December 2004

Data Flow Diagrams versus Use Cases - Student Reactions

Robert Nelson
Penn State University

Ido Millet
Penn State Erie

Follow this and additional works at: http://aisel.aisnet.org/amcis2004

Recommended Citation
http://aisel.aisnet.org/amcis2004/348

This material is brought to you by the Americas Conference on Information Systems (AMCIS) at AIS Electronic Library (AISeL). It has been accepted for inclusion in AMCIS 2004 Proceedings by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.
Data Flow Diagrams versus Use Cases
– Student Reactions

Robert Nelson
Penn State Erie
rjn2@psu.edu

Ido Millet
Penn State Erie
ixm7@psu.edu

ABSTRACT
Data Flow Diagrams and Use Cases are two popular methodologies in teaching as well as in practice. We recently used both methodologies in teaching four sections of our required Systems Analysis course. Questionnaire results indicate that students prefer Data Flow Diagrams.

Keywords
Process Modeling, Use Cases, Data Flow Diagrams

INTRODUCTION
The Data Flow Diagram (DFD) technique had been introduced in the late seventies (DeMarco, 1978, Gane and Sarson, 1979) and has become a popular process modeling tool for information systems. Research has shown that DFDs are also one of the most common tools taught in Systems Analysis and Design courses (McLeod, 1996).

While many believe that object-oriented design methodologies provide an “easier modeling process” and “improved communication” among developers as well as between developers and users (Johnson et al., 1999), empirical studies seem to disagree. Empirical research by Vessey and Conger (1994) shows that DFDs are easier to learn and to use, at least by novice users. Similarly, Agarwal et al. showed that DFDs produce higher-quality solutions in process oriented tasks and are not inferior to object-oriented methodologies even in object-oriented tasks (Agarwal and Atish, 1996).

In our own Systems Analysis course we have been using a simplified version of DFDs as proposed by Millet (1999). The CASE tool we have been using for DFDs is Sybase’s ProcessAnalyst. In the Fall 2003 semester, we added the Use Case methodology and Rational CASE tools to the course. We used questionnaires to evaluate student responses to the DFD and Use Case methodologies in order to investigate novice user perceptions of these two competing methodologies.

To our knowledge, this is the first empirical investigation comparing the Data Flow Diagram and the Use Case methodologies. Since both methodologies aim to model the services provided by a system, such a comparison is both meaningful and warranted.

RESEARCH DESIGN
Two sections of our Systems Analysis Fall 2003 course were introduced to structured analysis techniques as well as object-oriented methodologies. We assigned each section, one with 30 students and the other with 19, to two different treatment groups. As shown in Table 1, Group 1 (DFD First) was introduced to data flow diagram concepts during Lecture #1. In the next class (Lab #1), this group was given a lab session and an assignment demonstrating the principles of data flow diagrams (DFD) using Sybase’s ProcessAnalyst DFD CASE tool. During Lecture #2, Group 1 was given a lab session and an assignment demonstrating Use Case diagrams using Rational Rose as the CASE tool. During the 5th class meeting, students were asked to complete a questionnaire (See Appendix A) comparing the two methodologies.

As shown in Table 1, the same approach but in reverse sequence was taken with Group 2 (Use Case First). This group was exposed to Use Cases before they were exposed to Data Flow Diagrams.
QUESTIONNAIRE

The questionnaire, consisting of student classification by semester standing, major and five knowledge assessment questions, was administered to two systems analysis classes. Both classes were composed of MIS students, both majors and minors. Most students were juniors or seniors because the systems analysis course is recommended in the 6th or 7th semester of the MIS curriculum. The questions required students to respond to a Likert-type scale ranging from ‘1’ to ‘7’ with ‘1’ designated as “Strongly Disagree” to ‘7’ indicating “Strongly Agree”. Students were asked to provide comments for each of the five knowledge assessment questions.

RESULTS

As we write this version of the paper, we can only analyze the results from two sections (49 students) of our Fall 2003 Systems Analysis course. Additional results will become available at the end of the Spring 2004 semester as a repeat of the same study concludes for two more sections.

