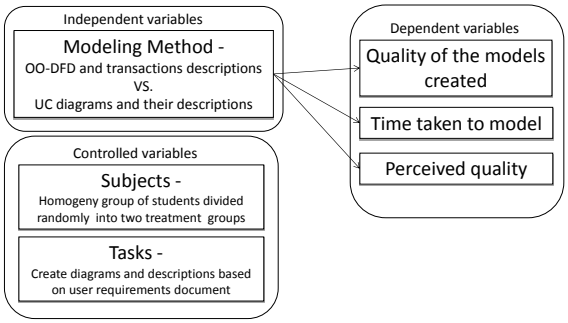


*Figure 4. The quality of model creation research model*

The null hypotheses for this experiment model are:

$H_4^o$: There is no difference between the two methods regarding the quality of the diagrams and descriptions created by analysts.

$H_5^o$: There is no difference between the two methods regarding the time it takes to create the diagrams and descriptions.

$H_6^o$: There is no difference between the two methods regarding the quality of the model as perceived by the analysts.

### 5.3 The Subjects

The participants of the experiment were 3rd year students of Software Engineering. During the same semester they studied two courses in which they were taught the two methods. In the course Information Systems Analysis & Design they studied FOOM, including OO-DFDs and transactions. (This course was lectured by a lecturer who has several years of experience teaching these subjects.) In the course Object-Oriented Analysis & Design they studied UML, including the UC method. (This

course was lectured by one of the authors of this paper.) The experiment was conducted as a mid-term exam but participation was voluntary; the participants were motivated by adding a few bonus points to the course grade.

Fifty three students participated in the experiment. A week prior to the date of the experiment we randomly assigned each participant to one of two groups: group A will experiment with the OO-DFD transactions method only; and group B which will experiment with the UC method only. (Hence, a subject will perform the two tasks with the same method.) Actually we had 28 in the transactions group and 25 in the UC group. Each participant was told in advance to which group (i.e., method) he was assigned. In addition, we gave the participants examples of diagrams and descriptions of the method assigned to each of them. These materials were given prior to the experiment in order to let the participants prepare better for the experiment.

### 5.4 The Tasks

In the first task - comprehension of models - each subject received diagrams and respective descriptions of an information system, according to his/her group. The system for this task was Greeting Cards Ordering. The domain of this system is not familiar to the students and thus there is no concern that the participants will answer the questions due to background knowledge of the domain (Burton-Jones et al., 2009; Parsons and Cole, 2005). For the OO-DFD method, we created a flat OO-DFD of the system that consisted of 5 transactions, and we created a structured description for each of the transactions. Similarly, we created an equivalent UC model that included respective UC diagram and descriptions. The diagram and descriptions were examined by experts of the two methods to verify their correctness. Along with the diagrams and descriptions, each participant received the same questionnaire consisting of 22 statements. For each statement each subject had to mark if it is "true", "false", or "can't tell". Table 3 shows a few examples of statements. Recall that the methods are not fully “information equivalent” (Siau, 2004; Parsons and Cole, 2005; Burton-Jones, 2009). For example, data classes appear in transaction diagrams but not in UC diagrams, but references to classes or data stores will appear in US descriptions because otherwise we cannot assume that they express the same functionality of the users' requirements. We overcame the problem by asking questions that do not refer to components/term that are specific to one of the models. For example, instead of referring to certain data classes that can appear in a transaction only, we just refer to data stores. (Note that the purpose of this task is to measure how well users understand a model presented to them; we do not deal with the question how well a given model represents users' requirements of a system.) Once completed the questionnaire, each subject was given a 1–7 ordinal scale question and a post-hoc question where he/she was asked to express his opinion about the comprehensibility of the model he/she used.

