Insights from the Defect Detection Process of IT Experts: A Case Study on Data Flow Diagrams

Gul Tokdemir
Computer Engineering Department
Cankaya University
Ankara,Turkey
e-mail: gtokdemir@cankaya.edu.tr

Nergiz Ercil Cagiltay
Software Engineering Department
Atilim University
Ankara,Turkey
e-mail: necagiltay@gmail.com

Ozkan Kilic
Informatics Institute
Middle East Technical University
Ankara, Turkey
e-mail: ozkankilic@gmail.com

***Abstract*— Design diagrams employed in software development process deliver groups of associated information about the software to be developed. They enhance the perception of the software engineers helping them better understand the software system at various levels of system development process. Today's fast-changing business environment necessitates the reflection of these changes into the operational software systems. Hence, the changes needed in software systems require software engineers to understand the system design diagrams and update them according to the changes. Therefore, it is very important for software engineers to understand and construct the design representations reflecting the software requirements correctly for the success of a software project. In the literature, there are not many studies conducted to better understand the behaviors of software engineers during designing and understanding these representations. Hence, the main aim of this study is to analyze the defect detection process of software engineers during their understanding of Data Flow Diagram (DFD) representations which are used to reveal system processes at different levels of abstraction and data flow requirements between them. Mainly, the question which type of defects can be detected easily is aimed to be answered. The results of this study show that missing information type defects (Missing Process-MP and Missing Dataflow-MD) are harder to detect than the incomplete or incorrect type (incorrect or missing Information-I) of defects.**

Keywords-DFD; software design; diagrammatic reasoning; defect detection.

I. Introduction

Diagrams can be more influential than sentential representations depending on the usage [1], as they communicate, and leverage knowledge that is crucial for solving problems [2]. Diagrams provide condensed information; hence, they are very effective in information systems for transferring information between stakeholders of the system during the system design phase. Moreover, during the software engineering lifecycle phases, they may offer reductions in cost and enhancements in understanding of the system.

During software development, engineers need to understand the system design from the diagrams, transform the system view into programs by viewing whole system, and check for consistency and errors resulting from misunderstanding of the design. As the understanding level of the engineers gets higher, their error correction performance is expected to increase. Finding and correcting these design errors or inconsistencies have a paramount effect in successful system development on time and within the predicted cost.

The aim of this study is to analyze the defect detection process by the software engineers during their DFD reviewing process. We believe that such analysis would provide insights about the design diagrams and software engineer's defect detection process. The results of this study are expected to provide insights to the researchers, software companies, and to the educators to improve DFD cognitive process. The State of the art section below contains related studies found in the literature, Methodology section explains the experiment, Result section analyzes the experiment results and Discussion and Conclusion section talks about the insights gained through this study.

II. State of the Art

Studies report that 40–50% of the development effort is being spent for fixing errors that could be detected and fixed early in the software development process [3]. Hence, defect detection performed early in the software development process is, an essential task as undiscovered defects may cause critical problems later in the process. In this regard, there are many studies mentioning defect detection activity as important, because, as they disseminate to the subsequent development phases, recovery would be more costly and difficult [4][5][6].

Studies also report that, by using model-based approaches, the defect detection rate could be increased in the early stages of the software life-cycle [7][8][9]. Accordingly, many researchers analyzed engineers'

 ISBN: 978-1-61208-367-4

perception of design diagrams and defect detection process of software engineers in ERD [10], DFD [10], and UML [4] and their cognitive processes [11]. For instance, Hungerford et al. [10] states that practice and proficiency in diagrams improve defect detection process of software engineers. Kumaresh and Baskaran [5] report that analysis of the defects at early stages of the software development lifecycle reduces development time, development cost and the resources required for the process.

Even though the DFD modeling language is over 30 years old, because of its usage history and familiarity among the software developers, many researchers, today, based their studies on this notation [12][13]. Additionally, since most of the current software systems' documentations are based on the DFD notations, for maintenance procedures the technicians are still required to better understand this notation. For instance Yuwen and Wang [14] report the drawing of DFD is the key technology in the development of system analysis and design [14]. According to them, DFD is not only the key composing part of the logic model in new system, but also the key basis in the system physical designing [14].