Appendix B provides a summary of the results so far. The information is presented in a way that facilitates comparisons within group (DFD First or Use Case first) as well as overall. The DFD methodology received higher average evaluations for all questions (overall and within treatment groups) except for the ‘DFD First’ treatment group, which rated Use Cases slightly higher on the ‘Ease of Use’ question. However, probably due to the small sample size, paired t-tests indicate that only a few of the differences are statistically significant.

The results indicate that the Use Case methodology is not rated as significantly better than the DFD methodology on any of the five questions. This holds true not only overall but also within each of the two treatment groups. The DFD methodology is rated as significantly better than the Use Case methodology on the questions of ‘Ease of Understanding’ (t = 1.94; p = .06), ‘Help in Extracting & Validating Requirements from Users’ (t = 2.04; p = .05) and ‘Help in Communicating Requirements to the Programmer’ (t = 5.6; p = .00).

The most extreme differences in ratings across the two methodologies were in relation to their effect on helping systems analysts communicate requirements to programmers. The students clearly perceived Data Flow Diagrams as a better mechanism in that respect. Perhaps this could be explained by the fact that DFDs allow systems analysts to model required system services in a hierarchy of functionality that terminates with relatively small chunks written descriptions (primitive specifications). In addition, as reflected by their comments, students seem to perceive the explicit depiction of data flows as an advantage.

The Use Case methodology was perceived by the students as relatively difficult to understand (mean = 5.2) rather than difficult to use (mean = 5.8). Another interesting finding is that exposing students to the DFD methodology first, made it easier for them to understand and use the Use Case methodology. This suggests that Systems Analysis courses should cover the Use Case methodology after covering the DFD methodology (Shelly et al., 2001).
STUDENT COMMENTS

As part of the questionnaire, we requested students to provide written comments about each question. The following list provides sample comments we received for each question:

Question #1 – Easy to Understand

- “The DFD methodology is fairly easy to understand and they allow a system to be planned out and understood with significantly less explanation.”
- “DFDs seem to be much easier to follow. The flows are straight forward.”

Question #2 – Ease of Use

- “Both are easy to use. I like using DFDs because of the different levels; and you can see more clearly what it should look like.”
- “The software packages seem very easy to use and an easy way to develop a visual flow diagram.”

Question #3 – Help Users Communicate Requirements to the Analyst

- “I think the DFDs explain the process much better and explain what is expected of all the components of the system.”
- “I understand things visually; I think the DFDs overall do a better job of drawing out the system for the analyst.”

Question #4 – Help Analyst Extract & Validate Requirements from Users

- “DFDs show more detail and requirements that the Analyst can explain to the user and the user can determine what is or is not necessary.”
- “DFDs were more technical and provided more information which allows for better communications of requirements.”

Question #5 – Helps Analyst Communicate Requirements to the Programmer

- “The DFD would be better for the programmer to understand. The Use Case is extremely vague while the DFD is more in depth.”
- “I think the DFD would provide more detail which would give the programmer a better picture of what is taking place.”

These comments seem to reflect positive but realistic views about each methodology.

CONCLUSION

Our students found Data Flow Diagrams easier to understand and better at helping systems analysts communicate with users and developers. However, they didn’t see the two methodologies as significantly different in ease of use or in their ability to help users communicate with systems analysts.

These findings indicate that rather than replacing DFDs with Use Cases, there is clearly a place for both methodologies in the IT curriculum. This, obviously, raises the question of how to structure our courses to accommodate both methodologies [Gabbert, 2000]. According to comparisons across the two treatment groups (DFD first versus Use Case first) exposing students to the DFD methodology first should be the preferred approach.

By the time of the presentation, additional results will become available from the spring 2004 semester. Based on the results of including the additional data, we will suggest future research areas to investigate.

REFERENCES


APPENDIX A: MISBD 430-01 – METHODOLOGY QUESTIONNAIRE, FALL 2003

Semester Standing:  □ Sophomore  □ Junior  □ Senior

My Major:  □ Accounting  □ Finance  □ ECNS/BECON  □ Management  □ Marketing  □ MIS  □ Undecided  □ Other

<table>
<thead>
<tr>
<th>Use Cases</th>
<th>DFDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strongly Disagree</td>
<td>Neutral</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

This methodology is **easy to understand**.

Comments (what aspects in what methodology are difficult or not intuitive?)

This methodology/software is **easy to use**.

