Supporting Project Tasks, Resources, Documents, and Defects Analysis in Software Project Management

A dissertation presented to
the faculty of
the Russ College of Engineering and Technology of Ohio University

In partial fulfillment
of the requirements for the degree
Doctor of Philosophy

Khaled M. Jaber
August 2016
© 2016 Khaled M. Jaber. All Rights Reserved.
This dissertation titled
Supporting Project Tasks, Resources, Documents, and Defects Analysis in Software Project Management

by
KHALED M. JABER

has been approved for
the School of Electrical Engineering and Computer Science
and the Russ College of Engineering and Technology by

Chang Liu
Associate Professor of Electrical Engineering and Computer Science

Dennis Irwin
Dean, Russ College of Engineering and Technology
Software development consists of different phases: initiation, planning, execution, monitoring and control, and closing. At the initiation phase a project is approved and a project manager, PM, is assigned. At the planning phase PM defines the project schedule, cost, tasks (work items), resources (team), and assigns project tasks to resources. At the execution phase, project tasks are implemented. At the closing phase, the project is delivered to customer. Across all phases, PM continue to monitor, analyze, manage, and control the execution of the project. The objective here is to keep the project under control and deliver the project on time and within planned budget.

This dissertation addresses the issues of managing project tasks, resources, documents, and software defects. PMs utilize project management software to manage project schedule, tasks, and resources. These systems provide visualizations to display project information (e.g., task name, resource name, task duration, task start date, defect ID, defect description, defect severity, etc.). To help PMs analyze and manage project schedule, tasks, and resources, they currently utilize common two-dimensional (2D) visualization methods such as Gantt charts and tables/spreadsheets. They also utilize defect tracking systems and common 2D visualization to analyze and manage the software defects found during the development of software systems.
The common 2D visualizations currently supported by project management and defect tracking systems have these limitations: it is difficult to see the entire schedule in a single view especially in the case of large data, they do not display analysis information, and they do not support interacting, e.g., rotating the view, with the displayed data to ease the comprehension of the data. This dissertation develops an approach that presents project tasks, resources, and defect information in three-dimensional (3D) visualizations to overcome the above limitations. To assess our approach, we conducted empirical studies using participants from both academia and industry on real-world projects. We developed a prototype tool named 3DProjView for the study. The studies showed that subjects using 3D visualizations achieved higher accuracy and spent less time analyzing project tasks, resources, and software defects.

Across all phases of software development, project stakeholders develop and share documents/artifacts such as project charter, project plan, requirements document, design document, code peer reviews, testing documents, etc. The project team uses these documents to implement/solve project tasks. Currently, project documents/artifacts reside at different repositories and they are not linked together which makes it difficult to implement project tasks. This dissertation develops a traceability system that links the system artifacts together and provides different views of traceability links based on stakeholder role. To help us develop the traceability system, we conducted a survey at an industrial firm evaluating the linking between system artifacts. We developed a tool called TraceLink to prototype and evaluate the traceability system. We conducted an empirical study using participants from the industry and students to evaluate the use of the traceability system to support software maintenance. The results showed that the
traceability system helped achieve high accuracy in finding and linking information together.
DEDICATION

To my parents, family, and people at Balata, refugee camp, in Palestine.
ACKNOWLEDGMENTS

First, I want to thank God, ALLAH, who helped me achieve this success. Next, I want to thank my family for their patience through this long journey.

I am also grateful to my advisor Dr. Chang Liu and committee members Dr. David Juedes, Dr. Jundong Liu, Dr. Frank Drews, Dr. Ronaldo Vigo, and Dr. Kelly Johnson for their support. Finally, I want to thank Dr. Bonita Sharif for her support and all subjects who participated in the experiments conducted in this dissertation.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>3</td>
</tr>
<tr>
<td>Dedication</td>
<td>6</td>
</tr>
<tr>
<td>Acknowledgments</td>
<td>7</td>
</tr>
<tr>
<td>List of Tables</td>
<td>11</td>
</tr>
<tr>
<td>List of Figures</td>
<td>12</td>
</tr>
<tr>
<td>List of Acronyms</td>
<td>14</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>15</td>
</tr>
<tr>
<td>1.1 Statement of the Problem</td>
<td>17</td>
</tr>
<tr>
<td>1.2 Purpose of the Study</td>
<td>17</td>
</tr>
<tr>
<td>1.3 Significance of the Study</td>
<td>18</td>
</tr>
<tr>
<td>2 Literature Review</td>
<td>19</td>
</tr>
<tr>
<td>2.1 Software Visualization</td>
<td>19</td>
</tr>
<tr>
<td>2.2 Project Management Visualization</td>
<td>24</td>
</tr>
<tr>
<td>2.3 Project Defects Visualization</td>
<td>27</td>
</tr>
<tr>
<td>2.4 Project Artifacts Traceability</td>
<td>29</td>
</tr>
<tr>
<td>3 Project Tasks and Resources Visualization</td>
<td>36</td>
</tr>
<tr>
<td>3.1 Project Model</td>
<td>39</td>
</tr>
<tr>
<td>3.2 3D Visualizations</td>
<td>40</td>
</tr>
<tr>
<td>3.2.1 3D Visualization of Project Resources</td>
<td>41</td>
</tr>
<tr>
<td>3.2.2 The 3D Visualization of Project Tasks</td>
<td>44</td>
</tr>
<tr>
<td>3.2.3 The 3D Visualization of Historical Information</td>
<td>47</td>
</tr>
<tr>
<td>3.3 3DProjView Tool Support</td>
<td>50</td>
</tr>
<tr>
<td>4 Software Defects Visualization</td>
<td>52</td>
</tr>
<tr>
<td>4.1 Project Model – Software Defects</td>
<td>54</td>
</tr>
<tr>
<td>4.2 The Analysis Rules</td>
<td>55</td>
</tr>
<tr>
<td>4.3 The 3D Visualizations for Project Defects</td>
<td>57</td>
</tr>
<tr>
<td>4.3.1 The 3D Visualization of Project Defects</td>
<td>57</td>
</tr>
</tbody>
</table>
4.3.2 The 3D Visualization of Defects History Information ........................................ 59
4.4 3DProjView Tool Support ..................................................................................... 60

5 Software Artifacts Traceability ................................................................................. 62
5.1 Real Life Scenarios ............................................................................................ 64
5.2 Industry Survey .................................................................................................. 65
5.3 Traceability Model ............................................................................................. 69
5.4 TraceLink Tool Support ..................................................................................... 72

6 Empirical Study – Project Tasks and Resources ....................................................... 73
6.1 Experimental Design and Hypotheses................................................................. 74
6.2 Participants ......................................................................................................... 75
6.3 Subject Systems ................................................................................................ 76
6.4 Tasks ................................................................................................................... 80
6.5 Running the Study and Data Collection Method ............................................... 82
6.6 Results ................................................................................................................ 83
6.6.1 Accuracy and Time ..................................................................................... 83
6.6.2 Difficulty Level ........................................................................................... 86
6.6.3 Secondary Interaction Effects of Experience and Background .................. 86
6.6.4 Task Analysis .............................................................................................. 90
6.6.5 Post-questionnaire Analysis ........................................................................ 96
6.7 Discussion .......................................................................................................... 97
6.8 Threats to Validity .............................................................................................. 99

7 Empirical Study – Software Defects ....................................................................... 104
7.1 Experiment Design and Hypotheses ................................................................. 104
7.2 Participants ....................................................................................................... 106
7.3 Subject Systems ................................................................................................ 107
7.4 Tasks ................................................................................................................... 107
7.5 Running the Study and Data Collection Method .......................................... 108
7.6 Results ............................................................................................................... 110
7.6.1 Accuracy, Speed and Difficulty ............................................................... 110
7.6.2 Secondary Interaction Effect of Experience and Background ............... 111
LIST OF TABLES

Table 5.1: Description of stakeholders interviewed ........................................................ 69
Table 6.1: Experiment overview ....................................................................................... 75
Table 6.2: Project Visualizations used in the study .......................................................... 80
Table 6.3: Experiment Tasks ............................................................................................ 81
Table 6.4: Resulting p-values of the Mann-Whitney test (one-tailed).............................. 85
Table 7.1: Experiment overview ..................................................................................... 106
Table 7.2: Participants experience and background ....................................................... 107
Table 7.3: Experiment Tasks .......................................................................................... 108
Table 7.4: Visualizations utilized to solve study tasks .................................................. 109
Table 7.5: Results of ANOVA analysis for accuracy, time and difficulty .................... 111
Table 7.6: Results of ANOVA analysis, experience & background interaction on accuracy and time ........................................................ 112
Table 7.7: Results of ANOVA analysis for experience and background interaction on accuracy ........................................................................................................... 114
Table 7.8: Results of ANOVA analysis for experience and background interaction on accuracy .. ............................................................................................................................................................................. 115
Table 8.1: Experiment overview ..................................................................................... 119
Table 8.2: Number of links in the Room Management System ...................................... 121
Table 8.3: Tasks used in the study .................................................................................. 122
Table 8.4: Resulting p-values of the Mann-Whitney test (one-tailed)............................ 128
Table 8.5: Complex model for accuracy and speed dependent variables for all tasks... 129
Table A.1: Stakeholders’ interest in types of information in each software artifact ..... 158
LIST OF FIGURES

Figure 2.1: Mapped software metrics to glyphs a) House - b) Table- c) Spear-Metaphor. (Source Boccuzzo and Gall, 2007; Figure 2) ............................................................................................. 20
Figure 2.2: Paft of the city. (Source Wettel and Lanza, 2007; Figure 6) ............................................................................................. 21
Figure 2.3: Tree-map. (Source Bladh et al., 2004; Figure 1) ............................................................................................. 21
Figure 2.4: 3D visualization - Tree structure. (Source Gall et al., 199a; Figures 3, 6, and 9) ............................................................................................. 23
Figure 3.1: Example of Gant chart above produced using MS Project ............................................................................................. 39
Figure 3.2: Project information model ............................................................................................. 40
Figure 3.3: 3D visualization of project resources ............................................................................................. 43
Figure 3.4: 3D visualization for some tasks of a project ............................................................................................. 46
Figure 3.5: 3D visualization of project tasks ............................................................................................. 46
Figure 3.6: 3D visualization of historical information ............................................................................................. 50
Figure 4.1: Defect life-cycle ............................................................................................. 55
Figure 4.2: Zoom-in view for project defects ............................................................................................. 58
Figure 4.3: 3D view for multiple projects defects ............................................................................................. 60
Figure 5.1: The traceability meta-model ............................................................................................. 71
Figure 5.2: Five different views of traceability links ............................................................................................. 71
Figure 6.1: Boxplots of the distribution for accuracy and time for the two groups ............................................................................................. 84
Figure 6.2: Interaction effects of experience on accuracy and speed ............................................................................................. 88
Figure 6.3: Interaction effects of background on accuracy and speed ............................................................................................. 89
Figure 6.4: Average correctness per task ............................................................................................. 95
Figure 6.5: Average time per task ............................................................................................. 96
Figure 7.1: Average accuracy for study tasks ............................................................................................. 113
Figure 7.2: Average speed for study tasks ............................................................................................. 113
Figure 8.1: Descriptive statistics for the three dependent variables (all data) ............................................................................................. 125
Figure 8.2: Interaction effects between the Type (academia vs. industry) and Group ............................................................................................. 128
Figure A.1: A snapshot of TraceLink ............................................................................................. 157
Figure A.2: Room management application requirements, design, code, build, and testing information ............................................................................................. 160
Figure B.1: Project tasks ............................................................................................. 161
Figure B.2: Project resources ............................................................................................. 161
Figure B.3: Project costs ............................................................................................. 162
Figure B.4: Project history information ............................................................................................. 162
Figure B.5: Project tasks ............................................................................................. 163
Figure B.6: Project resources ............................................................................................. 163
Figure B.7: Project costs ................................................................................................. 164
Figure B.8: Project history information ......................................................................... 164
Figure C.1: Project-A defects ......................................................................................... 165
Figure C.2: Project-B defects ......................................................................................... 165
Figure C.3: Project-C defects ......................................................................................... 166
Figure C.4: Project-D defects ......................................................................................... 166
Figure C.5: Projects A & B defects ................................................................................ 167
Figure C.6: Projects C & D defects ................................................................................ 167
Figure C.7: Project-A defects ......................................................................................... 169
Figure C.8: Project-B defects ......................................................................................... 170
Figure C.9: Project-C defects ......................................................................................... 172
Figure C.10: Project-D defects ....................................................................................... 173
LIST OF ACRONYMS

2D    Two dimensional
3D    Three dimensional
BA    Business analyst
MS    Microsoft
PdM   Product managers
PM    Project manager
PMs   Project managers
To develop a software system, a project is initiated. A project is a temporary endeavor undertaken to create a unique product (PMBOK 2004). A Project manager is a person assigned to achieve the project objectives. A project has a life cycle that consists of five phases: initiation, planning, execution, monitor and control, and closing. At the initiation phase project sponsors decide whether or not to undertake a project. Each project is formally started, named and defined at a high level at the initiation phase. At the planning phase the project management plan is defined (PMBOK 2004). The plan contains cost, scope, time, quality, communication, risk and resources. In the planning phase project resources are selected and work items, tasks, are assigned to resources. At the execution phase the project deliverables are developed and completed according to the project management plan (PMBOK 2004). In the monitor and control phase the project performance and progression are measured (PMBOK 2004). Project managers at this phase monitor performance and control changes to the project management plan. At the closing phase project reaches its end point (PMBOK 2004). This phase involves releasing the final deliverables to the customer, handing over the documentation to operations team, closure of supplier contracts, release of resources etc. (PMBOK, 2004).

To help project managers manage the project phases, they utilize project management software. Microsoft Project is widely utilized project management software (Chatfield, C. and Johnson, T. 2010). Project management software displays project information using visualizations. PMs utilize these visualizations to analyze, manage, and control the project. To control the quality of developed systems, defect tracking tools are
utilized. These tools record software defects and display their information using visualizations. These visualizations are utilized by PMs to analyze software defects.

During the development process of software systems, stakeholders develop different documents/artifacts. Some of these artifacts, for example, are project plans, requirement documents, testing plans, design documents, code, code inspection records, and release records. Project documents/artifacts are linked to each other. When a stakeholder carries out a project task, stakeholder usually utilizes more than one artifact that are linked together.

This dissertation contributes to project management, defect management, and project documents traceability. It developed two approaches, one is used to manage and analyze project tasks and resources; and the other is used to manage and analyze software defects. Furthermore, the dissertation conducted two studies utilizing real systems and participants from industry and students. The first study compared the research approach to Gantt chart and Tables. It should more accuracy and less time analyzing project tasks and resources using the research approach. The second study, however, compared the research approach to 2D Bar charts and tables. It also showed more accuracy and less time utilizing the research approach to manage and analyze software defects. With respect to project documents traceability, the research developed a traceability system that links all project documents together. The traceability system is developed based on an industrial survey that was conducted by the research. The research also conducted a study that showed more accuracy in solving software maintenance tasks utilizing the research traceability system verse manual traceability.
1.1 Statement of the Problem

In software development, project managers manage the project schedule, work items (tasks), and software defects, and project team, resources, develop/implements the project tasks.

To help PMs manage the project schedule, tasks, and software system, they utilize common visualizations provided by existing project management software.

Common 2D visualizations utilized by project managers to analyze and manage projects do not show the entire data in a single view, do not include analysis information in the visualization, and do not support interacting with the data (e.g. rotate the view) to ease the comprehension of data.

To develop project tasks, project team develop documents/artifacts (e.g., requirements, architecture, design, code reviews, test cases, testing plans, etc.). These documents depend on each other. Test plan, for example, is linked to requirements; code is linked to design, etc.

Currently, system documentation/artifacts stored at different repositories and they are not all linked together. Such type of limitation impacts the accuracy of project tasks that involve linked documents.

1.2 Purpose of the Study

The following is the purpose of the research:

1. Investigate whether the use of a traceability linking system increases the accuracy and the efficiency (less time) of solving project tasks.

2. Investigate whether the use of 3D visualizations increases the accuracy and the efficiency of analyzing and managing project tasks and resources.
3. Investigate whether the use of 3D visualizations increases the accuracy and the efficiency of analyzing and managing software defects.

1.3 Significance of the Study

In practice, many projects fail due to bad project management. 54% of information technology failures are due to project management (IBM Systems Magazine, 2012). Thirty-eight percent of projects finished on time and within budget (Project Management Institute Report, 2012).

This research project has the potential to achieve the following:

- Helps PMs analyze and manage project schedule, tasks, resources, and software defects. Such types of activities are important to meet project deadlines,
- Improves the quality of the developed software systems by utilizing a traceability system that links project documents, and
- Improves project team collaborations
2 LITERATURE REVIEW

In this chapter, section 2.1 presents related work to software visualization. Section 2.2 presents related work to project management visualization. Section 2.3 presents related work to project defect visualization, and section 2.4 presents related work to artifacts traceability.

2.1 Software Visualization

Visualizations have been utilized to present different aspects of software systems. They are used to visualize software architecture, design, code, etc. Boccuzzo and Gall, 2007 provide a tool called CocoViz that uses three different metaphors (house metaphor, table metaphor and spear metaphor) to represent entities in software architecture, see Figure 2.1. The house metaphor represents software entities such as classes as houses. The width of the house represents the number of functions in the class while the height of the house represents the number of lines of code. The table metaphor also represents software entities such as classes as tables. The width of the leg presents the number of methods and the height presents number of lines of code. The spear metaphor is represented as a cylinder with two spikes represented as cones on both ends of the shaft. The height of the spear is mapped to the number of functions and the width is mapped to number of lines of code.
Figure 2.1: Mapped software metrics to glyphs a) House - b) Table- c) Spear-Metaphor.
(Source Boccuzzo and Gall, 2007; Figure 2)

Wettel and Lanza, 2007 provide a 3D visualization that uses the city metaphor to depict object-oriented software systems, see Figure 2.2. The classes of an object-oriented system are displayed as buildings and packages as city districts. Brown building represent classes and blue tiles represent packages. The color saturation is proportional to the nesting level of the corresponding packages. The height of the building represent the number of methods in the class and the width and length represent the number of attributes.
Bladh et al., 2004 present a tool called StepTree that display hierarchies such as a directory structure, see Figure 2.3. StepTree uses the 3D space and stakes each subdirectory on top of its parent directory. It displays file system metrics such as file size, modified files, and directories. They use a solid block to represent modified files, a wire frame to represent modified directories, with unchanged files not visible.
Gall et al., 1999a and 1999b and Riva et al., 2000 provided 3D visualizations for displaying the history of software releases. They developed three views, see Figure 2.4. The first view is a 3D view displaying a tree structure of the system using spatial layout. The second view is also a 3D view presenting historical information with respect to programs changed within a module across system releases. It uses bars along the coordinate z. Each bar represents a release. The bar contains blocks that have a size and color. The block represents a module and the size of the block is proportional to the percentage of programs that have changed. They use colors to show whether a release has changed and a scale that has different colors with a different value for each color. The color of the block depends on the value on the color scale. The third view is a 2D view, and it uses rows and columns to show the release at which a program changed. Each row represents a release and each column represents a program. Gall et al., 1999a and 1999b and Riva et al., 2000 developed a tool called 3DSoftView to display the three visualizations.
Trumper and Dollner, 2012 presented three perspectives for software engineering processes and existing visualizations that support them, based on visits to industrial companies. The perspectives are management, quality, and architecture perspectives. They used the term software maps “for visual techniques that combine thematic information about software development processes (evolution), software quality, structure, and dynamics and display that information in a cartographic manner”. They proposed that software maps provide a complementary picture of underlying software engineering data and process, and they help making decisions.

The above research employs 3D visualization to support different areas in software engineering. Boccuzzo and Gall’s, 2007 visualization supports software architecture. Wettel and Lanza, 2007 and Gall et al., 1999 and Riva, 2000 visualizations...
support code. Bladh et al.’s, 2004 and Trumper and Dollner, 2012 work supports software development. Our research supports project management. We utilize 3D visualizations to display project tasks and resources information using a hierarchal structure of boxes. We use the three dimensions of box, colors, and textures to display the information.

2.2 Project Management Visualization

Visualizations have been used in several areas of project management. We present below visualizations to support different areas in the project management of software. Zhang and Zhu, 1997 developed a project schedule system called SWAV. In their work, they provide an example that uses rectangles that contain bars to show the utilization of three resources and cash flow of a project life cycle. The rectangle represents the duration unit while the bars inside the rectangle represent resource and cash flow information.

Burkhard et al., 2005 conducted a study that compared the effectiveness of using project Tube Map and Gantt chart for inter-functional communication in large projects. The Tube Map visualization presents a tube that contains lines and stations. Each line represents a target group and each station represents a milestone. The results indicate that project Tube Map is more effective than the Gantt chart for communication in long-term projects and helped in catching attention and illustrating the “big picture”. In contrast, the Gantt chart was more effective in a clear-structured approach and for displaying task duration.

Luz and Masoodian (2010) present an empirical study that evaluated the effectiveness of using Gantt charts and space-filling (mosaic) visualizations to help users maintain focus and context awareness. A mosaic visualization is similar to a Gantt chart
because the x-axis shows a calendar and the y-axis shows activities. However, a mosaic allocates space proportional to number of activities scheduled for a time interval. The study showed that the collapse mode (hiding sub-tasks and showing only summary tasks), in mosaic benefitted users more than the collapse mode in Gantt chart.

