Interactive Visual Support for Understanding the Structural and Behavioural Aspects of Embedded Systems

Vom Fachbereich Informatik
der Technischen Universität Kaiserslautern
zur Erlangung des akademischen Grades
Doktor der Ingenieurswissenschaften (Dr.-Ing.)
genehmigte Dissertation
von

Ragaad AlTarawneh

Datum der wissenschaftlichen Aussprache: 24. Juli 2015

D386

Declaration:

I hereby declare that my Ph.D. thesis, entitled ***Interactive Visual Support for Understanding the Structural and Behavioral Aspects of Embedded Systems***, is an independent piece of research compiled by me under the supervision of Prof. Dr. Achim Ebert from the Computer Graphics and HCI Group of the Department of Computer Science at the University of Kaiserslautern, Germany and Prof. Dr.-Ing. Peter Liggesmeyer from the Software Engineering: Dependability Group of Department of Computer Science at the University of Kaiserslautern, Germany.

This thesis has not been previously submitted for the award of any degree, diploma, associateship, fellowship, or its equivalent to any other university or institution. Throughout this thesis, sources of information have been credited to the best of my knowledge and abilities.

Ragaad AlTarawneh,
Kaiserslautern,

Dedication

This work is dedicated to my late father
Mohammed Irsheid Al-Tarawneh

Acknowldgment

First of all, I would like to express my full appreciation and gratefulness to my supervisor Prof. Dr. Achim Ebert for his guidance and directions during my PhD study. I am thankful to him for his flexibility and his tolerance during the working process. I appreciate the freedom he gave and the contentious support during my staying in Germany. Then, I would like to express my gratefulness to Prof. Dr.-Ing. Peter Liggesmeyer, as he hosted me almost for one year in his group. This made it easy for me in contacting directly with the researchers in his Software Engineering: Dependability group, which helped me a lot in understanding the research questions and the problem statements regarding my targeted application domain. I am also grateful for his directions that corrected my research path throughout my PhD journey. I am also thankful to him for inviting me in his group monthly meetings, for keeping track on my progress, and for considering me as a member of his research group.

Further, I would like to express my continuous thanking and appreciation to Prof. Dr. Hans Hagen for his generosity to accept me as a PhD student in his group. I am grateful for all the efforts he made to create the chance and the financial support that helped me for my staying in Germany and to proceed my PhD study.

I also would like to thank my colleagues Jens Bauer and Dr.-Ing. Patric Keller for the technical and scientific support during working in the Vier4ES project in its second phase. Moreover, I give full credit attached with many special thanks to Dr. Shah Rukh Humayoun for his support during conducting the evaluation studies and papers preparation times. I appreciate the times he corrected my language mistakes and his careful reviewing to almost each single publication I had during my PhD journey, including this thesis.

This PhD journey was full of challenges that hinder my working progress from time to time. I might never come over from these challenges without the significant support from the two persons, whom I consider as my family in Germany. Roger Daneker and Mady Gruys (the names' order is irrelevant). Thank you both for being there in my bad times, thank you for listening to my sad stories, thank you for standing next to me during my weakness times, and thank you for sharing with me the happiness moments. I also would like to thank my friends

(Lama Osman and Nebal Nayef) for their support and encouragement during my study time.

Moreover, I would like thank to all my colleagues in the Computer Graphics and HCI group for being collaborative and supportive during my stay here. Special thanks go to Dr.-Ing Max Langbein for the long time he spent with me to formally define some ideas and concepts I used in this work. Many thanks also go to all people from the Software Engineering: Dependability group, who supported me during the evaluation time and they were never late in giving me the feedback whenever I needed. I would also like to thank the people from the Robotics Research Laboratory headed by Prof. Dr. Karsten Berns, as they provided the required data and gave the required feedback whenever I asked for it. I would also like to mention the nice collaboration with the people from the Software Engineering: Processes and Measurement group, headed by Prof. Dieter Rombach. I would like to mention each one of you by name but the space would not be enough to do so.

This whole thing would never saw the light without the support, the love, and the care of my mother, my brothers, my sisters, my nieces, and my nephews. I really feel very grateful for the chance, and the trust you gave me to finish my study abroad. I have gone through this journey to make you all proud of me and to draw a nice big smile on your faces.

Finally, this whole PhD work would have never happened without the courage of my late father. I still remember the moment you agreed to give me the chance to study abroad, taking on your side the community-pressure, and allowing me to fulfill my dreams. This has been the main motivation for me to work hard and to keep continuing my tough journey. I really wanted to show you the result, but your time in this life just passed before I finished. This work is fully dedicated to your pure soul.

Abstract

Information Visualization (InfoVis) and Human-Computer Interaction (HCI) have strong ties with each other. Visualization supports the human cognitive system by providing interactive and meaningful images of the underlying data. On the other side, the HCI domain cares about the usability of the designed visualization from the human perspectives. Thus, designing a visualization system requires considering many factors in order to achieve the desired functionality and the system usability. Achieving these goals will help these people in understanding the inside behavior of complex data sets in less time.

Graphs are widely used data structures to represent the relations between the data elements in complex applications. Due to the diversity of this data type, graphs have been applied in numerous information visualization applications (e.g., state transition diagrams, social networks, etc.). Therefore, many graph layout algorithms have been proposed in the literature to help in visualizing this rich data type. Some of these algorithms are used to visualize large graphs, while others handle the medium sized graphs. Regardless of the graph size, the resulting layout should be understandable from the users' perspective and at the same time it should fulfill a list of aesthetic criteria to increase the representation readability. Respecting these two principles leads to produce a resulting graph visualization that helps the users in understanding and exploring the complex behavior of critical systems.

In this thesis, we utilize the graph visualization techniques in modeling the structural and behavioral aspects of embedded systems. Furthermore, we focus on evaluating the resulting representations from the users' perspectives.

The core contribution of this thesis is a framework, called **ESSAVis** (**E**mbedded **S**ystems **S**afety **A**spect **Vis**ualizer). This framework visualizes not only some of the safety aspects (e.g. CFT models) of embedded systems, but also helps the engineers and experts in analyzing the system safety critical situations. For this, the framework provides a 2D*plus*3D environment in which the 2D represents the graph representation of the abstract data about the safety aspects of the underlying embedded system while the 3D represents the underlying system 3D model. Both views are integrated smoothly together in the 3D world fashion. In order to check the effectiveness and feasibility of the framework and its sub-components, we conducted many studies with real end users as well as with general users. Results of the main study that targeted the overall ESSAVis framework show high acceptance ratio and higher accuracy with better performance using the provided visual support of the framework.

The ESSAVis framework has been designed to be compatible with different 3D technologies. This enabled us to use the 3D stereoscopic depth of such technologies to encode nodes attributes in node-link diagrams. In this regard, we conducted an evaluation study to measure the usability of the stereoscopic depth cue approach, called the stereoscopic highlighting technique, against other selected visual cues (i.e., color, shape, and sizes). Based on the results, the thesis proposes the **Reflection Layer** extension to the stereoscopic highlighting technique, which was also evaluated from the users' perspectives. Additionally, we present a new technique, called **ExpanD** (**Expan**d in **D**epth), that utilizes the depth cue to show the structural relations between different levels of details in node-link diagrams. Results of this part opens a promising direction of the research in which visualization designers can get benefits from the richness of the 3D technologies in visualizing abstract data in the information visualization domain.

Finally, this thesis proposes the application of the ESSAVis framework as a visual tool in the educational training process of engineers for understanding the complex concepts. In this regard, we conducted an evaluation study with computer engineering students in which we used the visual representations produced by ESSAVis to teach the principle of the fault detection and the failure scenarios in embedded systems. Our work opens the directions to investigate many challenges about the design of visualization for educational purposes.

Zusammenfassung

Informationsvisualisierung (InfoVis) und Human-Computer-Interaction (HCI) sind eng verbunden. Visualisierung unterstützt die menschliche Wahrnehmung durch eine interaktive und sinnhaltige Verbildlichung der Rohdaten. Der Bereich der HCI hingegen befasst sich mit der Benutzbarkeit des Visualisierungsprozesses aus Sicht des Menschen. Der Designprozess für ein Visualisierungssystem erfordert also die Berücksichtigung vieler Faktoren, um die gewünschte Funktionalität und Bedienbarkeit zu erreichen. Wenn all diese Faktoren berücksichtigt wurden, hilft dies dem Menschen, komplexe Daten schneller zu verstehen.

Graphen sind vielgenutzte Datenstrukturen, die dabei helfen, die Beziehungen zwischen den Datenelementen komplexer Anwendungen zu verstehen. Begründet durch ihre Vielseitigkeit werden Graphen in vielen Informationsvisualisierungsanwendungen eingesetzt (Bsp: Zustandsdiagramme, soziale Netzwerke, usw.). In der Literatur gibt es viele Layoutalgorithmen, die dabei helfen, diesen wichtigen Datentyp zu visualisieren. Manche dieser Algorithmen sind auf die Visualisierung großer Graphen und andere auf die Visualisierung mittelgroßer Graphen spezialisiert. Unabhängig von der Größe der Graphen soll das resultierende Layout für den Benutzer verständlich sein, und gleichzeitig eine Liste von ästhetischen Kriterien erfüllen, um die Lesbarkeit der Darstellung zu erhöhen. Die Beachtung dieser zwei Kriterien führt zu einer Graph-Visualisierung, die dem Benutzer dabei hilft, komplexes Verhalten kritischer Systeme zu verstehen und diese zu erkunden.

In dieser Arbeit werden Graphvisualisierungstechniken benutzt, um Struktur und Verhaltensaspekte von eingebetteten Systemen zu modellieren. Außerdem befassen wir uns mit der Evaluierung der erzeugten

Visualisierungen aus Benutzersicht.
Der Hauptbeitrag dieser Arbeit ist das Framework **ESSAVis** (**E**mbedded **S**ystems **S**afety **A**spect **Vis**ualizer). Dieses Framework visualisiert nicht nur Sicherheitsaspekte, sondern hilft Sicherheitsingenieuren und Experten, die Systemsicherheit in kritischen Situationen zu analysieren. Hierfür bietet das Framework eine 2D*plus*3D – Umgebung. Dabei erfolgt die Graphendarstellung der abstrakten Daten der Sicherheitsaspekte des zugrundeliegenden Systems im 2D-Bereich. Im 3D-Bereich wird das 3D-Modell angezeigt. Die beiden Bereiche synchronisieren sich gegenseitig.

Um die Effektivität und Anwendbarkeit des Frameworks und seiner Komponenten zu testen, führten wir viele Studien sowohl mit echten Endbenutzern als auch mit Fachfremden durch. Das Ergebnis dieser Studien mit dem gesamten ESSAVis-Framework zeigt bei den Benutzern eine hohe Akzeptanzrate und eine Erhöhung von Genauigkeit und Performanz bei Benutzung der visuellen Unterstützung durch das Framework. Das ESSAVis-Framework wurde so konzipiert, dass es mit verschiedenen 3D-Technologien kompatibel ist. Das erlaubt uns, den stereoskopisch erzeugten Tiefeneindruck solcher Technologien zu benutzen, um Knotenattribute in Node-Link-Diagrammen zu kodieren. Wir führten mehrere Studien zur Anwendbarkeit des Tiefeneindrucks mit der Stereoscopic-Highlighting-Technik hierfür im Verhältnis zu anderen visuellen Kodierungen (z.B. mit Farbe, Form, Größe) durch. Darauf basierend wird in dieser Arbeit die **Reflection Layer** – Erweiterung der Stereoscopic-Highlighting-Technik vorgeschlagen, welche auch eine Benutzerevaluierung durchlief. Zusätzlich wurde die Technik **ExpanD** (**Expan**d in **D**epth) eingeführt, welche die räumliche Tiefe benutzt, um Strukturbeziehungen zwischen verschiedenen Detailstufen der Node-Link-Diagramme zu kodieren. Die Ergebnisse dieses Teils der Arbeit öffnen eine vielversprechende neue Forschungsrichtung im Informationsvisualisierungsbereich, in der Ersteller von Visualisierern für abstrakte Daten Nutzen aus der Reichhaltigkeit der heutigen 3D-Technologien ziehen.

Schließlich wird in dieser Arbeit die Anwendung des ESSAVis–Framework als Visualisierungswerkzeug in der Ingenieursausbildung vorgeschlagen, um komplexe Sachverhalte zu verstehen. Hierzu führten wir eine Studie mit Informatik Studenten durch, in der wir die visuellen Repräsentationen in ESSAVis benutzten, um Versagensszenarien in eingebetteten Systemen darzustellen und die Prinzipien der Fehlererkennung zu lehren. Damit eröffnet diese Arbeit eine neue Richtung im Gebiet der Visualisierung für die Lehre.

Contents

1 Introduction 1
1.1 Problem Statement and Motivation 1
1.2 Research Contributions 5
1.3 List of Publications 10
1.4 Thesis Structure 14

2 State of the Art 17
2.1 Safety Information Background 18
2.2 Visualizing Safety Information 20
2.3 Graph Visualization Techniques 21
2.3.1 Terminologies 22
2.3.2 Graph Layout Algorithms 25
2.4 Nodes and Edges Clustering 33
2.5 Interaction Techniques 37
2.6 3D Layout Examples 39
2.7 Stereoscopic Depth in Information Visualization 41
2.8 3D Modeling 43

3 The ESSAVis Framework Overview 47
3.1 The Designing Process 48
3.2 The ESSAVis Framework 52
3.2.1 The Cluster Expander Unit 54
3.2.2 The Layout Manager Unit 55
3.2.3 The 3D Interactive Explorer Unit 55
3.2.4 The Scene Manager Unit 56
3.2.5 The Textual Information Unit 57

4 Navigating Compound Graphs 59
4.1 Motivation Behind the CluE Algorithm 61
4.2 The CluE Algorithm 64
4.3 Results 66
4.4 Expanding Options 69
4.5 The Evaluation Study 71
4.6 Summary 72

5 A Generic Layering Framework for Visualizing Compound Graphs Interactively 75
5.1 Background and Problem Statement 78
5.2 The LayMan Framework 80
5.2.1 The Preprocessing Phase 81
5.2.2 The Layout Phase 85
5.3 Application of LayMan in Encoding CFT Models 86
5.3.1 Mapping the Safety Scenario Information from the CFT model to the Graph Representation 88
5.4 The Evaluation Study 91
5.4.1 Testing the LayMan Framework over Different Layout Algorithms 92
5.4.2 LayMan Scalability 95
5.5 Summary 97

6 Adding the Stereoscopic Depth in Graph Visualization . . . 101
6.1 Evaluating Stereoscopic Depth in Node-Link Diagrams . 106
6.1.1 Study Design and Execution 106
6.1.2 Results and Discussion 112
6.2 The Reflection Layer Extension 117
6.3 Evaluating the Reflection Layer Extension 119
6.3.1 Study Design and Execution 119
6.3.2 Results and Discussion 123
6.4 Stereoscopic Filtering Technique 125
6.4.1 Topological Feature-Based Layout 126
6.4.2 The Filtering Approach 126
6.4.3 Implementation 128
6.4.4 Results 129
6.5 ExpanD: A Stereoscopic Depth Expanding Technique for Compound Graphs 131
6.5.1 The ExpanD Approach 134

6.5.2 The Evaluation Study 137
6.5.3 Results and Discussion 142
6.6 Summary 151

7 Visualizing and Integrating the System 3D Model with the Abstract Representations of CFT Models 153
7.1 The RAVON Robot 154
7.2 The 3DintEx Tool 156
7.2.1 The VRML Parser 158
7.2.2 3DintEx Interaction Options 159
7.3 The Scene Manager (SceneMan) Tool 162
7.3.1 SceneMan Views Alignment Mechanism 164
7.4 ESSAVis Interaction Options 169
7.5 ESSAVis in A Large Visual Collaborative Environment . 174
7.5.1 The Collaborative Environment Mode 175
7.5.2 System Collaboration Aspects 176
7.6 Summary 179

8 The ESSAVis Framework Evaluation Study 181
8.1 Introduction to ESSaRel 182
8.2 Experiment Design and Execution 184
8.2.1 Goals and Metrics 184
8.2.2 Hypotheses 185
8.2.3 Execution 186
8.2.4 Participants 186
8.2.5 Study Procedure 187
8.2.6 Study Scenario and Tasks Description 189
8.2.7 Analysis Procedures 190
8.3 Results and Discussion 190
8.4 Conclusions and Findings 201

9 ESSAVis Usage as A Visual Training Tool in Education . . . 209
9.1 Related Work 211
9.2 The Static Visualization Perspective 213
9.3 The Dynamic Visualization Perspective 215
9.4 The Visualization Pipeline 218
9.5 The Challenges 220
9.6 The Evaluation Study 222
9.6.1 The Test Settings 223

9.6.2 The Static Visualization Perspective Test 223
9.6.3 The Dynamic Visualization Perspective Test 225
9.6.4 Discussions 227
9.7 Summary 228

10 Conclusions and Future Directions 231
10.1 Conclusions 231
10.2 Future Directions 235

References 239

1 Introduction

1.1 Problem Statement and Motivation

Embedded systems are widely used in our daily life, a few examples of these systems are control systems in cars, airplanes, railroad crossings, and washing machines in our houses (see Figure 1.1). Embedded systems are classified as electronic devices that incorporate a computer system in their implementations [108, 114, 160]. Generally, they are not centralized in one component but are distributed among a set of components, which represent the system parts. In fact, these systems are normally composed of two types of components, software and hardware. Both types of components collaborate with each other through a set of interfaces, and this all leads to highly complex structures [108, 114, 160].

Due to the frequent usage of these systems in our day-to-day activities, safety and reliability aspects are essential and critical from end users' perspectives. These aspects guarantee a working system without any unexpected errors and risks if they are correctly defined and implemented. Consequently, maintaining these systems for a safe situation requires an intensive study of those critical situations that might bring the underlying system to undesired states [93].

The process of detecting possible failure mechanisms in these systems is normally performed by *safety experts*, who are responsible for finding out the failure mechanisms. For example, they use Component Fault Tree (CFT) [93, 92] technique to trace all the possible reasons for each specific *top event*, where each top event depicts a specific failure scenario during the system life. Their objective is to find all possible safety critical components that could trigger the underlying top event.

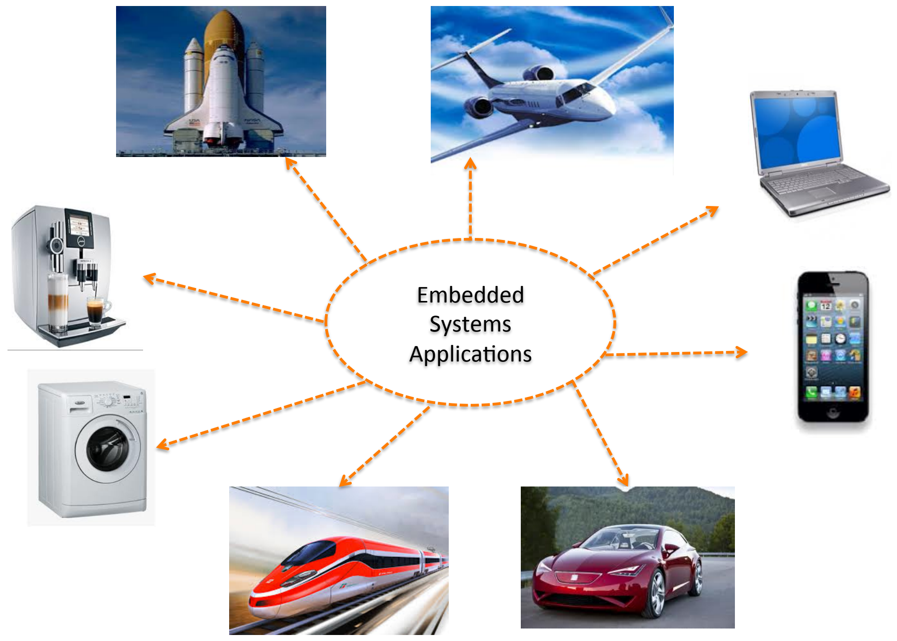


Fig. 1.1: Few examples of different embedded systems that are used in our daily life activities.

Safety experts design the corresponding CFT model based on all the possible failure relations between the system components related to the specified top event.

After building the CFT model, safety experts start analyzing the safety scenario for finding all the safety critical components in order to maintain them for keeping the system in a safe mode. The result of this analysis helps the *system engineers*, who are responsible for maintaining all the system components, for fixing the overall system in order to make it safe to be used by end users.

This iterative interaction process between safety experts and system engineers supports the maintenance process of these embedded systems. Involvement of different engineers and experts having different backgrounds in the analysis process, means a difference not only in the used terms to describe the system status but also a difference in seeing the system from their own perspective. Also, the interaction and collaboration between the two groups (i.e., the safety experts and the system engineers) is affected by differences in the input data format used by each group. Consequently, the overall speed of this maintenance process is impacted due to this gap, as we observed from several meetings we conducted with the end-users. The same observation was also discussed in the work of Ferrell T. et al, [64].

Information visualization can also play an important role in bridging the above-described gap between the two groups and in speeding up the maintenance process in embedded systems, as it eases the steps of finding important information from both the system and the viewer's perspectives. Also, it helps in reducing the errors that are made by humans while searching for the relevant information. For example, Weber in [158] mentioned that 30% of people's time is wasted on searching the important information. Further, the visualization helps in interacting with the data and automates the steps of the information-extracting process, which results in speeding up the overall maintenance process.

Information Visualization (InfoVis) has been used for several purposes, such as: to represent the structural and the behavioral aspects of complex data sets [98, 82], to synchronize multiple perspectives about the data using different interaction styles [131], to support intuitive interaction techniques in order to facilitate the information extraction process [131, 82], or to provide aesthetics representations of the data [125]. Generally, visualizations are generated by creating images of the underlying data sets from the target perspectives. Consecutively, viewers see the structural and the behavioral aspects of these data sets in much clearer and more intuitive ways. A critical task in information visualization field is to provide a closer match to the viewers' mental maps of the data sets in order to help them in understanding accurately the inside behavior of these complex data sets [131, 98, 81, 125].

Visualizing the safety aspects of embedded systems is comparatively a new field. As the complexity of embedded systems is increased the corresponding component fault tree size is also increased, which makes it difficult to handle the failure detection process.

In this thesis, we focus on utilizing the power of visualization in embedded systems application domains in order to help the safety experts and engineers in understanding the failure mechanisms of such systems rapidly and accurately and to get more insight about the underlying complex data. The goal of this thesis is to improve the understanding of the failure mechanisms in these systems and to reduce the communication gap between the system engineers and the safety experts. For this, we emphasize in providing many interaction facilities in order to help these experts and engineers to apply visual query about the underlying complex data. For example, we provide an integrated interactive visual framework called **ESSAVis** (**E**mbedded **S**ystems **S**afety **A**spects **Vis**ualizer). ESSAVis offers a 2D*plus*3D visual environment that on one side helps the engineers and safety expert in understanding the failure mechanisms of embedded systems, while increasing these engineers and safety experts' understandability of the structural relations between the different system components.

Recently, we witness a high acceptance of using 3D technologies in many of our daily life applications [5]. The cost of such technologies is becoming affordable over time due to the continuous decrements in the hardware cost. The quality of 3D stereoscopic pictures is progressing positively too, as the underlying technology for producing these devices is also advancing rapidly [5]. All these reasons make using these 3D technologies an active and attractive area in the visual research. In addition, many scientific visualization applications have already started using 3D displays. For example, visualizing the spatial relationships between different objects in the 3D datasets is one of the common examples of using 3D technology in the research.

In the case of information visualization field, data elements normally have non-spatial relations amongst them. This makes the idea of using 3D technologies for visualizing this kind of data an excluded possibility. However, Alper et al. [5] interestingly opened the doors of exploring the new possibility of utilizing the 3D technologies for visualizing some aspects about the data using the stereoscopic depth. They proposed to use the stereoscopic depth to highlight nodes in node-link diagrams. In this case, the node-link diagram is visualized as in 2D representations but it is rendered on a stereoscopic platform. This allows the highlighted nodes to be positioned in a plane closer to the viewer, which gives these highlighted nodes the required importance as they look bigger than the other nodes due to the depth cue. In our opinion, this idea is useful to bridge the gap between the 2D displays and the 3D technologies.

In the case of 2D displays, designers are forced to pack all the data elements on a 2D plane. This requires reducing the visual dimensionality, which leads to two main drawbacks: first it distracts our visual system as it is designed to perceive and interpret 3D spaces, second it can harm the scalability of visualizing large data [127]. However, in the case of 3D displays designers have the choice to layer the data in the 3D space offered by this technology in order to reduce the cluttering and potentially improves the understandability of the data [127]. Additionally, it compliments the human visual system, as human perceive data in the 3D world naturally. To achieve a pleasant visualization for non-spatial data (like graphs) on 3D platforms, designers need to start thinking in new directions for utilizing the stereoscopic depth offered by the 3D technology, similar to the direction proposed by [5, 127].

Therefore, the ESSAVis framework was also designed to be compatible with 3D stereoscopic technologies. This gives the opportunity to explore some possible directions to optimize the resulting visualization. For example, we use the stereoscopic depth to highlight some attributes of the nodes in the graph representation of the failure scenarios. This helps not only in saving other visual cues (e.g., color or shape) to encode other data attribute, but also helps in bridging the gap between the 2D graph representation and the 3D system model representation. In order to check the feasibility and effectiveness of the technique in general and specific to our application, we provided few extensions to the technique and conducted evaluation studies with different settings. The thesis also provides the details about our work and contributions in this direction.

1.2 Research Contributions

The thesis comes up with contributions towards three directions for achieving the final goal, i.e., enhancing the understanding process of safety failure mechanisms in embedded systems using visualization tools. In this regard, the thesis provides contributions in many directions to fulfill the desired goal. Figure 1.2 shows the thesis contributions in a nutshell. In the following, we summarize these research contributions one-by-one:

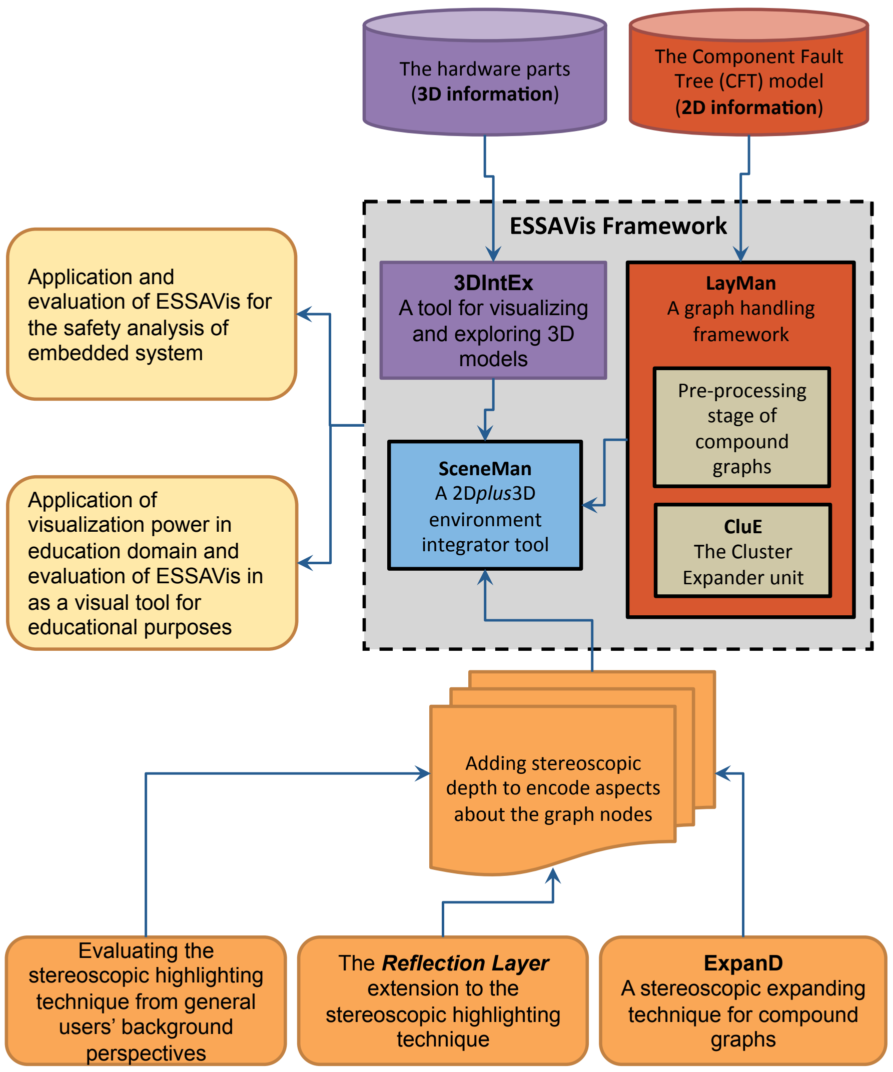

The hardware parts
(3D information)
The Component Fault Tree (CFT) model
(2D information)
ESSAVis Framework
3DIntEx
A tool for visualizing and exploring 3D models
LayMan
A graph handling framework
Pre-processing stage of compound graphs
CluE
The Cluster Expander unit
SceneMan
A 2Dplus3D environment integrator tool
Application and evaluation of ESSAVis for the safety analysis of embedded system
Application of visualization power in education domain and evaluation of ESSAVis in as a visual tool for educational purposes
Adding stereoscopic depth to encode aspects about the graph nodes
Evaluating the stereoscopic highlighting technique from general users' background perspectives
The Reflection Layer extension to the stereoscopic highlighting technique
ExpanD
A stereoscopic expanding technique for compound graphs

Fig. 1.2: An overview of the thesis contributions and their relations with each other.