However, in the literature, there are not many studies conducted to better understand the reviewers' performance during the defect detection process. For instance, Moser and Biffl report that the missing or incorrect type of information is often detected in a later engineering process step [15]. Hence understanding the defect types that cannot be detected easily could help the software system designers to better represent this type of information in their representations. Additionally, this information also can be used to better guide the reviewers in different phases of software development process accordingly.

Hence, in this study, defect detection process of software engineers during their DFD reviewing process is analyzed to obtain insights about the cognitive processes of the engineers. Mainly, three different types of defects, namely, Missing Process (MP), Missing Dataflow or information (MD) and incorrect or missing Information (I) have been seeded into the DFD representations. The following research question is aimed to be answered is 'Which types of defects (MD, I, or MP) are easy to detect in DFD representations?'

Data are collected through interviews and observations while the IT experts work on the corresponding materials in defect detection.

III. METHODOLOGY

The experimental study is conducted with 4 participants using a study material which is derived from the study of Hungerford et al. [10], which is adapted to the current settings of this study and translated into Turkish. Participants of this study were software engineers with average age of 32 (Table 1).

TABLE I. PARTICIPANTS' INFORMATION

Participant	Age	Experience in the field	Gender	Experience with DFD
P1	29	8	F	8
P2	28	7	M	1
P3	34	12	M	2
P4	35	12	M	3
Average	32	10		3

We have prepared two DFDs of the system with 17 defects seeded in total at two levels. The participants have been provided the system description one week before the experiment. During the experiment, participants were asked to find the defects seeded in the DFDs, based on the system description.

The defects are categorized into three types: MP, MD and I. Table 2 summarizes the number of defects in the DFDs according to each category defined here.

TABLE II. NUMBER OF DEFECTS IN EACH CATEGORY

Code	Description	# of Defects
MP	Missing Process	2
MD	Missing Dataflow/information	9
I	Incorrect/ Incomplete	6
	Total	17

Table 3 depicts the defects seeded into both DFDs with their defect types. Figure 2 shows the locations of the defects (Fig. 1) at level 1 and 2 (Fig. 2).

TABLE III. DEFECT EXPLANATIONS

Defect	Description	DFD	Defect Type
01	End of job proposal process (1.4) is missing	1	MP
02	The entity named accounting should be job costing section	1	I
03	Job request data should go from customer to 1.1. Job Evaluation Process	1	MD
04	Receipt information should go from process "1.5 Payment Monitor" to the Customer entity	1	MD
05	Job proposal data flow should go data store named D2, instate of entity named accounting	1	I
06	The data flow from data store D1 to process 1.2 should be part information not customer information	1	I
07	From entity named customer, to the missing process named end of job proposal (1.4), rejection information should go	1	MD
08	the missing process named end of job proposal (1.4) to the data storage named D2, end of job proposal information should go	1	MD
09	From process 2.1 to the process 2.2 purchase order information should go	2	MD
10	From the entity supplier to the process 2.2, approval date and time information should go	2	MD
11	The Data storage named D7 should be supplier account, not customer account	2	I
12	From the process 2.2 to the storage D5, instate of customer information, part	2	I

	information should go		
13	The direction of the data flow (order form) from the entity supplier to the process 2.2 is incorrect. It should be from the process 2.2 to the entity supplier	2	I
14	Process 2.3 delivery is missing	2	MP
15	From the data storage D8 to the missing process 2.3, order form information should go	2	MD
16	From the missing process 2.3 to the process 2.2, delivered part information should go	2	MD
17	From the missing process 2.3 to the data storage D1, delivered part information should go	2	MD

In Figure 1, there are five processes describing top level relationships and data flow between processes. These five processes define the top level diagram of an ERP sales function module of a company. They include request evaluation, proposal preparation, work order preparation, work order close-up and payment follow-up processes. These processes connected to each other through data flows. Moreover, data is accumulated in data stores called customer account, work order/proposal and personnel.