Biffl et al., 2005 conducted a study that compared the effectiveness of PERT charts and PlanningLines techniques, Aigner 2005, in presenting task relationships and temporal uncertainties. The PERT chart is used to represent project tasks. It shows tasks as rectangle nodes and dependencies as arrows connecting the nodes. The content of the rectangle shows key information such as name, start date, duration, and percent of completion to name a few. In the PlanningLines visualization a task is modeled as a set of related bars along a calendar scale. Each task has two encapsulated bars representing minimum and maximum duration that are bounded by two caps that represent start and end intervals. The study confirmed that PlanningLines are as simple and intuitive to use as the PERT chart. PERT charts are more appropriate than PlanningLines for answering detailed questions on single attributes of a project plan. PlanningLines are better suited than PERT to deal with temporal uncertainties regarding the duration, start, and end of activities/plans.

Aguirregoitia et al., 2010 used a 3D T-Cube and Metro Map visualizations to present development process information. T-Cube uses the six sides of a cube and color to present the information while Metro Map uses stations, traffic lights, color, and position. Both visualizations are two-dimensional. T-Cube uses cube and color to present information.
To manage stakeholders, Bourne and Walker, 2005 provide a stakeholder circle that maps and presents the power and the influence of stakeholders. The circle contains sections that represent stakeholders and the size of the section indicates the scale and the influence of stakeholders with color indicating the degree of impact.

To support portfolio management, Cable et al., 2004 adapted a well-known visualization technique called Treemaps, Bederson 2002, to display performance metrics for the entire portfolio of projects. The visualization uses the rectangle cube to represent projects. The size of the rectangle is used to be proportional to the budget and the color of the rectangle represents the performance index (e.g., cost performance index).

Chuah and Eick, 1997 provide two visualizations named TimeWheel and 3D-Wheel to present time information in a project. In TimeWheel, each object (e.g., a person, a project, and a release) is represented by a circle and each object attribute is represented as a time series laid out around a circle. The 3D-Wheel encodes the same data attributes as the TimeWheel but also uses the height dimension to encode time.

The above research utilized visualization to support different areas in project management. Zhang and Zhu’s work, 1997, Luz and Masoodian, 2010, and Biffl et al., 2005 utilized 2D visualizations to display schedule information. Aguirregoitia et al., 2010 visualization supports the software development process. Bourne and Walker’s, 2005 visualization presents stakeholders influence. Chuah and Eick, 1997 visualization supports time management. Our research differs from the above research in that we utilize a hierarchal structure of boxes to present project tasks and resources information. Furthermore, we define and apply analysis rules to display project tasks and resources status.
2.3 Project Defects Visualization

In the literature there are two important trends in the research of defect tracking; one trend addresses the visualization of bug information while the other addresses what makes a good bug report. Dal Sasso and Lanza, 2013 presented a 2D visualization that displays bug reports through a web-based tool called inBug. The visualization presents the life time of a bug using horizontal stacked bar charts to denote bug duration and different colors to denote status. It also presents the events occurred during the life time of the bug including description and time. D'Ambros et al., 2007 presented two 2D visualizations to help understanding bugs, matrix view and bug watch. The matrix view shows the bugs at each component of the system while the bug watch view shows the phases that a bug traversed. Knab et al., 2009 provided three visualizations that present bug information using box view, pie chart, and state transition view. The box view uses the width to represent the effort and the height to represent the actual effort. The pie chart shows the duration of each state of the bug. The state transition view uses nodes and arrows in which the width of the arrow denotes the number of bugs that exhibit the corresponding transition. Hora et al., 2012 provided two visualizations that present bug details, history view and snapshot view. The history view presents a chart that displays the number of bugs in class/package. The snapshot view presents measures and code. Couto et al., 2014 presented a tool that identifies the modules with more bugs, the average lifetime and complexity of the bugs. The tool provides a view that displays rectangles mapped to the classes of the system and the color denotes number of bugs at the class. It also provides a grid view that shows the number of defects at each class. Yeasmin et al., 2014 proposed a prototype that presents visualization that has bug reports
using topic analysis. User selects a collection of bugs and the system applies topic analysis and displays an area-graph. The x-axis under the graph represents time and the heights of the area at each time-segment represent the number of bugs at that topic.

D'Ambros et al., 2007 presented a tool that integrates information from the source code with information retrieved from CVS files and Bugzilla bug reports. They showed an example for a visualization produced by their tool that shows author information for a directory hierarchy of Gimp software. Each author is represented by a colored rectangle; the rectangle is proportional to the work done by the corresponding author. Anvik et al., 2005 provided pie visualization of two open bug repositories from the Eclipse and Firefox projects. Singh and Chaturvedi, 2011 compared tracking tools BugZilla, Jira, Trace, Mantis, BugTracker.Net, Gnats, and Fossil. They pointed out that bug tracking tool shall provide status in a graphical way in which color is used to show status and the graphical timeline shows current and remaining time to fix the bugs. Our research differs from the above research in which we utilize 3D visualizations to present defect information while they use 2D visualizations. Furthermore we display analysis information in addition to the displayed data; they only display data. Our approach utilize cylinders in 3D space to display defect information and apply analysis rules to the defect information and present the results of analysis utilizing colors, see chapter 5.

Other research in the literature studied the type of information that should be included in a bug report. Bettenburg et al., 2003 conducted a survey using ECLIPLSE developers to determine the information that are widely used by developers. The survey showed that the steps to reproduce and stack traces are important to developers, and inaccurate steps to reproduce and incomplete information are major issues to developers.
Bettenburg et al., 2008 conducted a survey using input from developers and users of APACHE, ECLIPSE, and MOZILLA to find out what makes a good report. They reported that most developers considered steps to reproduce, stack traces, and test cases are helpful to include in a bug report. They also developed a prototype tool, CUEZILLA that measures the quality of bug reports. Just et al., 2007 conducted a survey on common problems with bug reporting among developers and users of APACHE, ECLIPSE, and MOZILLA projects. They provide recommendations for bug reporting. Some of these recommendations are translating bug reports in foreign languages, providing a user interface that gives cues on what information is needed, and providing easy-to-use searching features. The research by (Bettenburg et al., 2003; Bettenburg et al., 2008; and Just et al., 2007) differs from our approach in which they focus on identifying what should be included in a bug report. Our approach focuses on display bug information; however, it is possible for our approach to display defect information identified by (Bettenburg et al., 2003; Bettenburg et al., 2008; and Just et al., 2007). For example, we utilize cylinders to present information. We use the cylinder height to present the number of defects found; it is possible to use the height to present the remaining time to fix a defect.

2.4 Project Artifacts Traceability

In this section, first we present existing work in software collaborations. Hildenbrand et al., 2008 evaluates collaborative approaches in requirement engineering, design and modeling processes, implementation, testing and maintenance. Whitehead, 2007 presents a list of goals for software engineering collaboration and surveys existing collaboration support tools in software engineering. He divides the collaboration tools
into four categories: model-based, process support, awareness, and infrastructure. Trude et al., 2009 presents empirical studies on collaborative software development. The studies are on the areas of collaboration during requirement engineering, collaboration during design, collaboration on distributed software development, existing collaboration tools, new collaboration tools, collaboration in open source, and collaboration in project management. Different tools are identified in (Trude et al., 2009; Hildenbrand et al., 2008; Whitehead, 2007) for collaborative software development. Some of these tools are configuration management tools (Pilatti and Prikladnicki, 2006), instant messaging tools (Handel and Herbsleb, 2002; Isaacs et al., 2002), source code annotation tools (Cadiz and Gupta, 2000; Storey et al., 2008), processes tools ((Pandey et al., 2003), awareness tools, (Sarma and Redmils, 2008; Biehl et al., 2007), and project management tools (Zhang et al., 2007; Tessem and Iden, 2008). None of these studies focus on artifact-based collaborations via traceability: a view we take in this dissertation.

Now we present representative related work in traceability, models, tools, link recovery, link evolution, link visualization, empirical studies, and case studies in industry. We also describe how this work complements, differs and adds to the current understanding of the community. In our prior work, traceability link definitions and a link model was developed (Maletic et al., 2005). A fine grained differencing mechanism to evolve traceability links was proposed in (Sharif and Maletic, 2007). Spanoudakis et al., 2004 and Lamb et al., 2011 describe types of traceability links between requirements, use cases and analysis object models. Their links are based on existing software systems and also take into account software product lines. The link types presented in (Spanoudakis et
al., 2004 and Lamb et al., 2011) can be used in the model presented here targeted towards stakeholder roles.

Mäder et al., 2012 conducted a study similar to the one presented in this dissertation. It assesses the effect of requirements traceability for maintenance tasks but uses only academic subjects, whereas our study uses subjects from academia and industry. Besides differences in the subject pool, they only trace requirements to code. Our study assesses traceability between requirements, design, code, builds, defects and test cases and supports many stakeholder roles. Similarities are that both studies use a prototype tool to test the effect of traceability on maintenance tasks, albeit with different subject pools. Both studies find that traceability links help accuracy and efficiency of maintenance tasks. In addition, our study finds that industry subjects who used traceability links (via the TraceLink tool) perform a lot better compared to their colleagues who did not use any traceability links. Recall that in our study, traceability links were only given to one group and they used the links via the prototype TraceLink tool we built.

Traceability link recovery is an important part of any traceability system since it supports existing systems that don’t have links defined a priori. DeLucia et al., 2008 discuss managing traceability for impact analysis and present the major challenges and research directions. There are many dissertations such as (Marcus and Maletic, 2003; Kagdi et al., 2007; Lucia et al., 2007; McMillan et al., 2009; Zou et al., 2010; Sundaram et al., 2010) addressing the topic of link recovery. Although this dissertation does not address link recovery, these are important to consider when developing a traceability system. Traceability tools such as ADAMS (DeLucia et al., 2006), Poirot (Lin et al., 2006), ACTS (Asuncion and Asuncion, 2010), traceMaintainer (Maderand and Gotel,
2009) and Traceclipse (Klock et al., 2011) have been developed for link recovery and management in academia. DeLucia et al., 2009 conduct a controlled experiment to assess the usefulness of an information retrieval based traceability recovery tool. Results show that the tool reduces the time spent vs. manual tracing. These studies compare different traceability techniques instead of assessing the effect of traceability itself on maintenance tasks. None of the studies look at the effect of traceability links on software maintenance tasks: the main goal of the study we present in this dissertation. A secondary goal of this dissertation was to gather data on what features are needed in a traceability system/tool that can be adopted successfully in industry.

There has also been some work on the visualization of traceability links. Marcus et al., 2005 was the first dissertation attempting to visualize traceability links with TraceViz, and presents a list of visualization requirements. More recently Chen’s work (Chen, 2010) focuses on visualizing traceability links in a project using a graph toolkit in Eclipse. VisMatrix (Duan and Cleland-Huang, 2006) generates a graphical representation of the requirements traceability matrix. None of the above work conduct empirical studies to determine the feasibility of these visual representations in practice. We consider this dissertation to be the first attempt to conduct an empirical study to understand what industry developers actually need in a traceability tool. It takes a similar but small scale empirical approach as Ramesh et al., 2001, by conducting a survey to determine what types of links practitioners are interested in. The work presented in this dissertation addresses the types of links sought by stakeholders (via an industrial survey) for collaboration. This is a first attempt at determining the types of links that are most useful to each different stakeholder. In addition, a study conducted on software maintenance
tasks was used to determine the benefits of a traceability links for artifact-based collaboration in practice.

Gotel and Finkelstein, 1994 were one of the first researchers to do an empirical study among software developers in a large industrial firm. Their main goal was to understand the scale of traceability problem. They did not consider the effect of traceability links on maintenance tasks. Ramesh et al., 2001 also conducted a study on industry practitioners by gathering data via questionnaires to better understand traceability links used. They define reference models based on their survey. Similar studies (Ahmad and Ghazali, 2007) like the ones by Gotel and Ramesh have been done on a smaller scale but none of them look into the effects of traceability links on software maintenance tasks versus not having the links. Our study seeks to provide via empirical evidence that when developers are given traceability links to work with, they are better at software maintenance tasks.

Asuncion et al., 2007 present an end-to-end traceability tool developed at a software firm that supported the entire life-cycle and focused on requirements traceability and the traceability process. Several guidelines were presented after deploying the traceability tool within the company. The above work complements the work in this dissertation. Neumuller et al., 2006 also developed and applied a traceability system in a small company and reported their findings. The findings from our study corroborate with (Neumuller et al., 2006); in both cases a need for better tools is called for. The study presented in this dissertation is a controlled experiment and differs from (Neumuller et al., 2006) because in this study, focused software maintenance tasks were used to determine the usefulness of the links via a prototype tool, TraceLink. Heindl et al., 2006 discussed
the application of requirements traceability to risk assessment and developed a cost-benefit model to help PMs. Klimpke et al., 2009 conducted a case study at five industrial firms to determine how traceability was realized in practice. They also pointed out what needed to be considered in order to adopt traceability in industry. They found that companies used traceability in an ad hoc manner since there was a lack of existing tools that were customizable for their needs. Even though the study presented in this dissertation was done at only one firm, the findings were similar to those of Klimpke et al.

There have been several recent dissertations related to improving the performance of requirements to code traceability. We are describing some of them next. Yu et al., 2012 present an invariant traceability framework to merge changes that occur between user-modified code and template-generated code in model-driven development. They focus on maintaining the bidirectional traceability links as software changes; this dissertation is concerned with showing the benefit of having the links while developers perform maintenance tasks. The goal in this dissertation was not to create a traceability tool but to provide evidence that traceability links actually do help in software maintenance tasks. Charrada et al., 2012 present an approach to automatically detect outdated requirements when source code changes. The links are updated when the system evolves over time. This is important because a traceability system is only as good as the validity of its links to requirements. Mahmoud et al., 2013 use refactorings to improve the performance of requirements-to-code traceability. They showed that as a system evolves, when corrupted textual and lexical structure was restored with refactorings, a positive impact on traceability was reported. Niu et al., 2013 and Dekhtyar et al., 2011 study the human analyst’s behavior in automated tracing. They note that the quality of the requirements
traceability matrix is important. They examine both the traceability matrix quality and the rational decision making process of human analysts while they determine validity of links.

The papers mentioned in the above paragraph are aimed at supporting the evolution of traceability links and/or requirements as software systems evolve or understanding how analysts validate traceability links. They do not show evidence that these links are actually useful to developers in software maintenance tasks. All previous evidence was purely anecdotal. Since our goal was to determine the effect of link presence, we considered the actual tool used to be of secondary importance in this dissertation. For this reason, we created a simple prototype that provided us with enough linking to run the study. Any tool that provides traceability link information would suffice since the tool itself is not being evaluated, but the presence of traceability links is.
3 PROJECT TASKS AND RESOURCES VISUALIZATION

In the planning phase of a project, PMs develop the project schedule. The schedule contains work items, tasks and their sequence, relationships between tasks, and resources assigned to carry out the tasks. A project task may have one or more subtasks. Examples of tasks could be developing the requirements document for a project or setting up a database. A task of setting up a database can have sub-tasks such as defining tables, schemas and views. A task that has sub-tasks is referred to in Microsoft Project (MS Project) as a summary task (Chatfield, C. and Johnson, T, 2010)). Resources are assigned to tasks or sub-tasks. It is possible for a sub-task to be a summary task, but deep nesting is not recommended because it introduces complexity. When assigning resources to tasks, PMs determine whether the resource will work full time or part time; MS Project uses the term “units” to indicate full or part time work. For example, if a PM wants to indicate that all person time is devoted to the project, the “unit” value for that person is 100% which is also the maximum one can assign. If the person works 8-hour days, he or she will be over allocated if any combination of assignments exceeds 8 hours (100%) of the available work day. If the person works 8-hour days and has 75% unit value of his or her time devoted to the project, then he or she will be over allocated if any combination of assignments equals more than 6 hours (75%) of the available work day.

Currently, many PMs utilize Gantt charts to display and analyze the project schedule. The Gantt chart is a two-dimensional chart; it displays project tasks as bars in a timeline and also shows the sequencing and dependencies between them as links. Progress is sometimes shown by shading the bar proportionally to the percent of duration or work completed (see Figure 3.1). PMs use Gantt charts, for example, to see task
duration, the expected time to finish the task, the percentage of completion, and remaining duration. Spreadsheets or tables are also used by PMs to display project information. They are, however, developed in an ad-hoc way and there is no agreement on the format nor data presented by them. MS Project, for example, uses them to display tasks and resources information. It has a resource sheet and resource usage sheet that displays information on resources. The resource sheet contains information about the resource such as the name, the max unit of work that the resource can spend on the project, the cost per hour, etc. The resource usage sheet, however, shows the amount of work the resource spent on each task assigned to him or her at any point in time since the start of the project. PMs use spreadsheets/tables to track and analyze the status of resources. MS Project also contains a complete list of information that a project can have for a resource or a task (Chatfield, C. and Johnson, T, 2010).

Previous work identified and tried to overcome some limitations with the Gantt chart. Leach (2010) pointed out that in a Gantt chart, it is very difficult to see the entire schedule, key performance information, past history, trends and future trends for prediction. To overcome some of these limitations, it was suggested that the area underneath the timeline be used to show key information, and zooming and animation be used to show the change in schedule over time. Leach proposes using the Gantt chart to show key performance information with respect to the schedule and budget. We believe budget information should be presented in spreadsheets or other charts (e.g., pie charts). Liston et al., 2000 suggested overlaying information on the Gantt chart to show differences and variance in schedule. Tory et al., 2013 used “shadows”, sidebar views, and TbarView to show the variances and overlapping of two schedules. The TbarView
has a file selection panel and a widget panel that displays the compared schedules. It also has a Tbar, comparison bar, that has a box that users drag to select the range of schedule they need to compare. They also use different line styles, line color, and icons to display new constraints between project tasks.

One of the objectives of this dissertation is to develop an approach that generates 3D visualizations to help PMs manage and analyze project tasks, resources and history information. We first outline the model used in our approach followed by three types of visualizations used for the analysis of project tasks, resources, and historical information. The visualizations are called 3D visualization of project resources, 3D visualization of project tasks, and 3D visualization of history information. Examples of these visualizations are shown in figures presented below. They were generated using a prototype tool 3DProjView (presented in Section 4.3). The section below outlines the method used to develop the three visualizations.

1. Define a model that captures the relationships between project tasks, resources and their historical information,

2. Use a landscape view in 3D space that contains a hierarchy layout of boxes,

3. Use the x, y, and z dimensions of boxes to display metric information for project tasks and resources. Use color and texture to display status information, and

4. Use interactive methods such as zooming in on items of interest, detail-on-demand, and rotating the view to interact with the visualization.

In this chapter, section 3.1, below, describes the model used by the above approach. Section 3.2 describes the visualizations. Section 3.3 presented the 3DProjView tool.
Figure 3.1 presents a model that captures the relationships between project tasks, resources and history information. The model is developed using UML notation (Arlow and Neustadt, 2005). In the model, a project contains tasks and resources. Task may have sub-tasks. Task or sub-task is assigned to one or more resources, and a resource works on or more tasks or sub-tasks. The type of relationship between (task and sub-task), (project and task), and (project and resource) is called aggregation (contains or has), and between (task and resource) is called association. Project tasks and resources each have multiple instances that contain history information. They contain large amounts of information such as a name, planned start date, planned duration, and planned finish date, actual start date, actual duration, actual finish date, remaining duration, percent of completion,
planned cost, etc. We refer the reader to (Chatfield and Johnson, 2010) for a complete list of attributes a project can have. The information that our approach is interested in for project tasks and resources are text and quantitative values (i.e., ordinal and categorical attributes are not used in our approach at this time and is reserved as part of our future work).

Figure 3.2: Project information model

3.2 3D Visualizations

In this section, three 3D visualizations are developed that are based on the project information model described above in section 3.1. The visualizations are 3D visualization of project resources, 3D visualization of project tasks, and 3D visualization of historical information.
3.2.1 3D Visualization of Project Resources

This visualization presents resource information and the relationship between resource and its assigned tasks. A task can be a summary task that has sub-tasks. A resource can be assigned to one or more tasks or sub-tasks of a summary task (see Figure 3.2). For example, suppose that a Java application has task T1 and a summary task that has two sub-tasks T2 and T3. Task T1 is for designing the user interface, task T2 and task T3 are set for development of client-side code and server-side code respectively. Resource “A” assigned to T1 and T2, and resource “B” assigned to T3.

To simplify the illustration, we use Figure 3.3 that shows an example of the 3D visualization of project resources. The visualization uses the hierarchical layout of boxes in 3D space. The first level represents a resource and the second level represents tasks assigned to the resource. It is possible to use the second level to denote a summary task and the third level to denote sub-tasks. We use two levels because PMs are mostly interested in the tasks assigned to resource regardless of their type (summary task or sub-task). The level one box dimensions (x, y, and z) and colors do not convey information; future work may use them to denote other project level information. The texture, of a first level box, however, denotes that the resource is over allocated. Figure 3.3 shows a project that has seven resources and their assigned tasks. Six of these resources are represented by six light blue long boxes, one for each. The seventh resource is represented by a long box that has grey texture; the texture denotes that the resource is over allocated. The name of the resource is written on the sides of the box. Each resource box contains colored boxes, level 2, representing the tasks assigned to the resource. The box’s x-dimension denotes the duration planned to complete the task, the y-dimension denotes the
percentage of completion, and z-dimension denotes the remaining duration, to complete the task.

Colors denote status information; they are defined based on rules that we developed and further explained below. For example, suppose a PM reviews project resources that are assigned to tasks on Monday and Friday.