- **Contribution 1 –** *The ESSAVis Framework:*

  Designing and developing the ESSAVis framework to support the understanding process of the safety failure mechanisms in embedded systems [8, 11, 12].

In this direction, the thesis provides several contributions as listed in the following:

– **Contribution 1.1 –** *Navigating through Compound Graphs:*

  The thesis provides a novel interaction approach, called CluE, for navigating through the compound structures of the compound graphs. One of the main feature of this approach is a generic algorithm that expands/contracts compound nodes in terms of the used layout algorithm [21].

– **Contribution 1.2 –** *A Generic Layering Framework for Visualizing Compound Graphs Interactively:*

  Designing and developing the Layout Manager unit, which first parses the CFT model and then converts it to the data structure that is necessary to calculate the safety aspects of the systems. It visualizes the abstract representations of the failure mechanism in 2D visual forms using the compound graph representation. Further, it produces a compact and area aware graph representation of the CFT model using the multi-level graph layout algorithm concepts [11, 20, 21].

– **Contribution 1.3 –** *Constructing the System 3D Model:*

  Designing and developing a mechanism to read the scene-graph data structure for building the 3D model of the underlying system. The thesis provides the detailed description of this contribution in Chapter 7, where we highlight the interaction techniques that help users to gain the required insight about the underlying system. All these provided options allow interactive exploration of the structural and behavioral aspects of the system model in 3D environments [7, 14]. This mechanism was built using the Vrui framework [102], which makes it portable and scalable on a range of VR environments (e.g., desktop 3D displays, PowerWall displays, or CAVE systems).

- **Contribution 1.4 –** *Integrating between Different Visual Views:*

  Designing and developing a mechanism to integrate between the different views seamlessly in the 3D world. The provided mechanism uses a novel algorithm to integrate between the different views using different styles like the *side-to-side* view, the *layered* view, or the *big-small* view [7, 14]. The detailed description of this contribution is provided in Chapter 7.

- **Contribution 1.5 –** *ESSAVis in Collaborative Environments:*

  Testing of the ESSAVis framework on a collaborative setup in a multi-users mode [6]. In this regard, the collaborative setup provides the visual views using a large shared screen while different users quipped with smart mobile devices interact individually with the visual scene.

- **Contribution 1.6 –** *ESSAVis Evaluation Study:*

  Designing and conducting a controlled evaluation study to compare between ESSAVis and ESSaRel [137] in terms of *accuracy*, *efficiency* and *user acceptance level* in analyzing the safety aspects of failures occurred in an embedded system represented as CFT models. The study was conducted with 25 participants from different scientific backgrounds, also including the real end users (i.e., the safety experts and the system engineers). The thesis provides the details of this conducted study and the analysis of gathered results in Chapter 8. The thesis uses different statistical tests to measures the significance of gathered results in order to show the feasibility and effectiveness of the developed ESSAVis framework [24].

- **Contribution 2 –** *Stereoscopic Highlighting Techniques:*

  Utilizing the stereoscopic depth for highlighting some aspects about the graphs nodes. This contribution was achieved through several evaluation studies including participants from different backgrounds

and using different graph sizes, node colors, and applications. This work is presented in Chapter 6.
The stereoscopic highlighting technique is a new emerging technique that supports using the depth cue in 3D devices as a highlighting technique for general node-link representations. One of the main abilities of this technique is isolating the specific graph portions to magnify them for a detailed exploration without resorting other highlighting attributes such as color or shape [5].
Using this technique in the ESSAVis 2D*plus*3D setup is helpful in integrating between the 2D representation and the 3D world. Therefore, we designed the system to convert the 2D representations to 3D representations on demand. In these situations, we use the stereoscopic depth to highlight some aspects about the nodes in the 2D node-link diagrams, such that nodes on focus are coming closer to the viewer rather than the other nodes. In this regard:

- First, we measured the accuracy of this technique by evaluating the ability of general users in reading the variations of depth values to encode some data aspects. To achieve this goal, we carried out a controlled user evaluation study in which we asked participants from different backgrounds to perform a specified set of tasks. The results show that using the depth cue in stereoscopic devices is readable for medium data sizes with at most three or four different layers. This can be useful in classifying the data set into a set of categories according to the depth value of the elemental layer [9].

- Second, we enhanced the stereoscopic depth detection with a *reflection layer* support [10]. We evaluated this reflection layer support from the normal users perspective in which we found out that this extra reflection layer extension improved the depth detection process with better accuracy. After this, we explored the possibility of using the stereoscopic depth is this domain in several directions. For example, we use the stereoscopic depth to visualize the filtering of sub-graphs within large graphs context [13].

  - Finally, we applied the technique for encoding structural relations between the compound nodes in clustered graphs using the **ExpanD** algorithm. In this case, we also conducted an evaluation study to measure the feasibility and the usability of our approach. The detailed description of this contribution is provided in Chapter 6.

- **Contribution 3 –** *ESSAVis Usage as A Visual Training Tool in Education:*

  We have successfully applied ESSAVis as a visual tool in the educational training process of engineers for understanding the complex concepts related to safety failure mechanism in embedded systems [16]. In this regard, we performed an evaluation study where we used the CFT representations produced by the ESSAVis framework to help students from the computer engineering background in order to understand the principle of the fault detection and the failure scenarios. Results of this conducted study show high acceptance rate from the participated students to use the *dynamic* visualization of the concept in understanding the failure scenarios. This conducted work opens the directions to investigate many challenges about the design of visualization systems for educational purposes. Also, this work investigates about those factors that can affect the students' learning process. This work is presented in Chapter 9.

1.3 List of Publications

The research contributions listed in the previous section have been published and presented in several prestigious venues related to the Human-Computer Interaction (HCI), the visualization domain, and the safety domain. The following is the list of all these publications:

1. **Ragaad AlTarawneh**, Shah Rukh Humayoun, and Achim Ebert: [17]
   A Hybrid Approach for Visualizing Spatial and Non-spatial Data Types of Embedded Systems
   In the proceedings of J. Abascal et al. (Eds.): INTERACT 2015, Part IV, LNCS 9299, 2015. *(To be published)*

2. **Ragaad AlTarawneh**, Shah Rukh Humayoun, Johannes Schultz, Achim Ebert, and Peter Liggesmeyer: [20]
LayMan: A Visual Interactive Tool to Support Failure Analysis in Embedded Systems.
In the proceedings of the 9th European Conference on Software Architecture (ECSA 2015), Dubrovnik-Cavtat, Croatia, September 2015. *(To be published)*
3. **Ragaad AlTarawneh**, Shah Rukh Humayoun, and Achim Ebert: [19]
Utilization of Variation in Stereoscopic Depth for Encoding Aspects of Non-spatial Data.
In the proceddings of the IEEE 10th Symposium on 3D User Interfaces (3DUI 2015), March 23-24, 2014, Arles, France.
4. **Ragaad AlTarawneh**, Shah Rukh Humayoun, and Achim Ebert: [18]
ExpanD: A Stereoscopic Expanding Technique for Compound Graphs
In proceedings of the 20th ACM Symposium on Virtual Reality Software Technology (VRST 2014), Edinburg, United Kingdom, November 11–13, 2014.
5. **Ragaad AlTarawneh**, Max Steiner, Davide Taibi, Shah Rukh Humayoun, and Peter Liggesmeyer: [24]
Does Visualization Speed Up the Safety Analysis Process?
In proceedings of A. Bondavalli et al. (Eds.): SAFECOMP 2014 Workshops, LNCS 8696, pp. 431–443. Springer International Publishing Switzerland, 2014. **[Book Chapter]**
6. **Ragaad AlTarawneh**, Jens Bauer, Shah Rukh Humayoun, Achim Ebert, and Peter Liggesmeyer: [8]
ESSAVis++: An Interactive 2Dplus3D Visual Environment to Help Engineers in Understanding the Safety Aspects of Embedded Systems
In proceedings of the 6th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS 2014), pp. 201–204, ACM, Rome, Italy, June 17-20, 2014.
7. **Ragaad AlTarawneh**, Andreas Griesser, Jens Bauer, Shah Rukh Humayoun, and Achim Ebert: [14]
3DintEx: A Tool to Explore Interactively the Structural and Behavioral Aspects of System Models in 3D Environments
In proceedings of the 9th IEEE Symposium on 3D User Interfaces (3DUI 2014), pp. 141–142, IEEE, Minneapolis (MN), USA, 2014.

8. **Ragaad AlTarawneh**, Jens Bauer, and Achim Ebert: [6]
   A Visual Interactive Environment for Enhancing Collaboration between Engineers for the Safety Analysis Mechanisms in Embedded Systems
   In proceedings of the companion publication of the 17th ACM conference on Computer supported cooperative work and social computing (CSCW Companion '14), pp. 125–128, ACM, New York, NY, USA, 2014.
9. **Ragaad AlTarawneh**, Jens Bauer, Shah Rukh Humayoun, Achim Ebert, and Peter Liggesmeyer: [7]
   Enhancing Understanding of Safety Aspects in Embedded Systems through an Interactive Visual Tool
   In proceedings of the companion publication of the 19th ACM International Conference on Intelligent User Interfaces (IUI Companion '14), pp. 9–12,ACM, New York, NY, USA, 2014.
10. **Ragaad AlTarawneh**, Johannes Schultz, and Shah Rukh Humayoun: [21]
    CluE: An Algorithm for Expanding Clustered Graphs
    In proceedings of the 7th IEEE Pacific Visualization Symposium (PacificVis), pp. 233–237, IEEE, Yokohama, Japan, 2014.
11. **Ragaad AlTarawneh**, Achim Ebert, and Eduard Kosel: [13]
    The Stereoscopic Filtering Technique
    Poster in IEEE VIS 2013, Atlanta, Georgia, USA, 13-18 October, 2013.
12. **Ragaad Altarawneh**, Jens Bauer, Patric Keller, and Achim Ebert: [11]
    ESSAVis: a 2Dplus3D visual platform for speeding up the maintenance process of embedded systems
    In proceedings of the 27th International BCS Human Computer Interaction Conference (BCS HCI '13). British Computer Society, Swinton, UK, 6 pages, Article 43, 2013.
13. **Ragaad AlTarawneh**, Jens Bauer, Shah Rukh Humayoun, Patric Keller, and Achim Ebert: [10]
    The Reflection Layer Extension to the Stereoscopic Highlight Technique for NodeLink Diagrams: An Empirical Study
    in proceedings of ISVC '13, Advances in Visual Computing, Lecture Notes in Computer Science, Springer, Volume 8034, pp. 1–12, 2013. **[Book Chapter]**

14. **Ragaad AlTarawneh** and Shah Rukh Humayoun: [16]
    A Two-Perspective Visualization Approach for Utilizing Visualization Power in Computer Science Education
    In proceedings of the 3rd ACM Computer Science Education Research Conference (CSERC' 13), pp. 85–91, Open University, Arnhem/Nijmegen, the Netherlands, 4-5 April, 2013.
15. **Ragaad AlTarawneh**, Jens Bauer, Shah Rukh Humayoun, Patric Keller, and Achim Ebert: [9]
    The Extended Stereoscopic Highlighting Technique For NodeLink Diagrams: An Empirical Study
    In proceedings of the 14th IASTED International Conference on Computer Graphics and Imaging (CGIM 2013), 8 pages, February, 2013 Innsbruck, Austria.
16. **Ragaad AlTarawneh**, Jens Bauer, Patric Keller, Achim Ebert, and Peter Liggesmeyer: [12]
    ESSAVis: A Framework to Visualize Safety Aspects in Embedded Systems
    In proceedings of the SIGRAD 2012, Interactive Visual Analysis of Data, pp. 59–66, Växjö, Sweden, 2012.
17. **Ragaad AlTarawneh**, Patric Keller, and Achim Ebert: [22]
    A General Introduction To Graph Visualization Techniques
    In proceedings of the IRTG 1131 Workshop 2011, OASICS, Vol. 27, pp. 151–164, VLUDS, 2011, Kaiserslautern, Germany

Moreover, the involvement in some other projects influenced partially (directly or indirectly) the work presented in this thesis. The following list presents some of the published publications resulted from that involvement:

1. Shah Rukh Humayoun, **Ragaad AlTarawneh**, and Yael Dubinsky: [84]
   Formalizing User Interaction Requirements of Mobile Applications
   In proceedings of the 28th British HCI Conference (BCS-HCI 2014), Southport, September 9-12, 2014.
2. **Ragaad AlTarawneh**, Max Langbein, Shah Rukh Humayoun, and Hans Hagen:
   TopoLayout-DG: A Topological Feature-Based Framework for Visualizing Inside Behavior of Large Directed Graphs [23]
   In proceedings of the 13th SIGRAD 2014 Conference of the

Swedish Eurographics Chapter, Gothenburg, Sweden, 2014. **[Best Student Paper Award]**.

3. **Ragaad AlTarawneh** and Shah Rukh Humayoun: [15]
   Evolving Mobile Prototypes Towards The Best-Suited Design and Interaction Schema Using The Genetic Algorithm
   International Journal of Mobile Human Computer Interaction (IJMHCI), 6(2), pp. 1–9, 2014. **[Journal Article]**
   (The initial submission at PID-MAD 2013 workshop, held on 27th of August in Munich (Germany) in conjunction with ACM MobileHCI 2013, received the ***best paper award****.)*
4. Shah Rukh Humayoun, **Ragaad AlTarawneh**, Achim Ebert, and Yael Dubinsky: [85]
   Automate the Decision on Best-Suited UI Design for Mobile Apps
   In proceedings of the 1st ACM International Conference on Mobile Software Engineering and Systems (MOBILESoft 2014), ACM, New York, NY, pp. 66–68, June 2–3, 2014.

1.4 Thesis Structure

The main theme of this thesis is to explore the possibility of utilizing the visualization power in depicting the failure scenarios in embedded systems. Due to the heterogeneity of such systems – as they are composed of software and hardware components – we visualize different aspects of these systems in order to cover both kinds of components as well as their relations.

After providing the problem domain and the overall introduction in this chapter, we cover the required background and the state-of-the-art literature, in Chapter 2. This chapter covers the state-of-the-art from different perspectives due to the diversity of the thesis contributions. After this, we focus on providing an overall picture of our ESSAVis framework in Chapter 3, in order to help the readers in understanding the overview of different parts and features of the framework. These parts and features as well as the specific utilization of the framework are then covered in details in the later chapters of the thesis.

In order to elaborate different units and features of the ESSAVis framework, we start with the framework contributions towards navigating through compound graphs using the CluE (Cluster Expander) algorithm in Chapter 4. The CluE algorithm is used in the **LayMan** (**Lay**out

Manager) unit of the ESSAVis framework, which is then presented in Chapter 5. In this chapter, we explain the steps that are required to produce the abstract representation of the failure mechanism in an embedded system. The LayMan unit produces a 2D representation of the failure scenario. However, in order to integrate this 2D representation with the 3D environment of our ESSAVis framework, we propose using the stereoscopic depth provided by the 3D technologies to encode some aspects about the abstract graph representation. Therefore, we focus on our contributions regarding the stereoscopic highlighting technique in Chapter 6.

Embedded systems are composed of software and hardware components; in order to visualize the hardware components we designed the ESSAVis framework to accept the geometry model of the underlying system. The construction of the system 3D model in ESSAVis and the integration between this 3D model and the abstract model of the failure mechanism presented in the earlier chapters are then covered in Chapter 7. After covering all the units and features of our ESSAVis framework in previous chapters, we focus on the user evaluation study in Chapter 8 that we conducted to compare between ESSAVis and ESSaRel [137] (which has previously been used in industry, as mentioned in CESAR Project Report, p. 43 [45]) in terms of *effectiveness*, *efficiency*, and *user-satisfaction level*.

One of the possible applications of our ESSAVis framework is to use it as a training tool for beginner engineers in the safety analysis domain. In Chapter 9, we focus on this aspect of the ESSAVis framework.

Finally, we present the conclusions of the whole thesis in Chapter 10. Also, we sketch the directions for the future works that could be done on the foundations of the work presented in this thesis.

2

State of the Art

Due to the complex nature of embedded systems, visualizing them is not a trivial process. In our research we aim at providing an insight about different perspectives of these complex systems. For example, we visualize the failure mechanisms of these systems, which can be modeled using graph models. Therefore, we use graph visualization techniques in order to provide the required insight about these failures. Another perspective is the 3D model of the system, which provides an insight about the structural aspects of the underlying system. Therefore, we use 3D visualization techniques in order to provide an insight about the required structural or behavioral aspects of the underlying system. Moreover, the research questions, which we discovered (see Chapter 3), are highly related to the application domain. Hence, our proposed solution was emerged from the application point of view, which is the *safety analysis process* in embedded systems.

Because of looking on the solution from different perspectives of the underlying system, we cover the state of the art from different point of views. In Section 2.1, we provide standard definitions from the literature and background information regarding the safety of a system. In Section 2.2, we mention some tools that visualize some safety information. In Section 2.3, we focus on graph visualization techniques and explain few of the layout algorithms in different categories. Also, we describe few of the interaction techniques that are used to interact the final representations and to explore the data. In Section 2.6, we cover the 3D visualization and the stereoscopic depth in information visualization.

2.1 Safety Information Background

In this section, we provide some background information on safety of a system. Here, we mention few terminologies that are necessary to know in order to understand about the safety of a system.

Faults, Failures, Hazards, and Accidents

In order to understand the construction of the failure scenario, which is used to describe the safety scenarios in embedded systems, it is necessary to distinguish between these terms as they are often misunderstood. Here we provide standard definitions of these terms from the state-of-the-art literature.

- **Deviation:**
  "A deviation is an event or a state that diverges from the correct, planned or usual function. The function can be a process, a technical function, or a human or organizational activity" [77]
- **Failure:**
  "Failure is an event that occurs when the delivered service deviates from correct service. A service fails either because it does not comply with the functional specification, or because this specification did not adequately describe the system function" [28]
- **Fault/Error:**
  "A fault is a static system event that may lead to a system failure" [28]
  Some of the fault definitions distinguish between the *fault* and the *error*; however, the most critical thing is to distinguish between the *fault* and the *failure*. A *fault* is a deviation from the system specification like the incorrect design, while a *failure* is a result of happening of one or more faults [28]. A fault may lead to a failure and may not; however, a failure happens only due to occurring of one or more faults. Also, a fault may occur as a consequence of one or more other faults [28].
- **Hazard and Risk:**
  "The term hazard is often used to denote a possible source or cause of an accident. While, the risk is the possibility of an undesired consequence to occur" [77]

- **Accident:**
  "The accident is defined as the event that causes unintentional damage or injury" [77]
  It is important to mention that an accident normally destroys or affects things such as human lives, human health, economical goods, or the environment.

Fault Tree Analysis

The Fault Tree Analysis (FTA) [152, 28] technique is a *top-down* approach to support the modeling of the safety scenarios in complex systems [152, 28, 112]. In a FTA, the relations between the set of faults, called the *basic events*, leading to a specific undesired event, called the *Top-Event*, are depicted in a graphical representation (see Figure 2.1(a)). A *basic event* is defined as one of the reasons causing the occurrence of current Top-Event with a certain failure probability [152, 112, 93]. In a FTA model, sets of basic events are represented as the tree leaves and the Top-Event is represented as the tree root. A FTA model can provide quantitative results such (e.g., the occurrence probability of the Top-Event) or qualitative results (e.g., the Minimal Cut Sets (MCS) information). A MCS signifies a set of events where the non-occurrence of even one event prevents the occurring of the Top-Event. The MCSs are ranked according to the number of events comprising them or according to their possible occurrence probabilities. In the case if MCS consists of just one event then it is called as a *single point failure*. There is a need to give special attention to such failures in order to ensure that they do not occur or to minimize the chances of occurring.

Component Fault Trees

The Component Fault Tree (CFT) technique provides an extension to the FTA technique by introducing additional information about the system structure. CFT is used to depict the failure relations among the system parts as a directed acyclic graph. The modeling of CFTs is a modularization technique to handle FTs for huge systems that consist of more than one component (see Figure 2.1(b)). The components are connected in a functional network via signal ports and the Top-Events of the CFTs correspond to the failure modes of the output signals. CFTs are similar to the classical FT with some differences. They can contain more than one top event. Also, one basic event can be connected to more than one logical gate. CFTs have been developed by P. Liggesmeyer, O. Mäckel, and B. Kaiser [93].

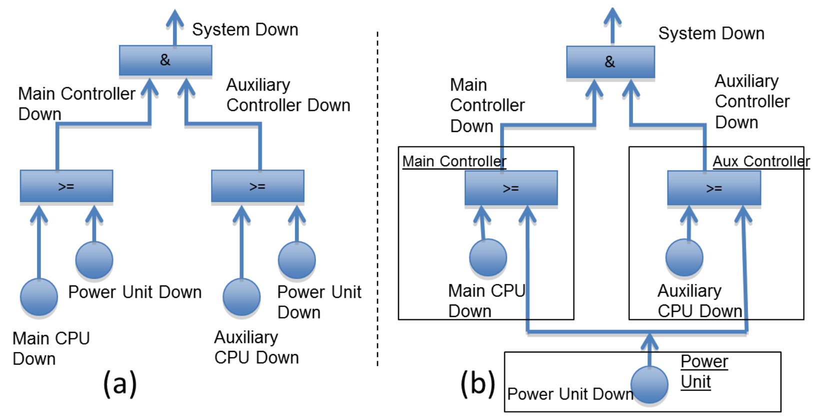


Fig. 2.1: The fault Tree concept Vs. the component fault tree concept

2.2 Visualizing Safety Information

To generate a fault tree model manually, highly skilled and experienced engineers and safety experts analyze the system. To achieve this, they need insights about the system and an overview of the failure modes and their consequences at a particular time. Performing this process manually requires plenty of concentration, but it is still an error-prone process due to the difficulties in understanding the whole system. To tackle the above-mentioned concern, many tools have been proposed in the literature to help the safety experts in modeling and understanding the safety mechanisms of complex systems. Many existing tools propose visualizing this by showing a tree structure of the failure. Most of these tools visualize the fault tree in 2D representations such as ESSaRel [137, 45], UWG3 [93], and Cecilia OCAS [34].

In the above-mentioned tools, the node-link diagram is used to show the relations between the system parts, while simple primitive shapes are used to show components of the Fault Tree (FT): e.g., small circles for representing basic events and a small rectangle for showing the gate (the logical connector) between two basic events. These tools also use color and/or text to depict other types of information, such as the gate type.

These kinds of tools are useful for modeling the failure relations between the system parts, but they do not provide options for analyzing the failure path or the set of critical parts in the underlying system. In spite of the facility provided for editing and modifying the FT structure in these tools, they generally lack the ability to analyze the FT itself and to present an overall view of the current failure mechanism.

Amongst these tools, ESSaRel provides a textual description of some safety aspects of the FT model. For example, the set of minimal cut sets in the current scenario is included together with information about the related set of components and the failure connections among them, all this data can be grouped into one xml file. However, this textual description is unreadable in most of the cases due to the data size and the file format.

PLFaultCAT [55] is another tool for analyzing the safety aspects of software systems in general [4]. PLFaultCAT is used to reduce the efforts needed by software engineers to safely reusing the software requirements and to customize the product-line for Software Fault Tree Analysis (SFTA) during the product-line engineering. In Yang et. al [161], a visualization system was proposed to support the engineers in identifying proper safe solutions for the system visually. Their proposed visualization integrates the fault tree with a plot representing the cause-effect relationships between the solutions of the system failures and the resulting risk reduction of the system. Further, they tried to associate the component fault tree view with the plot diagram to allow maintaining helpful context information about the current state of the system. SViT (Safety Visualization Technique) is another interesting visualization system, proposed in [103], that shows the safety status of a digital home. This system was proposed to help the homeowners for judging the current safety level of their homes and to find the reasons behind this level.

2.3 Graph Visualization Techniques

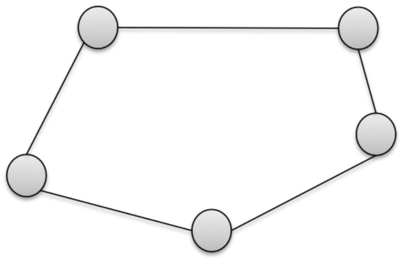
Graph visualization is a process of presenting the structural information in graphical form. It is used to represent diagrams of abstract data and general networks. It is one of the common techniques to visualize the behavior of the data in many application domains, e.g., visualizing the structures of software systems, representing different molecules in bio-informatics applications, showing the relations among people in social networks, etc.

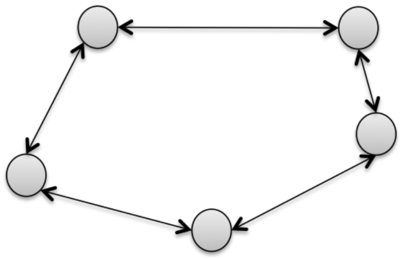
Using the graph visualization techniques requires the structural relations among the data elements. These techniques help in discerning the relations among the graph elements by representing them with a set of edges. Therefore, for large structured data sizes graph visualization techniques with the help of some interaction techniques are useful in exploring large data sets. Further, they provide intuitive methods to query the underlying data visually. In the forthcoming subsections, first we introduce some terminologies and then we provide an overview of a set of main graph visualization algorithms and their general features with some examples.

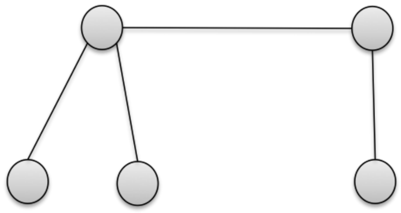
2.3.1 Terminologies

A graph is an abstract data type that is used to represent the set of relations between a set of data entities. Due to the diversity of this data type, graphs appear in numerous applications of the information visualization (InfoVis) field such as VLSI (circuit schematics), state-transition diagrams, social networks, and so on. As the size of this data type grows, the task of exploring and navigating through it becomes critical and difficult. Further, there are sets of requirements that should be met during the process of visualization. Therefore, many techniques have been introduced in the state-of-the-art literature that target at visualizing graphs efficiently and effectively. In the forthcoming subsections, we provide a brief survey about some of the important graph visualization techniques in order to gain an insight about them. Further, we briefly introduce few important interaction techniques that are useful in InfoVis domain.

The main aim of InfoVis techniques is to help in converting the abstract information from a textual description into some useful visual representations from the viewers' perspective, in order to support them to observe and navigate through the internal structure of the underlying data set [74]. We distinguish between different data sets based on the relations between their elements. In this regard, if the data elements do have inherent relations among each other then they are modeled as graphs. In this case, a proper graph visualization method is then


(a) An example of an undirected graph


(b) An example of a direct graph


(c) An example of a tree graph

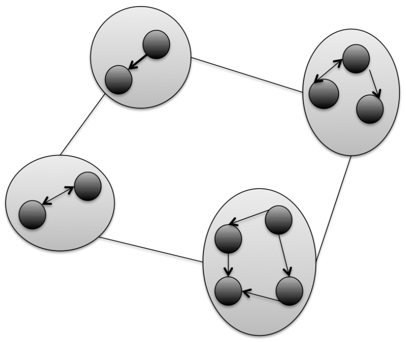

(d) An example of a compound graph

Fig. 2.2: Graph drawing examples

used to visualize such data types. If the data elements do not have any inherent relations among each other, then such data set is considered as a *discrete* data set, which means that the data entities' relations do not follow a specific pattern [81]. In our work, we interest ourselves only with the data sets that can be modeled as graphs. Therefore, we concentrate more on graph visualization techniques. Here, we provide a formal definition of a graph by Di Battista et al. in their book on graph drawing [30].