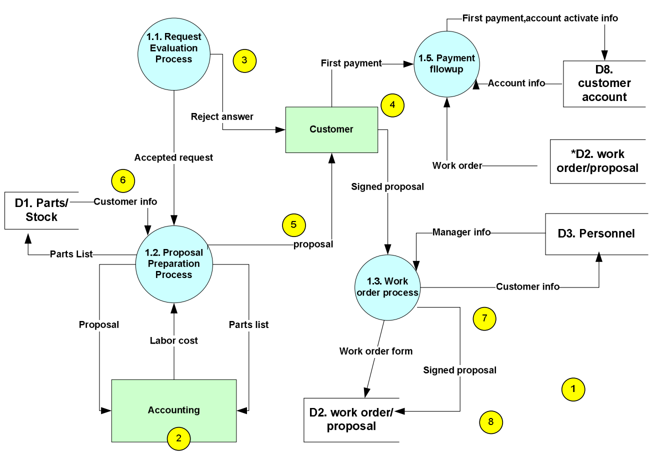


Figure 1. Defects' Placement in DFD_1

Similarly, Figure 2 depicts three sub-processes of proposal preparation process and their data flow. It has three processes which define second level DFD of proposal preparation process. They include parts/stock, order and delivery operations processes. These processes connected to each other through data flows. Moreover, data is accumulated in sx data stores called parts/stock, work order/ proposal, order form, customer account and supplier info.

As seen from Figures 1 and 2, the defects were seeded into two DFD diagrams and the participants were asked to detect them and take notes. During this process, the participants were allowed to check the system description. In the following section, the results of the defect detection process are provided.

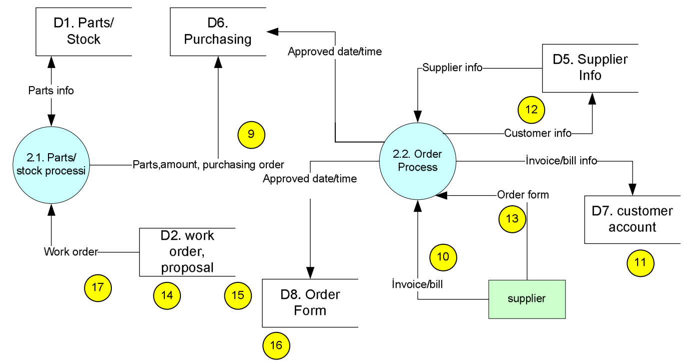


Figure 2. Defects' Placement in DFD_2

In this study, data is collected through Defect Detection Report used by the reviewers, observation notes and semi-structured interview sessions conducted by each reviewer. The defect detection report has the defect numbers and the explanation for the defects found. By using this form, the reviewers were asked to note each defect that they detect and describe their opinions about this defect as explained in the explanation document provided in Appendix A. The observations were conducted by one researcher and observation notes were taken during each reviewer's defect detection process. The durations spent for detecting each defect were recorded during the observation sessions and later synchronized with the reported defects in the Defect Detection Report. Additionally, by the same researcher, a semi-structured interview session was conducted by each reviewer individually. The interview sessions took around 30 minutes. The semi-structured interview questions were formed as below:

1. Which types of defects were easy to detect for you?
2. Which defects were hard to detect for you?
3. Which factors do you think helped you to detect the defects easily?
4. Which factors do you think maked it hard to detect the defects?

This study is conducted with the contribution of four participants who were asked to detect 17 defects seeded in two DFDs. Since the main research question of this study is based on the defects, the results of this study based on 68 cases (17 times 4). Additionally, this study aims to focus on the behaviors of the participants in order to uncover the complexity of human behavior in such a framework and present a holistic interpretation of what is happening during the review process. Nielsen and Landauer [16] also report that studying with four or five subjects is enough to understand and explain more than 80% of the phenomena. Accordingly, in this study, the participants' behaviors are analyzed in depth from different dimensions. Since each participant studied individually, we believe that this number of subjects could provide a view for understanding the phenomena.