- If the percent of completion of a task has not changed from Monday to Friday, no progress has been made and we color the task yellow.
- If a task starts after the planned start date, this indicates that the task is slipping its planned start date. Furthermore, if a task’s finish date is greater than the planned finish date, this indicates that the task is slipping its finish date. We use the color red to indicate that the task is slipping its planned start and/or finish dates.
- If the percent of completion is increasing from Monday to Friday and the planned finish date has not changed, this indicates that the task is progressing as expected and we color the task green.
- If a task is completed on the planned finish date, this indicates that the task is completed on time and we color the task pink.
- If a task is completed after its planned finish date, then the task is completed late and we color the task olive.
- We use the color orange to indicate that the task has not started yet.

In this visualization, if you hover over a task, a text box is displayed in the right upper corner. In Figure 3.3, the text displayed in the right upper corner is as a result of hovering over one of Kim’s tasks. In the case of a task that is colored red, the PM can
hover over the task to see whether the task slipped due to its start and/or finish dates.

Finally, the colors, texture, and the x, y, and z dimensions in the 3D visualization of project resources are configurable. They can be changed automatically and saved in an XML configuration file (see section 3.3).

Figure 3.3: 3D visualization of project resources

In Figure 3.3, (1) Light blue boxes are resources (space is needed to clearly distinguish boundaries); Gray box is an over allocated resource. (2) Y-dimension denotes percent complete. (3) X-dimension denotes planned duration. (4) Z-dimension denotes
remaining duration. (5) Colors denote task status. Pink color denotes that a task was completed on time, Olive denotes task completed late, yellow denotes no progress, red denotes slipping planned start, finish or both, green denotes that task is performing as planned, and orange denotes a task not started.

3.2.2 The 3D Visualization of Project Tasks

This visualization presents tasks information and the relationship between project and its tasks. A project has one or more tasks. A task can be a summary task that has sub-tasks. To simplify the illustration, we direct the reader to Figure 3.4. The figure presents an example showing four views (front, back, top, and diagonal) for the same set of tasks in a project; these views are produced using the zoom in-out and rotation features of the 3DProjView prototype tool. The visualization uses hierarchical layout of boxes in 3D space. The first level is a flat box that represents the project. The second level box could be sub-task or a summary task. The third level box could also be a sub-task or a summary task, and so forth. In our approach, it is possible to have several nested levels. However, in practice, nested levels are not recommended and PMs tend to avoid them because they introduce additional levels of complexity. The color and dimensions for level 1 and summary level boxes do not convey information; future work may use them to denote additional project attributes. In Figure 3.4, the flat green, level 1, box represents a project standing on it tasks T1, T2, T3, T4, T5 and T6. T6, however, is a summary task represented by a white box that contains sub-tasks T6.1, T6.2, T6.3, T6.4, and T6.5. The task or sub-task box dimensions, colors, and status rules are the same as in the 3D visualization of project resources above. Figure 3.4 presents schedule metrics and work progress.
The 3D visualization of project tasks is configurable and the metrics, colors and status rules can represent other information for project tasks. For example, to present cost status for project tasks instead of schedule status, the y-dimension is mapped to the percent of completion, the x-dimension is mapped to the planned cost, and the z-dimension is mapped to the remaining cost. With respect to cost status, we use these rules:

- If the task is completed and its cost was less than or equal to the planned cost, we consider the task completed within budget and color the task pink.
- If the task completed and its cost was greater than planned cost, we consider the task cost is over budget and color the task olive.
- If the task is in progress and its cost is less than the planned cost, we consider the task cost is as planned and color the task green.
- If the task is in progress and its cost is greater than planned cost, we consider the task is in progress and it is over budget; we color the task red.
- We use the color orange to indicate that the task has not started yet.

Figure 3.5 presents a project that has a large number of tasks (55). The intent of Figure 3.5 is to show large number of tasks in a single view and to also show colors and dimensions of boxes are configurable (see Section 4.1). The layout of the visualization is similar to the visualization in Figure 3.4.
Figure 3.4: 3D visualization for some tasks of a project

Figure 3.5: 3D visualization of project tasks
3.2.3 The 3D Visualization of Historical Information

This visualization presents historical information and variances for project tasks stored at baselines. A baseline is taking a snapshot of the project data at a certain point in time. Project managers save different baselines for the project data to compare progress. Figure 3.6 is an example of the 3D visualization of historical information; it shows four views (front, back, top and diagonal) for two baselines of a project using hierarchical layout of boxes in 3D space. Two levels are used in the 3D visualization of historical information. Level one contains one box representing the baseline. Level two contains boxes that represent tasks or sub-tasks that belong to the baseline. Level one boxes are arranged in rows and ordered based on time; for example, row 1 from the bottom, contains the current week’s baseline, row 2 contains last week’s baseline, and so forth. A box (task), at level 2 is located at the same location across all baselines (i.e., task 1, for example, is located at the same location at baseline 1, baseline 2, baseline 3, etc.). The x, y and z dimensions of the level 2 box presents task information with the color being either green or red. Red denotes that there is a variance in the task information presented at the x, y, or z dimensions. The color green denotes there is no variance. Our approach defines and uses a set of variance rules to determine the color. For example, in Figure 3.6 the x-dimension of the level 2 box represents the finish date for the task, the y-dimension represents the percentage of completion, and the z-dimension represents the start date. The following are the variance rules for tasks compared in two baselines as used in this example:
• Difference in finish duration ($\Delta fd (1, 2) = \text{finish date at baseline 2} - \text{finish date at baseline 1}$). If $\Delta fd (1, 2)$ is greater than zero, this means the task is late and the color is red.

• Difference in percent of completion ($\Delta pc (1, 2) = \text{percent of completion at baseline 2} - \text{percent of completion at baseline 1}$). If $\Delta pc (1, 2)$ is equal to zero, this means there is no progress and the color is red.

• Difference in start date ($\Delta sd (1, 2) = \text{start date at baseline 2} - \text{start date at baseline 1}$). If $\Delta sd (1, 2)$ is greater than zero, this means that the task slipped its start date and the color is red.

When the color of a task is red, a PM can hover over the task and in the upper right corner a detailed box shows the variances across the x, y, and z dimensions ($\Delta fd$, $\Delta pc$, and $\Delta sd$). In the figure tasks 15, 21 and 27 have a variance at the second baseline.

The 3D visualization of historical information is configurable and the metrics, colors, and status rules can represent other information for project tasks. For example, the y-dimension of task box can be % of completion, the x-dimension can be planned cost, and the z-dimension can be remaining cost. In this case we can use the following variance rule:

The difference in planned cost for task ($\Delta pc (1, 2) = \text{cost at baseline 2} - \text{cost at baseline 1}$). If $\Delta pc (1, 2)$ is greater than zero, this means that the task cost is over budget and the color is red.
d) Diagonal view

Figure 3.6: 3D visualization of historical information

In Figure 3.6, two baselines and variances for the tasks of a project are illustrated:

(1) Row 1 is the first baseline. (2) Row 2 is the second baseline.

3.3 3DProjView Tool Support

To prototype the approach defined by the research and to conduct the study in chapter 6, a prototype tool called 3DProjView is developed. The tool provides 3D visualizations that present project tasks, resources, and past history information. It is developed using the C# programming language. The tool reads an XML file to generate the visualization. The XML file contains layout information and metrics data. The layout information consists of the hierarchical structure, color, and texture. The metric information are the project tasks, resources or past history data that need to be mapped to x, y, and z box dimensions. For example, to create the visualization of project tasks in Figures 3.4 and 3.5, we exported from MS Project the metric data for percent of completion, planned duration and remaining duration for each task. We also exported from MS project, the layout information and summary tasks to sub-tasks relationships.
Next, we generated the XML file from the MS project exported data by using either of these two methods that we developed, a Python script or a C# module that has a GUI user interface. After generating the XML file, 3DProjView reads the XML file and creates the 3D visualizations. It uses a MS DirectX component to display the 3D visualization.

Similarly, to generate the 3D visualization for project resources in Figure 3.6, we used the same method utilized to generate Figures 3.4 and 3.5. We exported from MS project the resources, their tasks metrics, and the hierarchical structure and then used a Python script to generate the XML file from the MS Project export. All figures in Section 3.2 are generated using the same process described above.

To ease the understanding of the presented data, the tool provides the following interaction features: 1) Zooming in/out, 2) User can hover over a box in the view and get more detail information in the top right corner, 3) User can drag the figure and move it anywhere on the screen, 4) User can click the right mouse button to rotate the visualization around the x-axis 360 degrees; to stop rotation, the user clicks the right mouse button. While the visualization is rotating, the user can drag and move the figure to different area on the screen, and 5) Pressing the keys 0, 9, 8, 7, 6, and 5 shows the front, back, right side, left side, top and diagonal view respectively. For each of these views you can drag, zoom in/out, and hover over a box to see more details. It is important to note that 3DProjView is a prototype tool developed for the study; more capabilities can be added to the tool and left as future work.
SOFTWARE DEFECTS VISUALIZATION

During the development of software systems, different types of testing are performed to discover defects. For example, during the development of code, developers test their code modules/components trying to find defects. Testers utilize different types of testing such as system testing, performance testing, etc. to discover defects. System testing is performed to make sure that the system supports the requirements defined by the customer. Performance testing is performed to verify the expected performance of the system; for example, a system shall process user transaction within five seconds.

At the planning phase of a project, PM defines the project schedule. The PM’s goal is to meet the project deadlines. So during the development of the software, PMs want to make sure that discovered defects are resolved on time. Managing defects is an important activity performed by PMs as it impacts the schedule and the quality of software.

At development and testing, when a developer or a tester finds a defect, he or she uses a defect tracking tool to record and capture information about the defect. The following are some of the information captured for a defect by a defect tracking system called Bugzilla https://www.bugzilla.org/about/: Alias - a short name assigned to bug, Defect ID – a unique numeric id, Reporter – the person who filed the defect, Build – the build number where the defect was found, Creation date – when the defect was filed, Assignee – the person in charge of resolving the defect, Deadline – the date that the bug must be resolved, Summary – short description of the defect, Severity – how severe the bug is, Product – the product where the defect is found, QA contact – the name of the person who will verify the fix for this defect, and Status – indicates the current status of
the defect. After the defect information is recorded in a tracking tool, an email is usually sent to the PM. The PM meets with the development manager and project leads to review new defects.

Many defects tracking tools currently exist; some of these tools are open source Bugzilla https://www.bugzilla.org/about/ and Mantis https://www.mantisbt.org/. Other defect tracking tools are commercial, Rally https://help.rallydev.com/ and JIRA https://www.atlassian.com/software/jira. For example, Bugzilla is a widely used open source bug tracking tool. It allows individual or groups of developers keep track of outstanding problems/bugs with their product. It provides charts that present defect information; some of these charts are bar and pie charts. Mantis is another open source defect tracking tool. It allows tracking and managing of defects through the development life cycle. It also presents defect information using pie and bar charts. JIRA is a proprietary issue tracking tool, developed by Atlassian. It provides bug tracking, issue tracking, and project management functions. It allows generation of different charts that present various information including defect information. For example, JIRA can produce a chart for opened verses resolved defects, average number of days it took to resolve defect, etc. Some of the charts produced by JIRA are bar and pie charts. Rally is another example of bug tracking tool that is developed by Rally. It is a project and defect management tool. It allows developers to track defects found in their products. It generates charts that present bug information. Some of these charts are bar and pie charts.

This research develops a 3D visualization approach that displays defects status and analysis information in a software system. It also conducts an empirical study that compares the 3D approach to the 2D bar chart and table format. Different 2D charts are
supported by defect tracking systems; bar chart is one of the commonly used charts. The study results are applicable to 2D bar charts and tables.

One of the objectives in this research is to develop a visualization that display defect information as well as analysis information. To do that we utilize the following method:

1. Use a model that captures the defects information,
2. Apply analysis rules to the information captured by the model,
3. Display the information in steps (1 and 2) in a 3D landscape visualization, and
4. Utilize interaction features to interact with the 3D visualization.

The following sections describe the above steps in details. Section 4.1 presents the model, section 4.2 presents analysis rules, section 4.3 presents the 3D visualizations, and section 4.4 presents the support of the tool 3DProjView.

4.1 Project Model – Software Defects

Defects have a lifecycle; Figure 4.1 shows an example of the lifecycle. It has the states Submitted, Under-review, Assigned, Rejected, Resolved, and Closed. When a tester or a developer finds a defect, he or she uses a defect tracking tool to record the defect and sets its state to Submitted. The PM assigns a developer to review the defect and changes the state of the defect to Under-review. After the developer evaluates the defect, its state is changed to Rejected or Assigned. The developer assigned to fix the defect makes the code change and changes the state of the defect to Resolved. Tester tries the fix and if it passes, the defect state is changed to Closed; otherwise, tester re-assigns the defect and changes its state to Assigned. In the next section, we develop analysis rules that utilize the defect lifecycle and its states.
4.2 The Analysis Rules

Using the defect lifecycle, PMs continue to monitor each of the defect states Submitted, Under-review, Assigned, Resolved, Closed and Rejected. PMs want to make sure that a defect moves from the submitted state through its lifecycle to the closed or rejected state in a short period of time. Otherwise, if a defect stays in a particular state for a long period of time this may impact the project deadline. In this paper, we studied the defect lifecycle and developed analysis rules that indicate to the PM a problem or a potential problem that he or she needs to investigate. We describe these rules below and in section C we implement these rules in a 3D visualization. To simplify the presentation of the rules, we use the following terminology: S, U, A, R, C, and J refer to Submitted, Under-review, Assigned, Resolved, Closed and Rejected states respectively. The symbol # refers to the number of defects at a state. The terms $t1$ and $t2$ refer to two points in time in which $t1$ is before $t2$. $\Delta$ refers to the variance in number of defects at a state. For
example, $\Delta S = #St2 - #St1$ is the number of defects at Submitted state at time $t2$ minus the number of defect at Submitted state at time $t1$. Similarly, $\Delta U, \Delta A, \Delta R, \Delta C$ and $\Delta J$ is the variance in number of defects at Under-review, Assigned, Resolved, Closed and Rejected states respectively.

1. Given $\Delta S = #St2 - #St1$. $\Delta U = #Ut2 - #Ut1$. If $\Delta S = 0$ and $\Delta U = 0$ then no new defects are submitted. This indicates that testers are not making progress in finding new defects. This rule shows an issue/problem at the Submitted state.

2. Given $\Delta J = #Jt2 - #Jt1$. If $\Delta J > 0$ then rejected defects are increasing. This indicates that testers are not finding valid defects. This rule discovers an issue/problem at the Rejected state.

3. Given $\Delta R = #Rt2 - #Rt1$. $\Delta C = #Ct2 - #Ct1$. If $\Delta R > 0$ and $\Delta C = 0$ then there are defects in the Resolved state that are waiting to be tested and closed. This indicates that testers are not making progress in testing and closing defects. This rule shows an issue/problem at the Closed state.

4. Given $\Delta U = #Ut2 - #Ut1$. $\Delta A = #At2 - #At1$. $\Delta J = #Jt2 - #Jt1$. If $(\#Ut2 > 0)$ and $(\Delta U >= 0)$ and $(\Delta A = 0)$ and $(\Delta J = 0)$ then there are defects in the Under-review state but no progress has been made to move them to Rejected or Assigned state. This rule shows an issue/problem at the Under-review state.

5. Given $\Delta R = #Rt2 - #Rt1$. $\Delta C = #Ct2 - #Ct1$. If $(\#Rt2 > 0)$ and $(\Delta R = 0)$ and $(\Delta C = 0)$ then there are no defects resolved and moved to the closing state. This indicates that developers are not making progress in resolving defects. This rule shows an issue/problem at the Resolved state.
6. Given $\Delta U = \#Ut2 - \#Ut1$. $\Delta A = \#At2 - \#At1$. $\Delta C = \#Ct2 - \#Ct1$. If ($\#At2 > 0$) and $(\Delta U = 0)$ and $(\Delta C = 0)$ and $(\Delta A > 0)$ then number of defects at the Assigned state is increasing because testers tested some defects and they failed so they reassigned them to developers. This rule shows an issue/problem at the Assigned state.

4.3 The 3D Visualizations for Project Defects

This section presents two 3D visualizations that are developed using the model at section 4.1. These visualizations are the 3D visualization of project defects and the 3D visualization of defect history information.

4.3.1 The 3D Visualization of Project Defects

This section uses defects information, the model; in section 4.1, and the analysis rules; in section 4.2, and generates two 3D visualizations depicted in Figures 4.2 and 4.3. The first visualization, see Figure 4.2, is a 3D landscape view that displays the defects found in a project across different periods of time, say months. Each month is presented by a green section, and it contains cylinders that represent defects. The cylinders are placed in rows and columns. Each row represents a week in a month, and each column represents one of the six states of defect (Submitted, Under-review, Assigned, Rejected, Resolved, and Closed); the states in the figure are orders from left to right. The height of the cylinder denotes number of defects at a certain state; for example, the height of the cylinder at column 1 shows that number of defects at the Submitted state. Cylinders are colored to distinguish between the states. In the figure, the colors yellow, tan, pink, gold, green, and mint cream denote the Submitted, Under-review, Assigned, Resolved, Closed, and Rejected states respectively.
Next, we add the analysis information to the visualization. This approach applies the business rules defined in section 4.2 above to the defects data. The results of the rules indicate issues at particular states that need to be investigated by the PM. With respect to the rules defined in Section 4.2, if rule 1 holds, the Submitted states is colored red. Similarly the rules 2, 3, 4, 5, or 6 holds the states Under-review, Assigned, Rejected, Resolved, or Closed is colored red respectively. In Figure 4.2, the Submitted state at week four of September is red; this indicates that rule one at section 4.2 holds between the third and the fourth weeks of September. Similarly, the Resolved at the third week of October is red; rule five holds between the third and fourth weeks of October.

Figure 4.2: Zoom-in view for project defects
4.3.2 The 3D Visualization of Defects History Information

The second visualization developed by our approach is depicted in Figure 4.3. This visualization displays the defects history across multiple releases of the same project or multiple projects. In Figure 4.3, the defects of two releases are placed behind each other to simplify their comparison. The second visualization objective is to help PMs compare the progress in resolving defects and identify trends across multiple releases or project. For example, this visualization provides PMs the following insights:

1. Release or a project that has more red across a month or a period of time indicates worse turn around in resolving defects.
2. Release or a project that has more red at the states Submitted, Rejected, or Closed indicates problems with testers productivity (see rules 1, 2, and 3 at section 4.2 above). Similarly, more red at the states Under-Review, Resolved, or Assigned indicates problems with developers productivity (see rules 4, 5, and 6 at section 4.2).
3. Release that has more defects found, found and closed, and found and rejected.
4. Release that has more defects that are not resolved at the last month of the release indicates that the release may have slipped deadlines (i.e., delayed).

We developed a prototyping tool called 3DProjView to produce the 3D visualizations produced by the approach; see Section 4.4 for details. Figures 4.2 and 4.3 were produced by the 3DProjView. The colors, the height of the cylinders, and the location of sections are configurable by the tool. In Figure 4.3, for example, it is possible to place the two releases horizontally next to each other.
4.4 3DProjView Tool Support

To conduct the study, we developed a prototype tool called 3DProjView. The tool provides 3D visualization for defects information. It is developed using C# programming language. It uses an XML input file that contains the defects information, data which need to be displayed, the relationships between the data, and configuration information. The XML file is generated automatically using a python programming language script. For example, to generate Figure 4.2 above, we exported a number of defects at each state and at each week from a defect tracking tool at an industrial firm to CSV file. We added the relationships shown in Figure 4.1 between the states of defect. Furthermore, we add configuration information that defines sections, cylinders, colors, and layout. Then we ran the python script using the CSV as input to generate the XML file. Finally, the 3DProjview read the CSV file as input and generated the visualization.

Figure 4.3: 3D view for multiple projects defects
To ease the understanding of the presented data, the tool provides the following interaction features: 1) Zooming in/out, 2) User can hover over a box in the view and get more detailed information in the top right corner, 3) User can drag the figure and move it anywhere on the screen, 4) User can click the right mouse button to rotate the visualization around the x-axis 360 degrees; to stop rotation, the user clicks the right mouse button. While the visualization is rotating, the user can drag and move the figure to different area on the screen, and 5) Pressing the keys 0, 9, 8, 7, 6, and 5 shows the front, back, right side, left side, top and diagonal view respectively. For each of these views you can drag, zoom in/out, and hover over a box to see more details. It is important to note that 3DProjView is a prototype tool developed for the study.
SOFTWARE ARTIFACTS TRACEABILITY

Different stakeholders such as product managers, PMs, business analysts, developers, and testers are involved in the development of software systems. Product managers investigate, select, and develop products for an organization. They consider numerous factors such as the intended demographic, the products offered by the competition, and how well the product fits with the company’s business model. PMs are assigned by the performing organization to achieve project objectives. Business analysts or system engineers develop project requirements, developers implement the system’s requirements and testers verify these requirements.

Software traceability refers to the process of creating/discovering and maintaining links between the different artifacts i.e., how does a particular source code element relate to a corresponding design element. At the most general level, traceability links can be categorized into vertical and horizontal traceability links. Horizontal traceability refers to links within a model (intra-artifact links). Vertical traceability links are links across models. Under each of the two general link types we have causal links and non-causal links as defined in (Maletic et al., 2005 and Maletic et al., 2003). The main points of causal, and non-causal links are reiterated in the definitions that follow.

- **Definition: Causal Links** - Represent relationships that have an implied logical ordering between source and sink. For e.g., bug reports cannot be produced unless the source code is available. A causal link is always directional because it implies causality from source to sink; something happens and causes something else to happen. This establishes a partial ordering in time among the entities involved (Spirtes et al., 2001). When this partial order in time is violated, it is possible that the link between the source
and sink has been broken. Mathematically, the causal relationship is transitive, irreflexive, and anti-symmetric.