Definition 1 *Formally, a graph $G = (V, E)$ is a mathematical structure consisting of two sets, V the set of **vertices** (or nodes) of the graph, and E the set of **edges**. Each edge has a set of one or two vertices associated to it, which are called endpoints [30], see Figure 2.2a.*

- **Graph Drawing:**
  A graph $G(V, E)$ on a surface S means representing the G on the surface S, such that the points of S are associated to the nodes of G

and the curves of S are associated to each edge in G. The graph is *planar* if the drawing has no edges crossings. If the graph is drawn as a *tree* then the regions bounded by the edges are called faces and the infinite region surrounding the graph is called the out edge. The graph is called *orthogonal* if the edges are represented either horizontally or vertically [30, 68].

- **Directed Graphs:**
  These are the graphs in which the edges have direction notations, see Figure 2.2b. In this case, the out-degree of the node is the number of the out going edges from it whereas the in-degree of the node is the number of incoming edges to it. The graph is called *connected* when there is a path between any two nodes inside G. The directed graph is connected when each two vertices in G are reachable from each other via a directed path [30].
- **Tree:**
  A tree is an acyclic connected undirected graph, see Figure 2.2c. A directed tree is the directed graph with no cycles where each node inside it has exactly $indegree(v) = 1$ unless one single node called the *root*, which is connected to any other node in the tree and its in-degree equals to 0. The nodes with $outdegree = 0$ in the tree are called the *leaf* nodes. A *branch* is a set of nodes that are reachable by a directed path from the branch root and all the sources and the targets of edges inside this branch. The graph is *weighted* in the case when each vertex inside G has a real value represents its weight; otherwise the graph is *un-weighted* [30].
- **Compound Graphs**:
  A (directed) compound graph is a triple $C = (V, E, F)$ with the pair (V, E) forming a (directed) graph, called the *adjacency* graph, and the pair (V, F) forming a directed graph without cycles, see Figure 2.2d. In this thesis, we restrict our notion of a (directed) compound graph to triples $C = (V, E, F)$ where the pair (V, F) is a directed tree to which we refer as the *inclusion* tree. The edges $e = i, j \in E$ respectively $e = (i, j) \in E$ of a (directed) compound graph express adjacency relations and are referred to as *adjacency edges*. In the case of a directed edge $e = (i, j)$, we call the node i the *predecessor* of node j and node j as the *successor* of node i. We denote the set of predecessors for a node v as $pre(v)$ and the set of its successors as $suc(v)$. An edge $i = (u, v) \in F$ represents a nesting

relation, which means that *u* contains *v*. We call it as the *inclusion edge* [141, 68].

Many application domains use graphs as an abstract data type to represent their structure. For example, in social networks people in one group represent the graph vertices and the different set of relations among them are represented by sets of edges. In biology and chemistry, graphs are widely used to represent the molecular map, the genetic map, and the protein production path [73]. In the domain of software engineering, graphs are used to represent the structures of the software systems and to represent the internal data structure of software systems [99]. In object-oriented domain, graphs are used to depict the relations between different objects, e.g., they are used in data-flow diagrams.

Further, any hierarchical structure can be represented as a tree, which is a subtype of graphs. An example for this kind of structure is the file system in any operating system. The organization diagram for any institute can also be represented as a tree. See [30, 81] for further details.

Although graph visualization techniques are widely used in many applications; however, there are a set of challenges to deal with it properly. For example, the size of the represented graph is an issue by itself. Giving a layout for a very large graph is possible, however, understanding or reading this layout is often very difficult especially for untrained viewers. This also relates to the limited ability of human cognitive power and the limited screen space. Moreover, providing suitable interaction techniques in order to help the viewers in interacting and navigating through the data is also one of the main challenges. There is a need to think about possible solutions to handle these challenges in order to provide pleasant and predictable layouts as well as advancing suitable interaction techniques so that viewers can explore and navigate through the data more efficiently and effectively.

2.3.2 Graph Layout Algorithms

Generally, graphs are visualized through two main approaches, the *node-link* diagrams or the *space-filling* techniques. The trivial way to display this kind of data is to use node-link diagrams. This representation includes trees and graph networks. In this representation, the relations among the data entities are depicted using a set of lines connecting them [30, 81].

In [89], a new type of visualization model was proposed to display the graph using the space-filling techniques or the space-nested layouts, in which the relations are represented implicitly.

The Node-Link Layout

The basic requirement of the node-link layout model is computing the positions of the set of nodes and the set of curves connecting them. A pleasant layout may comply with the following aesthetic criteria that attempt to increase the readability of the underlying graph [98]:

- Nodes and edges should be evenly distributed.
- Edges crossings should be minimized, in order to make a readable layout.
- Depict symmetric sub-graphs in the same layout if possible.
- Minimize the edges bending ratio.
- Minimize the edges lengths, which increases the readability of the graph and helps the viewers in detecting the relations among different nodes faster.
- If the data is inherently structured then distribution of the nodes at different layers. For example, in data flow diagrams it is recommended to separate between different layers in the graph in order to reflect the semantic behind the behavior of the data.

Many other aesthetic criteria have also been proposed in the literature (see [30, 98] for further details). It is worth mentioning that it is a difficult task to combine most of the criteria together, as some of them contradict each other while some of them are hard to implement and cannot be optimized efficiently. Because of these difficulties, many approximate solutions have also been proposed in the literature (see [30, 98] for further details).

Thus, layout algorithms in practice are the result of compromising among different aesthetics. Specifying the required aesthetic criteria is highly depended upon the application domain. Therefore, prioritizing a set of aesthetic criteria is normally considered as pre-request for any layout algorithm. A detailed explanation for prioritizing such criteria has been described in [125].

Spring Embedding Layout Algorithm:

The spring embedding layout algorithm, widely known as the ***force-directed layout***, was originally proposed by Peter Eades in 1984 [58]. Due to its simplicity and its ability to produce a symmetric layout, it is considered as one of the popular node-link layout methods. The spring layout algorithm represents the graphs as physical systems in which the graph nodes are considered to be charged particles connected to each other using a set of springs.

The first model, proposed by Peter Eades [58], represented the graph nodes as a set of rings connected using a set of springs [30, 98]. Each ring or node is associated with two types of forces: the *attraction* forces and the *repulsive* forces. Based on the distance and the properties of the connecting springs, as the energy of the spring system is reduced the drawing of a graph approaches optimality. The attraction force f_a is applied to the neighbor nodes while the repulsive force f_r is applied to all graph nodes. These forces are defined as follows [58]:

$$f_a(d) = k_a log(d) \tag{2.1}$$

$$f_r(d) = \frac{k_r}{d^2} \tag{2.2}$$

Here, k_a and k_r are constants and d is the current distance between two nodes. Figure 2.7b shows results from applying this algorithm to relatively small graphs. Although, the force-directed layout produces pleasant and symmetric layout for moderate graph sizes; however, it is considered as one of the expensive algorithms in terms of time complexity as it exceeds $O(n^2)$ [30, 98, 81], where n is the number of the nodes.

Further, force-directed layout shows a lack of predictability [30, 98, 81, 159], which means it produces different results for two different run times for the same graph. This final representation unpredictability causes a serious problem, because interacting with an unstable representation could lead to an issue of the mental map preserving of the resulted representation [159]. Despite these disadvantages, it has been widely used in many visualization frameworks [153]. However, the algorithm itself has been revisited and optimized many times [94, 67, 70, 65, 86, 30, 98, 50].

Topological Feature-Based Layout:

The feature-based graph drawing concept has been initially proposed

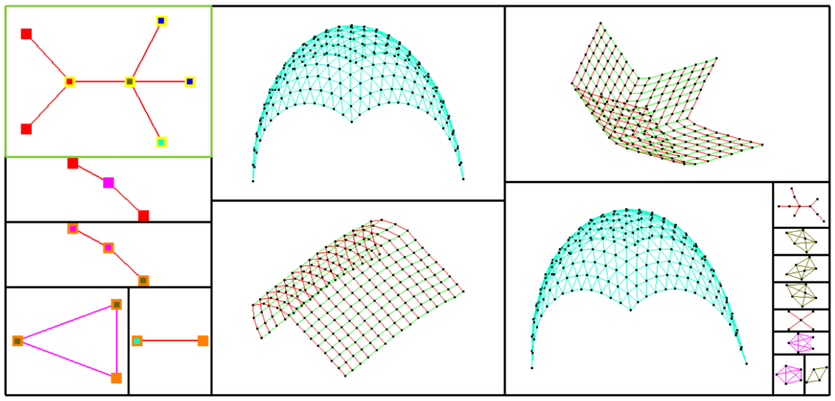


Fig. 2.3: Potential produced layout using the TopoLayout algorithm [25], produced using the libgraph package [78].

by Archambault et al. in [25]. The concept is called **TopoLayout**, which is a multi-level feature-based approach. The pipeline of this approach consists of four main steps.

The first step is called the ***decomposition*** phase in which the graph is decomposed into many sub-graphs based on the topological features for each internal sub-graph. For example, if a set of nodes in one sub-graph is topologically connected as a tree, then this set of nodes are grouped together into one node called a *meta-node* that represents them. TopoLayout detects seven topological features including *trees*, *complete graphs*, *biconnected components*, *clusters*, and the undefined structure that is called *unknown features* [25].

The second step is called the ***feature layout*** phase in which the meta-nodes or the grouped sub-graphs are layout using one of the layout algorithms tuned with its topological feature. The third phase is called the ***crossing reduction*** phase that aims at eliminating the crossings ratio in the resulted layout. The final ***overlap elimination*** phase aims at changing the node sizes in the final layout to ensure that nodes do not overlap each other.

The final result of TopoLayout is a tree representing the graph hierarchy, in which each node represents a sub-graph in the original graph and each meta-edge represents the relation between two sub-graphs in the original graph.

This layout technique helps in drawing relatively large graphs. Also, it provides viewers the details about the internal structure of the graph, which can be useful in extracting more information about the graph itself (see Figure 2.3).

GrouseFlocks [26] was introduced to provide an interactive platform to explore large input graphs through cuts of the superimposed hierarchy. The goal was to provide viewers the ability to see several different possible hierarchies of the same graph.

Planar Graphs:

A *planar* graph is a graph drawn without edges crossings. This class of graphs can be drawn in linear time. To produce a planar layout for a general graph, some pre-requests should be applied to these graphs [81], e.g.:

- *Planarity tests for graphs:* If it is possible to draw the given graph without edges crossings.
- *Finding a planar layout algorithm satisfying the required application constrains:* In order to know the suitable layout algorithm that satisfies a set of constrains related with a specific application.

Generally, a planar graph drawing is supported by two popular algorithms. The first one is called the Fraysseix, Pach, and Pollack (FPP) algorithm [53, 95, 132] that builds a drawing of G on a grid of size $(2n-4)*(n-2)$ and in $nlog(n)$ time. Later, the FPP algorithm was improved to run in linear time [95]. This second algorithm is proposed by Schnyder [132] and it finds a straight line embedding on a grid of (n^2) nodes and runs in linear time. Example of a planar graph is shown in Figure 2.6b.

The Sugiyama Algorithm:

Before we introduce tree layout methods, it is worth to mention that both layouts (i.e., the force-directed algorithms and the TopoLayout algorithm) work perfectly with undirected graphs. However, not many algorithms were designed for visualizing directed graphs. The Sugiyama algorithm [143] is one of the first algorithms that draws general directed graphs. The basic approach is to layer the graph nodes, which means assigning a layer for each node and then placing the nodes in the corresponding layers. The algorithm has two steps for reduce the edges crossings and the nodes overlapping. In general, directed graph layout algorithms are difficult to implement, mainly due to the complexity of the nature of directed graphs. Therefore, many of the Sugiyama algorithm steps are considered as NP-hard (based on [71]) and some of them are considered as NP-Complete (based on [62]).

Tree Layouts

Many layout algorithms have been proposed in the literature that focus on drawing the tree structures. This is mainly due to the tree structure's simplicity and popularity. Here we introduce briefly different layout algorithms targeting node-link trees and then explain some space-filling techniques.

Node-Link Tree layout Algorithms

One of the basic approaches to draw a tree is to use the node-link diagram in which the parent-child relations are depicted using explicit edges, as mentioned earlier in the previous subsection. The ***classical tree layout*** algorithm is one of the early methods that was initially proposed by Reingold and Tilford in [128] (see Figure 2.4a). It produces aesthetically pleasant trees in 2D and has a straight forward implementation. However, it is not a space efficient technique because of its preference for one of two dominating growth directions, i.e., *horizontal growth* or *vertical growth*. To cope with this problem, some compact tree layout algorithms have been proposed and implemented to produce a classical tree appearance in a more compact fashion [59, 30, 98, 159, 81, 50].

Another example for node-link tree layout is the ***radial layout*** algorithm, which was initially proposed by Eades in [59]. The radial layout, including its different variations, places the root in the middle of co-centric circles and distributes the children of a sub-tree into circular shape according to their depth in the tree recursively. This layout uses space more efficiently than the classical method, but it lacks the understandability compared to the classical tree layout, e.g., in this layout it is difficult to find the root of the tree (see Figure 2.4b) [30, 59, 120, 162].

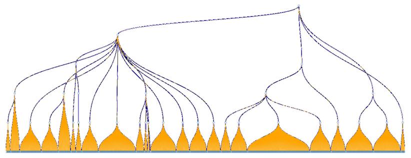
As a sibling of the radial layout, the ***balloon layout*** algorithm has been introduced by Carriere and Kazman in [44]. In this layout, sibling sub-trees are drawn in a circle centered at their parents. This layout is effective in showing the tree structure. Balloon layout can be obtained by projecting a cone tree onto a plane [129, 47, 30, 98, 159, 81] (see Figure 2.4c).

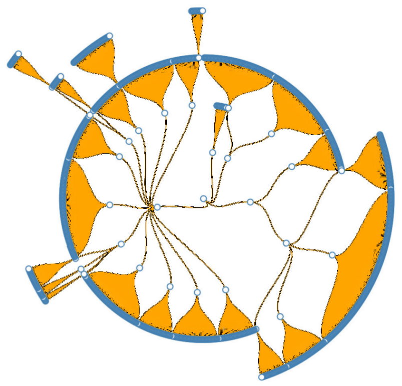
H-Tree produces a classical layout for binary trees and works perfectly for balanced tree. However, it is hard to find the root position [135] in it (see Figure 2.4d).

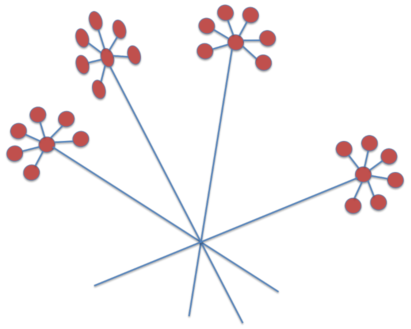
All tree layout algorithms produce predictable results in linear time, usually with a time complexity of either $O(n)$ or $O(nlogn)$ [30, 98, 159]. Doing a quick comparison among different tree layout algorithms, we observe that the classical tree layout perfectly depicts the hierarchy structure of the tree with sacrificing the screen space. While the radial layout, the h-tree layout, and the balloon tree layout use the screen space more efficiently; however, with difficulties in finding the tree root [30, 98, 159, 120].

Space-Filling Techniques

Space-filling techniques can be subdivided in two types: the ***Space-Division*** layouts and the ***Space-Nested*** layouts. In the Space-Division layouts, the parent-child relation is depicted implicitly by attaching the children to their parent. The *Sunburst* algorithm [138] uses radial or circular space-filling techniques. The general belief of the developer community is that radial layout methodology better conveys a hierarchy structure without sacrificing the efficient screen space usage [138, 90]. One of the problems of this layout is that it is difficult to distinguish between the child-parent relationships and the sibling relationships, as both of them are expressed using adjacency representations. Moreover, due to the different number of children for each parent, the nodes' sizes are difficult to control; therefore, the final layout can occupy a large space for a node that has many children, while other nodes in this case


(a) Classical tree layout example, produced using the D3 package [38]


(b) Radial tree layout example, produced using the D3 package [38].


(c) Balloon tree layout.

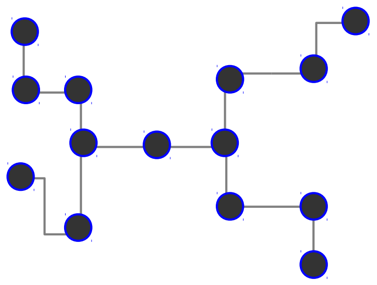
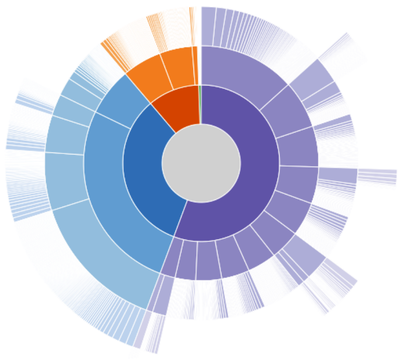
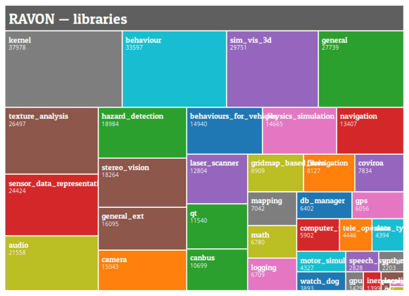

(d) H-Tree layout

Fig. 2.4: Tree Layout Examples.

may get a tiny thin rectangle that is not enough to show those nodes' labels or colors (see Figure 2.5a).

Whereas in Space-Nested layouts, the child-parent relationship is drawn using nested boxes. The idea is to place the children within their parent node. A good common example is the *Treemap* (see Figure 2.5b) [89, 136]. Nodes are represented as rectangles where each rectangle is subdivided into a number of sub-rectangles equal to its children number. The subdivision process is performed recursively. This technique is popular because it uses the screen-space efficiently and it shows the size of the leaves in the tree. However, this technique lacks the ability of showing the hierarchical structure of the tree. Also, due to the subdivision process it is highly possible to produce long and thin rectangles, which can lead to a problem in interacting with them, especially in selecting or highlighting such rectangles [89, 136, 159, 81, 150].


(a) SunBurst layout produced using the D3 package [38].



(b) TreeMap layout produced using the D3 package [38].

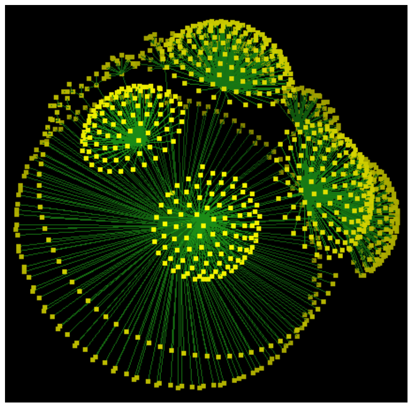
Fig. 2.5: Examples of space-filling layouts.

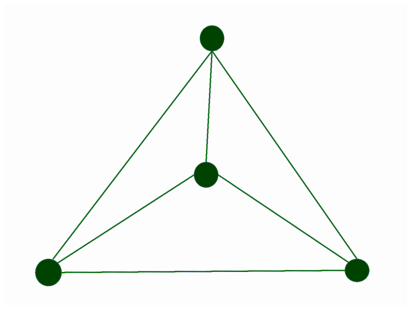
Matrix Visualization

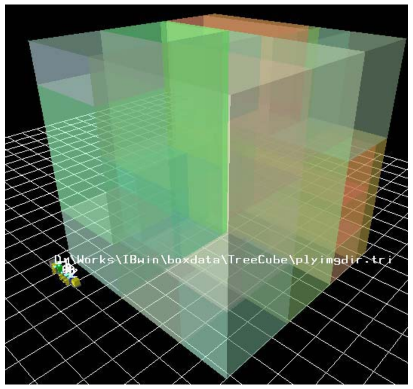
Matrix visualization is another technique that represents graph nodes relations implicitly (see Figure 2.7a). In this case, each row and each column corresponds to a node while the edge is represented using the intersection point between a row and a column. Edge attributes can be shown using different visual parameters such as color, size, and shape. This method is scalable since it can produce a pleasant layout graph with thousand nodes. However, it is difficult to detect the graph path in this representation. See [2, 80] for further details.

2.4 Nodes and Edges Clustering

Clustering techniques were introduced in graph visualization applications as a tool to reduce the visual cluttering in the final layout. This is achieved by producing an abstract view for the input graph. According to [100], reducing the number of visual elements not only increases the clarity of the scene but also increases the rendering performance. Clustering algorithms can be classified in two main types based on the criteria used in the clustering process. The first type is called the


(a) Hyperbolic tree layout, produced using the Walrus package [147].


(b) Planar graph example.



(c) TreeCube layout, (courtesy of [144]).

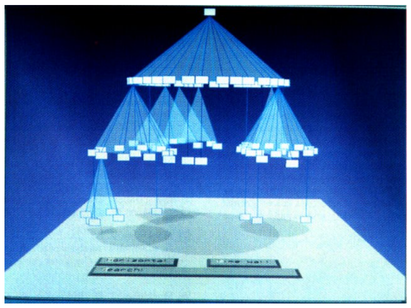
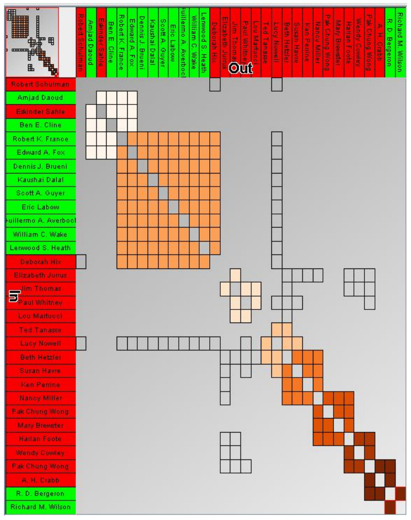

(d) Cone trees, (courtesy of [129]).

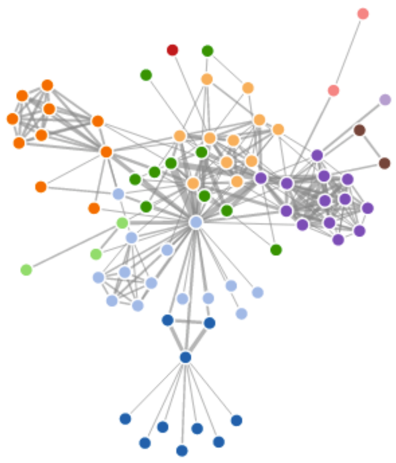
Fig. 2.6: Examples of graph layouts.

natural-clustering in which the structural information among the graph nodes is used to find a pattern of nodes having the same common criterion [130]. The second type is called the ***content-based*** clustering, in which the semantic meaning of the relations among the graph nodes is taken into account.

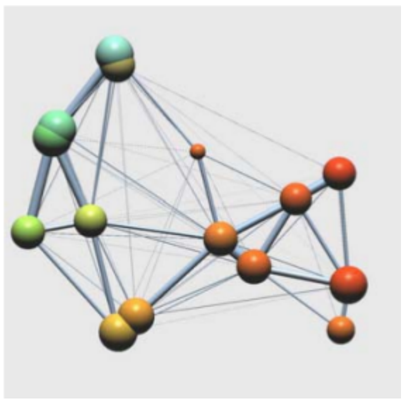
The clustering is an interactive concept of large graphs has been described in many previous works. Feng et al. introduced the clustered graph model in [63], which is considered one of the first work in this area. They investigated how the graph model can be drawn with minimum edges crossings or without edge-regions crossings. Also, they



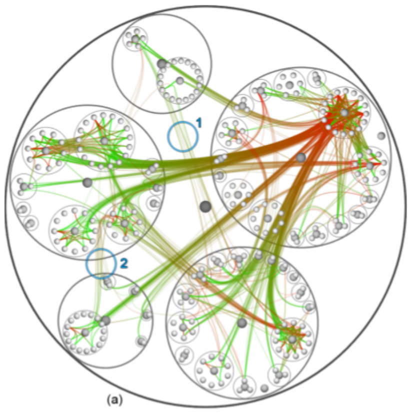
(a) Matrix visualization layout, (courtesy of [80]).



(b) Force directed layout example produced using the D3 package [38].



(c) Clustering example, (courtesy of [149]).



(d) Edge bundling example, (courtesy of [82]).

Fig. 2.7: Examples of graph layout techniques.

introduced efficient algorithms for testing C-planarity and for detecting C-planer embedding with this kind of graphs.

Another important work in this domain was introduced by Eades et al. in [61]. Their proposed approach allows exploring and navigating large graphs based on the following order: first producing a hierarchy of the graph using the clustering graph algorithm, then making layout the current level of detail using the force directed layout algorithm, and finally providing animation to transform one graph view to another one. In [83], Huang and Eades described the DA-TU system that combines animated clusters with online force-directed animated graphs for visualizing large graphs.

Another example is ASK-GraphView system, proposed by Abello et al. in [3], that allows clustering and navigating large node-links diagrams. Their proposed solution builds a hierarchy based on arbitrary weighted large graphs. Moreover, it suggests the navigation through top-down fashion in order to expand clusters on-demand. Archambault et al. proposed GrouseFlox approach in [26] that allows viewers interactively exploring large graphs using a cut operation of a superimposed hierarchy. This proposed approach creates hierarchy using a topological feature-based algorithm or any specific clustering algorithm. This solution was designed for exploring a graph hierarchy space. This was achieved through allowing viewers to explore different possible hierarchies for the same graph.

In [157], Martin Wattenberg described a method of clustering networks according to the attribute that is associated with network nodes. Since many graph visualization applications use the structure-based clustering algorithms; therefore, many structural characteristics have been used as clustering criteria such as the distances between graph nodes and node degrees. The ***structure-clustering*** is widely applied to preserve the structure of the original graph [130].

The interaction facilities become applicable with clustering, as it eases the applying of filtering techniques for the layout result. This results in an increased searching speed for specific data patterns. This is accomplished by partitioning the nodes into a set of groups, then filter them based on the specified criteria, and finally narrow the search domain to the remaining clusters (see Figure 2.7c for a clustering example). In [27], Auber et al. presented an interesting clustering technique for a special type of graph called *small-world networks*. In [30] and [98], authors provided an intensive overview about clustering, the covered application areas, and the set of heuristics for each clustering method.

Another method for reducing the cluttering ratio is the ***Edge Clustering*** approach. The goal of this approach is to free more space by grouping sets of edges that share the same end points. This process reduces the visual cluttering in the final representation. *Edge-bundling* techniques have also been proposed to increase the readability of the graph. This is achieved by reducing the visual cluttering from the adjacency edges (see Figure 2.7d) [82]. Another example is the *flow-map* method in which all edges that have the same source are grouped into one thick edge, generating a pattern of the *edge-flow* [123]. Although, this technique intuitively shows the flow of the data from a single source to different destinations; however, it is applicable only for specific graph visualization applications (e.g., the migration path from a single source). Also, in case of multiple sources this approach causes a visual cluttering among the different flow maps, which leads to difficulties in reading the underlying graph.

2.5 Interaction Techniques

The goal of the visualization techniques is to increase the comprehension level of the input data. This goal cannot be achieved most of the times by only producing a static image representing the data. In order to fully achieve this goal, there is a need to provide the ability of interacting with the data and the generated representation. Targeting this, many interaction and navigation techniques have been proposed in the state-of-the-art literature to facilitate the data exploration and interaction with the resulting representation [81, 136, 159]. In this subsection, we list a set of interaction concepts that have been applied together with the visualization layout algorithms. In [163], a summary of popular interaction techniques is presented :

- *Selection:* Giving viewers the ability to highlight and process specific objects.
- *Abstraction/Elaboration:* Changing the level of details of the representation scheme. This allows viewers to get different insights about the underlying data.
- *Reconfiguration:* Giving viewers the ability to change the layouts for the same representation scheme, e.g., sorting the graph nodes based on a specific criteria.
- *Encoding:* Switching between different layout methods, e.g., converting the node-link diagram into sunburst layout.
- *Exploration:* Giving viewers the ability to change and transfer the view point of the graph layout. Zooming and panning are examples of this category.

- *Filtering:* Removing unnecessary details and displaying the remaining items in a more visible fashion. The main concept is to filter the data nodes based on their attributes in order to make the querying process easy and fast. See [35] for further examples.
- *Connection:* Giving viewers the ability to highlight the paths between relevant objects to the focused object.

Zooming and Panning

Zooming and *Panning* are basic interaction tools for exploring large amount of information. Panning means moving the camera across the scene while zooming allows viewers to switch between the abstract and the detailed views.