 ISBN: 978-1-61208-367-4

IV. RESULTS

Table 4 shows the duration in seconds that each participant (D_{Pij}) spent during each defect detection process.

TABLE IV. DEFECT DETECTION DURATION DATA

Defect Type	Defect	Dp_{1j}	Dp_{2j}	Dp_{3j}	Dp_{4j}	AD_i
MP	14				993	993
I	13	386	60	678	70	299
MD	16				256	256
MD	07			145	236	191
I	12		114	347	69	177
MD	09		147			147
I	06	162	88	70	214	134
MP	01	133	113		114	120
MD	04	103				103
MD	03			89		89
I	02		6	163	47	72
I	11		45			45
I	05		36			36
MD	08					
MD	10					
MD	15					
MD	17					

As an example, in this table, D_{p1} is calculated from the observation data which shows the duration in seconds that the participant P1 spend time for detecting the defect i (D_i). It is the duration starting from the time point of last defect detection process until the defect detection of D_i. AD_i is the average of the durations spent by each participant to detect defect i (D_i). As seen in Table 4, the defects D_{08}, D_{10}, D_{15} and D_{17} were never detected. It is interesting that the defect type of all of these defects that were not recognized by any of the reviewers was MD type. On the other hand, most of the defects of type I, detected in relatively less time spent (D_2, D_5, D_{11}). Similarly, the participants spent more time for detecting defects D_{14} and only one participant could be able to detect this defect.

We have analyzed this data according to the defect types, as shown in Table 5. Accordingly, the detection rate for missing Information (I) type of defects is calculated as 16/24=0.67. Hence, defects of type I and MP were detected mostly; on the other hand the defects of type MD were detected seldom.

TABLE V. DETECTED DEFECT TYPE

Defect Type	Total Possibilities	Total Detected	Detection Rate
I	24	16	0.67
MP	8	4	0.50
MD	36	6	0.17

The detection frequency F_i of defects is shown in Table 6. In this table, F_i represents the frequency of a detected defect by participants. Its value is calculated by adding 1 point for each defect's detection for defect i (D_i). For example, if the defect is detected by only one participant this value is 1, if it is detected by three participants the Fi value for that defect is calculated as 3. As seen from Table 6, four defects 08, 10, 15 and 17 were never detected.

TABLE VI. DEFECT FREQUENCY F_i

Defect Type	Defect	F_i
I	06	4
I	13	4
MP	01	3
I	02	3
I	12	3
MD	07	2
MD	03	1
MD	04	1
I	05	1
MD	09	1
I	11	1
MP	14	1
MD	16	1
MD	08	
MD	10	
MD	15	
MD	17	

The average frequency of defect detection according to the defect types are given in Table 6. As seen from this table, the MD types of defects are detected less frequently, and the defect of type I detected most frequently. Parallel to this finding during the interviews, three reviewers (P2, P3, P4) reported that missing type of information were hard to detect. For instance, P3 reported that "the missing procedures were very hard to detect for me". Similarly, during the interviews, two reviewers (P1 and P2) reported that data flows were easy to understand. For instance P2 reported that "Detecting the data flow directions were easy. I easily detected the incoming and outgoing data. It was also easy to decide the data flow to each data store and which data should be read from a data store. Detecting the data, that supposed to go to a data-store but not shown in the design, was also easy". Moreover, we have asked participants about the factors that helped them to find the defects easily. They noted that the diagrams used to describe process were easy to detect. They stated that the data flows and external storages were difficult to follow in the diagrams. They said bigger and more detailed shapes with color would have increased the understandability of these diagrams.