- **Definition: Non-Causal Links** – These represent relationships that must agree with each other, but the causality cannot be clearly determined. For e.g., multiple versions of the same document in different languages must agree, but there need not be a causal relationship among them. A non-causal link is un-directional. Changes to the source or sink of a non-causal relationship can cause the logical semantics of the link to become invalid. Mathematically, the non-causal relationship is transitive, reflexive, and symmetric.

Most horizontal links are explicit, for example, two classes in a UML class diagram have a link since they have an association between them. But some links might exist within a model even though they are not explicitly linked. We call these hidden links i.e., links within a model that help in traceability. For e.g., three classes in a design model work towards a common feature of the system but there is no obvious linking between them on the UML class diagram. These links could help in feature location or impact analysis tasks. Such a link (or set of links) could be a causal or non-causal. We consider links across models and hidden links (within a model as well as across models) to be most important for traceability. Traceability link semantics such as those presented by Zisman et al., 2004 and Lamb et al., 2011 are viewed as complementary to the above definitions.

The general consensus is that traceability between different artifacts helps reduce development and maintenance time and cost thereby improving the quality of the system (Klimpke and Hildenbrand, 2009). However, there is no published evidence of the effect of traceability links in software maintenance tasks in industry. Our research focused on
developing a traceability model that supports artifacts traceability across the whole lifecycle of the software development of the projects (requirements, design, coding, testing and maintenance).

In this chapter, section 5.1 presents real life problems that we found at an industrial firm due to the lack of artifacts traceability. Section 5.2 presents a survey that we conducted at the firm to develop a traceability system. Section 5.3 presents the model used by the traceability system, and section 5.4 presents a tool called TraceLink that we developed to prototype the traceability system.

5.1 Real Life Scenarios

This section describes interesting scenarios that have occurred in real projects that have caused problems for the software firm interviewed in this study. They have occurred because there is no (visual/textual) traceability system that links the different repositories of the project and no automatic notification when artifacts in the repository changed. These scenarios were brought to light by conducting interviews during the survey with PMs, testers and developers.

Scenario One: A project was developed in which there was no traceability links between the project’s builds and test. During the project, many internal builds are developed that needed to be tested by the testing team; the last build of the internal builds is the release build. The procedure utilized between developers and testers was that the developers load the internal build on a specific machine and notify testers of the new build through email. Sometimes developers forget to notify testers of new builds and the testers become aware of the build late in the testing cycle. Some builds were tested late and that caused the project’s schedule to be delayed.
Scenario Two: A project was developed in which two organizations were involved in developing the project. The organizations use an XML file as an interface between them. One organization develops the file while the other uses it in one of its software components. The process used by organizations was that when a new version of the XML file is developed, the organization using the XML file is notified through email. Developers have reported that many new versions of the XML file were developed and the using organization weren’t aware of them on time. Such situations affected the project’s schedule.

Scenario Three: In another project, a device that has a network interface card (NIC) was developed. The NIC is utilized to get the internal data off of the device. Customers buy these devices and use them in their network. The purpose of the NIC is to present the device data to users. Sometimes device software is updated in which new data in the device is added or existing data is deleted. In this case, a new version of the NIC software is developed to pull the new data. In practice what happens is that customers update the device software but don’t update the NIC software and vice versa. Here, linking is needed in which a customer can see a link to all versions of the software for both NIC and the device itself.

In each of the above scenarios, a traceability system that is well maintained would have reduced the time taken and mitigate the problems that occurred in communication, development and maintenance.

5.2 Industry Survey

We studied the linking system utilized at a large industrial firm. The objective was to identify the type of artifacts used at industry and the traceability links interested to
different stakeholders. At the firm, they develop a set of artifacts, described below, stored at different repositories. These repositories are utilized throughout the development cycle of software systems. The firm uses both the waterfall model and Scrum agile methods via the Rally tool (https://help.rallydev.com/rally-reports-and-charts) to develop their software systems.

The following are managed by the repositories utilized at the company:

- Project management artifacts such as the project plan, project charter, and statement of work.
- Requirements artifacts.
- Design artifacts such as use cases, sequence, class, state, and activity diagrams.
- Source code.
- Code Inspection records.
- Tests artifacts such as test cases and their execution results.
- Defects records.
- Builds and Releases artifacts contain builds and releases information.

At most established industrial firms and including the one used in this dissertation, the repositories that store the above artifacts are not linked together. Some of the above systems are purchased from different third party vendors. Companies that develop software projects are always in a need to link some or all of these repositories to help developers, testers, and PMs manage projects and meet deadlines.

In order to provide insight into the first research question, a survey was conducted at a large industrial firm. Three product managers (PdM), four (PMs), two business
analysts (BA), nine testers (T), and seven developers (D) were used in the survey to gain input into the types of traceability links they were interested in Table 5.1 shows the number and range of experience among the stakeholders at the industrial firm. These stakeholders were asked to point out the type of links that would be helpful for them to perform their corresponding roles in the organization and the most important fields of each type of artifact that they are interested in Table A.1 at Appendix-A presents a form (excluding the x’s) that was presented to each project stakeholder. It contains for each artifact the fields that belong to that artifact and cross references that links the artifact to other artifacts. Stakeholders pointed out the fields and the links that they are interested in by marking an ‘x’ near the specified row. These results are summarized by majority vote in Table A.1 at Appendix-A. An ‘x’ refers to the most interesting fields to each stakeholder. The fields that are in bold represent fields that if changed, the appropriate stakeholder needs to be notified.

We summarize Table A.1 below.

- **Requirements Artifacts**: Developers, testers, PMs, PMs, and business analysts are interested in the ID/version number, description, author, and date/version number fields.

- **Design Artifacts**: Developers are interested in ID/version number, name, description, author, date and status. PMs are interested in status field.

- **Code Artifacts**: Developers are interested in the fields ID/version number, name, date, owner, and description.

- **Build Artifacts**: Developers and testers are interested in the fields ID/version number, location, and date. PMs are interested in ID/version number and date.
- **Release Artifacts:** Developers and testers are interested in the fields ID/version number, location, date, and release notes. Product managers, PMs, and business analysts are interested in the fields Id/version number, date, and release notes.

- **Test case artifacts:** Testers are interested in the fields ID/version number, title, name, test phase, status, last attempt, expected duration, actual duration, tester(s) names, expected results, and obtained results in the test case artifact.

- **Project Management artifacts:** PMs, product managers, and business analysts are interested in ID/version number, name, author, date, and description fields.

- **Defect Artifacts:** Developers, testers, and PMs are interested in ID/version number, description, create date, reported by, assigned to, status, attachments, and logs/screenshots. Testers and PMs are also interested in severity, estimated hours, planned release, lifecycle fixed in, and actual field hours.

- **Code inspection artifacts:** Developers are interested in ID/version number, project name, description, review date, participants, material, defect logs, and status. PMs are interested in ID/version number, project name, and status.

There are two important findings based on this survey. *First,* the type of traceability links among system’s artifacts is based on the role of the stakeholder. *Second,* the type of stakeholder role also determines which field a stakeholder is interested in. It is important to point out that one of the PMs at the firm mentioned that he is interested in all linking between repositories to show traceability when an audit occurs.
Table 5.1: Description of stakeholders interviewed

<table>
<thead>
<tr>
<th>Stakeholders</th>
<th>Range of Experience (years)</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developers</td>
<td>[5-10]</td>
<td>7</td>
</tr>
<tr>
<td>Testers</td>
<td>[3-5]</td>
<td>9</td>
</tr>
<tr>
<td>PMs</td>
<td>[7-10]</td>
<td>4</td>
</tr>
<tr>
<td>Product Managers</td>
<td>[5-10]</td>
<td>3</td>
</tr>
<tr>
<td>Business Analysts</td>
<td>[3-5]</td>
<td>2</td>
</tr>
</tbody>
</table>

5.3 Traceability Model

Based on the industrial survey, a traceability meta-model was developed. Different stakeholders in the software development cycle are interested in different traceability links between different artifact repositories. A traceability model must support traceability links that are specific to each stakeholder role. A meta-model that requires support for links specific to each stakeholder role is shown in Figure 5.1. In this meta-model, a TraceModel consists of one or more TraceLinks. A TraceLink contains two TraceElements (artifacts). A TraceElement has type and status attributes. The type is either “source” or “target”. The status is set to “modified” when one or more of the important fields in TraceElement are modified. Stakeholder has a Stakeholder role that traces one or more specific TraceLinks. For example, a Stakeholder Role can be a project manager, a developer or a tester. Each of these stakeholders traces specific links based on their roles.

Using the meta-model, the overall traceability model for the industrial firm is reflected in Figure 5.2. The model when implemented will overcome some of the issues
that have occurred at the firm due to lack of traceability. The links are defined based on
stakeholders’ roles. For example, using the model, a project manager can trace a
requirement through design, code, code inspection, build, test, and defects. A developer
can trace requirements to design, code, code inspection, build, and defects. The
information in Figure 5.2 should be taken into account when developing a traceability
tool.

The following summarizes the five views in Figure 5.2:

1. Product managers are interested in the links between these artifacts (project
   management, requirements) and (requirements, release).

2. Developers are interested in the following links (requirements, release), (requirements,
   design), design, code), (code, code inspection), (code, build), (code, defect), (release,
   build), (build, defect), and (defect, test).

3. Testers are interested in the following links (requirements, release), (requirement, test),
   (release, build), (build, defect), (build, test), and (test, defect).

4. Business analysts are interested in the links between these artifacts (project
   management, requirements) and (requirements, release).

5. Project manager are interested in the links between all artifacts.

While implementing the traceability model, reference fields for each artifact in the
repository are used to point to another linked artifact in the same or external repository. To
support the traceability across the artifacts utilized at the firm, all cross references shown
in Table A.1 need to be used. The traceability system shall support notifying stakeholders
interested in the change. For example, stakeholders interested in an artifact
change, register for the artifact. When the owner of the artifact modifies the artifact,
he/she sets the status field in *TraceElement* to “modified”. The system automatically checks for each artifact that has a status field set to modified and sends notifications to registered stakeholders.

---

**Figure 5.1: The traceability meta-model**

---

**Figure 5.2: Five different views of traceability links**
In Figure 5.2, PrM refers to project manager, the edge names represent stakeholders interested in links between the two endpoints/artifacts.

5.4 TraceLink Tool Support

A prototype link tracing tool, TraceLink was developed that reflects the overall traceability model for the industrial firm as shown in Figure 5.2. See Figure A.1 at Appendix-A for a screenshot of the tool. The traceability links between the repositories are stored in an XML file. The “Show Linked Artifacts” button displays all artifacts that are linked to the one selected on the left pane in the window. In Figure A.1, all artifacts linked to Req 2.1.1, for example, are displayed in the right pane. It is also able to show which repositories are linked.

When the user clicks the “Check for Changed Artifacts” button, the tool shows in the right pane all artifacts that have their status field set to “modified”. For each of these artifacts, it lists the impacted artifacts that could be within the same repository or another repository. In the future, instead of users clicking the “Check for Changed Artifacts” button, the tool will automatically notify registered users of a modified artifact via email. Note that the goal of the study was not to determine if TraceLink was useful, rather it was to determine if the presence of traceability links (via the TraceLink) tool had an effect in solving the tasks. We use TraceLink as a means to view traceability links in the experimental group as described below in the experimental design.
EMPIRICAL STUDY – PROJECT TASKS AND RESOURCES

The purpose of the study is to provide a quantitative evaluation of the effectiveness and efficiency of our approach when compared to state-of-the-art project management approaches namely: Gantt chart and tables. We use Gantt charts and tables as a control so as to make a comparison of the added benefit of the proposed approach for analyzing project status, progress, forecast, and identifying potential problems. Project status describes where the project stands now; project progress describes what has been accomplished by the team so far; and project forecast predicts the future status and progress (Cable et al., 2004). To conduct the study, we developed a prototype tool called 3DProjView (See Section 3.3).

The study addresses the following research questions:

RQ1: Does the use of 3DProjView increase the accuracy of analyzing project tasks and resources compared to Gantt charts and tables?

RQ2: Does the use of 3DProjView reduce the time in analyzing project tasks and resources compared to Gantt charts and tables?

RQ3: Do the potential benefits of using 3DProjView in terms of accuracy and time depend on the user’s experience (i.e., beginner versus advanced)?

RQ4: Do the potential benefits of using 3DProjView in terms of correctness and time depend on the user’s background (i.e., academic versus industry practitioner)?

RQ5: What are the types of tasks that perform better using 3DProjView versus Gantt charts and tables?
6.1 Experimental Design and Hypotheses

Following the template by Wohlin et al., 1999, the experiment seeks to evaluate the effect of using 3D visualizations via 3DProjView, for analyzing project tasks and resource information with respect to effectiveness (accuracy) and efficiency (speed) from the point of view of the researcher in the context of industry practitioners and software engineering students. The detailed null hypotheses are given below. The alternative hypotheses are 1-tailed predicting 3DProjView performs better. The hypothesis testing is done at 95% significance (alpha = 0.05).

H₁: There is no significant difference in accuracy when analyzing project tasks and resource information using 3D landscape visualization via 3DProjView versus using Gantt chart and tables. µ (Accuracy_{3DProjView}) = µ(Accuracy_{Gantt Chart/Tables})

H₂: There is no significant difference in speed when analyzing project tasks and resource information using 3D landscape visualization via 3DProjView versus using Gantt chart and tables. µ (Speed_{3DProjView}) = µ(Speed_{Gantt Chart/Tables})

H₃: There is no significant difference in difficulty when analyzing project tasks and resource information using 3D landscape visualization via 3DProjView versus using Gantt chart and tables. µ (Diff_{3DProjView}) = µ (Diff_{Gantt Chart/Tables})

H₄: User experience (beginner vs. advanced) does not significantly interact with 3DProjView or Gantt chart and tables to have an effect on task accuracy or speed.

H₅: User background (academic vs. industry) does not significantly interact with 3DProjView or Gantt chart and tables to have an effect on task accuracy or speed.

An overview of the experiment is shown in Table 6.1. The main factor in this study is the method used to display and analyze project tasks and resources information.
While analyzing the results we also looked at secondary factors such as the effects of subjects’ experience (advanced vs. beginner) and background (academia vs. industry) on accuracy, speed, and difficulty. The difficulty level refers to a rating of easy, medium or hard given by the participant for each of the fourteen analysis tasks described in Section 6.4.

Table 6.1: Experiment overview

<table>
<thead>
<tr>
<th>Goal</th>
<th>Study the effect of using 3D landscape visualization via $3DProjView$ to display and analyze project tasks and resource information.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Independent variables</td>
<td>Method used to display and analyze project tasks and resources information: 3D landscape view via $3DProjView$ versus Gantt chart and tables via MS Project.</td>
</tr>
<tr>
<td>Dependent variables</td>
<td>Accuracy, speed, difficulty level</td>
</tr>
</tbody>
</table>
| Secondary factors | Experience (advanced, beginner)  
Background Type (industry, academia) |

6.2 Participants

A total of 42 participants were involved in the study. There were 32 software engineering students and 10 industry practitioners. Six of the industry practitioners were project managers and four were project leads. We split the participants into two groups. Only one group, Group 1, used $3DProjView$ to solve the tasks. The other group, Group 2, used Gantt charts and tables within MS Project. To ensure a variety of abilities in both groups, the students were randomly placed into two groups making sure that an equal
distribution of abilities was maintained. Industry participants were also randomly assigned into one of two groups of five people each. For industry participants, we categorized people with 5 or more years of experience in software projects as advanced and the rest were assigned to the beginner experience level. For academic participants, we looked at student grades and students that were getting an A or B to date in the software engineering course were put into the advanced experience level, with the rest falling into the beginner category. Both the control and experimental groups had twelve subjects categorized with advanced experience and nine subjects categorized as beginners with respect to experience for a total of 21 subjects in each group. Six of the subjects from the industry are very skilled practitioners who have been managing software development for at least 5 years. The participants were not in any way related to the development or planning history of the two projects used in the study. This is important to maintain confidence in our results.

6.3 Subject Systems

Two real software projects from industry are used in this study. The first project, Project 1, contained 55 tasks and 21 resources. The second project, Project 2, contained 70 tasks and 7 resources. The duration of each project was one year. Due to the confidential nature of the industrial projects, we are unable to disclose project details.

We now present the type of visualizations each group used during the study. Table 6.2 shows the two groups of participants and the types of visualizations they had access to. The third column in the table lists the source of the visualization. The visualizations are further described below.
3D Visualizations used by the 3DProjView group. The visualizations were produced using the 3DProjView tool (see section 3.3 for details on how they are developed).

- **3D visualization of resources:** This visualization shows the project resources and the tasks assigned to each resource. In the visualization, each resource is represented by a long box. Boxes standing on it represent tasks assigned to the resource. Colors represent task status: pink denotes that task completed on time, Olive denotes a task was completed late, yellow denotes no progress (percent of completion is not increasing), red denotes slipping planned start or finish date, green denotes that a task is performing as expected (i.e., the percent of completion is increasing), and orange denotes not started. See Figure B.2 at Appendix-B.

- **3D visualization of tasks:** This visualization shows the project tasks and their status with respect to schedule and work completed. In the visualization, a green flat surface denotes the project. Each task in the project is represented by a box. The y-dimension of the box denotes percentage of completion, the x-dimension denotes planned duration, and the z-dimension denotes remaining duration to complete the task. The color and dimensions of the task boxes remain the same as for 3D visualization of resources. See Figure B.1 at Appendix-B.

- **3D visualization of project cost:** This visualization is the same as the above 3D visualization of tasks except the metrics data that are mapped to the dimensions x, y, and z of task boxes, the color, and status rules are different. The x-dimension denotes planned cost for the task, the y-dimension denotes the percent of completion, and the z-dimension denotes the remaining cost. The colors present cost status. Pink denotes task completed
within budget, olive denotes task completed over budget, green denotes task is in progress and cost is as planned, red denotes task is in progress and over budget, and orange task has not started yet. See Figure B.3 at Appendix-B.

- **3D visualization of historical information**: This visualization presents four baselines of a project. Each baseline presents the tasks information at specific point in time. The y-dimension of the task box denotes the percent of completion. The x-dimension denotes finish date, and z-dimension denotes start date. Color represents task status: green denotes that the task is making progress, and red denotes that the task has slipped its start and/or finish dates. See Figure B.4 at Appendix-B.

Visualizations used by the MS Project 2010 group. We used version 2010 of MS Project. These are the types of visualizations/views available to users in this group.

- **Project Resources**: The resource usage sheet provided by MS Project was used to show project resources. This sheet shows for each resource, the tasks that are assigned to the resource and how much time is used and remaining for each task. The left side of the visualization shows project resources and for each resource lists the tasks owned by the resource. On the right, a calendar shows the planned and actual work hours for each task. This is a standard view displayed by MS Project. See Figure B.6 at Appendix-B.

- **Project Tasks**: A Gantt chart is presented on the right side of the presentation, and on the left a table contains the project tasks. The columns include the name of the task, the duration of the task, the start and finish date of the task, and the remaining duration for the task. It uses a line inside the bar to present the percent of completion. See Figure B.5 at Appendix-B.
• **Project Cost:** To show project cost, we used a table view from MS project that lists project tasks. For each task, it shows the planned cost, the actual cost, and the remaining cost. See Figure B.7 at Appendix-B.

• **History Information:** To show the schedule variances for project tasks, we used a table view from MS project that lists the project tasks, and for each task it lists four baselines. At each baseline it lists, the percent of completion, the start date and finish date. See Figure B.8 at Appendix-B.
Table 6.2: Project Visualizations used in the study

Each group had 21 subjects (12 advanced and 9 beginners)

<table>
<thead>
<tr>
<th>Group</th>
<th>Visualization Views</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group 1</td>
<td>• Project Resources - 3D Visualization of resources (showing their tasks status and progress)</td>
<td>• 3DProjView tool</td>
</tr>
<tr>
<td>3DProjView</td>
<td>• Project Tasks - 3D Visualization of tasks (showing schedule info. and work progress for project tasks)</td>
<td>• 3DProjView tool</td>
</tr>
<tr>
<td></td>
<td>• Project Cost - 3D Visualization of tasks (showing cost info. and status for project tasks)</td>
<td>• 3DProjView tool</td>
</tr>
<tr>
<td></td>
<td>• History information – 3D visualization of historical information (showing schedule info. and variances for tasks)</td>
<td>• 3DProjView tool</td>
</tr>
<tr>
<td>Group 2</td>
<td>• Project Resources - Resource usage - Table (showing their tasks status and progress)</td>
<td>• MS Project (resource usage sheet)</td>
</tr>
<tr>
<td>Control: MS Project 2010</td>
<td>• Project Tasks - Gantt chart and table (showing schedule info and work progress status for project tasks)</td>
<td>• MS Project (Gantt Chart)</td>
</tr>
<tr>
<td></td>
<td>• Project Cost – Table (showing cost info. and status for project tasks)</td>
<td>• MS Project (cost table)</td>
</tr>
<tr>
<td></td>
<td>• History information - Table (showing schedule info. and variances for tasks)</td>
<td>• MS Project (variance table)</td>
</tr>
</tbody>
</table>

6.4 Tasks

Based on consultation from a few project managers at an industrial firm and the first author’s 15 year industrial project management experience, we defined in a set of
analysis tasks (shown in Table 6.3) typically performed by project managers. The tasks are associated with four main areas or concerns of analysis: 1) assess current status, 2) forecast future status, 3) assess the progression of progress and trends, and 4) identify and manage potential problems. The table also lists the visualizations, described above to solve the task.

Table 6.3: Experiment Tasks

Project 2 was used for T5, T6, and T7. Project 1 was used for the rest.