The *geometric* zooming is used to adjust the screen transformation and to show up the graph content. The *semantic* zooming means that the displayed information is changed when approaching a particular area of the graph. Both zooming and panning are complement to each other. A common example is the geographical maps like the ones provide by Google-Earth. In this case, let we suppose a viewer zooms into an area next to Frankfurt in Germany. If the viewer needs to transfer the view to another area (e.g., Al-Karak in Jordan), he/she zooms out first and then pans to Jordan in the middle-east region, and finally again zooms into the Karak city in south of Jordan. Doing this procedure without panning in the middle needs longer time to find the destination [81, 159].

In [69], Furnas and Bederson introduced an elegant model to explain how zooming and panning work together. Their proposed concept is called *space-scale* diagrams. It defines an abstract space by creating multiples copies of the original 2D image, then stacking them up to construct an inverted pyramid on which each copy is placed on a certain magnification level. The space-scale diagram can be used not only for the geometric zooming but also for the semantic zooming [69, 81].

Focus+Context Techniques

The main goal for the *focus+context* technique is to give viewers the ability to see the primary object in a full view (i.e., the focus), together with the an overview of the surrounding details which corresponds to the context.

In general, losing context is considered to be an issue in information visualization applications. In order to alleviate this problem, focus+context techniques give viewers the ability to focus on some details without losing the global context [131]. This concept does not replace the zooming and the panning methods, but rather complement them. The majority of visualization application systems implement both techniques together as provided interaction options.

Overview+detail is one of the earliest focus+context approaches, in which separate display regions for different resolutions are used in a way that enable viewers to switch between different displays frequently [111].

The fish-eye technique is one of the most popular focus+context techniques [131], in which an area of interest becomes larger than the rest of the underlying graph. At the same time, it shows the other regions of the layout successively with few details. In the fish-eye approach, the task of computing the hyperbolic coordinates is faster than the layout algorithms, which is considered as an issue for the interaction with the visualization [81, 159]. The distortion appears as a negative consequence of this technique, which leads to destroy many aesthetics criteria controlling the layout algorithm, e.g., unwanted edges crossings might appear [81, 159].

2.6 3D Layout Examples

Many layout algorithms have also been proposed in the literature targeting the 3D space rather than 2D plane. The trivial reason behind it is that human beings are familiarized with the 3D in real world. It becomes familiar for them to explore and read data in 3D spaces. Further, the third dimension adds more space, which helps in displaying large graphs sizes. For example, Tanaka et al. proposed the *Treecube* technique in [144], an extension of the traditional treemap technique, that uses nested cubes to represent the parent-child relationships (see Figure 2.6c). The *Hyperbolic* layout algorithm appeared for the first time in [105, 106] and has successively been used by [117, 116, 115]. The idea behind this algorithm was to distribute the data entities over the hyperbolic space. Figure 2.6a) shows a hyperbolic tree layout for a walrus-directory graph.

Hyperbolic trees are valuable to visualize hierarchical structures such as file directories, web sites, classification hierarchies, newsgroup structures, etc. While the traditional methods to visualize these data-set like the paging technique (i.e., dividing the data into several pages and displaying one page at a time), zooming, or panning show only part of the information at a certain level of details. Using the hyperbolic trees may help in showing details and context at once [37] . The graph can be easily laid out in the hyperbolic plane using uniform edge lengths. Indeed, this property of hyperbolic geometry is one of the main reasons behind its usage. Also, it reduces the node-to-node overlapping. Therefore, this kind of layout algorithm is good for Trees, DAGs, and even Cycles [117, 116, 115]. However, it requires many 3D interaction techniques like rotation or/and zooming. It also overlaps between nodes and produces a cluttered visualization in case of large number of edges between nodes. This method can be displayed in 2D or 3D providing a distorted view of the tree, which makes the interaction task with large trees easier [81]. Further, the force-directed techniques can also be generalized easily to make them suitable for the 3D space [51]).

Cone Tree technique was originally developed to layout trees in 3D space. It places the nodes at the top of the cone with its children arranged evenly in the cone base. This layout contains many layers where each one represents a tree level, in which all cones have the same height. The cone-base diameter for each layer is reduced in bottom-up fashion. This helps the lowest layer to fit into the width of the box containing the full cone tree (see Figure 2.6d) [129].

The above-mentioned examples are based on using the animated 3D visualization and the lightening to show the depth perception. The main drawback of using such visualizations is the extra occlusions that can hinder the visibility of the layout elements. Moreover, these examples do not consider the variations in depth values while positioning the nodes in the 3D space, which effects the level of the data perception [48].

Cockburn and McKenzie [48] did a study for evaluating the efficiency of 2D, 2.5D, and 3D layouts in physical and virtual platforms. In their study, they concluded that 3D stereoscopic effect is necessarily for leading to more effective use of spatial memory for high-level tasks, like building the viewers' mental map of document arrangements. As a result of this finding, many scientific visualization applications are getting benefits from 3D display devices because many of these applications' objects inherently are 3D objects [76]. However, the above-mentioned examples did not exploit the stereoscopic depth cue nor they provided a quantitative study to measure the usability of their approaches.

Based on [107], 3D visualization techniques face multiple challenges: Firstly, objects in 3D may occlude each other which causes an issue while exploring the data set. Secondly, providing a suitable layout algorithm that assures less object-overlapping and reduces the edges crossings is also considered as a big challenge. Thirdly, supporting interaction techniques making the data exploration task easier is also a big challenge by itself. Finally, choosing an appropriate metaphor that increases the information understanding process is hard to find. Many real-world metaphors have also been used to present data into a 3D world (see [101] for further examples).

2.7 Stereoscopic Depth in Information Visualization

Many examples have been provided by [49] for showing the usability of using the 3rd dimension in encoding different aspects of the data. This can be achieved by isolating a subset of a 2D graph representation to a separate layer, thus a 2.5D would be created. In this case, the third dimension can be used to encode the relations between the different layers rather than the spatial relations among nodes [40]. Another example was presented in [40], where the third dimension was used to show the evolution of the node-link diagram over time, where each layer encodes a 2D representation of the graph at a certain time step.

In [40], the third dimension was used to depict the hierarchical nesting of clusters in the graph using a set of transparent layers. However, above mentioned examples did not exploit the depth cue nor they provided a quantitative criteria to measure the approach usability. While in [5], the third dimension was used as a tool to support the low-level graph analysis tasks. In their work [5], Alper et al. combined the usage of stereoscopy and 3D graphs and then applied them to a node-link diagram. The depth cue was used as a highlighting technique to show the importance degree of the graph elements by rendering them closer to the view-point.

Many efforts have also been done in showing the efficiency of using the 3D graph visualization with high-resolution stereo displays. According to the study results by Ware and Franck in [156], 3D graph visualization is recommended only when stereoscopic cues and real time rotations are supported.

Many applications use the stereoscopic effect as a technique to separate the set of images that share the same texture of the background. In addition to that, depth cues have also been used to separate the overlapped labels in the 3D world in order to reduce the cluttering effect [122]. In [129], Robertson et al. used the variations in depth to reduce the cluttering of information by filtering the information based on the depth values. Whereas in [56], Deller et al. evaluated the pre-attentive visual features as a highlighting technique in virtual reality environments. Authors showed that the stereoscopic depth has the additional advantage of facilitating an intuitive interaction technique, as it positions the relevant information towards the viewers or pushes them away according to their importance degree.

The main motivation behind using the stereoscopic depth in graph visualization is its ability of reducing the edges crossings in node link diagrams. This finding is according to the study that was conducted by Ware et al. in 2005 [155]. They claimed that 3D node-link diagrams make it easier to identify the adjacency of certain elements. This also makes the task of counting nodes accessible from a particular node, quite easier. However, due to the perspective view, rendering a graph in 3D environments can lead to wrong conclusions about the importance degree for some nodes. This is because some nodes are rendered closer to the virtual camera viewpoint than the others, which results in a visual emphasis of the closer nodes. This emphasis comes from the graph layout algorithm or from the viewer's viewpoint rather than from the data set. Moreover, 3D layouts require a high amount of viewpoint navigation. For example, when a node of interest is rendered at a further depth level then the viewer needs to rotate, zoom, or change the view angle to get a better view. This process affects on viewers to keep the mental map of the layout because the graph may appear differently each time [5].

In [5], authors interestingly opened the doors of using the stereoscopic depth to highlight the importance degree of some nodes in 2D graph representations in a way that the highlighted nodes were brought to a new depth layer closer to the viewer rather the remaining graph, thus a 3D representation was created. Authors also provided a qualitative user evaluation study to evaluate the viewers' abilities in detecting the variations in depth. The results of their study showed a promising way to proceed in their direction in order to provide more utilization of 3D technologies in graph visualizations.

2.8 3D Modeling

VRML: Virtual Reality Modeling Language

Virtual Reality Modeling Language (VRML) is one of the standard languages that have been emerged to describe 3D shapes and interfaces on the World Wide Web [41, 146, 52, 32]. It has been used in many applications such as web-based entertainment, CAD system visualizations, 3D user interfaces developing, designing collaborative platforms, interactive visualization for education purposes, virtual experiences in museums, etc. The key feature of VRML is its ability in describing the complex object structures in simple textual forms.

VRML was designed to fulfill a set of the following requirements [33]:

- *Creation and Modification:* This requires giving developers the ability to create and modify VRML files as well as the ability to convert the file format to any other 3D format easily.
- *Integration and Reusability:* This requires giving developers the ability to integrate different 3D objects in a VRML world and the ability to reuse them in different projects.
- *Extensibility:* It gives the possibility to specify new objects' types that were not explicitly defined in the VRML standard.
- *Capability of implementation:* This means to be capable of implementation on a wide range of systems
- *Performance:* It requires the ability to be scalable and interactive with wide range of computing platforms with the same performance.
- *Scalability:* This means providing support for the development of large dynamic 3D world.

VRML supports many primitives, e.g., cylinders, cones, cubes, and spheres have already been implemented and supported by it. VRML also support different nodes types, e.g., group nodes, shape nodes, transformation nodes, etc. It links between these nodes in a parent-child fashion in a manner that each group node consists of many children nodes that specify its geometry, orientation and color.

- *Group Node:* It represents an abstract group for describing the grouping of other nodes.
- *Transform Node:* It represents a node for describing the transformation information of other nodes.
- *DirectionLight Node:* This node represents the light source position and its orientation.
- *PointLight Node:* It is a light node that illuminates in all directions.
- *SpotLight Node:* It is a light node that illuminates in a specific direction and a specific area.
- *Inline Node:* This node reads a file and attaches it to the scenegraph.

However, VRML also describes the leaf nodes using the following syntax:

- *Shape Node:* A node including the geometry.
- *Background Node:* It defines the background of the scene.
- *Sound Node:* It is for including the sounds in the 3D scene.

The VRUI Toolkit

The Vrui (Virtual Reality User Interface) toolkit was originally developed by Oliver Kreylos at the University of California, Davis [102]. Using the Vrui toolkit, one can design and implement scalable and portable applications that can be run on different virtual reality (VR) environments, from a normal desktop environment with custom input device such as space balls to cover immersive environments ranging from large single screen to multi-screens like TiledWall or full immersive platform like CAVE systems. One of the key features of Vrui is its ability to accept any input device, as it enables developers to map the available input devices to their applications using simple configuration files [102].

The richness and the portability of the Vrui toolkit enabled us to design and implement our ESSAVis platform in a portable fashion, which we will discuss in the forthcoming chapters. Therefore, it can run on different platforms starting from a single screen as in the case in laptop or desktop to large screen platforms as in the case of TiledWall, PowerWall, or CAVE systems. Moreover, different input devices can be mapped with our ESSAVis framework such as the normal mouse pointer, the keyboard keys, or in some cases the tablet devices.

3D Model Visualization

Exploring 3D models of mechanical systems have also been studied in many previous works. For example, producing an exploded view of the system model has been explored using different forms like in 3D-CAD or volumetric data [43] and in 2D image data [110]. Other works (e.g., [4], [109], and [145]) used spatial blocking information and the size of model parts in order to calculate the relations and directions of system parts. However, all these previous works focused on exploring 3D models just from the structural point of view.

3

The ESSAVis Framework Overview

The idea of providing **ESSAVis** (**E**mbedded **S**ystems **S**afety **A**spect **Vis**ualizer) framework for the purpose of visualizing the structural and behavioral aspects of embedded systems is the core and the novel contribution of this thesis. The challenge was to overcome the high complexity of this kind of systems, which mainly comes from the heterogeneity nature of embedded systems. Therefore, many other directions were emerged from this core point.

We came up with the solution of visualizing different targeted data sets, where each data set covered one aspect of the system. For this, we provided the 2D **LayMan** framework (see Chapter 5) as a novel compound graph layout framework to be a sub-part of ESSAVis targeting to visualize the behavioral aspects of the underlying system. We went a further step towards the compound graph navigation, when we proposed the **CluE** algorithm (see Chapter 4) as a novel general algorithm for expanding or contracting the compound nodes in compound graphs. We covered the visualization of the hardware side of embedded systems using the 3D visualization techniques (see Chapter 7). To achieve this goal, we designed and developed a novel **3DintEx** tool.

Then we started thinking on the possible ways to integrate between the 2D abstract view of the system behavior and the 3D model view of the system structure. Therefore, we designed and developed the **SceneMan** unit (see Chapter 7) that integrates between these two views seamlessly. The resulted platform was a 2D*plus*3D visual environment. All these developed units and tools are integrated tightly and seamlessly in our ESSAVis framework in order to provide an overall general framework for visualizing the structural and behavioral aspects of embedded systems

This developed 2D*plus*3D visual environment in ESSAVis framework is portable with different 3D platforms, which gives us the ability to utilize the stereoscopic depth offered by the 3D technology to encode some aspects of node-link diagrams. We investigated the stereoscopic depth technique intensively and extended it in different directions to enrich our ESSAVis framework (see Chapter 6). This helped us for a better integration between the 2D abstract view and the 3D system model view that are rendered in a 3D world.

The goal of this chapter is to introduce the overall image of the ESSAVis framework. First, we present the story of the designing process of ESSAVis that we followed to reach to the final design of the framework. Then we introduce briefly the main features of the ESSAVis framework together with the list of main units and tools inside it. Each of these units and tools are described in detail in the later chapters of the thesis.

3.1 The Designing Process

Before we start to motivate the design decisions of our developed ESSAVis framework, it is important to mention that this work was done in the context of the **Vier4ES** [1] project in the project's second phase. The main goal of Vier4ES project was to visualize the safety aspects of embedded systems in virtual realty technologies.

Mainly, we dealt with two groups of end users, the *safety experts* and the *system engineers*, where each group had their own expectations about the possible solution of this problem. Most of the times, in our work, the safety experts belong to the **Software Engineering: Dependability** group at the University of Kaiserslautern while the system engineers belong to the **Robotics Research Lab** in the same university.

The first step in this work was to understand the questions needed to be answered. After several meetings with the both groups (i.e., the safety experts and the system engineers), we came up with the following users requirements:

- *What are the critical system artifacts and how do they contribute to a system failure?*
- *How severe is their influence in terms of probability of occurrence?*

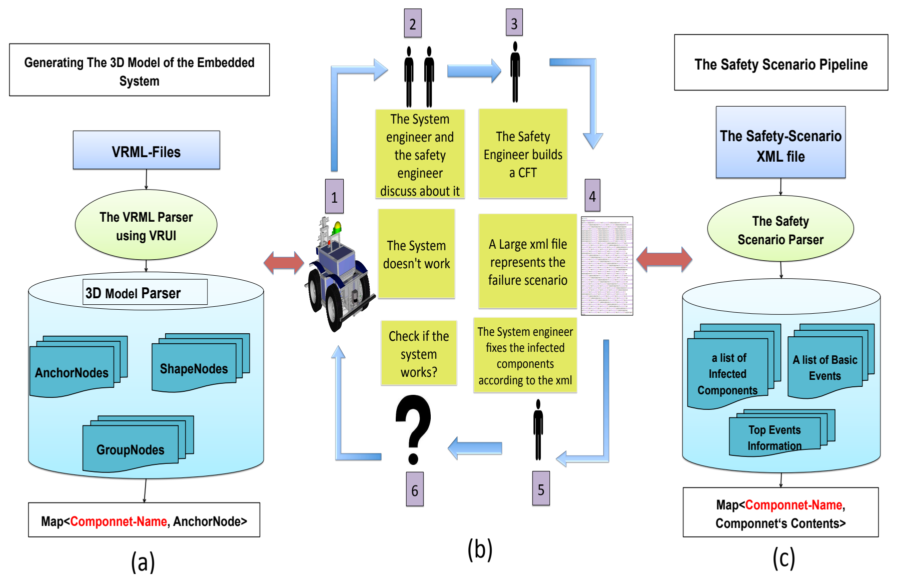

Generating The 3D Model of the Embedded System
VRML-Files
The VRML Parser using VRUI
3D Model Parser
AnchorNodes
ShapeNodes
GroupNodes
Map<Componnet-Name, AnchorNode>
(a)
1
2
3
4
5
6
The System engineer and the safety engineer discuss about it
The Safety Engineer builds a CFT
The System doesn't work
A Large xml file represents the failure scenario
Check if the system works?
The System engineer fixes the infected components according to the xml
?
(b)
The Safety Scenario Pipeline
The Safety-Scenario XML file
The Safety Scenario Parser
a list of Infected Components
A list of Basic Events
Top Events Information
Map<Componnet-Name, Componnet's Contents>
(c)

Fig. 3.1: (a) Generating the 3D model pipeline. (b) The safety cycle analysis. (c) The safety scenario pipeline where the two red arrows indicate connections between the safety cycle and the two pipelines

- *Which artifacts influence each other and how strong are those influences?*
- *Do the causes occur within the software or the hardware components or in both?*

Figure 3.1(c) shows the work flow – that we observed during the data collection time – between the different end users in our application domain. Safety experts normally work with the Component Fault Tree (CFT) model (as shown in Figure 2.1), which is a 2D representation of the failure scenario in the system at a certain time. They demanded us to provide them a 2D representation of the Fault Tree (FT) model, alongside the visual cues of important safety information embedded in this 2D representation. However, the system engineers' demands were completely different. Normally, they deal with the hardware model of the system; hence, they care more about the structural information of the system components. More precisely, they demanded us to map the safety information with the infected component(s) in the 3D model as well as to provide the interaction facility with the system 3D model directly.

For our case study, we used safety scenarios of the RAVON robot [124], developed in the Robotics Research Lab of the University of Kaiserslautern, that can detect obstacles on the road. In this case, the safety experts were responsible to provide the safety scenarios of the RAVON according to the system engineers' feedback. We concluded from the conducted several meetings with these both end user groups that an integrated visual solution (i.e., a 2D*plus*3D visual interactive platform) would be a better solution for both groups. Our idea was to provide both groups' perspectives in one visual platform that would also give them the ability to collaborate with each other in a more natural way.

This suggested solution solves not only the conflict between these both end users' perspectives, but also it shows the complex nature of embedded systems as such visual platform uses different data sets' types. For example, the 3D geometry information about the system hardware model or the graph representation of the software system attached to the hardware components, which is normally a 2D representation. In addition to deal with these data sets, we also decided to provide the system safety model in a 2D visual representation, as safety experts are used to use this kind of representations in their daily-life working routines. Therefore, we believed that a combined approach using the 2D and 3D data is a reasonable solution for such application domains. However, this combined approach was not decided immediately, as we went through different stages to come up with this final solution.

Ideas Generating

Here, we discuss briefly the ideas generation of our developed ESSAVis interactive visual framework. In ESSAVis, we proposed a novel approach to integrate between the fault trees and the underlying embedded system component parts in one visual environment. The goal was to give more insight about the underlying embedded system status such that both the safety experts and the system engineers can collaborate together in an efficient way.

Targeting this goal, the first step was to provide a pure 3D representation, integrating both the safety information and the system information (as shown in Figure 3.2). This pure 3D representation gives the system engineers the ability to find out the infected component(s) easier and faster, as it highlights them visually in an intuitive way and gives a textual description about them directly.

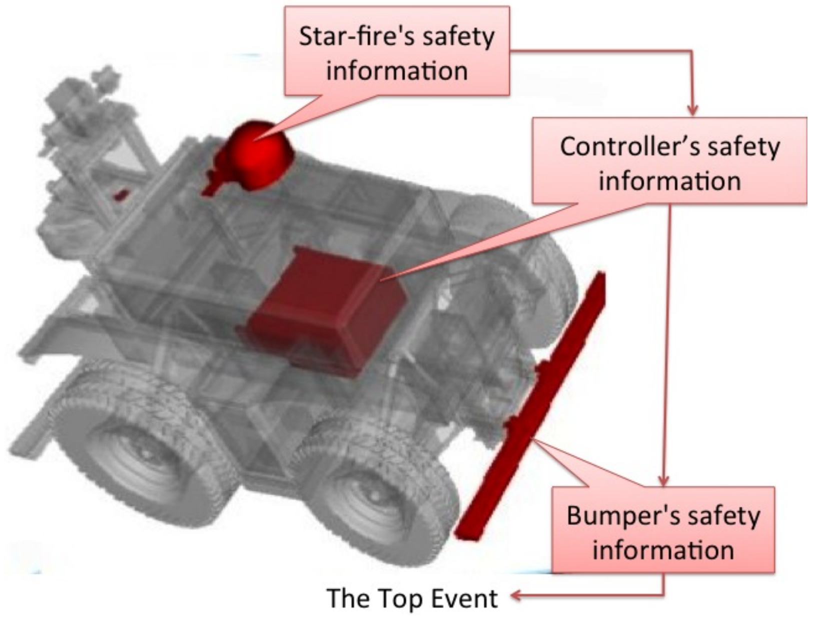

Star-fire's safety information
Controller's safety information
Bumper's safety information
The Top Event

Fig. 3.2: The first idea model, where the safety information embedded with the 3D model of RAVON.

However, this solution does not provide the safety experts what they expect, as it does not provide them the relevant safety information and the failure relationships between the components. Further, it does not show the failure path between the system components, as it is hard to visualize this kind of information in a 3D model due to the cluttering problem that might appear. Targeting this concern, we arranged focus group meetings with both groups to discuss the possible solution that would satisfy the both groups. Finally, we came up with a solution of integrated interactive visual views, which was about providing another view to show the abstract information of the failure path between the system components, side-by-side with the first view.

The proposed integrated interactive visual views solution offers two views, as shown in Figure 3.3, where the first one provides the system 3D model while the second one provides an abstract representation of the failure scenario in the system life at a specific time. This solution

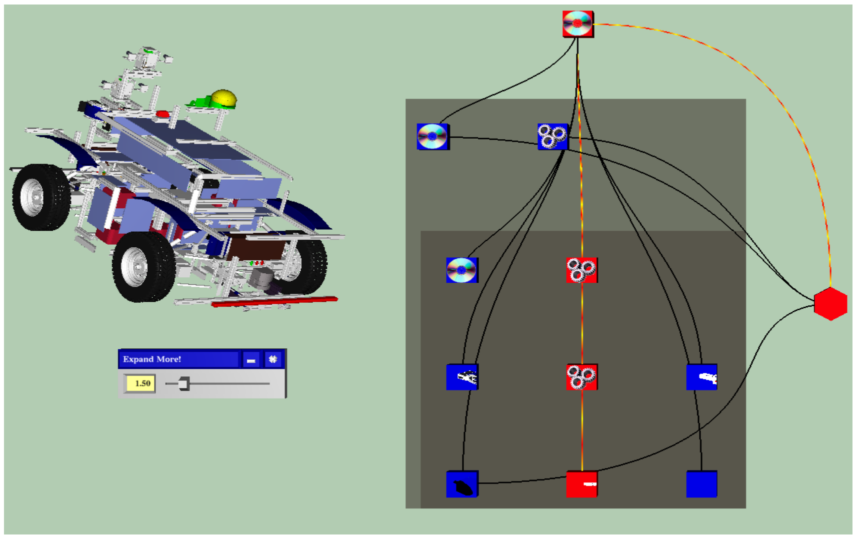


Fig. 3.3: A snapshot showing ESSAVis two main views side-by-side: the *abstract graph representation* view in the right-side of the figure, and the *system 3D model* view in the left-side of the figure.

offers the possibility to switch between the two views on users' demand. Also, it gives both groups the ability to collaborate through a common visual platform to achieve the required tasks together. After we finalized the solution with both end-user groups, we specified the work steps and the set of goals for the proposed 2D*plus*3D integrated visual environment.

3.2 The ESSAVis Framework

We present here briefly an overview of our ESSAVis framework and its capabilities in supporting the analysis of safety aspects of embedded systems through interactive visual support.

The key contribution of ESSAVis is its ability of accepting different data sets, and then integrating between them seamlessly. This principle is shown in Figure 3.4. After integrating the input data sets in different visual representations, ESSAVis allows users to interact with it through a number of interaction facilities. The framework provides two main views, i.e., the *abstract graph representation* view and the *system 3D model* view (see Figure 3.3), and combines them seamlessly for describing the safety status of the underlying embedded system.

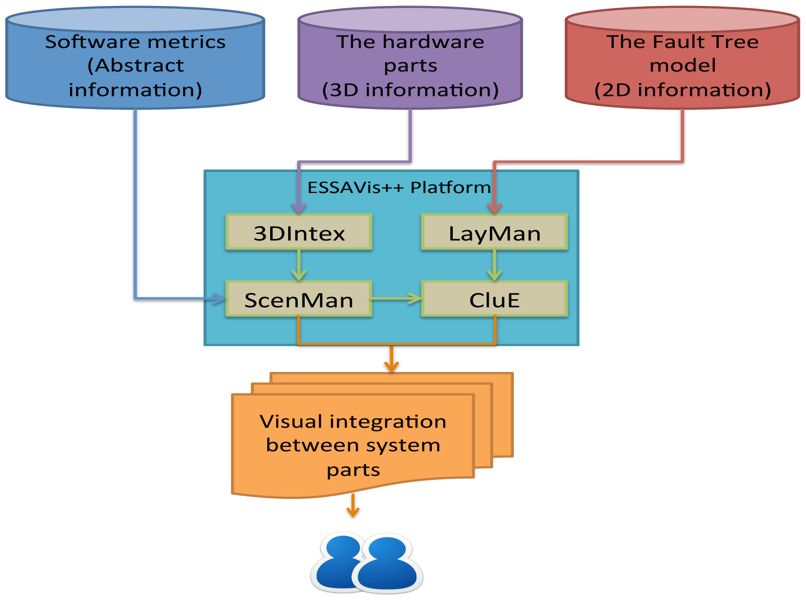


Fig. 3.4: An overview of the possible data sets, acceptable by ESSAVis. Different data sets are shown on the topside, while the middle part shows ESSAVis units that process these data sets and collaborate with each other for visualizing the safety status and system 3D model to the users

Here, each view (i.e., the abstract graph representation view or the system 3D model view) represents the safety status from a different user group's perspective, e.g., system engineers feel more interest towards the system 3D model view while safety experts feels more interest towards the abstract graph representation view. The integration between these two views allows them to understand the whole safety situation of the system, not only just from their own perspective, more easily and accurately.

Compared to traditional safety analysis tools, we added many new interaction techniques and visualizations in order to allow users to extract more information about the safety status of the underlying system. For this, ESSAVis integrates tightly and seamlessly a set of new tools and units for providing the desired visual support and the interaction options.

Additionally, ESSAVis was designed to be portable with different 3D technologies (e.g., 3D desktop displays, Power-Wall displays, and CAVE systems). This feature allowed us to think in the direction of utilizing the stereoscopic depth offered by these technologies. In this regards, we used the stereoscopic depth as a visual cue to encode some aspects of the nodes in the graph representation of the CFT model. Adding, the stereoscopic depth cue saves other visual cues (e.g., color, shape, or texture) to encode other aspects of data. We added this feature in ESSAVis to be an on-demand option and it works well when ESSAVis is rendered in a stereoscopic display.

In order to support a variety of interaction techniques, ESSAVis was built using different units and tools where each one is responsible for some specific functionality. These units that accept and process the different data sets in order to provide the required visual support, the different interaction techniques, and the integration between the two views in a 3D platform. In the following, we briefly focus on these different units of ESSAVis platform to show the visual support and the interaction options provided by them.

3.2.1 The Cluster Expander Unit

ESSAVis uses its **CluE** (**Clu**ster **E**xpander) unit to help the users in navigating through the graph representation of the failure mechanism. In this regard, the unit uses its algorithm, also called the CluE algorithm, in order to provide the facility to users to expand or contract any of the compound nodes on demand. The selected node representing a system component is then expanded to show its internal structure, in addition to the failure relations between the sub-components inside it. This helps in navigating the failure mechanism structure in a way reflecting both, the structural and the failure relations in the graph view. This leads to keep the overall view of the graph and to show extra level of details on demand. This facility also helps the users in reserving the context information about the failure structure. Chapter 4 provides the detailed description about this unit.