V. DISCUSSION AND CONCLUSION

In this study, an experiment is conducted to analyze defect detection performance of software engineers in reviewing DFD diagrams. During the experiment, we had provided materials to the participants, one week before the experiment (Appendix A) and requested to find defects on DFD diagrams compared to the explanations given. They were asked to think aloud. We have recorded defect detection duration of each participant. The results of this study show that, missing information type defects (MP and MD) are harder to detect than the incomplete or incorrect type (I) of defects. Hence the defect detection frequency of defects in average is higher for of type I defects (2.67) that that of type MP (2.00) and type MD (1.20) defects. Similarly, the detection rate of type I defects (0.67) is higher than that of type MP (0.50) and type MD (0.70) defects.

 ISBN: 978-1-61208-367-4

According to the results of this study, the software system designers may reconsider their designs especially for the defects of type missing information, which are harder to be detected in the future and may increase the cost of software projects. We believe that further analysis of the DFD defect detection process is expected to provide more insights to the researchers, software companies, and to the educators to improve DFD cognitive process.

REFERENCES

[1] J.H. Larkin and H.A.Simon, "Why a diagram is (sometimes) worth ten thousand words," Cognitive Science, 1987, vol. 11, pp. 65-99.

[2] J. Zhang, "The nature of external representation in problem solving," Cognitive Science, 1997, vol. 21 i2. 179-217.

[3] B. Boehm and V.Basili, "Software defect reduction top 10 list", IEEE Computer, vol. 34, pp. 135–137, January. 2001.

[4] O. Laitenberger, C. Atkinson, M. Schlich, and K. El Emam, "An experimental comparison of reading techniques for defect detection in UML design documents," Journal of Systems and Software, August. 2000, vol. 53 n.2, pp. 183-204.

[5] S. Kumaresh and R. Baskaran, "Defect analysis and prevention for software process quality improvement," International Journal of Computer Applications, 2000, vol. 8 i7. 42L 47.

[6] G. Travassos, F. Shull, M. Fredericks, and V.R. Basili, "Detecting defects in object-oriented designs: using reading techniques to increase software quality," ACM SIGPLAN Notices, October. 1999, vol. 34 no. 10, pp. 47-56.

[7] R. Alur and A.Chandrashekharapuram, "Dispatch sequences for embedded control models", In Proc. 11th IEEE Real-Time and Embedded Technology and Applications Symp. 2005, vol. 11, pp. 508–518.

[8] L. Kof "Scenarios: identifying missing objects and actions by means of computational linguistics", In Proc 15th International Requirements Engineering Conference, pp. 121–130, 2007.

[9] L. Kof, R.Gacitua, and M. Rouncefield, P.Sawyer, "Ontology and model alignment as a means for requirements validation", in International Conference on Software Engineering, pp. 46–51, 2010.

[10] B.G.Hungerford, A.R.Hevner, and R.W.Collins,"Reviewing Software Diagrams: A Cognitive Study,", IEEE Transactions on Software Engineering, February, 2004, vol. 30 no. 2, pp. 82-96.

[11] K.A. Ericsson and H.A.Simon, Protocol Analysis: Verbal Reports as Data. revised edition, Bradford Books/MIT Press, Cambridge, MA 1993.

[12] F. Chan, "The Role and Mechanism of Analogical Transfers in Novices' Data Flow Diagram Problem Solving: The Effects of an Explicit Hint and Alternative Training Methods, Senior Honors Thesis, University of Hawaii, 2014.

[13] V. Repa, Object-Oriented Analysis with Data Flow Diagram. InInformation Systems Development (pp. 419-430), 2013, Springer, New York.

[14] S. Yuwen and K.Wang, A Method of Data Flow Diagram Drawing Based on Word Segmentation Technique. In Frontier and Future Development of Information Technology in Medicine and Education, pp. 3269-3274, 2014, Springer, Netherlands.

[15] T. Moser and S.Biffl, "Semantic tool interoperability for engineering manufacturing systems" In Proc. Emerging Technologies and Factory Automation (ETFA), IEEE Conference, pp. 1-8, 2010.

[16] J. Nielsen and T.K. Landauer, "A mathematical model of the finding of usability problems", Proc. ACM INTERCHI'93 Conference, pp. 206-213, 1993.

APPENDIX A

Problem Definition

Assume Mavi Company has business in pipe sector. The company's work and process descriptions are given below.