<table>
<thead>
<tr>
<th>Task</th>
<th>Concern</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1: Might the overall project schedule slip?</td>
<td>Forecast future status</td>
</tr>
<tr>
<td>T2: List two tasks that may slip their planned finish date.</td>
<td>Identify potential problem</td>
</tr>
<tr>
<td>T3: What should you do to prevent the tasks you identified in question (T2) from slipping their planned finish date?</td>
<td>Manage potential problem</td>
</tr>
<tr>
<td>T4: List two tasks that are currently slipping their planned finish date.</td>
<td>Assess current status</td>
</tr>
<tr>
<td>T5: Which resource may need help?</td>
<td>Identify potential problem</td>
</tr>
<tr>
<td>T6: What can a Project Manager do to help the resource you identified in question T5?</td>
<td>Manage potential problem</td>
</tr>
<tr>
<td>T7: Compare Henry’s progress to John’s progress. Who has the better performance?</td>
<td>Assess current status (resources productivity)</td>
</tr>
<tr>
<td>T8: Might the project run over the planned budget?</td>
<td>Forecast future status</td>
</tr>
<tr>
<td>T9: List two tasks that may run over their planned budget.</td>
<td>Identify potential problem</td>
</tr>
<tr>
<td>T10: Overall, across the four baselines, is the percentage of completion for project tasks increasing or holding (remains the same)?</td>
<td>Assess the progression of progress and trends</td>
</tr>
</tbody>
</table>
Table 6.3: Continued

<table>
<thead>
<tr>
<th>T11: Overall, across the four baselines, is the finish date for project tasks staying the same, holding, or slipping (delayed)?</th>
<th>Assess the progression of progress and trends</th>
</tr>
</thead>
<tbody>
<tr>
<td>T12: Across baselines 2, 3, and 4, which baseline has the worst progression in tasks percentage of completion compared to the previous baseline?</td>
<td>Assess the progression of progress and trends</td>
</tr>
<tr>
<td>T13: Across baselines 2, 3, and 4, which baseline has the worst progression in tasks duration compared to the previous baseline?</td>
<td>Assess the progression of progress and trends</td>
</tr>
<tr>
<td>T14: Which tasks continued to slip their finish date since the start of the project?</td>
<td>Assess the progression of progress and trends</td>
</tr>
</tbody>
</table>

6.5 Running the Study and Data Collection Method

Two weeks before the study, the subjects from academia were given a 1-hour lecture about project management and a demo about MS Project. Students that were in the 3DProjView group were given a tutorial about the 3DProjView tool. Furthermore, they were given a quiz-like assignment to make sure that they read the materials ahead of time. The study was conducted at two different locations (computer lab, industrial firm) and took approximately one hour. The subjects from industry did the study at their usual workspace. Since they were familiar with MS Project and used it on a daily basis, they (participants in the 3DProjView group) were only given a demo on the 3DProjView tool. Both groups used an electronic question/answer survey tool built in-house that recorded their answers for each task. The subjects entered their solution and the level of difficulty (easy, medium, hard) they faced for each of the tasks in the space provided to them. The
question/answer survey tool recorded the time it took to answer each task and the total
time. All the data generated from this survey tool was later tabulated for analysis.

6.6 Results

In this dissertation, we proposed five research questions and corresponding
hypotheses for the first four RQs. We addressed each of these hypotheses for the first four
RQs and provide a task analysis addressing RQ5 (more information related to RQ5 is also
presented in the discussion section).

6.6.1 Accuracy and Time

Each question was given a score based on the correct answer given by subjects,
denoting accuracy. The projects used in this study are real industry projects that were
already complete. The answers to these questions come directly from analyzing those
projects that were completed. The industry project managers were consulted to provide
accurate answers to these questions. That forms our ground truth against which we test
accuracy.

Since the data (see Figure 6.1 for the distribution) was not normally distributed,
we used non-parametric tests (in particular the Mann-Whitney test) to determine
significance. In the Figure, we see an outlier with respect to tasks accuracy when using
the 3DProjView tool; the outlier could be due to a participant from Group 1 did not know
how to use the tool. With respect to speed, there is also an outlier using the Gantt chart
tables. The outlier could be due to the fact that one student from Group 2 was not as
familiar with Gantt chart as participants from industry.

Using Mann-Whitney test, all values are reported at 95% confidence. Table 6.4
shows the p-values for the Mann-Whitney test for each of the individual tasks as well as
all the tasks accumulatively. With respect to all the tasks (last row), we see that
3DProjView performs significantly better ($p < 0.0001$) than Gantt chart and tables. In particular, 3DProjView was 40% more accurate. Task T9 seemed to benefit the most from the use of 3D landscape visualization via 3DProjView followed by T3, T13, T14, and T2. Tasks T7 and T8 were very close to achieve a significant rating ($p=0.05$). One reason for the other tasks not showing any difference in accuracy could be due to the fact that they were inherently harder tasks. Considering all tasks, we are able to reject $H_1$ with 3DProjView performing significantly better.

![Figure 6.1: Boxplots of the distribution for accuracy and time for the two groups](image)

With respect to speed (time), we are also able to reject the null hypothesis $H_2$, showing there is a significant difference ($p$-value $< 0.0001$) in speed when analyzing project tasks and resources using 3D landscape visualization via 3DProjView, resulting in
less time. Tasks T1, T2, and T8, followed by T9 and T11 benefited the most from using 3D landscape visualization via 3DProjView. In addition, the percentage decreased overall in time for 3DProjView was approximately 39.6%.

Table 6.4: Resulting p-values of the Mann-Whitney test (one-tailed)

Tasks are grouped by the concern they represent

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Accuracy</th>
<th>Speed</th>
<th>Difficulty</th>
<th>Concern</th>
</tr>
</thead>
<tbody>
<tr>
<td>T2</td>
<td>0.0322*</td>
<td>0.0001*</td>
<td>0.0001*</td>
<td>Identify potential problem</td>
</tr>
<tr>
<td>T9</td>
<td>0.0007*</td>
<td>0.0032*</td>
<td>0.0006*</td>
<td>Identify potential problem</td>
</tr>
<tr>
<td>T5</td>
<td>0.7013</td>
<td>0.515</td>
<td>0.2859</td>
<td>Identify potential problem</td>
</tr>
<tr>
<td>T3</td>
<td>0.0008*</td>
<td>0.9989</td>
<td>0.0012*</td>
<td>Manage potential problem</td>
</tr>
<tr>
<td>T6</td>
<td>0.3958</td>
<td>0.2406</td>
<td>0.3324</td>
<td>Manage potential problem</td>
</tr>
<tr>
<td>T8</td>
<td>0.0556</td>
<td>0.0001*</td>
<td>0.0049*</td>
<td>Forecast future status</td>
</tr>
<tr>
<td>T1</td>
<td>0.3958</td>
<td>0.0001*</td>
<td>0.0144</td>
<td>Forecast future status</td>
</tr>
<tr>
<td>T4</td>
<td>0.2141</td>
<td>0.0001*</td>
<td>0.6185</td>
<td>Assess current status</td>
</tr>
<tr>
<td>T7</td>
<td>0.0565</td>
<td>0.0892</td>
<td>0.0352*</td>
<td>Assess current status (resources productivity)</td>
</tr>
<tr>
<td>T10</td>
<td>0.8546</td>
<td>0.6881</td>
<td>0.6745</td>
<td>Assess the progression of progress and trends</td>
</tr>
<tr>
<td>T11</td>
<td>0.0933</td>
<td>0.0241*</td>
<td>0.0449*</td>
<td>Assess the progression of progress and trends</td>
</tr>
<tr>
<td>T12</td>
<td>0.145</td>
<td>0.515</td>
<td>0.1034</td>
<td>Assess the progression of progress and trends</td>
</tr>
<tr>
<td>T13</td>
<td>0.0008*</td>
<td>0.6837</td>
<td>0.1068</td>
<td>Assess the progression of progress and trends</td>
</tr>
<tr>
<td>T14</td>
<td>0.0041*</td>
<td>0.1425</td>
<td>0.0324*</td>
<td>Assess the progression of progress and trends</td>
</tr>
<tr>
<td>All</td>
<td>&lt;0.0001*</td>
<td>&lt;0.0001*</td>
<td>&lt;0.0001*</td>
<td></td>
</tr>
</tbody>
</table>
6.6.2 Difficulty Level

Subjects were asked to rate the difficulty of each task right after they completed it during the study (the timer was paused during this time). Both groups were asked to rank the difficulty of the tasks on a three point scale (1 = easy, 2 = medium, and 3 = hard).

See the fourth column of Table 6.4. We were able to reject null hypothesis H3 over all tasks because there was a significant difference (p-value < 0.0001) in reported difficulty level. Tasks T2 followed by T9, T3, T8, T14, T7, and T11 were found to be less difficult when using 3DProjView as reported by subjects. No difference in difficulty was reported in tasks T1, T4, T5, T6, T10, T12, and T13. For all participants, we had 10 tasks ranked easy, 3 tasks ranked medium and 1 task ranked hard. With respect to 3DProjView, 9 tasks ranked easy, 4 ranked medium, and one ranked hard. For the Gantt chart and tables group, 5 tasks ranked easy, 8 tasks ranked medium and one ranked hard. It is clear from the above numbers that the Gantt chart and tables group found the tasks not as easy to solve when compared to the 3DProjView group.

6.6.3 Secondary Interaction Effects of Experience and Background

In the study, we had two secondary variables: Experience (advanced, beginner) and Background (industry, academia). We now present analysis of the interaction of these factors with the main factor (group – 3DProjView vs. Gantt chart and tables) on the results. Figure 6.2 and Figure 6.3 show the interaction plots for the Experience and Background secondary variables respectively.

We generated two models (one for correctness and the other for time) using linear mixed models regression to determine interaction effects of the secondary variables. For correctness, the results indicate that user experience does not significantly interact (p-
value=0.495) with our main factor. This means that 3DProjView benefitted both experience levels. In terms of time, our model does not find any significant interactions of experience levels either (p-value=0.582) which also means that both experience levels benefitted equally well from 3DProjView. Although we see that beginners were slightly more accurate than advanced users, this difference is not significant and quite negligible. In terms of time, advanced users were slightly better than beginners taking less time overall, but again this result is not significantly different.

3DProjView outperformed Gantt chart and tables in both correctness and completion time, regardless of the experience level as seen in Figure 6.2. In both correctness and completion time, the users of 3DProjView has less variability than the users of Gantt chart and tables; this shows more consistence performance for 3DProjView than Gantt chart and tables. The performance of the two levels of users of 3DProjView was very close in terms of correctness. This is an interesting insight that supports that 3DProjView was effective for both advanced users and beginners.

Tukey’s test (critical value d = 3.799) on analysis of differences between categories for both correctness and time shows significant differences between groups (supporting our previous hypotheses H₁ and H₂) but no difference in the interaction of experience with the main factor. Given the above analysis, we cannot reject H₄.
Similar to the Experience variable, we also generated two models for the Background variable using linear mixed models regression. For correctness, the results indicate that user background does not significantly interact (p-value=0.187) with our main factor. This means that 3DProjView benefitted both background levels as well. In terms of time, we see that background is very close to being significant (p-value=0.05).

We used Tukey’s analysis of differences to obtain greater resolution of patterns, and saw that industry users in the Gantt chart and tables group were placed significantly different from industry users of 3DProjView, academic users of 3DProjView, and academic users of Gantt chart and tables. The latter three fell into one group with no difference in means among them. One possible explanation for this could be that industry users were more familiar with using Gantt chart and felt secure spending more time with it even though the additional time taken did not have a significant effect on their accuracy.

Figure 6.2: Interaction effects of experience on accuracy and speed.
Similar to experience, we also see that the gap in correctness between industry users and students (see Figure 6.3 for interaction plot) is much less when using 3DProjView – an important observation. In terms of accuracy, we find that students were slightly more accurate (almost negligible difference) with 3DProjView. An even interesting observation is that in terms of time, industry users spent way more than academic users in the Gantt chart and tables group. This seems counterintuitive at first. However, as mentioned above, we have to remember that industry users seemed more “at-home” with MS Project’s Gantt chart and tables because this is something they use on a daily basis.

Based on the analysis above for the Background secondary variable we have to conclude that with respect to time, background does significantly interact with the main factor with respect to a subset of the data namely industry users that use Gantt chart and tables. Given this finding, we can reject $H_5$ for the dependent variable time. Overall
however, the effect of interaction of background is close to being significant (p-value=0.05). A bigger sample and further replication is needed to further strengthen this claim. In the study, $H_1$, $H_2$, $H_3$, and $H_5$ are rejected while $H_4$ cannot be rejected.

6.6.4 Task Analysis

One of the research goals of our study (RQ5) was to identify the types of tasks for which 3DProjView provides an advantage over the Gantt chart and tables. To answer this question, we compared for each task described in Section 6.4, the performance of our participants in terms of average accuracy and speed using 3DProjView versus using Gantt charts and tables and provide reasoning about potential causes of the difference in performance. See Figures 6.4 and 6.5 supporting task analysis discussions. This section is purely qualitative in nature as we do not compare results of statistical tests. Rather we try to provide explanations for a difference in averages (for correctness and time) shown in Figures 6.4 and 6.5. We do not claim any statistical significance in this section. All hypothesis testing and statistical testing was done in the prior three sections.

T1: Might the overall project schedule slip?

In terms of correctness and completion time 3DProjView is better. The project tasks in the 3D visualization showed all project tasks in a single view. Gantt chart subjects had to scroll to see all project tasks and that might have impacted their performance. Since identifying whether a task is slipping its planned duration is important to PMs, the 3D visualization colored each task slipped its planned duration by red color. With respect to Gantt chart users, they have to compare the duration for each task.

T2: List two tasks that may slip their planned finish date.
In correctness and completion time, 3DProjView outperformed Gantt chart. To solve T2, subjects needed to compare the remaining duration and percent of completion for tasks that are in progress. The difference in performance is probably due to the ease of visualizing and comparing box dimensions. In the case of Gantt chart, subjects needed to compare the lines inside the bars associated with each task to solve T2.

T3: What should you do to prevent the tasks you identified in question (2) from slipping their planned finish date.

Similar to T2, 3DProjView outperformed Gantt chart in both correctness and completion time. To solve this task, subjects needed to find a resource that has a small work load to help the resource in T3. The difference in performance is probably due to the fact that the 3D visualization grouped all tasks performed by a resource together and used color to show the tasks status while in Gantt chart some resources for tasks are spread across multiple screens.

T4: List two tasks that are currently slipping their planned finish date.

Subjects using 3DProjView performed better in both correctness and completion time than subjects using Gantt charts. The project tasks 3D visualization used the color red to represent tasks slipping their planned finish date and that helped in achieving better results.

T5: Which resource may need help?

3DProjView performed better in terms of correctness and completion time than Gantt charts and tables. T5 is close to T2, but in T5 subjects needed to find the resource that has more tasks that are slipping or close to slip their planned finish date. The difference in performance is due to the fact that the project resource 3D visualization
used colors to present task status. Moreover, 3DProjView allowed zooming in/out and rotating the view. We also notice that both performed close in terms of speed. This was because the user had to compare all resource task status to determine the one who had the most work load. It takes time to answer this question accurately. We find three subjects in particular took the longer but got the question correct.

T6: What can a Project Manager do to help the resource you identified in question 5?

In both average correctness and completion time, 3DProjView performed better than Gantt charts and tables. T6 is similar to T3, but T6 uses both Gantt chart and project resource sheet (table). Subjects utilizing 3DProjView performed better because the project resource 3D visualization grouped each resource tasks together, used colors to present status, and allowed interacting with the visualization (e.g., zoom in/out and rotating the view).

T7: Compare Henry’s progress to John’s progress. Who has the better performance?

3DProjView outperformed Gantt chart and tables in both correctness and completion time. The difference in performance is due to the same reasons listed for T6 above.

T8: Might the project run over the planned budget?

3DProjView outperformed tables in both correctness and completion time. Similar to project tasks and resources 3D visualizations, the project cost 3D visualization uses boxes to present project tasks and showed all of them in one view. In addition, colors
were used to present the costs status, and finally interacting with the visualization was allowed. All of these features helped achieving better performance over tables.

T9: List two tasks that may run over their planned budget?

In correctness and completion time, 3DProjView outperformed tables. To solve this task, subjects needed to compare the percent of completion to the remaining budget for each task. The difference in performance is due to the fact that the project cost 3D visualization used the dimension of the box to present percent of completion and remaining planned budget; this helped subjects identify tasks that are close to run over budget.

T10: Overall, across the four baselines, is the percentage of completion for project tasks increasing or holding (remains the same)?

3DProjView performed lower in terms of accuracy than tables when we visually compare the averages in Figure 6.4. This is the only task where tables performed better in terms of accuracy. We believe this could be because of the way the four baselines were shown to the user. There was a lot of occlusion in the 3D view and it requires the user to do a lot of mental comparisons. In terms of average time (Figure 6.5), we don’t see too much of a difference.

T11: Overall, across the four baselines, is the finish date for project tasks staying the same, holding, or slipping (delayed)?

3DProjView outperformed tables in both correctness and completion time. The better performance is due to the fact that the 3DProjView subjects looked for tasks colored pink.
T12: Across baselines 2, 3 and 4, which baseline has the worst progression in tasks percentage of completion compared to the previous baseline?

*3DProjView* performed better in both correctness and completion time than tables. The better performance is because *3DProjView* subjects looked for the baseline that had more tasks colored yellow. When we look closely at the average time, we don’t see too much difference in the bars between *3DProjView* and Gantt + Tables, but a bigger difference in accuracy between the two. This indicates that for history comparisons, *3DProjView* is more accurate even though the time taken is very close to the Gantt + Tables group.

T13: Across baselines 2, 3 and 4, which baseline has the worst progression in tasks duration compared to the previous baseline?

In correctness and completion time, *3DProjView* performed better than tables. The better performance is because *3DProjView* subjects looked for the baseline that has more tasks colored pink. The same observation as in T12 can be made in T13 related to how accuracy is highly impacted in a positive way when using *3DProjView* without much difference in the time taken in each group.

T14: Which tasks continued to slip their finish date since the start of the project?

*3DProjView* outperformed tables in both correctness and completion time. To solve T14, subjects compare the finish date for project tasks across the four baselines. The better performance is due to the fact that *3DProjView* subjects looked for tasks colored pink across the four baselines.

Based on the above qualitative analysis above, we find that *3DProjView* performed better on average when participants were asked to explicitly name tasks that
were slipping (T14). See Figures 6.4 and 6.5. In addition, 3DProjView also showed a large difference in average accuracy compared to Gantt + charts, when the task related to assessing the progression of progress and trends using baselines (T12, T13). Another important one that showed a big difference in average accuracy and average time were tasks T7 and T8 that related to assessing current status of resource productivity and forecasting future status.

Figure 6.4: Average correctness per task
6.6.5 Post-questionnaire Analysis

The industry participants that used the 3DProjView tool were given an additional post-questionnaire and were also interviewed after the study. The goal was to get their feedback on the effectiveness of using 3D visualization to present project information. Some of the important highlights are presented below.

- Four out of five participants reported that displaying the project data using 3D visualization made it easy to understand the data. One participant stated that he could easily focus in on what was needed for the task. Another participant stated that it was much easier to understand with a visual representation than reading numbers off a page.

- Five participants reported that displaying the project data using 3D visualization helped them answer the study questions.

- Five participants reported that the tool was easy to use.
• Some industry participants stated that they needed time to remember shortcut keys to make it effective to use it. They suggested adding a clickable arrow to allow rotation of the model, possibly 90 degrees at a time. That would help someone pick it up and not have to memorize key strokes in order to rotate the model.

6.7 Discussion

The focus of this study is not business intelligence nor is it to add more information on Gantt charts as proposed by Leach, 2010. The main focus of this study is to provide PMs a holistic view on the status of a project and answer important questions about the project status without flipping around many different tables and/or charts. We used 3D to benefit from using the z dimension to present all data in a single view. Using 2D to show all the data especially in the case of large data can clutter the view. The 3 dimensions helped us show the front, back, top, right and left sides, and the diagonal view of the data. Even when occlusion was present, the features that were needed to understand the project as a whole stood out due to interactivity features available to users. Furthermore, using 3D space allowed us to zoom into specific parts of the data to further analyze it. In this case, we believe 3D actually helped rather than hindered performance. The interaction was quite basic and did not need users to memorize any complex key or mouse strokes. The visualization itself was kept simple enough for the average PM to understand and use.

The results of the study indicate that using 3D landscape visualization to analyze project tasks and resources performance information leads to significantly more accuracy, less time used, and less difficulty while solving tasks. We split the tasks into four concern areas (See Section 6.4). Based on Table 6.4, we see that we have significant results in all
four areas. The area most benefitted from the visualization was the “identify potential problems” concern. We see that in both tasks T2 and T9 we have significant results in all three: accuracy, speed, and difficulty levels. This implies that the visualization was extremely useful to users in identifying potential problems. If problems are identified early, projects have a higher chance of getting done on time. The “manage potential problem” concern is one of the harder concerns to deal with because it depends on the skill of the user/project manager. In this category we find that in one of the tasks, (T3) the visualization made the task significantly easier. Another more involved concern that also depends on the skill of the user is the “forecast future status” concern. Here, we also find that even though accuracy of the forecast using the visualization was not significantly different from the Gantt chart group, the speed in which they gave the answer was. This indicates that 3DProjView performed more efficiently than Gantt chart and tables. Finally the last concern (going in order as listed in Table 6.4), is “assessing of status and progress.” Within this concern category, there were five tasks (out of seven) that had a significant difference in either accuracy, speed, or difficulty level – making majority of the tasks benefit from the visualization. Revisiting RQ5 from the task analysis in Section 6.6.4, we can state that all four task categories benefit from the 3D visualization. Based on the qualitative analysis shown in Section 6.6.4, we find the following set of tasks benefitted most: when participants were asked to explicitly name tasks that were slipping, when the task related to assessing the progression of progress and trends using baselines and when assessing current status of resource productivity and forecasting future status.