3.2.2 The Layout Manager Unit

ESSAVis provides an abstract representation of the underlying safety scenario through one of its component, called the **LayMan** (**Lay**out **Man**ager) unit. This unit first parses the safety scenario modeled as a CFT model [93] and then arranges it on the designed data structure. After this step, it converts the read data into a compound graph model where the system components are the graph nodes, the failure relations between these nodes are the adjacency relations in the graph model, and the structural relations between these nodes are the structural edges in the graph model. After building the graph model, this component uses one of the layout algorithms to calculate the graphical representation of the graph. LayMan component uses different layout algorithms (e.g., the orthogonal layout algorithm, the Sugiyama layout algorithm, the force-directed layout algorithm, etc.) for the final graphical representation. It uses the OGDF framework [148] for calculating the final graph representation. Further, it uses the CluE unit to iteratively visualize the different levels of details in the underlying compound graph. Chapter 5 provides the detailed description about this unit.

3.2.3 The 3D Interactive Explorer Unit

ESSAVis provide the **3DintEx** (**3D int**eractive **Ex**plorer) unit in order to render the system 3D model. This component parses the geometry of the CAD model and then it renders it in the 3D world. This unit is also responsible for reducing the complexity of the CAD model in order to accelerate the rendering and the interaction steps. Additionally, it provides a number of interaction options for picking and selecting one or more components of the underlying system 3D model. Moreover, it provides the facility to expand the 3D model on demand using the explosion-slider interaction facility. This option is triggered intuitively such that users can control the distance between the components in the CAD model on demand. This is especially useful for the system engineers in picking one of the inside component to find further details about the picked component. The 3DintEx unit was built using the Vrui framework [102]. Chapter 7 provides the detailed description about this unit.

3.2.4 The Scene Manager Unit

ESSAVis provides two main views, the system 3D model and the abstract graph representation view of the safety scenario. In order to synchronize between these two views, ESSAVis uses one of its unit, called the **SceneMan** (**Scene Man**ager), for connecting between them in such a way that if the user selects one node in the graph representation then the corresponding component in the system 3D model is also highlighted. This helps users in connecting between the two views such that on one hand they can find the location of the critical component from the abstract view to the actual system 3D model on demand, while on the other hand they can find the failure relations of the selected component in the 3D model using the abstract view. This component is also responsible for aligning the two views in the 3D space without extra cluttering. Moreover, it provides different options to allow users to change the configuration of the scene on demand. Chapter 7 provides the detailed description about this unit.The SceneMan unit also provides the facility of visualizing the software metric information embedded in the graph view of the failure mechanism (see Figure 3.5). This option helps system engineers and safety experts to highlight those software components that might be in a critical safety situation.

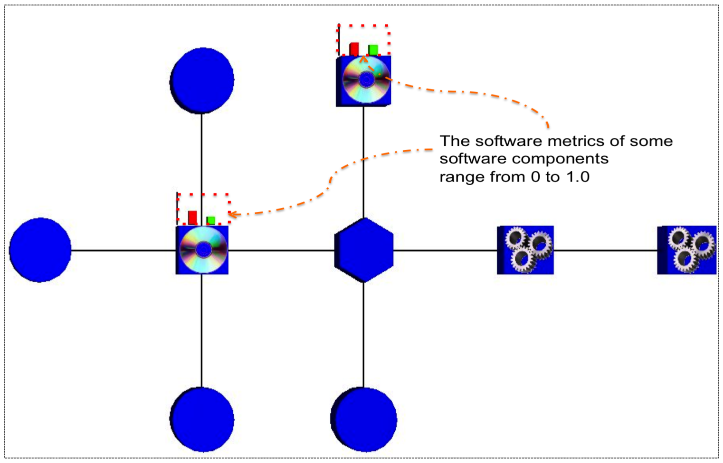


Fig. 3.5: Visualizing the software metrics on demand.

3.2.5 The Textual Information Unit

In order to extract information about any picked component from any of the two provided views, ESSAVis uses its **TextInfo** (**Text**ual **Info**rmation) unit to provide a dialog box option to show the safety information of the corresponding node on demand. The displayed information includes the criticality value of the selected node, the parent node, and the node's name. This facility is triggered in two ways either by hovering the mouse over the selected nodes or by selecting the "Show Textual information" option from the main menu option.

4

Navigating Compound Graphs

In this chapter, we present an algorithm, called ***CluE*** *(The* ***Clu****ster* ***E****xpander of Compound Graphs), that expands cluster nodes in compound graphs. CluE was designed to work with multiple layout algorithms (e.g., the orthogonal layout algorithm [72] or the Sugyima layout algorithm [143]). It keeps the un-expanded nodes in their relative layers with slight changes in their original positions. This helps the viewers in maintaining their mental map of the underlying graph. We have successfully applied the CluE algorithm for navigating through compound graphs representing the failure mechanisms in embedded systems. We also performed a brief evaluation study in order to measure the viewers' acceptance level with regard to the application of CluE for compound graphs. Results of the study show a high acceptance ratio from participants having different backgrounds, which indicates the intuitiveness and feasibility of our approach.*

Generally, creating abstract representations of large data sets for limited display sizes is considered to be one of the challenging tasks. To handle this challenge, one of the solutions is to producing a graphical representation of the containment relationships between the graph nodes. This is one of the common ways to tackle this kind of challenging task in the state-of-the-art literature [68, 97].

Normally, diagrams of such abstract representations offer a hierarchical structure of the graph components, where nodes may contain other nodes and/or edges. These kinds of abstract representations are popular in many application domains [68] (e.g., UML diagrams of software

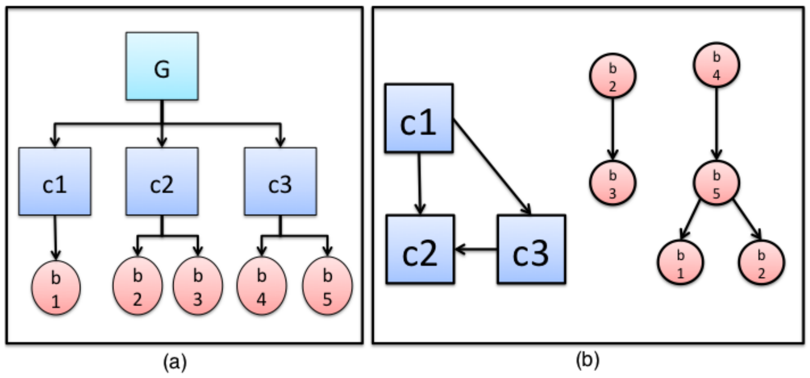


Fig. 4.1: The original graph, showing two types of relations between the graph nodes: *(a)* the structural relations, and *(b)* the adjacency relations.

systems, realizing the relations between the molecules in bioinformatics applications, etc.). Besides keeping the adjacency relations between the graph nodes, one important factor in these kinds of abstract representations is that they also show at the same time the structural relations between the graph nodes. In such cases, this phenomena is called *hierarchy crossing edges*, especially when we talk about the *compound graphs* [97].

Hierarchical graph structures or compound graphs are widely used in information visualization due to their popularity in many application domains [126, 141]. Normally, they are common in those application domains where grouping of data elements, which belong to the same cluster or share the same attribute, is a natural part of the underlying data. This grouping process helps in producing an abstract representation of the underlying graphs, called as *clustered graphs* (see Figure 4.1 and Figure 4.2).

These clustered graphs' representations offer many possibilities for trained users in understanding complex diagrams. For example, it speeds up the understanding process of the software systems architecture as it provides a logical grouping for the packages and classes. However, exploring these complex diagrams is considered a tedious task in the lack of proper interaction mechanisms. Therefore, it is important to provide an interactive navigation mechanism for exploring the clustered nodes in such representations, in order to get more insight about the adjacency relations between the graph nodes at different levels

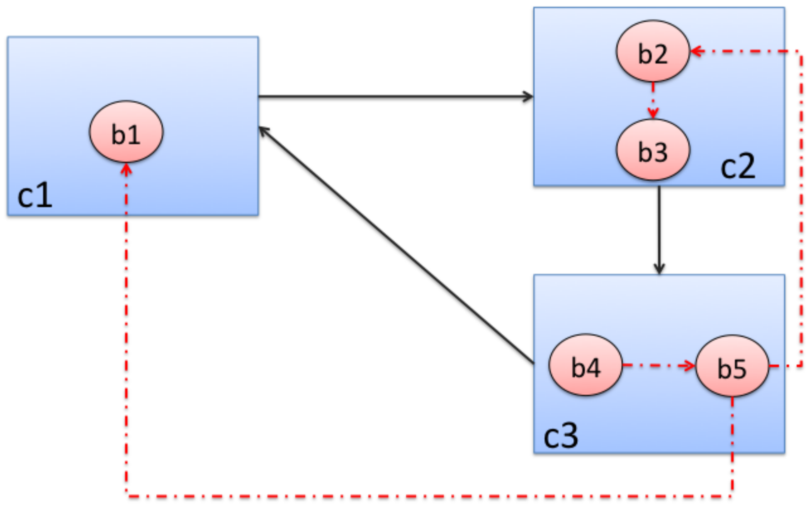


Fig. 4.2: The compound graph representation of the graph presented in Figure 4.1.

of details in the graph. Expanding or contracting cluster nodes is one of the traditional mechanisms to explore such structures for increasing the comprehension level of the data.

In this chapter, we present a novel algorithm, called **CluE** (The **Clu**ster **E**xpander of Compound Graphs), that aims at expanding the compound nodes in clustered graphs. The CluE algorithm works with any hierarchical layout algorithm (e.g., the orthogonal layout algorithm [72], the Sugiyama algorithm [143], the classical tree layout [128], etc.). The CluE algorithm offers an intuitive approach for maintaining the viewer's mental-map for compound graphs. Also, it takes care of the bounded region of the underlying graph. Therefore, it produces an area-aware layout of the expanded cluster based on the number of children of that cluster [25]. All these features of the CluE algorithm facilitate the graph exploration and navigation in order to get more insight about the graph structure on-demand.

4.1 Motivation Behind the CluE Algorithm

The clustering and interactive concept of large graphs has been described in many previous works. Feng et al. introduced the clustered graph model in [63], which is considered one of the first work in this area. They investigated how the graph model can be drawn with minimum edges crossings or without edge-regions crossings. Also, they introduced efficient algorithms for testing C-planarity and for detecting C-planer embedding with this kind of graphs.

Another important work in this domain was introduced by Eades et al. in [61]. Their proposed approach allows exploring and navigating large graphs based on the following order: first producing a hierarchy of the graph using the clustering graph algorithm, then making layout the current level of detail using the force directed layout algorithm, and finally providing animation to transform one graph view to another one. In [83], Huang and Eades described the DA-TU system that combines animated clusters with online force-directed animated graphs for visualizing large graphs.

Another example is ASK-GraphView system, proposed by Abello et al. in [3], that allows clustering and navigating large node-links diagrams. Their proposed solution builds a hierarchy based on arbitrary weighted large graphs. Moreover, it suggests the navigation through top-down fashion in order to expand clusters on-demand. Archambault et al. proposed GrouseFlox approach in [26] that allows viewers interactively exploring large graphs using a cut operation of a superimposed hierarchy. This proposed approach creates hierarchy using a topological feature-based algorithm or any specific clustering algorithm. This solution was designed for exploring a graph hierarchy space. This was achieved through allowing viewers to explore different possible hierarchies for the same graph.

In a CFT model: the set of vertices V contains the list of components, the set of inclusion edges E contains the structural relations between these components, while the set of adjacency edges F contains the failure relations between the system elements. Moreover, a component may contain other components or leaves, where leaves in this context are the basic events in the underlying system. As the complexity of an embedded system is increased the corresponding CFT model size is also increased, which makes it difficult to handle the failure detection process. The information visualization field can play an important role in speeding up the system developing process, because it eases the steps of finding important information from both the system and the user perspectives.

However, as the size of the graph is increased, the traditional layout algorithm fails in providing a readable visualization. Therefore, there is a need of abstracting the graph to reduce the visual cluttering and to enhance the information extracting process. In addition to the abstraction step, expanding and contraction operations over the produced cluster are required in order to navigate the hierarchy and to get more insight about the deeper level of details. Our CluE algorithm expands the compound nodes in the CFT model in order to help the users (e.g., safety experts or system engineers in our case) for getting a better understanding of the underlying system's failure mechanism. Moreover, we provide a pleasant animation for showing the changing process of graph, which is useful for maintaining the user's mental map of the graph.

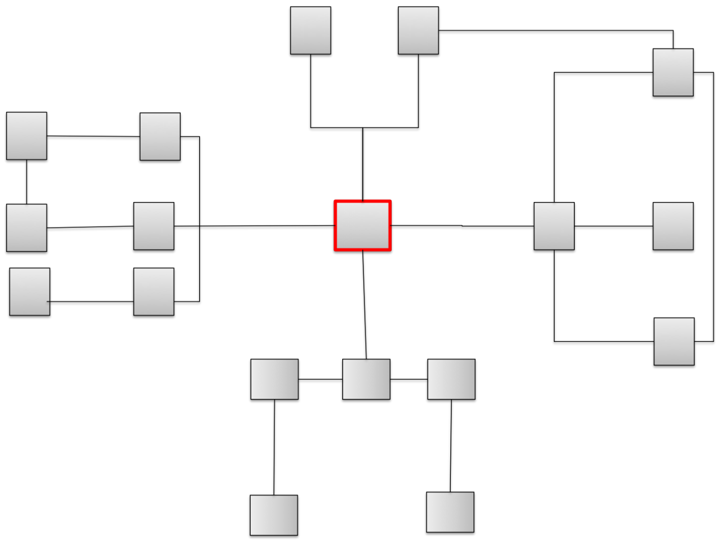


Fig. 4.3: The original graph, showing the original nodes in the graph where all nodes are clustered as they contain other graph elements.

In Figure 4.3, we show an example of original graph where all nodes have the same level of details. In this case, all nodes are clustered as they consist of other graph elements (i.e., like in our case each component may consist of a set of other components together with a set of basic events). In this figure, we show an example of the expected behavior of the graph upon expanding the highlighted node. Once a

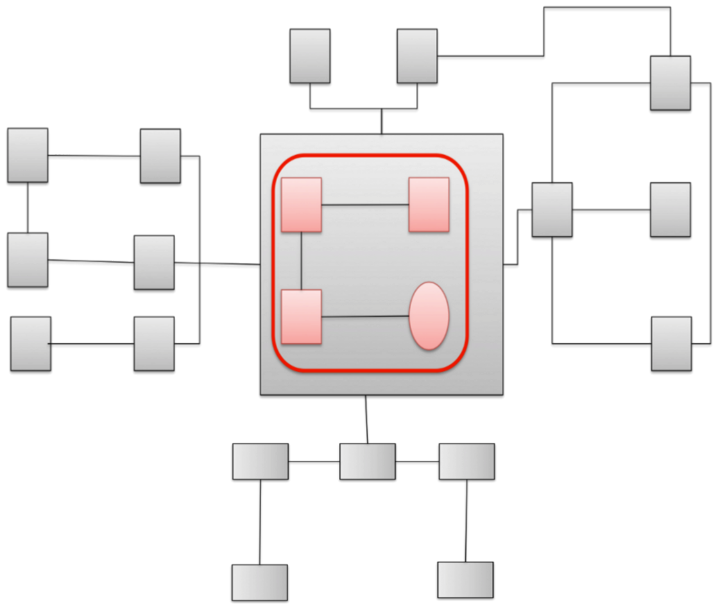


Fig. 4.4: All the non-expanded components are shifted according to the new selected component's size in all directions based on their initial positions to the selected component.

node is selected to be expand, CluE calculates the number of its internal elements and the number of edges to be added to the current view. This step is required as it helps in estimating the new size of the expanded node. The new node's size is saved for the next step in the algorithm. In Figure 4.4, we show the graph behavior after expanding the selected node. This expanded node's size is used to help the algorithm to place the graph nodes in their new locations. All none-expanded nodes are shifted in all directions to give the chance of the new elements to be layout.

4.2 The CluE Algorithm

In this section, we explain the working of our **CluE** (The **Clu**ster **E**xpander of Compound Graphs) algorithm that aims at expanding the compound nodes in clustered graphs. Algorithm 1 provides the pseudo code of the CluE algorithm. We assume that the information about the underlying graph's vertices and edges is available, in a pre-stage before

```
Procedure CluE
Input: All nodes and edges with their current positions and sizes, and the current node N
Output: New nodes' position and sizes
NL = Empty node list
EL = Empty edge list
for all child nodes C of the current node N do
    Add C to the node list NL;
    if C is expandable then
        Apply algorithm to C (i.e., to its all child nodes);
        Set size of C to the minimal bounding box of its children;
        Save this bounding box;
    end
end
for all edges E do
    if E is connected to any two children of the current node N then
        Add E to the edge list EL;
    end
end
Apply layout algorithm to the node list NL and the edge list EL
for all child nodes C of the current node N do
    if C is expandable then
        Move children of C by the difference between the saved bounding box of C and
        the current position (which is the result of the layout algorithm);
    end
end
```

Algorithm 1: The CluE pseudo code.

the execution of CluE. However, CluE cares only about the current node in the graph that is responsible about the current level of details. CluE takes this information and then it calculates the required size of the current level of details based on the number of children of the current node. In the beginning of execution, CluE takes the graph root and its children for calculating the required size. Then CluE calls the layout algorithm in order to calculate the coordinates of child nodes in the current level of details. CluE is called recursively based on the viewer's request. It is called when a node is triggered by the viewer to be expanded. After this, CluE performs the following steps:

CluE starts from a specified node N in the graph, which represents the current node to be expanded or the graph root in the case of initial stage. Then it finds all the child nodes of this node. First, it adds the child nodes to a list NL (Node List), that is empty at the initial stage. All the nodes in this list represent the current level of details and are connected with each other via the set of structural relations of F edges.

It also keeps the adjacency relations between these nodes in a list data structure *EL* (Edge-List), that is initially empty. CluE works iteratively, as it layouts the graph nodes level by level. It checks if one of the child nodes in *N* is needed to be expanded, then it calls itself to add the child nodes of this targeted node in the node list *NL*. After that, it estimates the size of this expanded node based on the number of its children and then it saves this size for later usage. At this stage, CluE adds the adjacent relations related with the internal structure of the expanded node to the edge list *EL*.

In the next step, CluE calls the layout algorithm. In fact, CluE does not rely on any specific layout algorithm, as it requires only the coordinates of nodes in the graph. Therefore, it works perfectly with many different layout algorithms (e.g., the orthogonal layout, the Sugiyama algorithm, the grid layout algorithm, or any multi-level layout algorithm). We have successfully tested CluE with all these mentioned layout algorithms and it performed well in all cases. After this step, CluE takes the coordinates of graph nodes and then it shifts these nodes in two directions according to the size of the bounding box of the expanded node, which was saved in the previous stage of CluE.

The main limitation of CluE is that it calls the layout algorithm whenever there is a need to expand a new node. This requires calculating new positions each time. However, CluE keeps fixed the layer of the nodes. This affects on maintaining the viewer's mental-map of the layout [79]. In order to help viewers in tracking the nodes at previous positions, we added a slow animation to show the slight changing in nodes' positions.

4.3 Results

In this section, we discuss results of our CluE algorithm when it is called on the abstract representation of a CFT model. Figure 4.5 shows the most abstracted level of a CFT model. In this figure, the presented graph shows those direct components that trigger the critical safety situation in the underlying system. The root of this model represents a critical safety situation, which is called the "TopEvent" in the safety analysis domain.

As a part of the safety analysis process, it is important to find those safety critical components that could lead to this unpleasant system

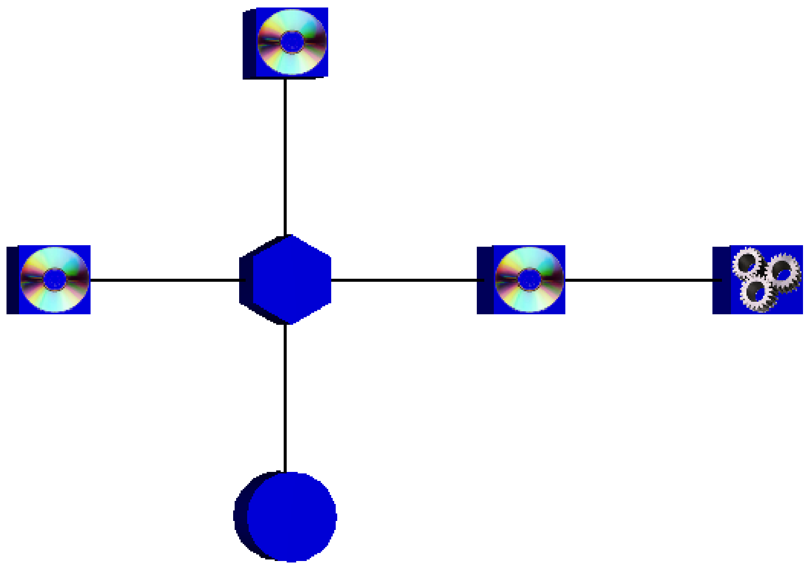


Fig. 4.5: The highest level of details of an CFT model without expanding any node.

situation. Therefore, CluE offers the facility to expand the targeted component and allows the viewers to go deeper in the graph hierarchy for showing the in-depth details on demand. For example in Figure 4.5, we show the highest level of details in the current CFT model for describing one possible safety situation in the underlying system. In this figure, the Hexagon represents the TopEvent, the blue circles represent the Basic Events (i.e., the simplest structure in a CFT model for representing leaves of the graph), the blue rectangles represent the system components, and the texture on each rectangle represents the type of this component (i.e., either software or hardware). As described earlier, a component may contain many sub-components or Basic Events; therefore, these structures are expandable.

When a viewer expands the targeted component a new view of the graph is displayed, which shows another level of details by providing the internal structure of this targeted component (see Figure 4.6).

Figure 4.7 shows the visual representation of structural relations between the different components, which are visualized implicitly using the containment relation representation such that the parent node contains the child nodes. Moreover, after expanding, extra failure relations that were not included in the previous view are then displayed. These

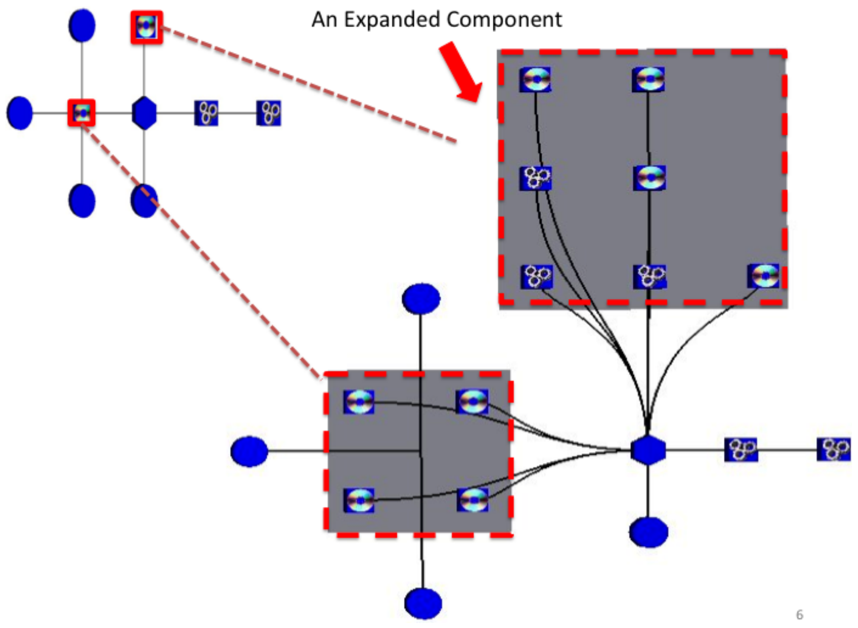


Fig. 4.6: The process of expanding the required nodes using the CluE algorithm. The abstract view is shown at the top left side of the figure with smaller size, while the expanded view is shown at the center of the figure.

failure relations are appeared on the visual representation, because CluE adds the adjacency relations upon their presence in the graph during the expanding process.

CluE scales well with different number of levels in the graph hierarchy, as we show in Figure 4.7 the three levels of details of the graph after expansion. We designed the visualization in such a way that the color saturation of the node depends on its level, e.g., a higher level of details looks more transparent than the lower level of details. This difference in color saturation helps in defining the structural relations between the components at different levels. We also use the spline curves for showing the adjacent relations between nodes at different level of details. This design decision was made because it helps in reducing the crossings rate between the adjacent edges [82].

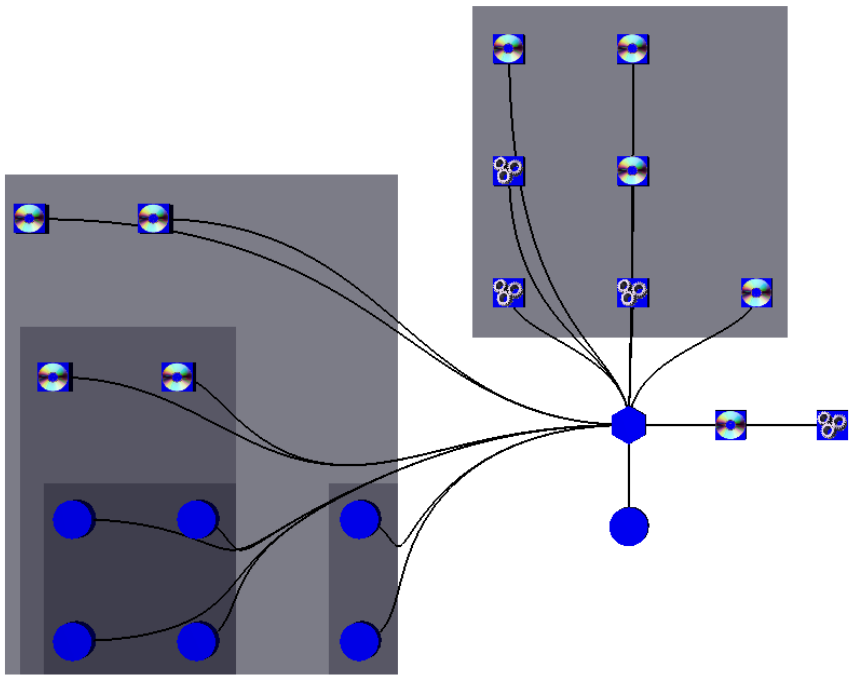


Fig. 4.7: An illustration of the graph representation after expanding three compound nodes. The color saturation is used to show the parent-child relationships between the components and their sub-components, while the failure relations are visualized using explicit edges between the graph nodes.

4.4 Expanding Options

In our implementation of the CluE algorithm, we allow viewers to interact with the underlying compound graph view through different interaction techniques in order to expand at different levels of details. The result is a set of options useful for viewers during navigation through compound graphs. Here, we provide brief details of these expanding options:

- *Expanding an individual node using the Mouse + the '2' button on the keyboard:*
  For this option, viewers are encouraged to hover the mouse over the targeted node in the current view and then press on the '2' button on the keyboard. The consequences of this action are the following: first CluE invokes the function to find the number of children of the selected node in order to calculate the size of this node after

expansion. Then it adds these children to the current view together with their adjacency relations. After performing this step, CluE invokes the layout algorithm to calculate the positions of the new nodes. After this step, CluE shifts all nodes in the current level of details based on the size value of the expanded node that was calculated in the initial step of CluE.

- *Expanding the selected node using the safety information menu:* In this option, Clue allows the viewers to select one node from the 'safety information menu' that includes all information about the current safety scenario. In this option, viewers can select any CFT element, either a compound node or a basic event node. For this, viewers are required to select one node from the menu and then either double click it using the mouse interaction technique or click of the 'Expand To Show' button in the menu. Upon selection, CluE calculates the path between the selected node and the current level of detail. This step is required for measuring the number of nodes that are needed to be expanded recursively. This is important to estimate the size of the nodes that will be expanded over the path. We designed this option to help in navigating the compound clustered graphs on demand. After calculating the positions of all nodes in the current view, CluE shifts all the previous nodes from the previous view based on the new sizes of the expanded nodes over the path between the selected node and the root of the graph.
- *Expanding all nodes in the graph:* Sometimes, viewers are interested in viewing the overall view of the graph. It is time consuming to expand nodes individually, especially when the graph size is large. Therefore, we provided an option to allow expanding the whole graph. This expansion starts from the least level of details in the graph, in order to allow CluE calculating the size of nodes after expansion recursively. The visualization of the expansion is performed using the top-bottom fashion. However, the calculations for the actual sizes and positions are performed in a bottom-up fashion. To help viewers in keeping the mental map of the changes that occur in the graph, we use slow animations to help in tracking the changes in the view. An example of the whole graph expansion is shown in Figure 4.8.