In the second task of the experiment - model creation - we used part of the IFIP Conference System requirements (Mathiassen et al., 2000). All participants received the same requirements document and were asked to model the system using the same method that they used in the comprehension task. Prior to the experiment, we have prepared "expert solutions"; these were used after the experiment by the graders of the models created by each of the participants. We also created a grading scheme for each model to serve as guidelines for the graders; but each solution was graded according to its correctness and completeness compared to the requirements document. We emphasise that the grades were given regarding to the degree in which the created models reflect the necessary requirements defined in the requirements document. For this task too, we prepared a 1–7 ordinal scale question and a post-hoc question to express the perceived goodness of the model used.

| |
|---|
| When a client's order is updated, first the updated order details are inserted and then the card details are read. |
| The client does not need to approve the chosen design details before ordering the desired amount. |
| A message is sent to the client if his order is not updated according to his request. |
| When a client wants to choose a card from the existing variety, the system will display to the client all the cards existing in the collection. |

*Table 3. Examples of the comprehension task statements.*

Grading of the comprehension task was easy because for each participant we only had to count how many correct answers he/she marked. For the model creation task we used two graders for each model. Each grader worked separately (using a separate copy of each exam book); i.e., each model (solution) received two independent grades on this task.

# 6 RESULTS

Before analyzing the results, we had to check that the various grades are normally distributed. For this, we used Kolmogorov-Smirnov test for each of the groups in every dependent variable. The results of these tests showed that only the perceived goodness scores were not normally distributed. Hence, for all variables besides these we could use t-test, while for the latter variables we used the Wilcoxon test.

Table 4 summarizes the results of the **comprehension** task. It shows that there were no significant differences in comprehension, but it took less time to comprehend the UC models, while users perceived that the OO-DFDs method is better.

| | **OO-DFD** | **UC** | **Statistical Analysis (α = 0.05)** | **Null Hypothesis** |
|---|---|---|---|---|
| Comprehension grade | 70.71% | 70.68% | p=0.98 (T-test) | $H_1^o$ is not rejected |
| Time to complete comprehension task | 43 minutes | **35** minutes | p=0.008 (T-test) | $H_2^o$ is rejected |
| Perceived comprehensibility of model | **5.18** | 4.28 | p=0.023 (Wilcoxon) | $H_3^o$ is rejected |

*Table 4. The comprehension task results.*

| | **OO-DFD** | **UC** | **Statistical Analysis (α = 0.05)** | **Null Hypothesis** |
|---|---|---|---|---|
| Creation grade | 73.87% | **87.52%** | p=0.001 (T-test) | $H_4^o$ is rejected |
| Time to complete creation task | 112 minutes | 108 minutes | p=0.53 (T-test) | $H_5^o$ is not rejected |
| Perceived quality of the method for model creation | 4.04 | 3.92 | p=0.96 (Wilcoxon) | $H_6^o$ is not rejected |

*Table 5. The model creation task results.*

Regarding the results of the **model creation** task; first, we had to make sure the grades given by the two graders were correlated. For this, we used Pearson correlation test, which is a mean to measure correlation among graders when the grades are in an ordered scale[3]. The p-value of the two tests was p-value = $0 < \alpha = 0.01$, meaning that the grades of the two graders were correlated. Thus, we could average the two grades of each participant. Table 5 summarizes the results of the creation task. It shows a significant difference in the quality of the created models in favour of the UC method, but no significant differences in the time it took to create the models, and in the users' perceptions about the creation of the models.

# 7 DISCUSSION AND CONCLUSIONS

## 7.1 Analysis of the model Comprehension Results

We found no difference in the comprehension of the two models. This result is in accord with our a-priory speculations, due to the conflicting factors. On one hand, the OO-DFD diagram and the transaction descriptions contain many details that may ease understanding, but on the other hand this

[3] We could not use Kappa's coefficient since it is suitable only for qualitative/categorical items; grades on a 0-100 scale are not categorical.

may confuse users and make the understanding of models harder. Contrarily, the UC diagram contains almost no details, and the UC description is less structured than the transaction description. The simplicity of UCs may ease their understanding but may also make it harder due to the lack of details. These contrary considerations explain the outcomes.