With respect to user experience, we did not find any difference between the two groups. However, in terms of background, we find industry users in the Gantt chart
group took significantly more time. We attribute this to the fact that these users were more familiar with MS Project and hence felt comfortable using it for a longer time to get to a solution (even though the actual accuracy of the solution was not significant). In summary, 3DProjView subjects achieved overall significant results in terms of accuracy, speed, and difficulty in analyzing project tasks and resource information. Another important result was that the visualization reduced the gap of performance between industry users and students (See Figure 6.3). This is an important finding because given the visualization, we can have students, who are usually novices, solve tasks faster and hopefully improve their skills that lead them to become experts in the future.

The observations presented here are useful to 3D visualization tool developers in order to get more industry acceptance of the tools they develop to present project information. It is also a call for improved visualizations for project information (The study shows this to be the case). It also incorporates different views from different stakeholders of a project.

6.8 Threats to Validity

The four main threats to validity are internal, external, construct and conclusion validity. With respect to internal validity, the study was between-subjects and did not suffer from learning effects between groups. Within each group, each task was from a different category and did not overlap so there were no learning effects involved there either. The study was done in a classroom/lab setting or an industrial setting with dedicated time allotted to it, so as to minimize any other external factors that might have an effect on the results. So even though we used two sites to conduct the study, similar working conditions were present at both sites. An alternative mechanism would have
been to conduct a more in-depth, exploratory study with the professionals, using think aloud protocols. We leave this a possible future extension of this research. We did however interview industry practitioners after the study and report on highlights of this interview in Section 6.6.5. A tutorial was given to all subjects prior to the study to make sure all subjects were at the same knowledge level. None of the subjects were familiar with the study systems.

A threat to validity that one might bring up in this category is that the comparison is not relevant and it compares two tools (Gantt charts and tables in MS Project and 3DProjView) which have been engineered to support different main goals. The Gantt chart, one might argue, is a construction tool and not a summarization or presentation tool. To address the above point, we point out that Gantt chart and tables provided by MS project are widely used to analyze project tasks and resources by PMs. One can build a graph using Excel to mimic 3DProjView but there is no agreement on how to build the graph nor on its data contents which will make it highly ad-hoc. We compare our approach to an existing widely used approach which is standard. Our goal is to eventually have MS Project support 3DProjView visualizations. If embedded into MS Project, it will be more accepted by PMs since they already use MS Project and don’t need to use yet another tool to assess the state of their project.

Another possible threat to validity is the possible amount of interaction between the two tools. For instance, it could be that data was not presented in the right table structure in MS Project (to answer the required questions) and users needed to do various operations to ‘create’ the right view. With respect to the point above that the data may not be presented correctly in MS project, MS project is a widely used product in project
management for a long time and uses standard table structures all of which were available to the users. Gantt chart contents and format is fixed across project management tools including MS project. We did not intentionally hide any data. These are Gantt charts that were in use at the industry firm. With respect to the point above that data may be laid out in the right way in the 3D tool, we did not intentionally place any one answer closer to the front view of the user. Even in 3DProjView, subjects had to spend time zooming in, rotating, comparing boxes dimensions, etc. to answer some of the study’s questions. We conjecture that we see a difference in time because even though interaction is needed in both tools, 3DProjView abstracts a lot of the data revealing only what the user wants to see at a certain point in time. In Gantt charts, these abstractions are missing (i.e., a lot more searching and mental filtering is needed to see the big picture).

One might also argue that the 3D visualization was pre-built to favor the respective tasks thereby making it hard to defend the comparison of a per-task fine-tuned visualization. Gantt charts and tables supported by MS Project are currently being used by PMs to answer similar questions used in our study. We wanted to compare our approach to what is currently being used. If we had to pre-tune the Gantt chart to support these tasks, we would have to add detail to it which would defeat the purpose of using a Gantt chart in the first place. This would be something other than the Gantt chart which a majority of PMs don’t use. Also, we would have to train PMs to be able to use this new version of the Gantt chart. If we did this, it is also not without criticism. One can argue that a better pre-tuned version is possible and so on and so forth. To avoid these issues, we wanted to use the current state of the art and since MS Project is widely used, we
decided to use their representation of the Gantt chart. Note that the participant worked inside MS Project and was able to manipulate things if they wished.

One can also argue that color coding is clearly an important aid and that if we somehow added information into the Gantt chart to show this, it would be equally fast as the 3D visualization. Color coding is clearly an important aid. One can thus pose the following question: Can we infer that, if we had the colorizing option in Gantt charts, all tasks that were arguably easier done with the 3D visualization would also succeed equally well/fast using the Gantt charts? Furthermore, one could argue that a simple 2D tree map with cells color coded as now (and the information currently placed in the height being added, for instance, into a 2D glyph drawn atop of the cells) would be as effective as the 3D visualization. To address the above point, we would like to clarify that we are comparing our approach to the commonly used Gantt chart. What is referred to above indicates that one needs to come up with a better representation in 2D of the Gantt chart. This is interesting and we think is an option for future research. We are not aware of such a 2D tool widely used in industry. This provides opportunity to see if 2D is enough, however we leave this as future work. Since all our data is made available, future researchers can run another study to compare 3DProjView with a 2D variant. We also note that the major visual cues users employed (for the 3D case) were spatial grouping and color coding however this alone was not sufficient to answer the questions. They also had to interact with the layout to make sure they checked all possibilities before giving the answer.

With respect to construct validity, we choose to measure accuracy, speed, and difficulty level. Since our study goal was to assess the effect of using 3D landscape
visualization, we measure performance on the tasks via accuracy and speed. Better performance indicates higher accuracy and lower speed i.e., time to complete task. The difficulty level gave us some indication of how difficult the tasks were for them to solve. The students were not told and did not know the hypotheses in advance. Some of the project leads from industry have done project management tasks before and their input may be affected by their previous role, however we tested for this in the interaction effects and present our results in Section 6.6.3. It would be nice to have the same number of industry experts as students but it was hard to find many project leads/managers volunteer to invest their time for this study. We found 10 practitioners who were willing to participate.

External validity deals with generalizing the results of the study. The study is conducted only at one industrial firm. Replicating this study with a larger sample from different organizations in different domains and with different systems and tasks can help reduce this threat. We leave this as future work. We did have different experience levels within our subject pool. Our results from the study are only generalizable to analysis tasks that fall within the four areas of concerns mentioned. Even though we used MS Project to represent the Gantt chart and tables group, any software with similar capabilities should give similar results.

With respect to conclusion validity, due to low sample size and non-normality of data we use the Mann-Whitney test for hypotheses testing. We also constructed models using linear mixed effects regression (to determine interaction effects), which is a robust method to determine significance for unbalanced groups.
7 EMPIRICAL STUDY – SOFTWARE DEFECTS

The purpose of our study is to provide a quantitative evaluation of the effectiveness and efficiency of our approach in analyzing defects information compared to 2D bar charts and table format. Defects tracking systems utilize different 2D charts to display and analyze defects information; bar chart is commonly used. We compare our approach to both bar charts and tables; the results of the study are applicable to bar charts and tables. It is important to point out the term 2D chart in the study refers to bar charts; Figure C.7 at Appendix – C is a bar chart.

In this study, we address the following research questions:

RQ1: Does the use of 3DProjView increase the accuracy of analyzing software defects status compared to 2D charts and tables?

RQ2: Does the use of 3DProjView reduce the time in analyzing software defects status compared to 2D charts and tables?

RQ3: Do the potential benefits of using 3DProjView in terms of accuracy and time depend on the user's experience (i.e., low versus high)?

RQ4: Do the potential benefits of using 3DProjView in terms of correctness and time depend on the user's background (i.e., academic versus industry practitioner)?

7.1 Experiment Design and Hypotheses

This experiment seeks to evaluate the effect of using 3D visualization via 3DProjView, for analyzing project defects status with respect to effectiveness (accuracy) and efficiency (speed) from the point of view of the researcher in the context of industry practitioners and software engineering students. The detailed null hypotheses are given
below. The alternative hypotheses are 1-tailed predicting 3DProjView performs better.

The hypothesis testing is done at 95% significance (alpha =0.05).

H1: There is no significant difference in accuracy when analyzing project defects status using 3D landscape visualization via 3DProjView versus using 2D chart and tables.

\[ \mu (\text{Accuracy}_{3DProjView}) = \mu (\text{Accuracy}_{2D \text{ Chart/Tables}}) \]

H2: There is no significant difference in speed when analyzing project defects status using 3D landscape visualization via 3DProjView versus using 2D chart and tables.

\[ \mu (\text{Speed}_{3DProjView}) = \mu (\text{Speed}_{2D \text{ Chart/Tables}}) \]

H3: There is no significant difference in difficulty when analyzing project defects status using 3D landscape visualization via 3DProjView versus using 2D chart and tables.

\[ \mu (\text{Diff}_{3DProjView}) = \mu (\text{Diff}_{2D \text{ Chart/Tables}}) \]

H4: User experience (low vs. high) does not significantly interact with 3DProjView or 2D chart and tables to have an effect on task accuracy or speed.

H5: User background (industry vs. students) does not significantly interact with 3DProjView or 2D chart and tables to have an effect on task accuracy or speed.

Table 7.1 contains the main factors in the study. The method (3DProjView versus 2D bar charts and tables) is the independent variable. Three dependent variables, speed, accuracy, and difficulty are used in the study. The difficulty level refers to a rating of easy, medium or hard given by the participant for each of the ten analysis tasks defined in section D below. While analyzing the results we also looked at secondary factors such as the effects of subject's experience (low vs. high) and background (academia vs. industry) on accuracy and speed.
Table 7.1: Experiment overview

<table>
<thead>
<tr>
<th>Goal</th>
<th>Study the effect of using 3D landscape visualization via 3DProjView to display and analyze project defects status.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Independent variables</strong></td>
<td>Method used to display and analyze project defects status: 3D landscape view via 3DProjView versus 2D bar charts</td>
</tr>
<tr>
<td><strong>Dependent variables</strong></td>
<td>Accuracy, speed, and difficulty level</td>
</tr>
<tr>
<td><strong>Secondary factors</strong></td>
<td>Experience (high, low); Background type (industry, academia)</td>
</tr>
</tbody>
</table>

7.2 Participants

Forty participants were involved in the study: 30 software engineering students and 10 practitioners (six project managers and four project leads) from the industry. We split the participants into two groups, Group 1 and Group 2. Group 1 used 3DProjView to solve the experiment tasks while Group 2 used 2D charts and tables using Microsoft Excel software. To ensure a variety of abilities in both groups, the subjects were randomly placed into two groups making sure that an equal distribution of abilities was maintained. Table 7.2 presents the experience and background of participants. For industry participants, we categorized people with 5 or more years of experience in software projects as high and the rest were assigned to the low experience level. For academic participants, software engineering graduates students were put into the high experience level, and undergraduate students in a software engineering class were put into the low category. Each group has 6 subjects with high experience and 14 low. Six of the subjects from the industry are very skilled practitioners who have been managing software development for at least 5 years.
Table 7.2: Participants experience and background

<table>
<thead>
<tr>
<th>Experience</th>
<th>Industry (10)</th>
<th>Students(3)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>&gt;=5 years</td>
<td></td>
<td></td>
</tr>
<tr>
<td>experience</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt; 5 years</td>
<td></td>
<td></td>
</tr>
<tr>
<td>experience</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Group 1 (20)</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Group 2 (20)</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

7.3 Subject Systems

The study used four real software projects from industry. To preserve the company privacy, we named the projects Project A, Project B, Project C, and Project D. Projects A and B are considered medium size and they have 76 and 89 defects respectively. Project C and D are large size projects and they have 112 and 227 defects respectively. With respect to the visualizations utilized by each group see section 7.5.

7.4 Tasks

The goal in the study is to define tasks that allow participants to analyze the defects information of a system and identify problems in resolving these defects. We identified in Table 7.3 ten analysis tasks that assess the defects status and potential problems in resolving the defects. The first four tasks T1, T2, T3, and T4, analyze the defects status and discover problems with developer or testers across a week period of time. Tasks T5, T6, T7, and T8, however, analyze the defects status and identify problems across a month period. Finally Tasks T9 and T10 compare the progress of
resolving defects across two projects. The next section describes each visualization utilized by participants to answer each of the study tasks.

Table 7.3 - Experiment Tasks

<table>
<thead>
<tr>
<th>#</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>Comparing weeks 9/12/2011 and 9/19/2011, what is the problem with testers?</td>
</tr>
<tr>
<td>T2</td>
<td>Comparing weeks 10/10/2011 and 10/17/2011, what is the problem with developers?</td>
</tr>
<tr>
<td>T3</td>
<td>Comparing weeks 10/17/2011 and 10/24/2011, what is the problem with developers?</td>
</tr>
<tr>
<td>T4</td>
<td>Comparing weeks 11/7/2011 and 11/14/2011, what is the problem with testers?</td>
</tr>
<tr>
<td>T5</td>
<td>In Aug. 2011, what is the problem with testers?</td>
</tr>
<tr>
<td>T6</td>
<td>In Sept. 2011, what is the problem with testers?</td>
</tr>
<tr>
<td>T7</td>
<td>In Oct. 2011, what is the problem with developers?</td>
</tr>
<tr>
<td>T8</td>
<td>Comparing Sept. 2011 and Oct. 2011, which month has worse progression and why?</td>
</tr>
<tr>
<td>T9</td>
<td>Comparing project A to project B which project has worse progression and Why?</td>
</tr>
<tr>
<td>T10</td>
<td>Comparing project C to project D which project has worse progression and Why?</td>
</tr>
</tbody>
</table>

7.5 Running the Study and Data Collection Method

The subjects from academia were given a 1-hour lecture about project defects management. Both industry and student subjects were given a demo on how to use the 3DProjView tool. The study was conducted at two different locations (classroom, industrial firm) and took approximately one hour. The subjects from industry did the study at their usual workspace. All subjects from both groups were given hard copy consent forms; they provided their written consent to participate in the study. Both groups were given a hard copy of the questions and for each question they recorded the start and finish date. They entered their solution, the start and end time, and the level of
difficulty (easy, medium, and hard) they faced for each of the tasks in the space provided to them. They carried out the tasks using the visualizations in Table 7.4; the table lists the visualizations utilized to solve each task.

Table 7.4 - Visualizations utilized to solve study tasks

<table>
<thead>
<tr>
<th>Task #</th>
<th>Visualizations used by Group 1</th>
<th>Visualizations used by Group 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>Project A defects information displayed in Figure C.1 at Appendix - C.</td>
<td>Project A defects information displayed in Figure C.7 at Appendix – C.</td>
</tr>
<tr>
<td>T2</td>
<td>Same figures as in T1.</td>
<td>Same figures as in T1.</td>
</tr>
<tr>
<td>T3</td>
<td>Project B defects information displayed in Figure C.2 at Appendix - C.</td>
<td>Project B defects information displayed in Figure C.8 at Appendix – C.</td>
</tr>
<tr>
<td>T4</td>
<td>Same figures as in T3.</td>
<td>Same figures as in T3.</td>
</tr>
<tr>
<td>T5</td>
<td>Project C defects information displayed in Figure C.3 at Appendix - C.</td>
<td>Project C defects information displayed in Figure C.9 at Appendix – C.</td>
</tr>
<tr>
<td>T6</td>
<td>Same figures as in T5.</td>
<td>Same figures as in T5.</td>
</tr>
<tr>
<td>T7</td>
<td>Project D defects information displayed in Figure C.4 at Appendix - C.</td>
<td>Project D defects information displayed in Figure C.10 at Appendix – C.</td>
</tr>
<tr>
<td>T8</td>
<td>Same figures as in T7.</td>
<td>Same figures as in T7.</td>
</tr>
<tr>
<td>T9</td>
<td>Projects A and B defects information displayed in Figure C.5 at Appendix - C.</td>
<td>Same figures as in tasks T1 and T3.</td>
</tr>
<tr>
<td>T10</td>
<td>Projects C and D defects information displayed in Figure C.6 at Appendix - C.</td>
<td>Same figures as in tasks T5 and T7.</td>
</tr>
</tbody>
</table>
7.6 Results

The study proposed four research questions and corresponding hypotheses for the first four questions. We studied the impact of accuracy, speed, and difficulty and the interaction of experience and background on the method utilized (3DProjView versus 2D charts and tables). Based on this design, the suitable parametric test for hypothesis testing is Multi-factor Analysis of Variance (ANOVA). Multi-factor Analysis of Variance (ANOVA) and post ANOVA Least Square Mean (LSM) analyses were conducted to statistically evaluate the results; a linear Completely Random Design (CRD) model. We performed the test using the Statistical Analysis Software (SAS) (SAS Institute Inc., 2004).

7.6.1 Accuracy, Speed and Difficulty

Table 7.5 presents the results of the ANOVA analysis for accuracy, speed and difficulty. At 95% confidence level (p-value<0.05), the effects of used method (3D visualization versus 2D chart and tables) was significant with respect to accuracy, speed and difficulty. Considering the accuracy p-value 0.0297, we are able to reject the null hypothesis H₁ with 3DProjView performing significantly better. With respect to speed (time), we are also able to reject the null hypothesis H₂ showing a significant difference (p-value 0.0168) in speed when using 3DProjView to analyze project defects. Moreover, the results show a significant difference (p-value 0.0267) in reported difficulty level that allows us to reject the null hypothesis H₃.
Table 7.5 - Results of ANOVA analysis for accuracy, time and difficulty

<table>
<thead>
<tr>
<th>Effect</th>
<th>Num DF</th>
<th>Den DF</th>
<th>F Value</th>
<th>Pr &gt; F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>method</td>
<td>1</td>
<td>32</td>
<td>5.18</td>
</tr>
<tr>
<td>Time</td>
<td>method</td>
<td>3</td>
<td>32</td>
<td>6.36</td>
</tr>
<tr>
<td>Difficulty</td>
<td>method</td>
<td>3</td>
<td>32</td>
<td>5.39</td>
</tr>
</tbody>
</table>

In Table 7.5, Num DF is the number of degrees of freedom in the model. Den DF is the number of degrees of freedom associated with the model errors; F Value is the F statistic for the given predictor and test statistic; Pr > F is the p-value associated with the F statistic of a given effect and test statistic.

7.6.2 Secondary Interaction Effect of Experience and Background

In the study, we had two secondary variables: Experience (high, low) and Background (industry, academia). Table 7.6 presents ANOVA results of the experience and background interaction on accuracy and speed. At 95% confidence level, the effect of experience on accuracy (p-value 0.6686) and on speed (p-value 0.7058) is not significant; therefore we couldn’t reject the null hypothesis $H_4$. Similarly, the effect of background interaction on accuracy (p-value 0.3053) and on speed (0.4437) is not significant; so we could not reject $H_5$ hypotheses.
Table 7.6 - Results of ANOVA analysis, experience & background interaction on accuracy and time

<table>
<thead>
<tr>
<th>Interaction</th>
<th>Effect</th>
<th>Num DF</th>
<th>Den DF</th>
<th>F Value</th>
<th>Pr &gt; F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experience on Accuracy</td>
<td>method</td>
<td>3</td>
<td>32</td>
<td>0.52</td>
<td>0.6686</td>
</tr>
<tr>
<td>Experience on Time</td>
<td>method</td>
<td>3</td>
<td>32</td>
<td>0.47</td>
<td>0.7058</td>
</tr>
<tr>
<td>Background on Accuracy</td>
<td>method</td>
<td>1</td>
<td>36</td>
<td>1.08</td>
<td>0.3053</td>
</tr>
<tr>
<td>Background on Time</td>
<td>method</td>
<td>1</td>
<td>32</td>
<td>0.60</td>
<td>0.4437</td>
</tr>
</tbody>
</table>

7.7 Discussion

The results of the study indicate that using 3D landscape visualization leads to significantly more correct answers and less time when analyzing project defects. Looking into more fine-grained analysis, Figures 8.1 and 8.2 show the average of accuracy, speed and difficulty respectively for each task. 3DprojView subjects achieved better accuracy for tasks T1, T4, T5, T6, T7, T8, T9, and T10 (Figure 7.1). For task T2, subjects who utilized 2D charts and tables achieved slightly better accuracy, and they achieved the same accuracy for task T3, compared to the accuracy of subjects who utilized 3DProjView for those two tasks. It is possible that is due to the fact that task T2 requires less data (one week data) to analyze. However, the difference in accuracy for task T2 is not significant and quite negligible. With respect to average speed, Figure 7.2, 3DProjView achieved better results across all tasks.
Looking at the experience interaction on accuracy, we studied the interaction across the four subgroups: students with high experience (SH), students with low experience (SL), industry with high experience (IH), and industry with low experience (IL). The results are presented in Table 7.7 (a). They results show that the 3D subjects from industry and students with high experience achieved better accuracy. This is an interesting finding in which high experience resulted in better accuracy. The next
subgroup that achieved better accuracy is industry low. It is shown here that overall, industry participants achieved better accuracy. Within the 2D group, students with high experience achieved better. However, within industry participants, both high and low experience achieved the same; this can be due to that fact that industry participants are familiar with using Excel, bar charts, and tables. With respect to experience interaction with speed, Table 7.7 (b), among the participant’s subgroups, the first subgroup that achieved a lower time is the 3D industry with high experience participants; this is an interesting finding in which experience helped the industry participants to finish before any other subgroup. The second subgroup that achieved less time is 3D students with low experience. This result can be due to a wish to finish the study quickly. The third subgroup that achieved less time is 3D industry with low experience; next is 3D students with high experience.