There are several types of employees working for Mavi Company, such as managers, sales staff and security guards. Telephones are shared and several employees may have the same office address. Security guards may be assigned to both buildings and car parks. Sales staff provides consultation services to customers by phone or face to face. Customers are assigned to exactly two salespersons and employees work with other employees in teams.

Each department can have more than one unit of the company. Personnel works in the units and each employee can work in one unit. Unit numbers and unit names are only defined uniquely in that department.

Customers can make job requests to Mavi Company. Mavi Company may reject this request, or if accepts, it prepares a job proposal and sends it to the customer.

When a job proposal is prepared, necessary parts' information is retrieved from parts file. Unit labor costs for parts are retrieved from job costing section. In this way, prepared job proposal is sent to the customer. Customer may accept or reject the proposal. If the customer rejects it, job proposal is closed. If accepted, the proposal is signed and the first payment is withdrawal.

Accepted job proposal is used to create a work order to follow the request in the company. For each customer's each job proposal, a single account is opened. A manager is appointed for each work order. Some work orders may include several customers. Orders associated with each other, brought together more than one job are classified as a new project. First invoice is sent to the customer at this step.

After the work orders are prepared, the necessary parts are controlled from the stock. If the parts do not exist in the stock, purchase is made using the amount information. According to the purchasing information, suppliers are identified; invoice is prepared and sent to the supplier. When the supplier approves the invoice, date and time is recorded. Each manufacturer must have a separate account. The supplier should provide invoice for the manufactured parts. This information is used to update the supplier info. Invoices are controlled as the parts are delivered. After the delivery, part information is updated in the stock.

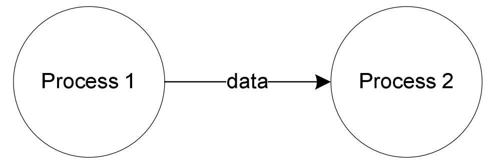
Special promotional campaign is created for important projects. These campaigns are handled either by Mavi Company, or by a local organization like a school or an art festival. Projects cannot be performed by both campaigns. Each campaign introduces a single project.

SOME DESCRIPTIONS ABOUT THE RESEARCH

1. Assume you are employed to analyze the software system of Mavi Company. In this document, you are given information about the business process of Mavi Company.
2. You are required to use this information in analyzing the system to find the possible errors/mismatches. These errors/mismatches may exist because of incomplete or incorrect requirements.
3. The errors/mismatches you found should be based on the system definition and the other supporting documents presented to you earlier. Assume the document describes the company processes correctly.
4. In this study, you are not required to create new solutions to solve the problems or not required to fix these problems.
5. You are given 2 hours to find the induced errors/mismatches. Please adjust your time accordingly.
6. Identify errors/mismatches and list them on the forms provided. To describe the error/mismatch, if possible, please specify the related process(es) and data-flow information. If not possible, please use most appropriate way to explain the error/mismatch.
7. You can use any method or technique to find the Identify errors/mismatches. However, during the process, please don't interact with anyone else.
8. In identifying the errors/mismatches, you can review the documents provided to you as you want.
9. Please, try to think loudly as you are analyzing the system design. While you are reading and interpreting the documents, try to talk loudly. please, please. In particular, when you identify errors/mismatches, please indicate your findings loudly.

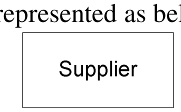
DFD Notations

1. The DFD diagrams used in this study are developed by Visio. The processes are represented by circles; the data flow is represented through arrows as described below.



2. Data storage is represented as below:

3. External entity is represented as below:



4. In this study, you are given Context diagram (Level-0 DFD) and DFD of two processes in detail (Proposal Preparation process and Stock control/ Proposal process).

5. There are 17 defects in DFD diagrams These can be missing process, Missing Dataflow/information, Incorrect/ Incomplete data flow type defects

6. The top level process definitions are given in the figure below. Proposal Preparation and Stock control/ Proposal processes' DFD will be given during the experiment.