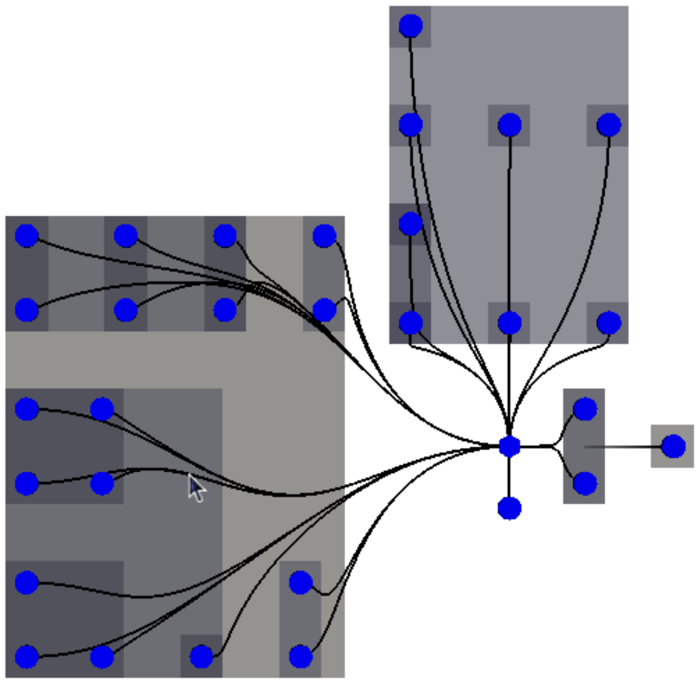


Fig. 4.8: An example of the graph after the expansion of all nodes. The color graduation shows the different levels of details of the graph.

4.5 The Evaluation Study

In order to evaluate the expanding and contracting facility through our CluE algorithm, we conducted a brief evaluation study. We invited 25 participants from different backgrounds for performing exploration tasks over the graph representation of a CFT model. The tasks mainly were focused on extracting some safety information about the corresponding components at different levels.

First, we trained each participant for using the developed tool that incorporates CluE algorithm. Then we provided the required background they needed to achieve the given tasks. We explained them the tasks and then asked them to provide us their feedback. After it, we also asked the participants to give us their feedback through a closed-ended questionnaires form. In this closed-ended questionnaire form, participants were asked to select a number between 1 and 5 (1 meant strongly disagree while 5 meant strongly agree) based on the likert scale to rate their performance.

In the feedback, 21 participants found the expanding and contracting node operation using the CluE algorithm intuitive and easy to follow. Their rating of their own performance was either 4 (i.e., *agree*) or 5 (i.e., *strongly agree*). While the remaining 4 subjects rated themselves either 2 (i.e., *disagree*) or 3 (i.e., *neither agree nor disagree*), as they found some difficulties in tracing the transitions of nodes. Regarding the trust level of the developed tool and algorithm, 19 participants assigned their trust level to be either 4 or 5, while the remaining 6 subjects requested more time in order to judge their trust level. Overall, the results of this evaluation study show the intuitiveness of our approach and indicate a high acceptance ratio from the participants' perspective. However, there is a need of extended detailed evaluation studies to generalize the findings.

4.6 Summary

In this chapter, we presented the CluE algorithm that expands cluster nodes in compound graphs. The main contribution of this algorithm is its ability of expanding the cluster nodes regardless of the used layout algorithm in the background for different graph sizes. It produces an area-aware layout, as it takes care of the size of the expanded node before arranging the child nodes in their final positions. Additionally, it keeps the expanded nodes in their corresponding layers. However, a slight difference in the expanded nodes' coordinates might occur.

In the implementation, we added a slow animation for showing gradually the graph nodes' transition process in order to help the viewers in maintaining their mental-map of the underlying graph. We have successfully applied the CluE algorithm in real graphs for navigating through compound graphs representing the failure mechanisms in embedded systems (see Chapter 5). The results of the brief evaluation study show a high acceptance ratio among the participants in the study, as they were able to distinguish between the structural relations and the failure relations between the nodes at different hierarchy levels. We also tested the algorithm in large graph sizes using different layout algorithm like the Sugiyama algorithm and the orthogonal layout algorithm. The algorithm performed very well in both cases; however, we observed slight changes in the edge-crossing ratios in these two mentioned layout algorithms.

The work in this chapter was applied in the context of navigating thorough the compound structures of component fault tree (CFT) models, which are used to model the safety mechanism of embedded systems. In the forthcoming chapter (i.e., Chapter 5), we show how we applied successfully this algorithm in the domain of safety analysis in embedded systems. In Chapter 5, we present in details about the layout management framework that uses the CluE algorithm in one of its steps in order to layout the compound graphs structures. This framework consists of two main phases, the *preprocessing* phase and the *layout* phase, where the CluE is utilized in the layout phase.

5

A Generic Layering Framework for Visualizing Compound Graphs Interactively

*Creating abstract representations of large data sets for limited display sizes is considered to be a challenging task. In this chapter, we present a generic layering framework, called **LayMan** (**Lay**out **Man**ager), that handles the visualization of compound graphs, in which nodes are connected with each other via two types of relations: the structural relations and the adjacent relations. LayMan consists of two main phases: the preprocessing phase where the necessary data structures are created to handle the visualization, and the layout phase where the underlying compound graph is visualized interactively. The key feature of the framework is its ability to work with different layout algorithms. We tested the framework in visualizing the failure behavior of embedded systems modeled using the Component Fault Tree (CFT) models. Further, we conducted an evaluation study to collect the participants' feedback towards the interaction facilities in the resulting visualization and the aesthetic aspects of the final layout. Results of this evaluation study show interesting findings regarding the expanding and contracting operations as most of the participants agreed on the intuitiveness of our approach. Moreover, results of the comparisons between the different layout algorithms indicate that participants in the study preferred most the orthogonal layout due to many reasons (e.g., its clearness and the less number of edges crossings).*

In the case of visualizing large data sets, the abstraction concept is one of the traditional solutions to handle this challenge [130]. Generally, containment relationships are used to visualize such abstraction. This visualized abstraction produces a hierarchical representation of the graph, representing the large data sets, in a way that nodes may contains other graph elements. This kind of graphs where the nodes contain other nodes and edges are called *compound* graphs [126].

A compound graph C normally consists of a set V of nodes, a set E of inclusion edges, and a set F of adjacency edges. The inclusion digraph T can be defined as the rooted tree and consists of a set of tuples, where each tuple is (V,E). The leaves of this inclusion graph T are simple vertices, while the remaining are compound vertices. A *cluster node* represents the sub-tree $T(v)$, rooted in vertex v. While the adjacency graph G is a directed graph and consists of a set of tuples, where each tuple is (V,F). In an adjacency graph, any adjacency edge connects a node to one of its descendants or ancestors in T. Note that the graph G also contains the cluster as vertices [126, 68, 141].

Compound graphs are widely used in modeling and data visualization. For example, they are used to realize the relations between the molecules in bioinformatics applications. In this case, grouped elements inside other elements show explicitly the structure of these kinds of elements. The result of such grouping is called *clustered* graphs [130].

In the case of visualizing compound graphs, the resulting visualization should follow certain sets of rules, as also mentioned by Sugiyama and Misue in [142, 143, 141]. The first rule assumes that the structural relations between different levels in the resulting visualization are denoted by full geometrical inclusion of the children shapes inside the geometrical shape of their parents. The second rule assumes that the edges outside of a container (which represents a compound node) should point towards the layout direction. However, in the case of cyclic graphs it is possible that the representing edges of these cycles may point to some other directions. Another rule is that nodes without any structural relations never overlap other nodes, which helps in reducing the number of edges crossings as much as possible. Further, nodes that are connected with each other by adjacency edges are supposed to be placed as close to each other as possible. However, the crossings between the adjacency relations should be reduced as well as the bends in the edges. Moreover, size of a node containing other nodes should be adapted to fit the contents. Lastly, it is recommended to produce the final layout as compact as possible. Precisely, these rules guarantee an increasing in the readability of the final graph representation, based on a study conducted by Helen et al. in [79].

In the literature, many approaches have been proposed to visualize compound graphs having pleasant visualizations. An overview of such approaches can be found in [68]. The layering technique is one of the common approaches for visualizing compound graphs [126, 68, 141]. This method is based on dividing the drawing space into layers, where each layer holds the related set of nodes in a way that all the edges from these nodes point to the same direction. Overall, this approach consists of four main stages: the first step is the cycle removal stage, then the layering stage where nodes are sorted into layers, then the algorithm stage to find the suitable ordering of the nodes, and finally the stage to minimize the edges crossings.

In general, the compound graph drawing techniques based on the layering concept can be divided into two groups: the first one is based on the local layering where one distinct set of layers is employed for each compound node, while the second one is based on the global layering where one set of layers is employed for all the nodes.

Utilizing this existing knowledge we present the **LayMan** (**Lay**er **Man**ager) framework, a layered based general framework for managing and visualizing compound graphs. In this chapter, we present the main steps that are required for managing the layout of this kind of graphs. The framework consists of two main phases, the *preprocessing* phase and the *layout* phase. Although, we inspired from the technique presented by Misue and Sugiyama in [141], we focus towards the optimization of the layout phase in order to produce a compact representation with less edges crossings ratio.

The input of our LayMan framework is a compound graph consisting of the set of vertices, the set of the structural relations between these vertices, and the adjacency relations in the graph. The framework passes this input graph to the preprocessing phase, which is a *bottom-up* approach to build the required hierarchy of the graph to represent the different levels of details. The framework then passes it to the second stage, i.e., the layout phase.

During the layout phase, LayMan creates the layout of the graph interactively, i.e., it creates different levels of details for the graph. Then it retrieves the level of details based on viewers' reactions via two main operations: the *expand* operation and the *contract* operation, which help in building the graph representation on demand. Once the selected node is expanded then the internal nodes are called for the layout phase. These internal nodes are then added to the current view for establishing the new level of details together with the previous visual elements in order to provide the new view [21], as referred in Chapter 4.

We have successfully applied this framework on visualizing the failure scenarios of complex systems like embedded systems. We also conducted an evaluation study to measure the participants' acceptance level towards our approach and the aesthetic aspects of the final layout. Results show high acceptance level among the participants towards the resulted visualizations. Also, they gave positive feedback regarding the intuitiveness of the interaction mechanisms provided by the LayMan framework.

5.1 Background and Problem Statement

Current embedded systems are mostly consisted of complex structures due to the tight integration between a number of hardware and software components. This tight integration between the hardware and software parts makes it difficult the task of tracing the system structure, if it is done manually. To help engineers and experts in understanding and analyzing the underlying system structure, a decomposition phase is normally applied for defining the system main parts [108]. In hierarchical model-based systems, this process is done by refining recursively the sub-components till the desired component.

In embedded systems, safety and reliability aspects are important for measuring the quality aspects of these systems. Many techniques have been proposed in the literature to support system developers, engineers, and safety experts for achieving these aspects. A few examples of these proposed techniques are [93, 92]: the Fault Tree Analysis (FTA), the Event Tree Analysis (ETA), and the Reliability Blocks Diagram (RBDs). FTA is one of the most common techniques for analyzing the safety aspects in embedded systems [39]. FTA depicts the chain, which leads to a specific failure, as a tree structure where to be examined the system failure is treated as the root of the tree while the main reasons of this failure are treated as the leaves of the tree. This graphical representation of the failure describes a hierarchical breakdown, with regard to the hierarchy of the failure influences rather than the system structure perspective. In cases of big trees, FTA replaces the sub-tree with a container, called *module*, which represents an independent sub-tree. However, FTA suffers from many problems, e.g., sometimes the tree structure itself is insufficient to model the failure propagation path for such kind of system's hazard that is caused by more than one paths. Consequently, these paths are split into several repeated events in order to preserve the underlying tree structure [39, 93].

To handle the limitations of FTA, Kaiser et al. proposed a new technique [93], called the Component Fault Tree (CFT), that extends the traditional FTs. In CFT, the notation of *component* is used instead of the *module* notation and these components are connected with each other via ports. It is important to note that in CFT model it is *not necessary* for a component to represent an independent sub-tree, which implies the existence of adjacent relations between the underlying component and other components at the same level or at different levels. Each component in a CFT model represents a real component of the underlying system. The whole CFT model is normally considered as a Directed Acyclic Graph (DAG) of the involved components in the safety scenario [93].

From the above explanation, we can conclude that CFT graph representations can be modeled as compound graphs [68], which are one of the most general graph models having the hierarchical structures. In a CFT model: the set of vertices V contains the list of components, the set of inclusion edges E contains the structural relations between these components, while the set of adjacency edges F contains the failure relations between the system elements. Moreover, a component can contain other components or leaves, where leaves in this context are the basic events in the underlying system.

As the complexity of an embedded system is increased the corresponding CFT model size is also increased, which makes it difficult to handle the failure detection process. The information visualization (InfoVis) field can play an important role in speeding up the system developing process, because it eases the steps of finding important information from both the system and the user perspectives. However, as the size of the graph is increased the traditional layout algorithms fail in providing a readable visualization. Therefore, there is a need of abstracting the graph to reduce the visual cluttering and to enhance the information extracting process [68, 91, 97, 61]. In addition to the abstraction step, expanding and contraction operations over the produced clusters are required in order to navigate the hierarchy and to get more insight about the deeper level of details.

In embedded system domain, *safety experts* are responsible to develop and model the CFT models in order to describe some failure scenarios in the underlying system. CFT models are useful not only in showing the failure relations between the system components but also the structural relations between these system components.

Producing a graph layout that respect the set of aesthetics criteria, as mentioned earlier in this chapter (e.g., minimizing the edges crossings and maximizing symmetry), have proven useful for comprehension of small class diagrams (typically 10–15 classes). However, this is inadequate in dealing with a large information space like large CFT models. Preserving the aesthetic criteria such as minimizing the edges crossings is considered to be computationally difficult problem [63]. The general solution to effectively visualize large CFT models with layout is unlikely. Therefore, it is desirable and meaningful to investigate visual abstraction for representing CFT models.

In this chapter, we propose a *focus+context* technique for CFT models, where the focus area is presented in details and visualized with standard CFT notations while the remainder of the model is abstracted at various levels of details in a style called the *onion notation* [91]. The formation of the onion graphs from the CFT notation and the onion notation provides a single view with a combined *focus+context*. A series of onion graphs support semantic zooming and incremental exploration [91].

5.2 The LayMan Framework

Our **LayMan** (**Lay**out **Man**ager) framework is a general framework for managing and visualizing compound graphs. In this section, we explain the two main phases of the framework (i.e., the *preprocessing* phase and the *layout* phase) and the steps that are required to manage and visualize compound graphs. Although, we inspired from the technique presented by Sugiyama and Misue in [142, 141], we focus towards the optimization of the layout phase in order to produce a compact representation with less edges crossings ratio. The input of our LayMan framework is a compound graph consisting with the set of vertices, the set of the structural relations between these vertices, and the adjacency relations in the graph.

The initial view of the graph represents the highest level of details or the *root component* level. In this case, the layout may contain compound nodes or leave nodes. Here, the compound nodes are the expandable nodes and are represented using the rectangular shapes, while the leaf nodes are nonexpendable nodes and are represented using the circular shapes. Each time a compound node is expanded, it adds new elements to the view. Further, the adjacency edges between these new nodes are also added to the view.

In this context, we use the spline representation [82] to represent the edges that pass through the different levels of the hierarchy. We connect the list of edges that go in one path, into one edge in order to reduce the time during the rendering stage. Then we represent them using the spline representation. Here we follow the edge bundling technique that was proposed by Holten in [82]. We calculate the list of edges that share the same compound node source but may come from different leaf nodes inside this compound node source (as shown in Figure 5.1). These edges share the same target. Then we calculate the least common ancestor node between any two sources for establishing the control polygon of the spline. In Figure 5.2 we show the effect on none bundling edges, while in Figure 5.3 we show the result of smoothing the edges in the layout using the heuristic of the edge-bundling technique. Further, in order to show the direction we use the color propagation animation to show the flow direction on-demand, as shown in Figuer 5.6.

In the forthcoming two subsections, first we describe the preprocessing phase that eases up the visualization process and then we explain how our LayMan framework uses the CluE algorithm, presented in Chapter 4, to iteratively visualize the different levels of details in the underlying compound graph.

5.2.1 The Preprocessing Phase

After importing the graph, our LayMan framework preprocesses it to ease up the visualization process. This preprocessing phase consists of five stages as following:

1. ***The Cycle Removal Stage:*** The main idea behind this step is to produce the layers within the graph, because all edges should be

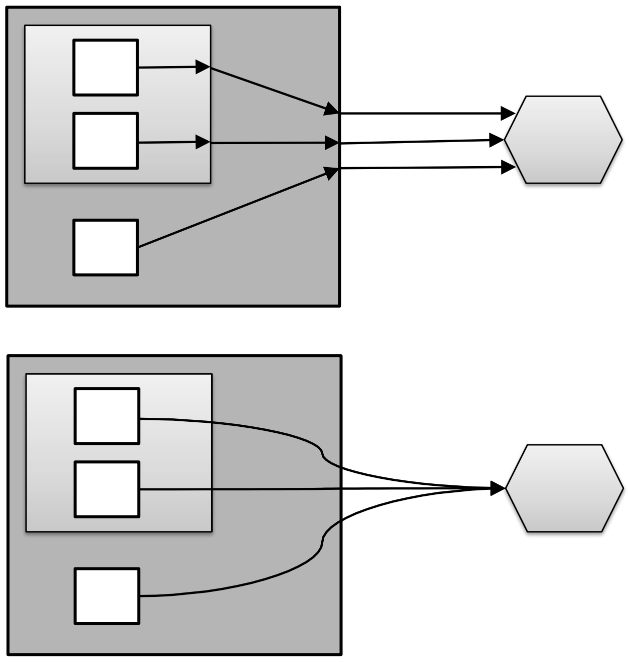


Fig. 5.1: The lower figure is the resulting figure after the application of LayMan edges treatment process on the upper figure. It shows that the list of edges that share the same path direction are combined into one segment represented as a spline.

pointing towards the layout direction. The cycle removal stage guarantees that the graph is ready for passing to the layering assignment. In this stage, cycles are broken up by reversing the constituting edges. Although, the optimal selecting of the set of edges to be reversed is considered as NP-hard; however, many heuristics have been proposed in the literature to cope with this issue [96]. The edges-reversion enables the framework to render the graph acyclically and to retrieve those edges that cause the cyclic behavior. This step is important as it keeps these edges in the graph; otherwise, if they are removed from the graph at this stage then it would be difficult to represent them during the edges routings and edges crossings minimization stages later in the process.

2. ***The Layers Assigning Stage:*** At this stage, each node in the graph is assigned to a layer such that its successors are placed in the upper layer. This helps in drawing the edges pointing to the same direction.

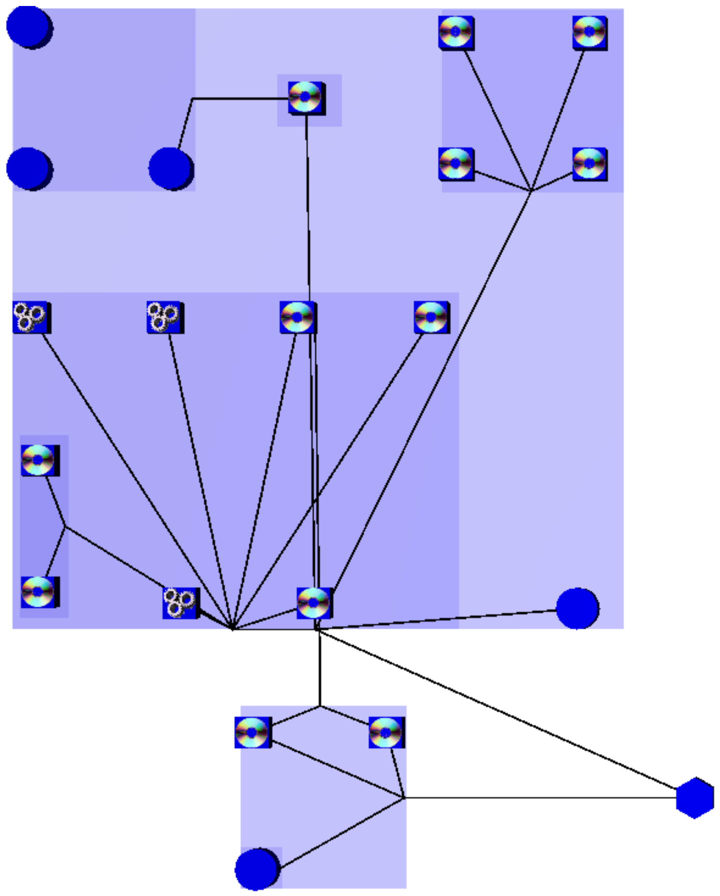


Fig. 5.2: The final result of the layout in the case of non-bundling edges, before using the spline representation to link between the different edges segments.

3. ***The Edge Preprocessing Stage:*** This stage aims at reducing the edges crossings and the nodes overlapping. Therefore in this stage, the framework strives to calculate the nodes ordering in each single layer in order to reduce the edges crossings. However, in order to optimize this stage we designed the framework to represent those edges that pass through different layers to render them as splines. The control points for these splines are the nodes' coordinates in the path to the root node.
4. ***The Node Placement Stage:*** At this stage, the y-coordinates of the nodes are determined as well as dummy nodes are inserted to split up long edges. This stage is relevant for the number of edge bends.

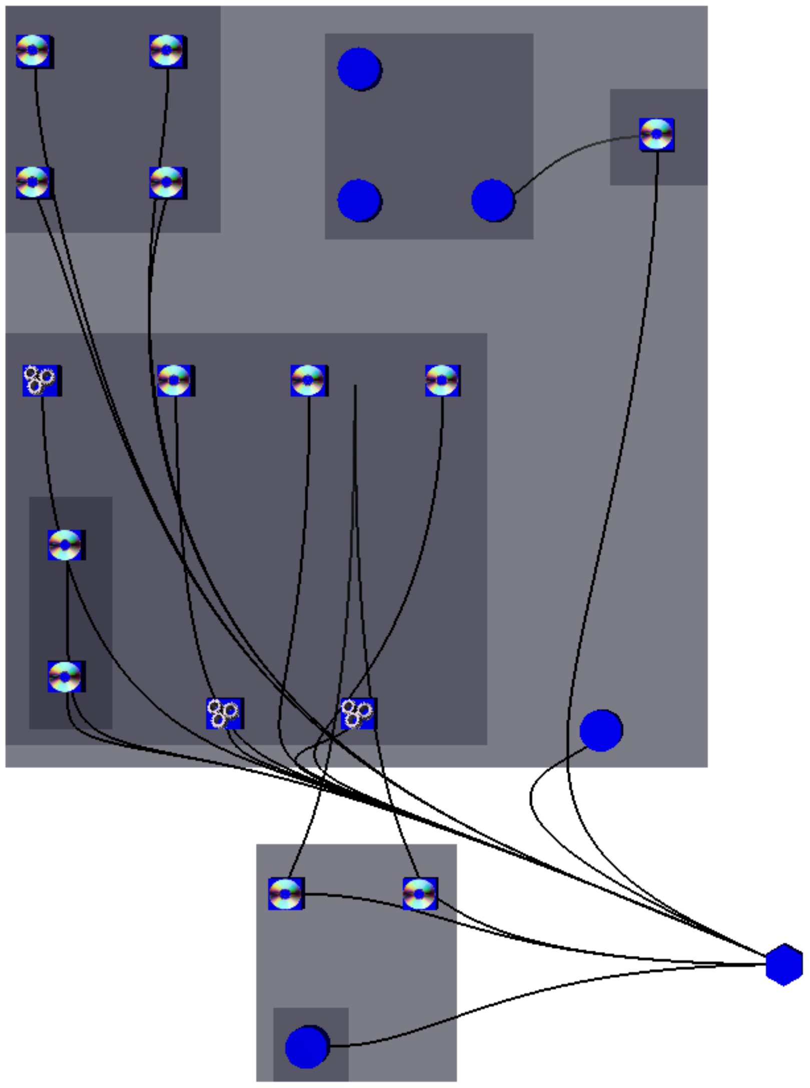


Fig. 5.3: The final result of the layout in the case of non-bundling edges, before using the spline representation to link between the different edges segments. Here, the spline curvature shows the edge direction.

5. ***The Edge Routing Stage:*** In this stage, the routing of the edges is determined and the final *x* coordinates of these nodes are assigned.

5.2.2 The Layout Phase

After the preprocessing phase, the graph is ready to be visualized. At this stage, the LayMan framework passes it to the to the *layout* phase, which is a *top-bottom* approach for building the hierarchy in contrast to the preprocessing stage that is a bottom-up approach. In this phase, the framework uses the CluE algorithm, described earlier in Chapter 4, to iteratively visualize the different levels of details in the underlying compound graph.

In this subsection, we explain how the LayMan framework uses the CluE algorithm, which aims at expanding the compound nodes in clustered graphs. Here we assume that the information about the underlying graph's vertices and edges is available in a pre-stage before the execution of CluE, as the algorithm cares only about the current node in the graph that is responsible about the current level of details.

At the beginning of execution, CluE takes the graph root and its children for calculating the required size. Then it calls the layout algorithm in order to calculate the coordinates of child nodes in the current level of details. CluE is called recursively based on the viewer's request. It is called each time whenever the viewer triggers a node to be expanded.

After this, CluE performs the following steps:

CluE starts from a specified node *N* in the graph, which represents the current node to be expanded or the graph root in the case of initial stage. Then it finds all the child nodes of this node. First, it adds the child nodes to a list *NL* (Node List), that is empty at the initial stage. All the nodes in this list represent the current level of details and are connected with each other via the set of structural relations of *F* edges. It also keeps the adjacency relations between these nodes in a list data structure *EL* (Edge-List), that is initially empty. CluE works iteratively, as it layouts the graph nodes level by level. It checks if one of the child nodes in *N* is needed to be expanded, then it calls itself to add the child nodes of this targeted node in the node list *NL*. After that, it estimates the size of this expanded node based on the number of its children and then it saves this size for later usage. At this stage, CluE adds the adjacent relations related with the internal structure of the expanded node to the edge list *EL*.

In the next step, CluE calls the layout algorithm. In fact, CluE does not rely on any specific layout algorithm, as it requires only the coordinates of nodes in the graph. Therefore, it works perfectly with many different layout algorithms (e.g., the orthogonal layout, the Sugiyama algorithm, the orthogonal layout algorithm, or any multi-level layout algorithm). CluE has been tested in our LayMan framework with all these mentioned layout algorithms and it performed well in all cases. After this step, CluE takes the coordinates of graph nodes and then it shifts these nodes in two directions according to the size of the bounding box of the expanded node, which was saved in the previous stage of CluE.

The main limitation of CluE is that it calls the layout algorithm whenever there is a need to expand a new node. This requires calculating new positions each time. However, CluE keeps fixed the layer of the nodes. This affects on maintaining the viewer's mental-map of the layout [79]. In order to help viewers in tracking the nodes at previous positions, our LayMan framework adds a slow animation to show the slight changing in nodes' positions.

5.3 Application of LayMan in Encoding CFT Models

In this section, we discuses briefly the idea of how we give the insight of the component type using the abstract representation of the CFT model.

Generally, embedded systems are consisted of different types of components, normally divided into two main groups: the *hardware nodes* and the *software components*. Therefore, it is important to show the component type in the abstract graph representation of the underlying safety scenario.

In our LayMan framework, the resulting abstract representation of the safety scenario also encodes the component type in visual form, as shown in Figure 5.4. To achieve this, we propose the solution of using the texture cue to show the node type. In this regards, we use the CD icon image to encode the software components inside the system (see Figure 5.5a), while the hardware components are encoded using the hardware icon image (see Figure 5.5b). Because the CFT model encodes the component type in an abstract way, that's why some components are a mixture of hardware and software subcomponents; therefore, it is difficult to distinguish their exact type. For these components, we encode their type using the question mark icon (see Figure 5.5c) in order to encourage the viewers for spending more time in understanding the type of these components and then to classify them.