Table 7.7 - Results of ANOVA analysis for experience and background interaction on accuracy

(a) Experience interaction on accuracy

<table>
<thead>
<tr>
<th>subgroup</th>
<th>Estimate</th>
<th>Standard Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D SH</td>
<td>8.3333</td>
<td>0.8559</td>
</tr>
<tr>
<td>3D IH</td>
<td>8.3333</td>
<td>0.8559</td>
</tr>
<tr>
<td>2D IH</td>
<td>7.3333</td>
<td>0.8559</td>
</tr>
<tr>
<td>3D SL</td>
<td>6.9167</td>
<td>0.4280</td>
</tr>
<tr>
<td>2D SH</td>
<td>6.6667</td>
<td>0.8559</td>
</tr>
<tr>
<td>3D IH</td>
<td>5.5000</td>
<td>1.0483</td>
</tr>
<tr>
<td>2D IL</td>
<td>5.0000</td>
<td>1.0483</td>
</tr>
<tr>
<td>2D SL</td>
<td>4.7500</td>
<td>0.4280</td>
</tr>
</tbody>
</table>

(b) Experience interaction on accuracy

<table>
<thead>
<tr>
<th>subgroup</th>
<th>Estimate</th>
<th>Standard Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D SL</td>
<td>22.5000</td>
<td>1.6708</td>
</tr>
<tr>
<td>2D SH</td>
<td>19.6667</td>
<td>3.3415</td>
</tr>
<tr>
<td>2D IL</td>
<td>19.0000</td>
<td>4.0925</td>
</tr>
<tr>
<td>2D IH</td>
<td>18.6667</td>
<td>3.3415</td>
</tr>
<tr>
<td>3D SH</td>
<td>15.3333</td>
<td>3.3415</td>
</tr>
<tr>
<td>3D IL</td>
<td>15.0000</td>
<td>4.0925</td>
</tr>
<tr>
<td>3D SL</td>
<td>13.4167</td>
<td>1.6708</td>
</tr>
<tr>
<td>3D IH</td>
<td>13.0000</td>
<td>3.3415</td>
</tr>
</tbody>
</table>
Looking at the background interaction on accuracy, the results in Table 7.8 (a) show that 3D students and industry participants achieved better accuracy. The 3D students and 3D industry achieved same accuracy; this shows with respect to the use of 3DProjView tool to solve the project tasks, the background is not important. This finding supports that 3DProjView is helpful in achieving high accuracy regardless of the participants' background. However, with respect to the 2D participants, participants with industry background achieved better accuracy; this finding is due to the fact that industry participants may be more used to bar charts than students. With respect to background interaction, the 3D participants with industry background achieved less time, Table 7.8 (b). This may be due to industry participants being more used to working with tools such as 3DProjView, 2D bar charts, and tables.

Table 7.8 - Results of ANOVA analysis for experience and background interaction on accuracy

(a) Background interaction on accuracy

<table>
<thead>
<tr>
<th>subgroup</th>
<th>Estimate</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D Students</td>
<td>7.2000</td>
<td>0.4306</td>
</tr>
<tr>
<td>3D Industry</td>
<td>7.2000</td>
<td>0.7459</td>
</tr>
<tr>
<td>2D Industry</td>
<td>6.4000</td>
<td>0.7459</td>
</tr>
<tr>
<td>2D Students</td>
<td>5.1333</td>
<td>0.4306</td>
</tr>
</tbody>
</table>

(b) Background interaction on speed

<table>
<thead>
<tr>
<th>subgroup</th>
<th>Estimate</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D Students</td>
<td>21.9333</td>
<td>1.4304</td>
</tr>
<tr>
<td>2D Industry</td>
<td>13.8000</td>
<td>2.4776</td>
</tr>
<tr>
<td>3D Students</td>
<td>15.8000</td>
<td>1.4304</td>
</tr>
<tr>
<td>3D Industry</td>
<td>12.6000</td>
<td>2.4776</td>
</tr>
</tbody>
</table>

7.8 Threats to Validity

The four main threats to validity are internal, external, construct and conclusion validity. With respect to internal validity, the study was between subjects and did not
suffer from learning effects between groups. Within each group, tasks were from a
different category and did not overlap so there were no learning effects involved there
either. The study was done in a classroom setting or an industrial setting with dedicated
time allotted to it, so as to minimize any other external factors that might have an effect
on the results. A tutorial was given to all subjects prior to the study to make sure all
subjects were at the same knowledge level. None of the subjects were familiar with the
study systems.

With respect to construct validity, we choose to measure accuracy, speed, and
difficulty level. Since our study goal was to assess the effect of using 3D landscape
visualization, we measured performance on the tasks via accuracy and speed. Better
performance indicates higher accuracy and lower speed i.e., time to complete task. The
difficulty level gave us some indication of how difficult the tasks were for them to solve.
The students were not told and did not know the hypotheses in advance. Some of the
project leads from industry have done project management tasks before and their input
may have been affected by their previous role, however we tested for this in the
interaction effects and presented our results in the above section H.2. It would be nice to
have the same number of industry experts as students but it was hard to find many project
leads/managers willing to volunteer to invest their time for this study. We found 10
practitioners who were willing to participate.

External validity deals with generalizing the results of the study. The study is
conducted only at one industrial firm. Replicating this study with a larger sample from
different organizations in different domains and with different systems and tasks can help
reduce this threat. We leave this as future work. We did have different experience levels
within our subject pool. Our results from the study are only generalizable to analysis tasks that fall within the tasks mentioned in Section $D$ above (i.e., tasks that assess the status of defects and the productivity of developers and testers in resolving defects) and the size of defects used in the study. Furthermore, the results are only generalizable to 2D bar chart and tables. Even though we used MS Excel to represent the 2D bar charts and tables group, any software with similar capabilities should give similar results. With respect to conclusion validity, Multi-factor ANOVA is a robust method to determine significance and the interaction between the study variables.
8 EMPIRICAL STUDY – ARTIFACTS TRACEABILITY

The goal of this study is to investigate the following question: Do traceability links help software maintenance tasks? The question seeks to determine if the presence of traceability links help in software maintenance tasks. This chapter presents details on the study conducted in industry and academia on the effect of traceability links in software maintenance tasks. To conduct the study, we used the traceability tool TraceLink defined in section 5.4.

8.1 Experimental Design and Hypotheses

Following the template by Wohlin et al. 1999, the experiment seeks to analyze the effect of using traceability links via TraceLink, for performing software maintenance tasks with respect to effectiveness (accuracy) and efficiency (speed) from the point of view of the researcher in the context of industry practitioners and software engineering students. The detailed null hypotheses are given below. The alternative hypotheses are 1-tailed predicting TraceLink performs better. The hypothesis testing is done at 95% significance (alpha =0.05).

H₁: There is no significant difference in task accuracy when performing maintenance tasks in the presence of using traceability links via TraceLink (versus not using it). \( \mu(\text{Accuracy}_{\text{with links}}) = \mu(\text{Accuracy}_{\text{without links}}) \)

H₂: There is no significant difference in task speed when performing maintenance tasks in the presence of using traceability links via TraceLink (versus not using it). \( \mu(\text{Speed}_{\text{with links}}) = \mu(\text{Speed}_{\text{without links}}) \)

H₃: Ability level does not significantly interact with TraceLink to have an effect on task accuracy, difficulty or speed.
The overview of the experiment is shown in Table 8.1. The main factor being analyzed is the link-tracing method with two treatments: without links (no TraceLink) and with links (links displayed using TraceLink). While analyzing the results we also looked at secondary factors such as the effects of subjects’ ability level (high vs. low ability) on accuracy, difficulty level and speed combined with the link-tracing method. The difficulty level refers to a rating of easy, medium or hard given by the participant for each of the five software maintenance tasks.

Table 8.1: Experiment overview

<table>
<thead>
<tr>
<th>Goal</th>
<th>Study the effect of traceability links in software maintenance tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Independent variables</strong></td>
<td>Tracing method: with links (using TraceLink), without links</td>
</tr>
<tr>
<td><strong>Dependent variables</strong></td>
<td>Accuracy, speed, difficulty level</td>
</tr>
<tr>
<td><strong>Secondary factors</strong></td>
<td>Ability level (high, low)</td>
</tr>
<tr>
<td></td>
<td>Type (industry, academia)</td>
</tr>
</tbody>
</table>

8.2 Participants

A total of 28 participants were involved in the study: 22 software engineering students and six practitioners (three developers and three testers) from industry. We split the participants into two groups. Only one group used TraceLink to solve the tasks. The other group was not provided with traceability links. To ensure a variety of abilities in both groups, the students were randomly placed into two groups. Industry participants were also randomly assigned into one of two groups of 14 people each. Both the control and experimental groups had eight high-ability subjects and six low-ability subjects. Six
of the subjects from industry are very skilled practitioners who have been developing and testing software for at least 5 years.

8.3 Subject System

A room management system (RMS) was used in the study. This system was developed by students in a software engineering capstone class and refined later by the authors of the dissertation to reflect a complete system. The system is about 10K lines of code. The system is responsible for managing shared office, meeting, laboratory, and teaching space and includes a web-based component and a physical device outside each room. The main features are reserving rooms, displaying reservations and editing reservations.

It is important to point out that the industrial survey provided two important outcomes: (1) the linking between project artifacts, for example, requirements is linked to test, test is linked to build, etc., and 2) five different stakeholders’ traceability views. In this study, we used the first outcome.

A total of 165 artifacts were part of the test, requirements, design, code, code inspection, build and defects repositories. Project management and release artifacts are not included in the study because there were not available for the RMS at the time we conducted the study. We manually traced 402 links between artifacts based on the results of the industrial survey (See Figures 5.1 and 5.2). These links are verified with industrial stakeholders. Table 8.2 shows the different links from source to target between the repositories in our system.
Table 8.2: Number of links in the Room Management System

The number of artifacts in each repository are Test (113), Req (9), Design (19), Code (8), Code Inspection (2), Build (2), Defects (12)

<table>
<thead>
<tr>
<th>Source</th>
<th>Target</th>
<th># Links</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test</td>
<td>Build</td>
<td>113</td>
</tr>
<tr>
<td>Test</td>
<td>Defect</td>
<td>12</td>
</tr>
<tr>
<td>Requirements</td>
<td>Test</td>
<td>111</td>
</tr>
<tr>
<td>Requirements</td>
<td>Design</td>
<td>39</td>
</tr>
<tr>
<td>Design</td>
<td>Code</td>
<td>83</td>
</tr>
<tr>
<td>Code</td>
<td>Code Inspection</td>
<td>16</td>
</tr>
<tr>
<td>Code</td>
<td>Build</td>
<td>16</td>
</tr>
<tr>
<td>Code</td>
<td>Defects</td>
<td>12</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td><strong>402</strong></td>
</tr>
</tbody>
</table>

8.4 Tasks

There were five tasks created for the RMS each under a different category. Task 1 (T1), was about adding new functionality. Task 2 (T2), consisted of changing existing functionality. Task 3 (T3), dealt with fixing incorrect behavior. Task 4 (T4), was responsible for porting existing functionality and Task 5 (T5), dealt with an IDE change. For example T1 was based on adding new functionality: *Customer requested the following new requirement for the RMA system: 2.1.8. For some rooms, the maximum time of reservation is 2 hours for any user. What are the impacted artifacts due to the new requirement?* All tasks, T1, T2, T3, T4, and T5 require tracing the impacted artifacts for RMS. The order in which task should be solved first by participants does not matter. There
is no dependency between the tasks. Table 8.3 shows the tasks used in the study. We have provided all the study material at http://www.csis.ysu.edu/~bsharif/tracelinkstudy and refer the reader there for more details.

Table 8.3: Tasks used in the study

<table>
<thead>
<tr>
<th>ID</th>
<th>Category</th>
<th>Task Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>Adding new functionality</td>
<td>Customer requested the following new requirement for the RMA system. What are the impacted artifacts due to the new requirement? 2.1.8 For some rooms, the maximum time of reservation is 2 hours for any user.</td>
</tr>
<tr>
<td>T2</td>
<td>Changing existing functionality</td>
<td>Customer requested a change to Requirement 2.1.2; the following is the modified requirement. What are the impacted artifacts due the new change? 2.1.2: The Web interface shall display the status of the room. If the room is currently occupied, the occupant decides whether to display his/her name. The time when the meeting will finish must be displayed. The Web interface shall display occupied and available status by a red or green light.</td>
</tr>
<tr>
<td>T3</td>
<td>Fixing incorrect behavior</td>
<td>Customer has found the following defect in which a user (not a level-1 user) is allowed to release a reserved room. A developer has investigated this defect and found that the reservation class code has a bug. What are the impacted artifacts due to this defect?</td>
</tr>
</tbody>
</table>
Table 8.3: Continued

<table>
<thead>
<tr>
<th>T4</th>
<th>Porting existing functionality</th>
<th>The development of a new system is starting, and the functionality of creating and deleting logins to level-1 users supported by the RMA system needs to be ported to the new system. What are the impacted artifacts?</th>
</tr>
</thead>
<tbody>
<tr>
<td>T5</td>
<td>IDE change</td>
<td>Microsoft Studio version 2008 was used to develop the Room Management Appliance system (RMAS). A new version of Microsoft Studio version 2010 is released. Due to the new release, the RMAS is recompiled and a new build 6.3 is created. What are the impacted artifacts?</td>
</tr>
</tbody>
</table>

8.5 Running the Study and Data Collection Method

A week before the study was conducted the participants were given a document describing the subject system. The document contained a description of all the repositories set up for the system as well as information within each repository such as requirements and design documents. This was done to familiarize the participants with the system. The students were also given a quiz-like assignment to make sure they read the materials ahead of time. They were also given a 15 minute lecture on the importance of software traceability and the problems/challenges facing industry.

The study was conducted at three different locations (classroom, computer lab, industrial firm) and took approximately one hour. The tasks were presented to the subjects on dissertation. They were asked to trace the artifacts that were impacted for each software maintenance task. Subjects were asked to note the start and end time for each task on dissertation as well as indicate the level of difficulty (Easy, Average, Difficult).
they faced for each of the five tasks. All the data was collected on dissertation questionnaires and later tabulated.

In addition to the tasks provided to all, people in group 1 (that were provided no links) were provided with an electronic file that contained all artifacts for the system (see Figure A.2 at Appendix-A for an example). People in group 2 were provided with the same material as group 1, and in addition they were also presented with traceability links between the artifacts via the TraceLink tool (see Figure A.1 at Appendix-A). The tool contains all artifacts and the 402 links listed in Table 8.2. A sample demo for group 2 on how to use the tool was given before the study began. The academia subjects took the study in a computer lab while industry subjects did the study at their workplace. After all subjects were done with the five tasks, they completed a post-questionnaire.

8.6 Results

Due to non-normality of the data (based on results from the Shapiro-Wilk test) and a sample size less than 30, we use the non-parametric Mann-Whitney test (for non-paired samples) to determine statistical significance between the two groups. Figure 8.1 shows boxplots for the three dependent variables: accuracy, speed (time), and difficulty level.
8.6.1 Accuracy, Speed, Difficulty, and Interaction effects

Each question was given a score based on the correct answer given by subjects. The score denotes the accuracy. Table 8.4 shows the p-values for the Mann-Whitney test for each of the five tasks as well as all the tasks accumulatively. With respect to all the tasks (last row), we see that the only significant variable is accuracy. We are thus able to reject the null hypothesis $H_1$, showing that there is a significant difference ($p$-value = 0.0001) in task accuracy when performing maintenance tasks in the presence of traceability links presented via TraceLink, resulting in higher accuracy. T4 seemed to benefit the most from the presence of traceability links via TraceLink ($p$-value = 0.0003) followed by T5 and T2. No significance was detected in T1 and T3. One reason could be due to the fact that T4, T5, and T2 were harder than T1 and T3. When all subjects are
considered, the users that were presented traceability links in *TraceLink* were 86.06% more accurate than subjects that did not use traceability links.

With respect to speed and difficulty, the only time we achieve significance is with T5. Thus we cannot reject H₂ which states that there is no difference in speed between the group presented with traceability links via *TraceLink* vs. the control group. It is important to mention that subjects using *TraceLink* pointed out that they did not have enough training on using the tool before performing the tasks of the study and that impacted their speed. They said that they were learning how to use the tool while they were solving the tasks. This may be one of the reasons why we did not see more improvements in speed. Not enough training was given to subjects due to their limited time. A future replication of the experiment with more training would help determine if this was the case. It is important to mention that industry subjects were 21% faster in solving the tasks using traceability links than their colleagues that did not use the tool that showed them traceability links in the system.

In order to determine interaction effects, a linear mixed-effects regression model is fit to each of the dependent variables: accuracy, speed, and difficulty level with respect to all tasks accumulatively. The explanatory variables were Group (with links, without links), Ability (high, low) and Type (industry, academia). Table 8.5 shows the model parameters for accuracy and time only, because difficulty level was not significant for any explanatory variable.

As we can see from Table 8.5, none of the interactions are significant. We do find that the presences of traceability links had a significant effect on the accuracy variable (p-value=0.009), which confirms the results from the Mann-Whitney test presented earlier.
Ability (high, low) and Type (academia, industry) are individually significant with respect to the time dependent variable. Based on the observations, we cannot reject the null hypothesis $H_3$ which states that Ability does not interact with the link tracing method to have an effect on time or accuracy.

There are some interesting observations to be made despite not finding any significant interaction effects. With respect to accuracy, there was a very small difference between Type (academia and industry) within the group that used traceability links via the tool. There was also a small difference between Ability (high and low) within the experimental group compared to the control group (presented with no links or tool). Both high and low ability subjects scored higher in the experimental group vs. the control group.

With respect to time, in the experimental group, students and industry subjects were also comparable (See Figure 8.2). Without traceability links, industry subjects took longer than students. This can be attributed to the fact that the subjects from industry took the study more seriously. If students did not get an answer within a certain time they just moved on to the next task. However, if they were given traceability links via the tool, they tend to find the task more solvable and spend time on it. Another observation is that subjects with high ability levels took longer than those with low ability levels, again due to the seriousness of the subjects wanting to perform well.
Figure 8.2: Interaction effects between the Type (academia vs. industry) and Group

Table 8.4: Resulting p-values of the Mann-Whitney test (one-tailed)

<table>
<thead>
<tr>
<th>Task</th>
<th>Accuracy</th>
<th>Speed</th>
<th>Difficulty</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>0.666</td>
<td>0.877</td>
<td>0.988</td>
</tr>
<tr>
<td>T2</td>
<td>0.002 *</td>
<td>0.089</td>
<td>0.143</td>
</tr>
<tr>
<td>T3</td>
<td>0.153</td>
<td>0.997</td>
<td>0.953</td>
</tr>
<tr>
<td>T4</td>
<td>0.0003 *</td>
<td>0.201</td>
<td>0.286</td>
</tr>
<tr>
<td>T5</td>
<td>0.007 *</td>
<td>0.016 *</td>
<td>0.014 *</td>
</tr>
<tr>
<td>All</td>
<td>0.0001 *</td>
<td>0.759</td>
<td>0.527</td>
</tr>
</tbody>
</table>
Table 8.5: Complex model for accuracy and speed dependent variables for all tasks

The first line indicates accuracy with speed parameters given in parentheses.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Standard Error</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>5.000</td>
<td>0.650</td>
<td>(&lt;0.0001) *</td>
</tr>
<tr>
<td></td>
<td>(31)</td>
<td>(3.801)</td>
<td>(&lt;0.0001) *</td>
</tr>
<tr>
<td>Type</td>
<td>-1.000</td>
<td>1.062</td>
<td>0.354</td>
</tr>
<tr>
<td>(Academia vs. industry)</td>
<td>(17.667)</td>
<td>(6.208)</td>
<td>(0.008) *</td>
</tr>
<tr>
<td>Group</td>
<td>3.333</td>
<td>1.187</td>
<td>0.009 *</td>
</tr>
<tr>
<td>(with links vs. without links)</td>
<td>(-10.333)</td>
<td>(6.940)</td>
<td>(0.148)</td>
</tr>
<tr>
<td>Ability</td>
<td>-0.067</td>
<td>0.881</td>
<td>0.940</td>
</tr>
<tr>
<td>(High vs. low)</td>
<td>(-11.433)</td>
<td>(5.147)</td>
<td>(0.035) *</td>
</tr>
<tr>
<td>Type * Group</td>
<td>-0.933</td>
<td>1.502</td>
<td>0.539</td>
</tr>
<tr>
<td></td>
<td>(17.933)</td>
<td>(8.779)</td>
<td>(0.051)</td>
</tr>
<tr>
<td>Group * Ability</td>
<td>0.400</td>
<td>1.245</td>
<td>0.750</td>
</tr>
<tr>
<td></td>
<td>(3.100)</td>
<td>(7.279)</td>
<td>(0.673)</td>
</tr>
</tbody>
</table>

8.6.2 Post-questionnaire Results

The post questionnaire asked the participants to rate their experience in the study. Both groups were given a common set of 7 questions. The questions were based on time needed, document understanding, clear objectives, clear tasks, diagram reading difficulty, task difficulty overall, and percentage of time spent reading the document. The group that was given traceability links answered four additional questions regarding percentage of time spent using the traceability links in the tool, tool usefulness, tool user friendliness and general comments on improving the way links are presented to them. On average, the
experimental group (given the links) felt that they had more time to complete the task than the control group without links. The requirements document and tasks were clear to all subjects. The Mann-Whitney test found a significant difference between the experimental and control groups when it came to percentage of time spent reading the document vs. using traceability links. The group that used the tool reported that the tool was very useful and easy to use. The percentage of time taken using the tool was 60% on average.