Yet, we found that it takes less time to comprehend model expressed in UC models. If we consider time to comprehend a model as an aspect of its utility, we can conclude that with this respect the UC model is more efficient.

Surprisingly, we found that users perceive the OO-DFD models as more comprehensive; this outcome is difficult to explain because it is in contrast with the outcome on the time dimension. One possibility is that the users thought that since the method is more detailed and demand more precision and more structure – it "must be better".

### 7.2 Analysis for the Quality of Model Creation Results

We found a significant difference between the two methods in favor of the UC method - in accord with our preliminary speculations. Since the creation of the transaction diagrams involved many details, and since the transaction descriptions are more structured, there are many more chances to commit errors when modeling with this method. Contrarily, not only that the UC diagrams are very simple, their descriptions too are written in a more natural language, thus enabling fewer mistakes to be made.

Regarding time to create the models, we found no significant differences. This result is somewhat surprising because we would expect a more demanding task to take more time. The reason for the (almost) indifference in time might be that the participants were allocated the same amount of time for the whole experiment (3 hours), and where advised to use the first hour for the comprehension task, so they reserved about the same amount of the remaining time for the creation task.

Regarding the perceived quality of method for model creation tasks, we found no significant differences. This is somewhat inconsistent with the equivalent outcome regarding perceived comprehensibility of the model, in which there was a significant difference.

### 7.3 An Improved Method for Modeling Functional Requirements

We have seen that there are differences between the two methods, and each has some advantages and disadvantages. Our observations led us to conclude that each of the methods can be improved by considering the advantages of the other. The OO-DFD method can be improved by simplifying the redundant, overly detailed and structured description of transactions, but retaining the structured description of the process logic of the transaction. The UC method can be improved by adding these missing components to the UC diagram, and referring to them in the UC description.

In the following we briefly describe an enhanced UC method that we term EUC – Enhanced Use Case, which combines the advantages of the two compared methods.

A. A EUC diagram will include components that exist in an OO-DFD transaction diagram, i.e., one or more functions, external entities signifying sources of input data and destinations of output, data classes from where the functions can retrieve data or where they can store data, and directed data flows between the respective components. Proponents of existing UC-driven methodologies may ask: where can the data classes come from at this stage? Two answers are possible: a) if we also adopt the FOOM approach, prior to creating the EUCs we create an initial class diagram and use these classes in the respective EUC diagrams; b) even if we do not adopt the FOOM approach, while creating a EUC diagram we may expect the modeler to define not only the functions of the EUC but also the required data classes, as well as the external entities.

B. There is no need to include the "traditional" actors in the EUC diagram; we can list them in the description of the EUC, where we also define other things that are not part of the diagram (e.g., pre- and post-conditions). Another reason for this change is that a EUC may sometimes be

operated by many different types of operators, each having different access privileges; there is simply not enough room to include such details in the diagram.

C. There is no need to include the "special" types of "uses" and "includes" UCs in the EUC diagram. Instead of them, we may have just chained functions; a data flow from one function to another means that the first triggers the latter and may pass some data/parameters to it.

D. The description of the EUC will have to distinguish between the external entities who are source of input or destination of output, and the operators of the use case at runtime. But contrary to the current too-structured description of a transaction, we adopt a less structured, more "natural" description, as in "traditional" UC descriptions.

We complete this "sketch" of the enhanced method by showing a possible description of a EUC. We assume that the EUC is the same as the transaction diagram in Figure 1; Table 6 shows its description.

| **Component** | **Description** |
|---|---|
| EUC name | Assign reviewers to a submitted article |
| Operators | PC Chair |
| Preconditions | There are submitted articles in the system that have not yet been assigned to reviewers |
| Postconditions | The chosen article is assigned to reviewers |
| Main success scenario | The System reads the details of reviewers from class “Reviewer” and non assigned articles from class “Article” and displays them to the PC Chair<br>The PC Chair selects and inputs to the system an article from the list and, the reviewers assigned to review it<br>The assignment details are saved in class “Article review”<br>The system sends the article's details and the last date to submit review reports to the reviewers |
| Extensions | None |