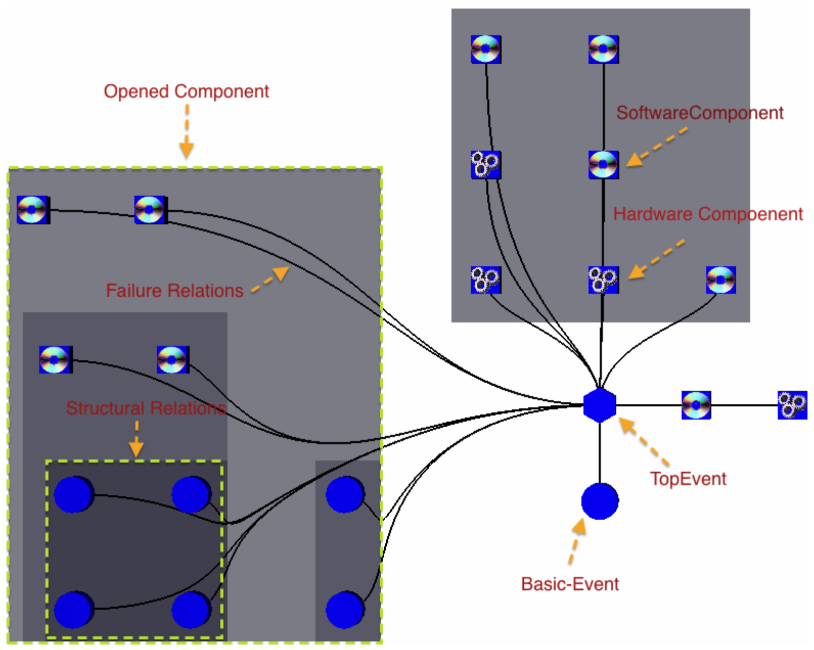


Fig. 5.4: An explanation of the graph representation by LayMan for a CFT model.

(a) Icon for a software component

(b) Icon for a hardware component

(c) Icon for an unknown node/component type

Fig. 5.5: Encoding nodes and components types

In CFT models, there are two types of relations between the CFT elements: the *structural* relations and the *failure* relations. To visualize the structural relations we use the containment representation in a way that each component contains its internal elements. On the other side, we use explicit edges to visualize the failure relations between the CFT elements. The color gradation is used to encode the level of the node in a manner that the darker nodes are at the least levels of details in the failure hierarchy. This was designed to achieve two goals: the first one is to reduce the edges crossings because the edges are encoded implicitly, and the second one is to help in producing a compact and area-aware representation from the aesthetic point of view.

The failure relations are encoded using the explicit line representation, as these relations are drawn in the view on-demand based on the number of expanded nodes. Further, splines are used to visualize these relations in order to reduce the crossings, as in [82] it has been proved that using curvy representations of edges produces less number of crossings compared to the straight line representations.

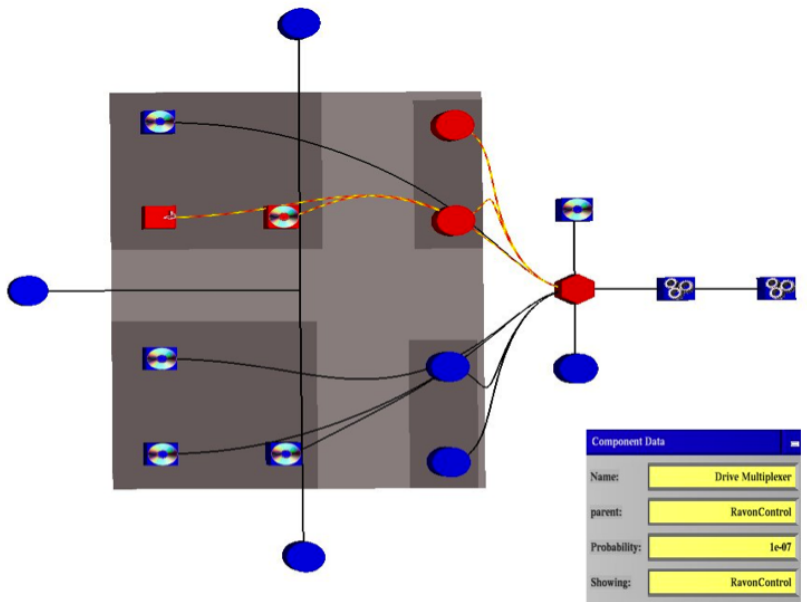
The *top event* node is encoded using the hexagonal shape at the center of the representation, while the *basic events* are encoded using the blue circle representation. The gate information is represented implicitly in the final representation. This was designed to reduce the number of objects in the scene as well as to reduce the extra cluttering in the final representation. However, the logical connection between the nodes can be extracted easily by hovering the mouse over the node, as then all the nodes are highlighted on-demand that are logically connected with the selected node.

The direction of the failure is encoded using the slow animation goes from the selected node towards the top event node. The animation is triggered on-demand and is presented using the color interpolation scheme, as shown in Figure 5.6.

Initially, all the nodes in the graph representation are represented in their default presentation, as explained above. However, in order to help users to highlight the failure points in the system we use the red color to encode those nodes that are participating in the current failure. This is illustrated in Figure 5.6, where all the red components and basic events are the participating elements in the current failure path.

5.3.1 Mapping the Safety Scenario Information from the CFT model to the Graph Representation

To help our real end users (i.e., the *system engineers* and the *safety experts*) to use the graph representation in order to analyze the safety



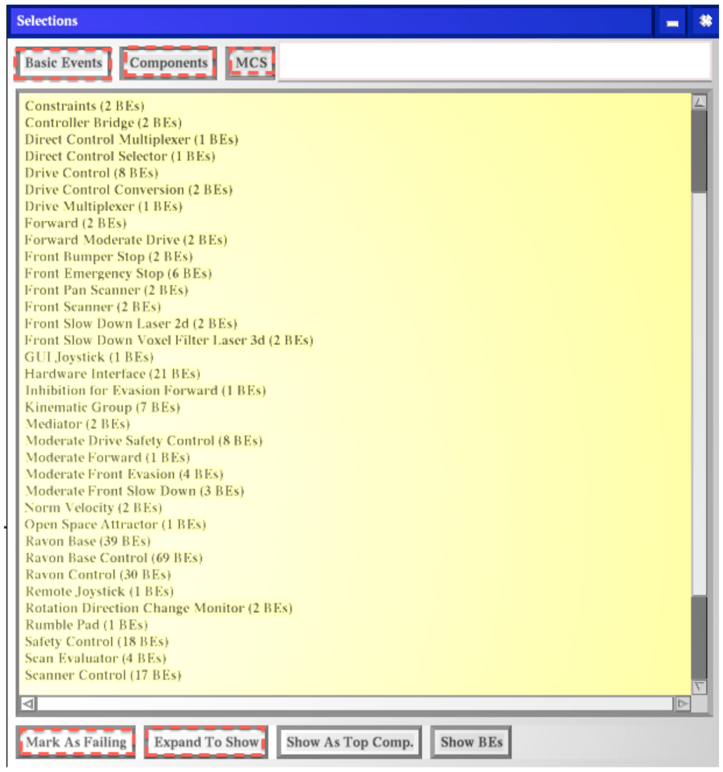
Name:	Drive Multiplexer
parent:	RavonControl
Probability:	1e-07
Showing:	RavonControl

Fig. 5.6: The color propagation animation over the adjacent edges is used to show the flow direction on demand.

aspects of embedded systems, our LayMan framework provides the following options to support the interaction with the system.

- ***The Safety Information Menu:***
  The LayMan framework arranges the important safety information in a menu called the *Safety Information Menu*. Users can trigger this menu by clicking the right mouse button. This menu contains five main tabs. The first tab shows the list of components in the current CFT model together with their number of basic events. The second tab contains a list of all the basic events and their probabilities in the current CFT model, while the third tab contains a list of all the minimal cut sets in the current CFT model with the option to sort them based on their probabilities or their sizes. In each of these tabs, LayMan provides two buttons to allow users to highlight the selected elements in the list and then expand them to show their place in the visualized CFT model. The last two tabs are responsible on synchronizing between the graph representation and the 3D model of the underlying embedded system. These two tabs will be

explained in details in Chapter 7. Figure 5.7 shows a screen-shot of this menu.



Constraints (2 BEs)
Controller Bridge (2 BEs)
Direct Control Multiplexer (1 BEs)
Direct Control Selector (1 BEs)
Drive Control (8 BEs)
Drive Control Conversion (2 BEs)
Drive Multiplexer (1 BEs)
Forward (2 BEs)
Forward Moderate Drive (2 BEs)
Front Bumper Stop (2 BEs)
Front Emergency Stop (6 BEs)
Front Pan Scanner (2 BEs)
Front Scanner (2 BEs)
Front Slow Down Laser 2d (2 BEs)
Front Slow Down Voxel Filter Laser 3d (2 BEs)
GUI Joystick (1 BEs)
Hardware Interface (21 BEs)
Inhibition for Evasion Forward (1 BEs)
Kinematic Group (7 BEs)
Mediator (2 BEs)
Moderate Drive Safety Control (8 BEs)
Moderate Forward (1 BEs)
Moderate Front Evasion (4 BEs)
Moderate Front Slow Down (3 BEs)
Norm Velocity (2 BEs)
Open Space Attractor (1 BEs)
Ravon Base (39 BEs)
Ravon Base Control (69 BEs)
Ravon Control (30 BEs)
Remote Joystick (1 BEs)
Rotation Direction Change Monitor (2 BEs)
Rumble Pad (1 BEs)
Safety Control (18 BEs)
Scan Evaluator (4 BEs)
Scanner Control (17 BEs)

Fig. 5.7: The Safety-Information-Menu containing the required safety information to help the users in getting more insight about the system status.

- ***The Textual Information Box Option:***
  In order to extract information about any picked component from any of the two provided views, the TextInfo unit inside LayMan provides a dialog box option to show the safety-information of the corresponding node (see Figure 5.6) on-demand. The displayed information includes the criticality value of the selected node, the parent node, and the node's name.

- ***The Mouse+Keyboard Interaction Option:***
  In this option, LayMan allows users to navigate through the graph representation, using different combinations between the mouse

and the keyboard keys to explore the graph hierarchy. For example, left mouse button + ‘2’ button on the keyboard are responsible on expanding or contracting the required node, left mouse button + ‘1’ button on the keyboard are used to pick a node in the system 3D model, and for zooming in or out users are required to keep pressing on the left mouse button and the ‘*A*’ button on the keyboard. Beside these, the framework provides many other combinations to enrich the interaction options with the CFT graph representation.

5.4 The Evaluation Study

We conducted an evaluation study in order to get the participants’ feedback in general over our presented LayMan framework. In this case, we interested ourselves in collecting this feedback to compare between the different layout algorithms that are used by LayMan framework in visualizing the final layout. However, for the study we selected the most common ones, i.e., the orthogonal layout algorithm [72], the Sugiyama layout algorithm [143], and the radial layout algorithm [59]. Further, we tested the expanding and the contracting options of the compound nodes using the CluE algorithm.

In our study, we invited 25 participants in order to perform the exploration test of the LayMan framework for visualizing the CFT models. Most of these participants were from the University of Kaiserslautern having a background in engineering or computer science. In these participants, 21 were males while the remaining 4 were females. The participants were divided into groups based on their backgrounds (4 were robotics experts, 6 were safety experts, 4 were software engineers, and the remaining 11 were visualization researchers).

The tasks were mainly focused on extracting some safety information about the corresponding components at different levels using different layout algorithms. The experiment was performed using a desktop environment, with the participants seated in front of the computer. They interacted with the platform with a normal mouse and a keyboard.

First, we trained participants how to use our platform and then we provided the required background they needed to achieve the given tasks. We explained them the tasks and then asked them to provide us their feedback. After it, we also asked the participants to give us their feedback through a closed-ended questionnaires form. Participants were asked to select a number between 1 and 5 (1 meant strongly disagree while 5 meant strongly agree) based on the likert scale to rate their performance. The sixth option was “Don’t know”.

Out of these participants, 21 found that the expanding and contracting node operations using the LayMan framework are intuitive and easy to follow. Their rating of their own performance was either 4 (i.e., *agree*) or 5 (i.e., *strongly agree*). The remaining 4 participants rated themselves either 2 (i.e., *disagree*) or 3 (i.e., *neither agree nor disagree*), as they found some difficulties in tracing the transitions of nodes. Regarding the trust level of the framework, 19 participants assigned their trust level to be either 4 or 5, while the remaining 6 participants requested more time in order to judge their trust level on the framework. Overall, the results of this evaluation study show the intuitiveness of our approach and indicate a high acceptance ratio among the participants. However, there is a need of extended detailed evaluation studies to generalize these findings.

5.4.1 Testing the LayMan Framework over Different Layout Algorithms

We also tested our LayMan framework over different layout algorithms for handling the complexities of the CFT model. In this subsection, we briefly introduce some of the layout algorithms that were tested within the framework. We compare between the results in terms of the number of edges crossings, layout compactness, and the node-to-node overlapping. In the evaluation study, we chose to test the framework over the orthogonal layout algorithm, the Sugiyama algorithm, and the radial layout algorithm. We tested these algorithms with the same-sized graph and the same number of level-of-details within the graph.

The Radial Layout Algorithm

The radial layout algorithm [59] is a specialized layout algorithm for the tree structures, as it places the nodes in radial spaces. It also works fine for the DAG structures. It produces a cluttered layout for general graphs because of the edges crossings between different hierarchy levels. This kind of layout algorithms is recommended when the data set is inherently containing tree structures [59].

We tested the LayMan framework over the radial layout algorithm (as shown in Figure 5.8). Participants in our evaluation study required more time to distinguish the root of the graph from the other nodes. Due to this fact, it is considered to be not efficient in safety critical applications like diagnosing the system failure using the CFT models. We also observed increasing the node-to-node overlapping and the high edges crossings ratio in the case of large-sized graphs.

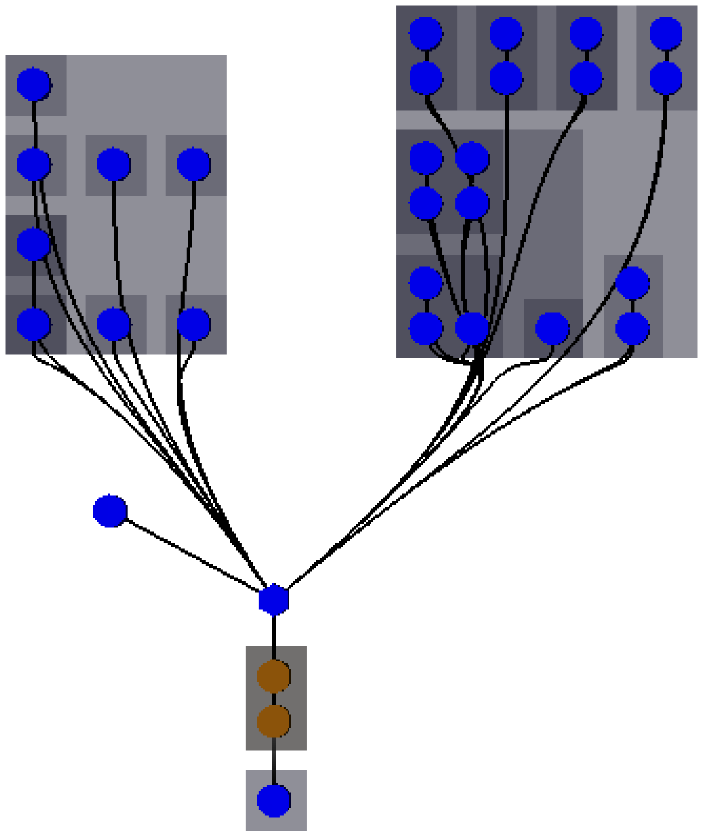


Fig. 5.8: A possible layout after applying the *radial layout* algorithm on a compound graph representing a CFT model.

The Sugiyama Layout Algorithm

The Sugiyama algorithm [143] is one of the earlier algorithms that draw general directed graphs. The basic approach is that it first layers the graph nodes, which means assigning a layer for each node and then placing the nodes in the corresponding layers. After this, the algorithm reduces the edges crossings and the nodes overlapping during the preprocessing steps.

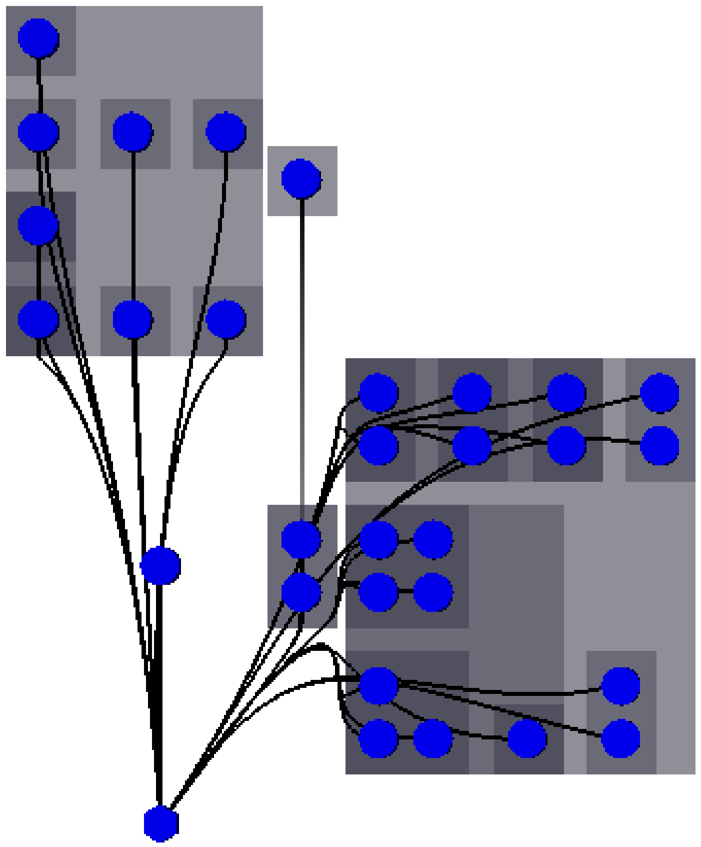


Fig. 5.9: A possible layout after applying the *Sugiyama* algorithm on a compound graph representing a CFT model.

We were expecting good results from this algorithm due to its ability of visualizing compound graphs. Figure 5.9 shows a possible layout using this algorithm for a compound graph representing a CFT model. Most of the participants in our evaluation study reported intuitiveness of the layout when they interacted with the visualized CFT model through this layout algorithm. Further, most of the participants agreed on increasing in the readability of the resulting visualized graph due to the less number of edges crossings and the decrement in node-to-node overlapping. Also, it was easy for them to find the root of the hierarchy, mainly due to the natural representation of the tree structure resulting from the layering steps of the algorithm.

The Orthogonal Layout Algorithm

The Orthogonal layout algorithm [72] is considered to be a general layout algorithm. It produces a pleasant and clear layout of complex networks and graphs. Originally, the algorithm represents the set of nodes as a set of rectangles and the edges between these nodes as chains of horizontal and vertical segments. This offers a nice representation with less number of crossings between the edges and helps the viewers easily tracing the relations between the graph nodes. In this case, a cluster helps in specifying a group of objects that are needed to be place together in the final layout so they can be defined easily [72]. Figure 5.10 shows a possible layout using this algorithm for a compound graph representing a CFT model.

Overall, this was the algorithm that got the highest number of votes over the two above-mentioned algorithms due to its clearness and the less number of edges crossings, the less number of edged bends, and the less node-to-node overlapping. Most of the participants mentioned that the orthogonal layout increases the readability factor of the graph. They also agreed on the layout intuitiveness and the clarity of resulted layout.

5.4.2 LayMan Scalability

In this subsection, we show results of using LayMan on large graphs by applying different layout algorithms. For this, we discuss the LayMan scalability from two perspectives: with respect to the graph size and with respect to using different layout algorithms.

We tested LayMan on a larger-sized graph and then we asked participants in the evaluation study about the interactivity speed when the whole graph was expanded. The underlying graph consisted of 100 compound nodes, a high number of adjacency edges between the nodes, and 6 different levels of details.

In this case, the participants reported that there was no delay in the expansion of nodes at different levels of details or expanding the whole graph. Further, the contraction speed was also high. Overall,

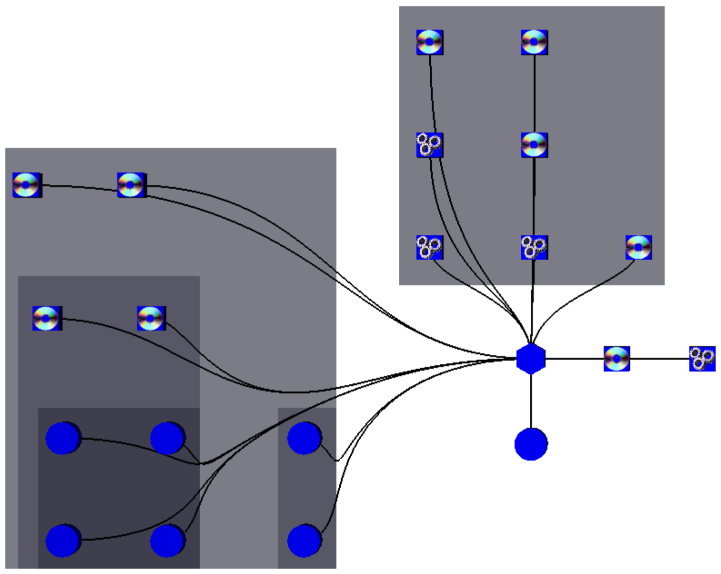


Fig. 5.10: A possible layout after applying the *orthogonal layout* algorithm on a compound graph representing a CFT model.

participants reported the intuitiveness of the expanding and contracting operations in large-sized graphs. However, we think that there is a need to test it on further large complex graphs with higher levels of details in order to make a more general statement about the scalability of the framework in terms of the graph size.

With respect to the layout algorithms used in the LayMan framework, we tested two main layout algorithms (i.e., the Sugiyama algorithm and the orthogonal layout algorithm) for visualizing large compound graphs. In these two algorithms, the framework performed very well without any problem. However, the resulted layouts were slightly different in each case due to the nature of the background layout algorithm.

Figure 5.11 and Figure 5.12 show examples of using the two algorithms respectively for a large-sized graph. Regarding the edges crossings, we think that our layout algorithm reduces relatively the edges crossings ratio because we use spline representation to represent the adjacency relations between the nodes. However, we observed that Sugiyama algorithm produces nicer representation with less number of edges crossings compared to the orthogonal layout algorithm.

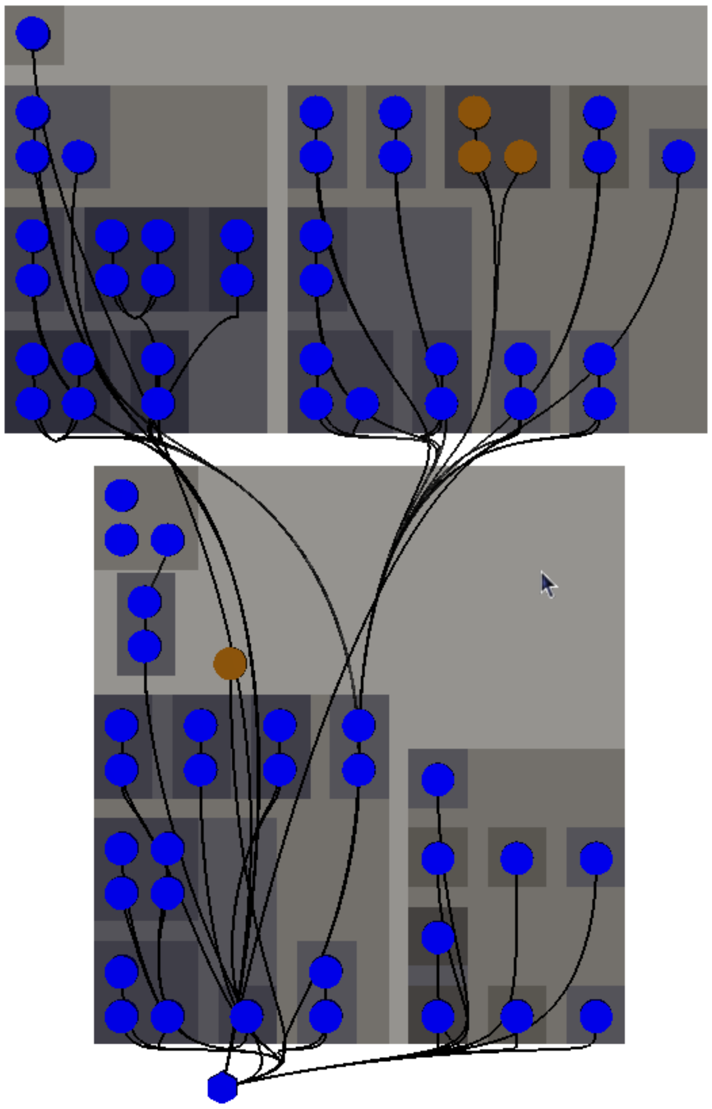


Fig. 5.11: The result of expanding large-sized graph using the Sugiyama algorithm in LayMan to layout the graph.

5.5 Summary

In this chapter, we presented a generic layering framework, called LayMan, that handles the visualization of compound graphs. The framework consists of two main phases, the preprocessing phase and the

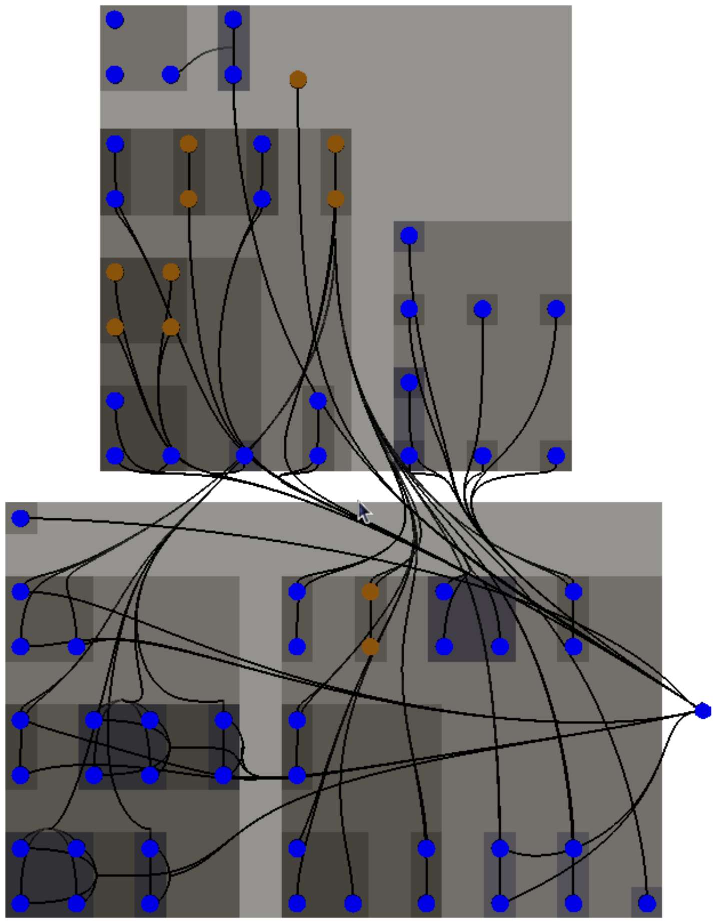


Fig. 5.12: The result of expanding large-sized graph using the Orthogonal layout algorithm in LayMan to layout the graph.

layout phase. We designed the preprocessing phase to create the hierarchy of the graph and to layer the nodes based on their structural behavior. For the layout phase, we visualize the processed graph using the CluE algorithm described in Chapter 4.

LayMan provides an interactive incremental multi-level layout process, where the initial layout shows the highest level of details or the root component level. Upon expanding a compound node the inside elements, including other components or adjacent edges, are added to the new level of details. This creates another view of the graph showing more details. All the new data elements are then passed to a layout algorithm in order to assign these elements' positions. In this regards, the framework is portable with different layout algorithms including the orthogonal layout algorithm, the Sugiyama algorithm, and the radial layout algorithm. The LayMan framework also uses a heuristic solution to decrease the edges crossings through the edge bundling technique and the spline representation style.

We have successfully applied the framework for visualizing the failure behaviors of embedded systems modeled as CFT models. We also conducted an evaluation study in order to collect participants' feedback during exploring the resulted layout using different layout algorithms in the framework. We also evaluated the intuitiveness of the expanding and the contracting operations from viewers' point of views. Results show that participants preferred the orthogonal layout algorithm in this domain due to its clearness and the less number of edges crossings, the less number of edged bends, and the less node-to-node overlapping. They ranked the Sugyiama algorithm as their second choice, as it provides a natural representation of the graph hierarchy and it was easy for participants to find the root node and the corresponding layer to it. Overall, participants agreed on the ability of the framework to produce a compact readable final layout with less edges crossings.