The industry participants were given an additional post-questionnaire to fill out and were also interviewed after the study. The questions were open ended to probe them with respect to their experience with maintenance tasks that benefit from traceability in their daily work as well as potential features of a traceability link presentation tool (discussed in the next section). Some of the important highlights are presented below.

• Four out of five developers acknowledged that they encountered the situations described in the software maintenance tasks in their daily work. This suggests that the tasks developed for the study were similar to real-world tasks developers encounter in industrial software.

• Two out of five developers reported that they like to see links directly between requirements and source code since design information is usually out of date as soon as a line of source code is added/modified.

• Four out of five developers reported that visual linking is very helpful. One of the four developers reported that providing visual linking for small size projects would not be that useful, but as the project size increases, its importance would also increase.
• Another developer reported that visual linking might help out in small projects, but when used with a large project, the view will be a large birds-nest picture which would be impossible to decipher. This shows that even though the developer appreciates traceability, he/she does not want to be overburdened with a hodgepodge of links.

• One of the developers reported that there is a need for required fields for the artifact in each repository that helps linking the repositories.

• Another developer stated that it would be nice if an item has tags (topics) and search functionality using tags. This was a very important observation since search trumps navigation.

8.7 Discussion

The results of the study indicate that using traceability links leads to significantly more correct answers for software maintenance tasks. With respect to speed, industry subjects who were given traceability links (via TraceLink) were faster than subjects that were not given any links (group with no TraceLink). In academia, however, subjects who did not have traceability links were faster. This can be due to the fact that industry subjects are more accustomed to using tools (given that the links were presented via the TraceLink tool) compared to students; in addition, more training was needed on the tool as pointed out by some subjects. With more training we could expect to see a significant difference in time needed to solve the tasks. This is left as a future exercise.

Results also show that the use of traceability links reduces the gap between high and low ability subjects as well as between academia and industry subjects. This is an important finding because given traceability links, we can have novices solve tasks faster and hopefully improve their skills that lead them to become experts in the future.
Note that the study conducted in this dissertation tries to determine the effect of the presence of traceability links in software maintenance tasks. The method of how the links are presented to the user is of lesser importance in the study. We chose to have the links presented in a prototype tool called TraceLink.

The observations presented in the rest of this section are useful to traceability tool developers in order to get more industry acceptance of the tools they develop to present traceability links to human analysts/developers. It also incorporates the different views from different stakeholders; another contribution of this dissertation. The following are interesting findings as stated by developers. One developer stated “I would only expect it to list the connections in a simple-to-use way making them easier to understand”. This implies that traceability needs to be effortless and almost second nature in order for it to be adopted. Another developer stated that “as an automated tester, all test cases are automatically generated. So I don’t need to trace as much as the questions in the study”. The automated tester believes that when creating the automated test suite, traceability is required, but after that tracing is not needed much. This could be the case in some instances.

One developer stated the following: “The developer is the one who is generating most of the information contained in these repositories. The design and requirements in many organizations are generally not well defined, and therefore do not really provide good information. At best, that information is helpful at the beginning of a development project, but are soon outdated because they are not kept up-to-date. A project manager or test engineer would use the tools in a different manner and seek different information than
a developer, but they are generally light users of the tools and generally are manipulating the data instead of generating the information”.

8.7.1 Developer Comments on TraceLink

This section presents what developers are interested to see in a traceability tool to help them in software maintenance tasks. Based on the input from developers in industry, some of the features that need to be supported by a traceability system are given below. These were all additional feature requests made by the subjects as they used TraceLink to solve the maintenance tasks. These observations are mentioned here with the goal of aiding future traceability tool developers.

- A simple search function similar to grep would be required. Note that TraceLink provided a way to find all links that were related to a given artifact however searching on specific keywords was not implemented.

- A traceability system should have more than one level of linking. For example, if artifact A is linked to artifact B and B is linked to artifact C, the tool shall support A to B and A to C linking. This is important during requirements gathering to test for contradicting requirements.

- A traceability system should support functionality to create different kinds of reports. For example, the tool should provide a table format. A report can be a table that lists builds, associated requirements, defects, test cases, etc.

- A traceability system should support artifact linkage in a graphical format in which filtering is a must to avoid being presented with an overwhelming number of links.
Traceability needs to be almost effortless and blend in with the software process in order for it to be adopted in industry.

All artifacts associated with a project should be linked to the project plan and provide easy navigation. When a developer gets assigned to work on a project, the developer sometimes reviews the project plan. The project plan mentions requirements, design, code, test, and other artifacts. It will be nice if the project plan has links that user can click on to get to the requirements, test cases, and other artifacts associated with the project. This type of traceability is not only important to developers, but also to PMs.

A traceability system should provide views that are specific to stakeholders. This is an important feature that tends to be overlooked. For example, a developer who needs to visit the links that are of interest to him/her doesn’t need to dive into an overall view that has links between all artifacts that is most useful to the project manager. Too many links impact developers’ time and accuracy because the view can get complex and errors may occur.

The above mentioned features are useful to traceability tool developers in order to have their tools more readily adopted by industry. The lesser the effort required by a stakeholder to use a traceability system, the easier it is to adopt. Note that TraceLink was not designed to provide a fully working traceability system, rather an electronic way of presenting subjects with traceability links.

8.8 Threats to Validity

The four main threats to validity are internal, external, construct and conclusion validity. With respect to internal validity, the study was between-subjects and did not suffer from learning effects between groups. Within each group, each task was from a
different category and did not overlap so there were no learning effects involved there either. The study was done in a classroom/lab setting or an industrial setting with dedicated time allotted to it, so as to minimize any other external factors that might have an effect on the results. A tutorial was given to all subjects prior to the study to make sure all subjects were at the same knowledge level we expected. None of the subjects were familiar with the room management system prior to the study.

With respect to construct validity, we choose to measure accuracy, speed, and difficulty level. Since our study goal was to assess the effect of traceability links on software maintenance tasks, we measure performance on the tasks via accuracy and speed. Better performance indicates higher accuracy and lower speed i.e., time to complete task. The difficulty level gave us some indication of how difficult the tasks were for them to solve. This combined with accuracy and speed was used in the analysis. The students were not told and did not know the hypotheses in advance. Some of the PMs were developers before and their input may be affected by their previous role. It would be nice to have the same number of industry experts as students but it was hard to find many developers volunteering to invest their time for this activity.

External validity deals with generalizing the results of the study. The study and survey is conducted at one industrial firm and on one system. The survey does reflect a single firm view. However, it is a step towards identifying links interesting to different stakeholders. It is quite challenging to find a bunch of firms willing to participate and invest time in surveys and studies such as the one we conducted. We welcome traceability researchers to build on top of the results in this dissertation and conduct such surveys with other industrial firms to advance this body of knowledge. Replicating this study with a
larger sample from different organizations in different domains and with different systems and tasks can help reduce this threat. We leave this as future work. Even though the subject system used is not a large open source system, the questions asked with respect to a feature that needs to be added/changed is comparable to similar issues in open source issue tracking systems. We derived our tasks after browsing the Bugzilla database for various projects. The developers also agreed in the post questionnaire that the tasks were realistic. Also, there are not many open source systems that have traceability matrices available for use which makes the choice of a subject system very narrow. Most traceability dissertations use the same academic datasets such as iTrust (Meneely et al., 2012) to validate their link retrieval/evolution algorithms. In this regard, we can consider the room management system dataset as yet another dataset with developer provided traceability links that could be used by traceability researchers in the future. Our results from the study are only generalizable to maintenance tasks that are similar (i.e., finding artifacts that need to be changed if a feature is added, changed, or behaves incorrectly). Our results from the industrial survey are generalizable only to industrial firms that follow similar process models. We do not claim that these results would apply to all settings. Further empirical studies are needed to validate this generalization.

With respect to conclusion validity, due to low sample size and non-normality of data we use the Mann-Whitney test for hypotheses testing.
9 CONCLUSIONS AND FUTURE WORK

Section 9.1 presents the conclusion of the dissertation. Section 9.2 presents future work, and section 9.3 is the contribution.

9.1 Conclusions

This dissertation addressed the issues: 1) managing and analyzing project schedule, tasks and resources, 2) managing and analyzing software defects, and 3) project documents traceability. For the first issue, the dissertation developed an approach to present project tasks and resources information in 3D landscape visualization views. The visualization views are created based on a model that captures the information and the relationships between the project and its tasks and resources. The dissertation also presents a study to assess the effect of using 3D landscape visualization views to present and analyze project tasks and resources information. A prototype tool namely 3DProjView was developed for the experiment that tests the hypotheses in the study. Forty-two subjects from industry and academia participated in the study. Results indicate on average, a significant increase in task accuracy (40%) and decrease in time (39%) when 3D landscape visualization was used to solve the tasks. Using 3D landscape visualizations (via 3DProjView) also reduced the gap between industry users and beginners. This implies that using 3D landscape visualization provides more gains for novices and makes them more productive making them comparable to industry users.

For managing and analyzing software defects issue, the dissertation also provided an approach to present project defects status information using 3D visualizations. The visualizations are created based on a model that captures the information and the relationships between the defects information. The research also presents a study to
assess the effect of using 3D visualizations to present and analyze project defects information. Forty subjects from industry and academia participated in the study. Results indicate a significant increase in task accuracy (32%) and decrease in time (35%) when 3D landscape visualization was used to solve the tasks. The data shows that neither experience nor background had a significant impact on the performance of participants who utilized 3D visualization. This suggests that using 3D visualization helped users with different level of backgrounds and experiences to achieve similar outcome. In other words, 3D visualization helped compensate the lack of experience or expertise in these tasks.

For project artifacts traceability, the dissertation developed a traceability system that links project artifacts and provides different views for stakeholders based on their role. A study was conducted to assess the effect of the presence of traceability links during software maintenance tasks. A prototype tool namely TraceLink was developed to test the hypotheses in the study. Twenty-eight subjects from industry and academia participated in the study. Results indicate a significant increase in task accuracy when traceability links were used to solve the tasks. Subjects using traceability links were 86.06% more accurate. Using traceability links (via TraceLink) also reduced the gap between high and low ability subjects as well as between academic and industry subjects. This implies that using a traceability system provides more gains for low ability subjects making them more productive. Difficult tasks benefitted more from traceability links than the ones that were easier to solve. This is all the more evidence to adopt traceability practices in organizations. In addition to the traceability system and the study, we conducted an industrial survey to help us develop the traceability system. The survey
determined the kinds of traceability links most needed by industry stakeholders for collaboration. Results from the survey include a traceability meta-model highlighting the strong relationship between the types of links needed and the stakeholder’s role of developer, tester, and project manager. This is another important contribution of the dissertation. Several real-life scenarios were also discussed due to the lack of traceability at the industrial firm surveyed.

Finally, the following lists limitations to the approach developed by the research:

- The 3D visualization approach currently allows displaying maximum of 3 metrics. For example, the approach maps metric to the three dimensions of the box.
- The 3D visualization approach currently supports cylinder and box metaphors. Other metaphors, for example, such as sphere, triangle, etc. are not supported.
- The approach currently provides analysis for these metrics for a task: percent of completion, start date, and finish date. Other metrics, for example, cost, work hours, etc. are not supported. With respect to software defects, the research provides analysis for the defect status (submitted, under-review, rejected, assigned, resolved, and closed). Other metrics, for example, estimated completion date, submitted date, etc. are not supported.

9.2 Future Work

For the traceability of software artifacts, future work will be to study traceability systems and interested links to different stakeholders at other industrial firms, as well as conduct a study that evaluates the benefits of a traceability system that provide views (defined in chapter 5 Figure 5.2), specific to stakeholders’ roles. Furthermore, a reevaluation of the H2 hypothesis (in chapter 6), needs to occur with more subjects and
more in-depth training on the traceability tool. These replications will strengthen the findings presented by our work.

For project management, future work will be to apply the approach to present and analyze more project management information and different areas in project management. For example, use the approach to display and analyze information such as planned start date, planned finish date, actual start date, actual duration, actual finish date, remaining duration, actual cost, etc. (See MS project for more information related to project tasks and resources) Furthermore, use the approach to visualize information related to team communication and quality. Other future work will be to incorporate the visualizations developed by this dissertation as a MS Project plug-in. Furthermore, add the following functionalities to the 3DProjView tool:

- Adding a clickable arrow to allow rotation of the model, possibly 90 degrees at a time. That would help someone pick it up and not have to memorize key strokes in order to rotate the model.
- Add filtering capabilities. Use colors to highlight some of the filtered results.
- Provide short-cuts keys.

For project software defects, future work can be to apply the approach to present and analyze more project defects information. For example, present the time it took to resolve a defect, who is assigned to a particular defect, etc. See Bugzilla https://www.bugzilla.org/about/ for a list of information for defects. Currently, we use cylinder to present this information; future work will utilize boxes to represent more information. Finally, future work will be incorporating the visualizations developed by the dissertation as a plug-in to other defect tracking tools.
9.3 Contributions

This dissertation contributes to project management, defect management, and project documents traceability. For project management, the research developed an approach to manage and analyze project schedule, tasks, and resources. It conducted a study that showed 40% more accuracy and 39% less time analyzing project tasks and resources using our approach versus Gantt chart and tables visualizations. With respect to project defects, the research also developed an approach to manage and analyze software defects. The approach improved the quality of software systems by resolving defects. A study was conducted by the research that showed 32% more accuracy and 35% less time analyzing software defects using the research approach versus 2D Bar chart and tables. With respect to documents traceability, the dissertation provides a traceability system that links project documents together. It conducted an industrial survey that identified different traceability views used by stakeholders based on their roles. These views allow a stakeholder to view the links that are interested to him or her instead of providing all links in the system to view. Furthermore, the dissertation conducted a study that showed 86% more accuracy when using traceability linking system to solve maintenance tasks versus manual traceability.
REFERENCES


Atlassian – JIRA Documentation. [https://www.atlassian.com/software/jira](https://www.atlassian.com/software/jira)


Bugzilla. [https://www.bugzilla.org/about/](https://www.bugzilla.org/about/)


http://www.steelray.com/ProjectScheduleVisualization.pdf


Mantis. [https://www.mantisbt.org/](https://www.mantisbt.org/)


APPENDIX A – ARTIFACTS TRACEABILITY STUDY ARTIFACTS

A.1 Group 1 Visualizations

Figure A.1: A snapshot of TraceLink
Table A.1: Stakeholders’ interest in types of information in each software artifact

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Dev</th>
<th>Tester</th>
<th>PMI</th>
<th>PBM</th>
<th>BA</th>
<th>Build</th>
<th>Dev</th>
<th>Tester</th>
<th>PMI</th>
<th>PBM</th>
<th>BA</th>
<th>Detect</th>
<th>Dev</th>
<th>Tester</th>
<th>PMI</th>
<th>PBM</th>
<th>BA</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID/Version Number</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>ID/Version Number</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>ID/Version Number</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Location</td>
<td>X</td>
<td>X</td>
<td>Description</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Author</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Date</td>
<td>X</td>
<td>X</td>
<td>Creation date</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Date or version number</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Reporte</td>
<td>X</td>
<td>X</td>
<td>Reported by</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reference (Ref) to Test</td>
<td>X</td>
<td>X</td>
<td>Ref to Defect</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Estimated hours</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ref to Design</td>
<td>X</td>
<td>X</td>
<td>Ref to Test</td>
<td>X</td>
<td>X</td>
<td>Planned release</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ref to PM</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Test Case</td>
<td>Dev</td>
<td>Tester</td>
<td>PMI</td>
<td>PBM</td>
<td>BA</td>
<td>Life Cycle Feed-in</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ref to Release</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Fields</td>
<td>Actual hours</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Design</td>
<td>Dev</td>
<td>Tester</td>
<td>PMI</td>
<td>PBM</td>
<td>BA</td>
<td>ID/Version Number</td>
<td>X</td>
<td>Status</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UML Diagram</td>
<td>Fields</td>
<td>Title</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Attachments - logs/screeshots</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use case ID/Version no., Name, Description, Author</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Name</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class Diagram ID/Version no., Name, Description, Author, Date, Status</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Test Phase - Ex. Unit testing, system testing, etc</td>
<td>X</td>
<td>X</td>
<td>Ref to Test</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sequence Diagram ID/Version no., Name, Description, Author, Date, Status</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Status</td>
<td>X</td>
<td>Ref to Build</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Collaboration Diagram</td>
<td>ID/Version no., Name, Description, Author, Date, Status</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Last attempt</td>
<td>X</td>
<td>Ref to code</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>State Diagram ID/Version no., Name, Description, Author, Date, Status</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Expected duration</td>
<td>X</td>
<td>Code Inspection</td>
<td>Dev</td>
<td>Tester</td>
<td>PMI</td>
<td>PBM</td>
<td>BA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Activity Diagram ID/Version no., Name, Description, Author, Date, Status</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Actual duration</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ref to code</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Tester(s)</td>
<td>X</td>
<td>ID/Version Number</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ref to requirements</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Expected results</td>
<td>X</td>
<td>Project name</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ref to requirements</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Results Obtained</td>
<td>X</td>
<td>X</td>
<td>Basic Information: title, descriptor, review deadline</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Code</td>
<td>Dev</td>
<td>Tester</td>
<td>PMI</td>
<td>PBM</td>
<td>BA</td>
<td>Test procedure</td>
<td>X</td>
<td>Participants</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Package [e.g. DLL]</td>
<td>Component or file</td>
<td>Fields</td>
<td>Material</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID/Version Number</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Ref Requirement</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>X</td>
<td>Ref to Build</td>
<td>X</td>
<td>X</td>
<td>Status</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Date</td>
<td>X</td>
<td>Ref to Defect</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Owner</td>
<td>X</td>
<td>Project Management (PM)</td>
<td>Dev</td>
<td>Tester</td>
<td>PMI</td>
<td>PBM</td>
<td>BA</td>
<td>Ref to Code</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>X</td>
<td>Fields</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ref to Design</td>
<td>X</td>
<td>X</td>
<td>Name</td>
<td>X</td>
<td>X</td>
<td>ID/Version Number</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ref to Code inspection</td>
<td>X</td>
<td>X</td>
<td>Author</td>
<td>X</td>
<td>X</td>
<td>Location</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ref to Build</td>
<td>X</td>
<td>X</td>
<td>Date</td>
<td>X</td>
<td>X</td>
<td>Date</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ref to Defect</td>
<td>X</td>
<td>X</td>
<td>Description</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Release Notes associated with the</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ref to Requirement</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Ref to PM</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ref to Release</td>
<td>X</td>
<td>X</td>
<td>Ref to Requirements</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Legend:
- X: Requested
- : Not requested
- : Not applicable
- BA: Business Analyst
- PMI: Project Manager
- PBM: Project Business Manager
A.2 Group 2 Visualizations

2. Requirements

2.1 Functional Requirements

2.1.1 Using the Web interface, there will be three levels of users: a user, a level 1 user, and an administrator. Level 1 users and administrators will have logins.

3. Design

3.1 Use Cases

3.1.1 The Web User’s Interaction with RMA System.
5. Builds

Two builds are used for the RMA System:

6.1 Build – This build implements requirements 2.1.1, 2.1.2, 2.1.3, and 2.1.4.
6.2 Build – This build implements requirements 2.1.5, and 2.1.6.

6. Code Inspection

Two code inspections are used:

7.1 Code Inspection – This code inspection for version 5.1 of source code.
7.2 Code Inspection – This code inspection for version 5.2 of source code.

7. Testing

Two builds 6.1 and 6.2 exist for the RMA system (see section 6). In the Table below, for build 6.2, test cases 8.2.1 through 8.2.46 are regression test cases; the rest of the test cases are new.

<table>
<thead>
<tr>
<th>Test Cases for Build 6.1</th>
<th>Test Cases for Build 6.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test No</td>
<td>Title – Description</td>
</tr>
<tr>
<td>8.1.1 Login Test Cases</td>
<td>8.2.1 Login as a user:</td>
</tr>
<tr>
<td>8.1.2 Login as a level-1 user using valid user name and invalid password.</td>
<td>8.2.2 Login as a level-1 user using valid user name and invalid password.</td>
</tr>
<tr>
<td>8.1.3 Login as a level-1 user using valid user name and valid password.</td>
<td>8.2.3 Login as a level-1 user using valid user name and valid password.</td>
</tr>
<tr>
<td>8.1.4 Login as a level-1 user using invalid user name and invalid password.</td>
<td>8.2.4 Login as a level-1 user using invalid user name and invalid password.</td>
</tr>
</tbody>
</table>

Figure A.2: Room management application requirements, design, code, build, and testing information
APPENDIX B – PROJECT SCHEDULE ATRIFACTS

B.1 Group 1 Visualizations

Figure B.1: Project tasks

Figure B.2: Project resources
Figure B.3: Project costs

Figure B.4: Project history information
B.2 Group 2 Visualizations

Figure B.5: Project tasks

Figure B.6: Project resources
Figure B.7: Project costs

Figure B.8: Project history information
APPENDIX C – SOFTWARE DEFECTS ARTIFACTS

C.1 Group 1 Visualizations

Figure C.1: Project-A defects

Figure C.2: Project-B defects
Figure C.3: Project-C defects

Figure C.4: Project-D defects
Figure C.5: Projects A & B defects

Figure C.6: Projects C & D defects
C.2 Group 2 Visualizations

(a) Table View

(b) Bar View
Figure C.7: Project-A defects

(c) Bar View

(a) Table View
Figure C.8: Project-B defects
(a) Table View

(b) Bar View
Figure C.9: Project-C defects

(c) Bar View

(a) Table View
Figure C.10: Project-D defects