*Table 6. EUC description.*

# 8 SUMMARY AND FUTURE WORK

We compared two methods for modeling the functional requirements of an information system. In a controlled experiment, we examined the differences between the two methods with respect to comprehension, quality, time and perceived goodness of the methods. The main results were that the quality of the models created with the UC method is significantly better than the quality of models created with the OO-DFD transactions method. In addition, we concluded that the OO-DFD transactions are overly detailed while the UC diagrams are too general and not detailed enough. This led us to propose an improved method – EUC - to define and describe the functional requirements. Note that the ECU method has not yet been tested empirically. In the future, we will include evaluation of the enhanced method compared to the original.

Like in other controlled experiment in software engineering, this one too has limitations[4]. For example, it involved relatively small tasks. In further research, we plan to repeat the experiment using tasks of different size and complexity to further verify the results and overcome various limitations.

# References

Burton-Jones, A., Wand, Y. and Weber, R. (2009). Guidelines for empirical evaluations of conceptual modeling grammars. Journal of the Association for Information Systems, 10, 495-532.

Cockburn, A. (2001). *Writing Effective Use Cases*. Addison-Wesley.

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

[4] The limitations are not detailed in the paper due to lack of space.

Gane, C. and Sarson, T. (1979). *Structured System Analysis: Tools and Techniques*. Prentice Hall.
Hahn, J. and Kim, J. (1999). Why are some diagrams easier to work with? effects of diagrammatic representation on the cognitive integration process of systems analysis and design. ACM Transactions on Computer-Human Interaction, 6 (3), 181–213.
Jacobson, I. (1987). Object oriented development in an industrial environment. In: Proceedings of OOPSLA '87, October '87, Orlando, Florida, 183-191.
Jeyayaj, A. and Sauter, V. (2007). An empirical investigation of the effectiveness of system modeling and verification tools. Communications of the ACM, 50 (6), 63-67.
Kim, J., Hahn, J. and Hahn, H. (2000). How do we understand system with (so) many diagrams? cognitive integration processes in diagrammatic reasoning. Information Systems Research, 11 (3), 284–303.
Mathiassen, L., Munk-Madsen, A., Nielsen, P. and Stage J. (2000). *Object Oriented Analysis and Design*. Marko Publishing.
Millet, I. and Nelson, R. (2007). Data flow diagram vs. use cases – student perceptions. Int'l Journal of Information and Communication Technology Education, 3 (1).
Moody, D. (2006). What makes a good diagram? Improving the cognitive effectiveness of diagrams in IS development. In: Proc. of the 15th Int'l Conference in Information Systems Development - ISD.
Moody, D. (2009). The "physics" of notations: toward a scientific basis for constructing visual notations in software engineering. IEEE Transactions on Software Engineering, 35(6), 756-779.
Parsons, J. and Cole, L. (2005). What do the pictures mean? Guidelines for experimental evaluation of representation fidelity in diagrammatical conceptual modeling techniques. Data & Knowledge Engineering, 55, 327-342.
Ramsin, R. and Paige, R. (2008). Process-centered review of object oriented software development methodologies. ACM Computing Surveys, 40 (1), 1-89.
Rockwell, S. and Bajaj, A. (2005): COGEVAL: applying cognitive theories to evaluate conceptual models. Advanced Topics in Database Research, 4, 255-282.
Shoval, P. (1998). ADISSA: architectural design of information systems based on structured analysis. Information Systems, 13 (2), 193-210.
Shoval, P. (2007). *Functional and oObject-Oriented Analysis and Design: An Integrated Methodology*. Idea Group Publishing.
Siau, K. (2004). Informational and computational equivalence in comparing information modeling methods. Journal of Database Management, 15, 73-86.
Topi, H. and Ramesh, V. (2002). Human factors research on data modeling: a review of prior research, extended framework and future research directions. Journal of Database Management, 13 (2), 3-19.