Comments (what aspects are difficult or not intuitive?)
<table>
<thead>
<tr>
<th>Strongly Disagree</th>
<th>Neutral</th>
<th>Strongly Agree</th>
<th>Strongly Disagree</th>
<th>Neutral</th>
<th>Strongly Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>

This methodology really helps **Users** to communicate and confirm requirements to the Systems Analyst.

Comments: why do you think one methodology would be better or easier for the user to communicate requirements to the Systems Analyst and then verify that the Systems Analyst understood the requirements?

This methodology really helps **Systems Analysts** to extract and validate requirements from the user.

Comments: why do you think one methodology would be better or easier for the Systems Analyst in understanding user requirements and verifying with the user that the requirements are correct?

This methodology really helps **Systems Analysts** to communicate requirements to the programmer.

Comments: why do you think one methodology would be better or easier for the Systems Analyst in communicating specifications to the programmer?
## APPENDIX B: QUESTIONNAIRE RESULTS

<table>
<thead>
<tr>
<th>1.1.1.1.1 Treatment</th>
<th>DFD First (n = 30)</th>
<th>Use Case First (n = 19)</th>
<th>ALL (n = 49)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1.1.1.1.1 Question</td>
<td>DFD</td>
<td>U.C.</td>
<td>DFD</td>
</tr>
<tr>
<td>1. Easy to Understand</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>5.6</td>
<td>5.3</td>
<td>5.6</td>
</tr>
<tr>
<td>(Variance)</td>
<td>(0.7)</td>
<td>(1.7)</td>
<td>(0.7)</td>
</tr>
<tr>
<td>Paired t-test α (two-tail)</td>
<td>0.31</td>
<td>0.08*</td>
<td>0.06*</td>
</tr>
<tr>
<td>t Stat</td>
<td>1.03</td>
<td>1.88</td>
<td>1.94</td>
</tr>
<tr>
<td>2. Easy to Use</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>5.8</td>
<td>5.9</td>
<td>5.8</td>
</tr>
<tr>
<td>(Variance)</td>
<td>(1.1)</td>
<td>(1.0)</td>
<td>(1.2)</td>
</tr>
<tr>
<td>Paired t-test α (two-tail)</td>
<td>0.41</td>
<td>0.86</td>
<td>0.76</td>
</tr>
<tr>
<td>t Stat</td>
<td>-0.83</td>
<td>0.18</td>
<td>-0.31</td>
</tr>
<tr>
<td>3. Helps Users Communicate &amp; Confirm Requirements with Systems Analyst</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>5.5</td>
<td>5.1</td>
<td>5.3</td>
</tr>
<tr>
<td>(Variance)</td>
<td>(1.5)</td>
<td>(2.9)</td>
<td>(1.6)</td>
</tr>
<tr>
<td>Paired t-test α (two-tail)</td>
<td>0.28</td>
<td>0.80</td>
<td>0.31</td>
</tr>
<tr>
<td>t Stat</td>
<td>1.09</td>
<td>0.25</td>
<td>1.02</td>
</tr>
<tr>
<td>4. Helps Systems Analysts Extract &amp; Validate Requirements from Users</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>5.4</td>
<td>5.2</td>
<td>5.5</td>
</tr>
<tr>
<td>(Variance)</td>
<td>(1.2)</td>
<td>(1.5)</td>
<td>(1.4)</td>
</tr>
<tr>
<td>Paired t-test α (two-tail)</td>
<td>0.41</td>
<td>0.04**</td>
<td>0.05**</td>
</tr>
<tr>
<td>t Stat</td>
<td>0.82</td>
<td>2.16</td>
<td>2.04</td>
</tr>
<tr>
<td>5. Helps Systems Analysts Communicate Requirements to the Programmer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>5.4</td>
<td>4.3</td>
<td>6.2</td>
</tr>
<tr>
<td>(Variance)</td>
<td>(2.3)</td>
<td>(2.6)</td>
<td>(1.1)</td>
</tr>
<tr>
<td>Paired t-test α (two-tail)</td>
<td>0.00***</td>
<td>0.00***</td>
<td>0.00***</td>
</tr>
<tr>
<td>t Stat</td>
<td>3.75</td>
<td>4.28</td>
<td>5.60</td>
</tr>
</tbody>
</table>

* p < .10  ** p < .05  *** p < .01