We have successfully applied the LayMan framework to visualize the CFT models that represent the safety mechanism of complex systems like embedded systems. This framework can also be used to visualize software systems or other this kinds of systems. The resulting layout can be used together with the hardware model of the system to give the full story about the system' safety status. The hardware visualization and the synchronization between the system 3D model and the graph representation of the CFT models are introduced in the forthcoming chapter, i.e., Chapter 7, of the thesis.

6

Adding the Stereoscopic Depth in Graph Visualization

The stereoscopic highlighting technique is a new emerging technique that supports using the depth cue in 3D devices as a highlighting technique for general node-link representations. One of the main abilities of this technique is isolating the specific graph portions to magnify them for a detailed exploration without resorting other highlighting attributes such as color or shape. In this chapter, we focus on our contributions towards this technique, especially the different evaluation studies that we carried out to measure the feasibility and usability of the technique from the viewers' perspectives. In this regard, first we present the evaluation study that we conducted to measure the accuracy of this technique by evaluating the ability of general viewers from different backgrounds in reading the variation in depth values to encode some data aspects. Based on the feedback from the this evaluation study, we extended the technique using the ***Reflection Layer*** *in order to enhance the viewers' abilities in detecting the depth cue more efficiently and accurately. We also carried out an evaluation study to find out whether our reflection layer increases the viewers' abilities in detecting accurately and efficiently the variation in depth values. Further, we present a new technique, called the* ***Stereoscopic Filtering Technique (StFT)****, based on the stereoscopic highlighting approach. This new technique utilizes the depth cue to filter large graph structures. One of its main abilities is that it isolates specific graph portions to magnify them for a detailed exploration without resorting other visual cues such as color or shape. It offers a* *natural way to provide a focus+context result as it brings some parts of the graph closer to the viewer while keeping the rest in the background. We use this technique to query about specific topological features about the graph such as trees, clusters, complete graphs, or unknown graph structures. Moreover, we present our* ***ExpanD*** *technique in which we apply the stereoscopic depth to encode the structural relations in compound graphs. Finally, we present another evaluation study that we conducted to measure the usability of our ExpanD technique. Results of this study show that using the depth cue in stereoscopic devices is readable for medium data sizes with at most three or four different layers. This can be useful in classifying the data set into a set of categories according to the depth value of the elements' depth layer.*

Recently, we witness a high acceptance of using 3D technologies in many of our daily life applications. For example, we can easily observe such technologies' widely acceptance in entertainment industry for movies and animations. Due to that, people are getting familiar with 3D movies and 3D games. Moreover, the 3D TV concept is quite common nowadays, as the cost of such technologies is also becoming affordable over time due to the continuous decrements in the hardware cost. Along this, the quality of 3D stereoscopic pictures is progressing positively too, as the underlying technology for producing these devices is also advancing at accelerating steps. All these reasons make using these 3D technologies an active and attractive area in the visual research. In addition, many scientific visualization applications have already started using 3D displays. For example, visualizing the spatial relationships between different objects in the 3D dataset is one of the common examples of using 3D technology in the research [122].

In the case of information visualization (InfoVis) field, data elements normally have non-spatial relations amongst them. This makes the idea of using 3D technologies for visualizing this kind of data an awkward possibility. However, Alper et al. in [5] interestingly opened the doors of exploring the new possibility of utilizing the 3D technologies for visualizing some aspects about the data using the stereoscopic depth. They proposed to use the stereoscopic depth to highlight nodes in node-link diagrams. In this case, node-link diagrams are visualized as in 2D representations but they are rendered on stereoscopic platforms. This allows the highlighted nodes to be positioned in a plane closer to the viewer, which gives these highlighted nodes the required importance as they look bigger than the other nodes due to the depth cue. In our opinion, this idea is useful to bridge the gap between the 2D displays and the 3D technologies.

In the case of 2D displays, designers are forced to pack all the data elements in a 2D plane. This requires reducing the visual dimensionality, which leads to two main drawbacks: first it distracts our visual system as it is designed to perceive and interpret 3D spaces, second it may harm the scalability of visualizing large data [127]. However, in the case of 3D displays designers have the choice to layer the data in the 3D space offered by this technology in order to reduce the cluttering and to improve the understandability of the data. This also helps in designing a scalable representation for big data. Additionally, it compliments with human visual system, as human perceive data in the 3D world naturally. To achieve a pleasant visualization for non-spatial data (like graphs) in 3D platforms, designers need to start thinking in new directions for utilizing the stereoscopic depth offered by the 3D technology, similar to the direction proposed by [5, 127].

Rendering a 2D node link representation in a stereoscopic platform supports the idea of having a natural *focus+context* interactive scheme. In this case, we utilize the stereoscopic depth to highlight or encode some data aspects and put the highlighted regions on the viewer's focus while taking the remaining parts of the graph on a deeper plane, which helps in keeping the context of the whole graph in the background.

For example, visualizing a dense graph using the node-link diagram in a pleasant way is considered to be one of the challenges in information visualization domain [156, 155]. Due to the edges crossings, the current state-of-the-art lacks a pleasant representation for large dense diagrams [5]. However, many heuristic solutions have been proposed to optimize this issue. One such solution is based on the 3D node-link layout in [154], where the authors argued that the rotation could be used to identify the adjacency relations of certain elements. In spite of this advantage of the 3D layout, the perspective view in the 3D world could lead to wrong conclusions about the importance degree of some nodes. This happens as a result of rendering some nodes closer to the virtual viewpoint than the others, which increases the visual emphasis of the closer nodes and consequently affects the viewer's perception. This problem appears mostly because of two reasons, either due to the graph layout algorithm or due to the viewer's viewpoint rather than the data set itself. Additionally, the 3D layout requires high amount of viewpoint navigation, e.g., when a node of interest is rendered at a further depth level the viewer needs to rotate, zoom in/out, or change the view-angle to get a better view of the required node. This affects on preserving the viewer's mental map because the graph appears differently each time to the viewer [5].

These problems can be reduced if the graph is rendered in a stereoscopic environment, because such environment could eliminate the perspective issues and the navigation problems. In [5], Alper et al. did a comparison of using the depth cue provided by stereoscopic displays as a highlighting technique beside other static visual cues on 2D and 3D graph layouts. They proved the ability of viewers to read the stereoscopic depth and used it as a highlighting technique for showing the important nodes in the underlying graph. They claimed that using the stereoscopic depth as a highlighting technique helps in saving other static visual cues for other usages, especially with highly attributed data. However, it can be used only as an ordinal property while the other static visual cues can be used to highlight nominal properties.

In this chapter, our main focus is on utilizing the stereoscopic depth to highlight some aspect about the data. In this regard, we extended the technique with a *reflection layer* in order to enhanced the viewers' ability in detecting the depth cue more efficiently and accurately. Further, we present our **StFT** (**St**ereoscopic **F**iltering **T**echnique) technique for utilizing the depth cue to filter large graph structures and **ExpanD** (**Expan**d in **D**epth) technique in which we apply stereoscopic depth to encode the structural relations in compound graphs. In order to check the feasibility and effectiveness of all these techniques, we carried out evaluation studies where participants from different backgrounds evaluated them in the controlled experiment environments. In this chapter, we provide details of these conducted evaluations study and discuss the results.

As mentioned above, using a 3D node-link diagram is an optimized solution for the edges crossings problem in the 2D layout [154]. This can be guaranteed by isolating the set of edges that overlaps in the 2D version. However, producing a 3D diagram makes it difficult to preserve the mental map of the graph, as it could produce more occluded layout that needs extra interaction tools to navigate through it.

The stereoscopic highlighting technique, proposed in [5], utilizes the stereoscopic depth to highlight the important regions in a 2D diagram by bringing them closer to the viewer. This technique saves the other highlighting techniques like the color or the shape to encode data attributes. The main concept is to use the variations in depth to highlight the multiple groups in the diagram. One of the biggest advantages of using the stereoscopic highlighting technique is its ability in achieving the *focus+context* views naturally, as it brings the region of interests to the foreground while keeping the rest of the graph in the background. Hence, the region of interest becomes larger and the rest of the graph appears less important as it is projected in the background. Therefore, it reveals less detail about the graph structure.

Further, the depth cue has few advantages over other static visual cues, e.g., when elements in a set are associated together at the same depth level it gives a spatial relation to those associated elements. Also, based on the Gestalt theory the spatial association among the data elements is a stronger indication than the colors [113]. However, according to [5], nearly $8 \sim 10\%$ of the population suffers from depth blindness problem. On the other hand, around 10% of men and less than 1% of women population also face some degree of color-blindness [57], which makes it also difficult for them to use the color as a cue.

This technique also offers an effective interaction approach by arranging the graph nodes into a set of layers according to their depth values. This can be used to convert 2D node-link diagrams into 3D node-link diagrams, in which case the depth is used in a discrete fashion to encode some data aspects. This can be useful in analyzing a set of nodes together by bringing them at the same depth layer. Moreover, stereoscopic environments allow a visual search at different layers simultaneously. This has been shown by the work of Nakayama and Silverman in [118], where they concluded that stereoscopy disparity allows for parallel conjunctive search. They proved the human ability in perceiving similar colors and motion attributes of node on different layers that are isolated using different depth values.

It is worth mentioning that in [5], Alper et al. measured the feasibility of using the stereoscopic highlighting technique. They provided an interactive framework to allow viewers to arrange the discrete layers in order to find out the set of neighbors of the highlighted node and to find out the number of all nodes that are along the shortest path between the selected two nodes. They claimed that viewers were able to find out the number of layers and the number of nodes that are associated to each layer. However, there is no any indication of the number of the layers that viewers were able to detect accurately or what was the number of nodes associated to each layer.

6.1 Evaluating Stereoscopic Depth in Node-Link Diagrams

To check the effectiveness and feasibility of using the stereoscopic highlighting as a highlighting technique in order to identify the important nodes in node-link diagrams, we carried out an evaluation study with participants from different backgrounds.

The goals of our evaluation study were:

- We aimed at evaluating the viewers' abilities in perceiving the variations in depth values. We wanted to know whether the depth variations could be used to encode information beside the color and the shape.
- We aimed to judge whether viewers are able to perceive the variations in depth more accurate and quicker with passage of time after getting experienced.
- Finally, we wanted to investigate the viewers' abilities to interpret the depth variations into some meaningful information related to the underlying diagram.

6.1.1 Study Design and Execution

The study design was following the Quasi-Experimental Design format [133]. It consisted of seven different configurations that were switched at random based in order to reduce the learning effect on participants during the controlled experiment. The participants started with a simple 2D node-link layout that was rendered into a 3D world, whereas all the nodes had the same shape, the same color, and the same depth value. Then we changed the depth values for the set graph nodes in order to create a 3D representation of the configuration. The participants saw only the front view of the layout and were not allowed to rotate the layout. In more advanced configurations, we added another visual cues such as the color or the shape to encode extra data attributes.

In this study, the participants evaluated our approach through task-based test technique [57] in a controlled experiment environment. At the end of the test, they provided their feedback through a closed-ended questionnaires form and an open-ended questionnaires form. Results of this study show the promising possibility of using highlighting technique in simple node-link diagrams accurately. However, they also show some limitations of this approach, as participants required some time and training to get used to this approach and the 3D environment. In the forthcoming subsections, we describe the evaluation experiments settings, details and the analysis of the results of each task, and the findings we obtained from this evaluation study.

The 3D Display Setting

We carried out the evaluation experiment tests using a Zalman stereoscopic 3D display equipped with 3D glasses. The Zalman 3D glasses are designed to create stereoscopic optical 3D image with reduced eye-strain. The display size was 60 cm with 16:9 wide as a display ratio. The resolution was 1920 x 1080 pixels with 5ms response time. The frequency rate was 38 $\sim$ 40 KHz in the Horizontal and 56 $\sim$ 75 Hz in the vertical mode. Each participant wore the polarized glasses by Zalman to view the 3D effects. They were seated directly in front of the 3D monitor. A precession was performed to insure that each participant was able to see the 3D effect accurately. We allowed each participant to adjust the height of the seat until they saw the 3D effects properly, with almost the same distance for all participants. Figure 6.1 shows a participant during the experiment wearing the Zalman 3D glasses and seated in front of the Zalman stereoscopic 3D display.

Hypotheses

Based on our observations, we were expecting the following results:

- **H1:** For all tasks in which the nodes representing components have the same colors and shapes, the three-layer configurations is the most readable configuration.
- **H2:** We expect that combining more than one visual cue with the depth cue does affect on the viewers' abilities in reading the required information.

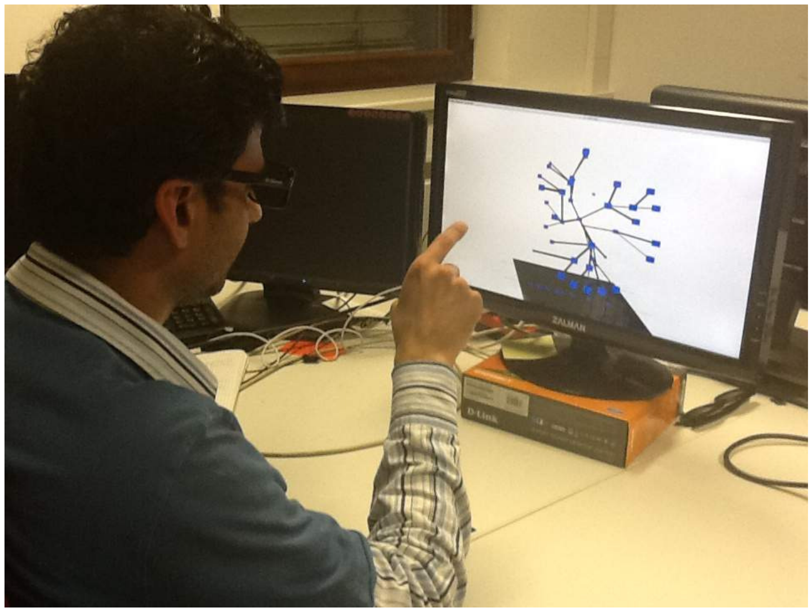


Fig. 6.1: A participant during the experiment.

- **H3:** Viewers get familiar with the variations in depth values by time. Therefore, the accuracy of the depth detection is increased by the passage of time.

User Groups

The participants in this evaluation study were from three backgrounds. The first user group, called the *safety experts*, consisted of participants having the background in the safety analysis of embedded systems. They were included in the study because this work was part of the ESSAVis framework in which they are one of the real end users. The second group, called the *Vis. Experts*, consisted of participants having the background in visualization area. They were included in the study in order to measure how the variations in depth values can be used in encoding some information aspects. While the third group, called the *non-experts*, consisted of participants from other backgrounds. They were included in the study in order to measure how accurately they detect the variations in depth and how they perceive it compared to the other two groups. At the beginning of each test, we asked the participant to give some personal information including age, gender, sight problem or any form of color blindness in order to analyze the results from different perspectives. We asked 20 participants (6 females, 14 males) to undergo our experiment. Out of these 20 participants; 7 were from the safety experts group, 7 were from the vis. experts group, and the remaining 6 belonged to the non-experts group. The participants' age ranged from 23 to 60 years with a mean age of 32 years.

The Experiment Layout

We implemented the visualization application using the Vrui toolkit [102], which provides a virtual reality development environment (see Chapter 7 for further details about the toolkit). The initial visualization consisted of a tree structure with 20 nodes in size (see Figure 6.2), representing a possible failure scenario using a component fault tree (CFT) model [93]. The resulted visual layout was done using the radial layout algorithm [59] in 2D. The tree nodes were represented using a set of cubes connected to each other through a set of tubes in the 3D world.

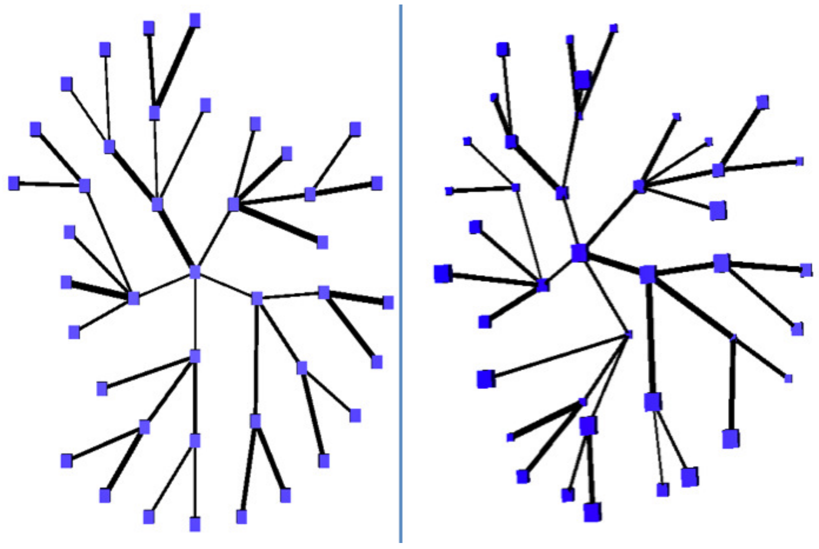


Fig. 6.2: The initial visualization of the abstract representation using the radial layout algorithm.

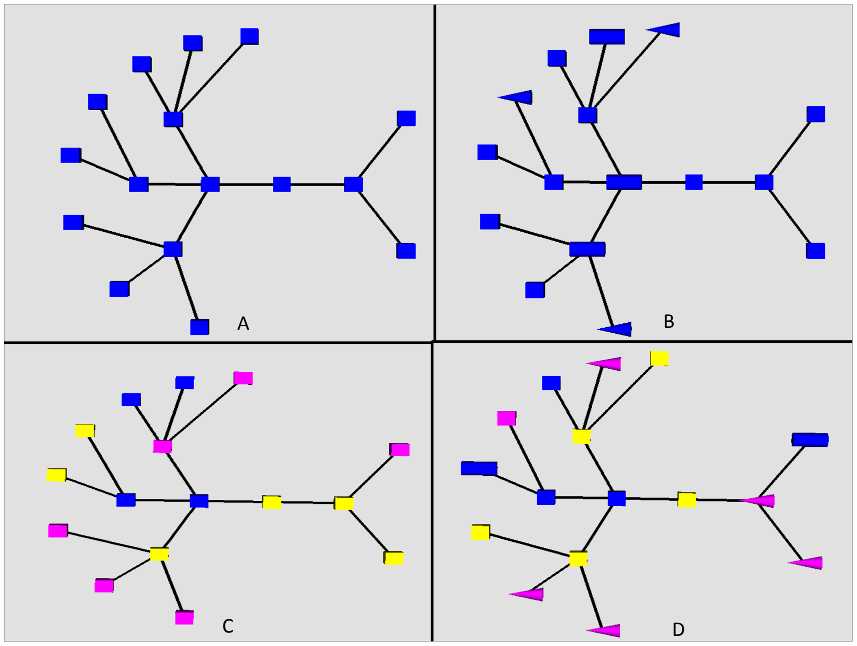


Fig. 6.3: Test Configurations: (a) all nodes with same color and same shape, (b) all nodes with same color but with different shapes, (c) all nodes with same shape but with different colors, and (d) nodes with different colors and different shapes

The experiment test in this study consisted of three tasks where each task was targeted to measure a specific goal. The description of each task is described as follows:

- **Task1 :**
  - *The Goal:* Task 1 targets at measuring the number of different layers that can be detected by the participants and the number of nodes associated with each layer.
  - *Features:* All nodes have the same size, color and shape (see Figure 6.3(a)). The nodes were shown in blue cubes in all configurations.
  - *Participated Configurations:* Configuration 1, 2, 3, 5, and 7.
- **Task2:**
  - *The Goal:* Task 2 aims at judging whether participants are able to perceive the variations in depth more accurate and quicker with passage of time after getting experienced.

  - *Features:* All nodes have the same size, color and shape (see Figure 6.3(a)). The nodes are shown in blue cubes in the configuration A of the Figure 6.3 with different depth values.
  - *Participated Configurations:* Configurations 3 and 7.

- **Task 3:**
  - *The Goal:* Task 3 aims at measuring whether the depth variations can be used to encode information beside other (*single or multi*) visual cue like shape or color (see Figure 6.3).
  - *Features:* All nodes have the same size with different visual cues (like shape, color, or both) and with different depth values.
  - *Participated Configurations:* The nodes in configuration 3 have different colors (i.e., blue, pink, and yellow) with same shape, the nodes in configuration 6 has different shapes (i.e., cube, cone, and cylindrical) and same color but with different depth values, and the nodes in configuration 8 have the same size having different colors with different shapes at different depth values.

Study Execution

For all participants we kept the same order of tasks, but the given sequence was not following the same hardness pattern as we intended to switch from easy task to a harder ones and vice versa. This was because we wanted to train the participants to read the configuration at random base. Through this, we could measure easily how participants react with the variations of the scenario. In order to avoid the learning effect for the participants, the configurations were different for each task and the animation was used to show changing process between the configurations in a real time. For each task, we gave participants the option of skipping the task if they feel very confusing.

The order of the tasks was carefully assigned as we started with an easy configuration to train participants to use variations in depth values as a cue to read some data attributes. After that we switched to a harder task, in which participants had to detect the critical nodes based on their depth values. We switched between four different layer depth values. The number of layers ranged from 2 to 5. Results show (see Figure 6.4 and Figure 6.5), that participants detected the three layers configuration more accurately than the four or five layers configurations. All participants detected very easily the two-layer configuration but the majority preferred the three-layer configuration because it conveys more information.

At the end of the experiment test, we gave each participant a closed-ended questionnaires form and an open-ended questionnaires form in order to get their feedback about the overall environment and the proposed approach. The closed-ended questionnaires form consisted of 12 questions where for each question participant had six options (*strongly disagree, disagree, neutral, agree, strongly agree*, and *don't know*) based on the likert scale. In addition to these two questionnaires forms, the safety experts group was given an addition questionnaires form in both categories targeting to the embedded systems environment. Each participant was given maximum 30 minutes of time excluding the training time.

6.1.2 Results and Discussion

The results of this study provide an indication of the possibility of accurately using the depth cue as a highlighting technique in stereoscopy devices. The average accuracy of all groups in all configurations (with same color and same shape) of task 1 is 86% (see Figure 6.4). This result is less than (0.94) that was achieved by [5]. This is because of using a different stereoscopic device technology, a different user group size (i.e., 20 participants from different backgrounds), and also as our test itself had different sets of tasks. The average time for all configurations (with same color and same shape) of task 1 and for all groups is 48.57 seconds (see Figure 6.5), which is higher than the average time (15.46 second) in [5], due to the difference in the test setting and the test configuration.

However, results show that participants get experienced with time as configuration 3 and configuration 7 had the same number of layers, but the order of their appearance in the experiment was different. The configuration 3 appeared before the configuration 7 and results indicate (see Figure 6.6 and Figure 6.7) that the accuracy and the average time of participants' picking of nodes and layers at different depth levels improved significantly over the time.

The goal of task 1 was to determine how strong the number of layers could affect the viewers' performance in detecting the different layers correctly. Configuration 2 had 4 layers while configuration 5 had 5 layers; therefore, participants needed more time to detect the variations

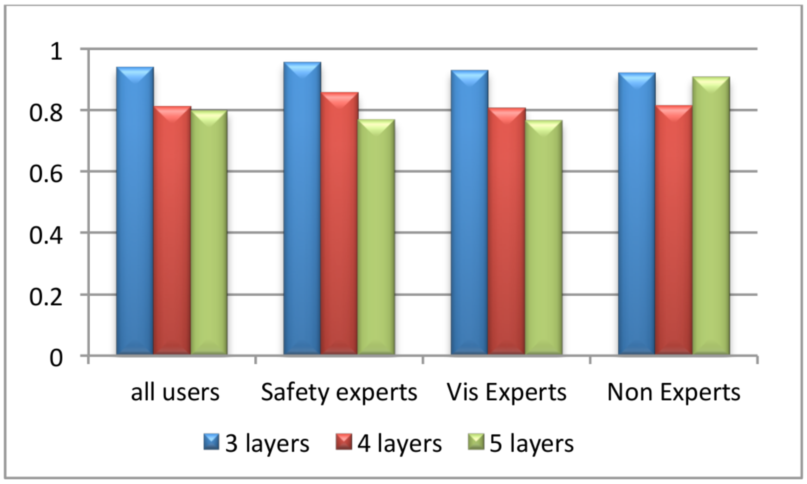


Fig. 6.4: Average accuracy for all configurations in Task 1.

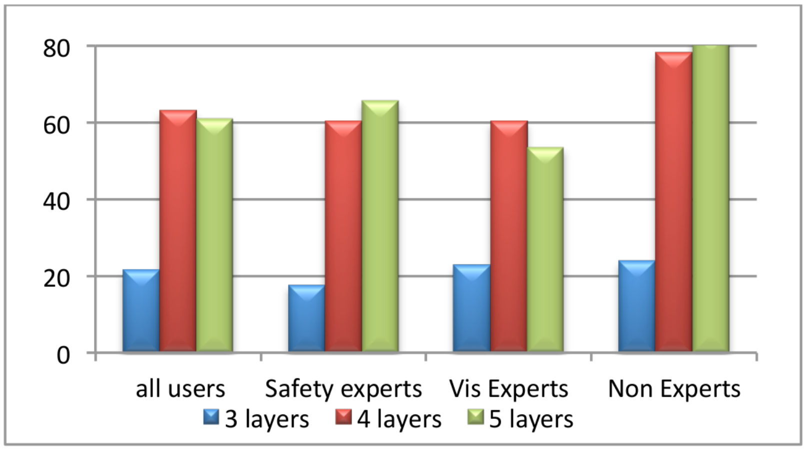


Fig. 6.5: Average time in seconds for all configurations in Task 1.

among the depth values with less accuracy. Results indicate (see Figure 6.4 and Figure 6.5 - where 4 layers bar is for the configuration 2 and 5 layers bar is for the configuration 5) that average accuracy is reduced as the number of layers goes beyond 3 layers. In the open-ended questionnaires, we asked the participants about the number of layers to detect easily and 95% participants chose 3 layers as the most readable configuration.

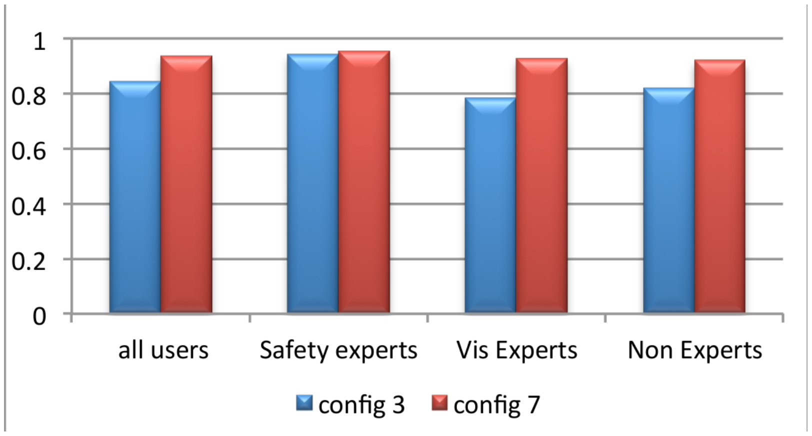


Fig. 6.6: Average accuracy for configurations 3 and 7 in Task 2.

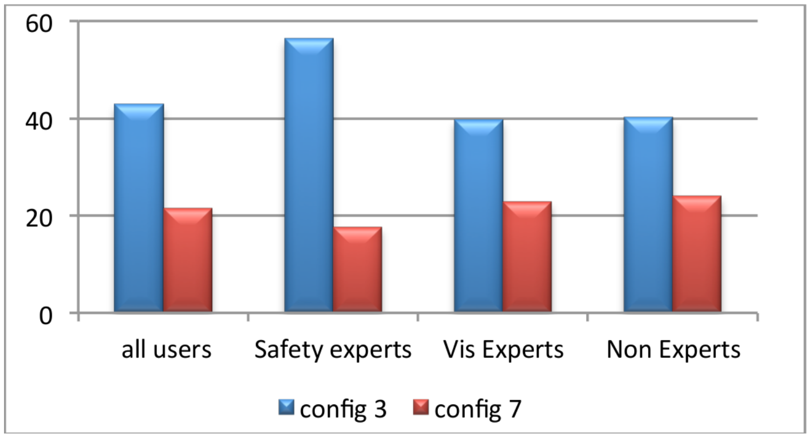


Fig. 6.7: Average time in seconds for configurations 3 and 7 in Task 2.

Configuration 4 was dedicated to measure the effects of adding different colors to the depth cue and configuration 6 was dedicated to measures the effect of different shapes but with unified color. In configuration 8, we combined three visual cues (i.e., color, shape and depth) together. The overall goal in these configurations was to measure the accuracy of the depth cue in presence of color and shape encoding. The performance of participants in task 3 indicates (see Figure 6.8 and Figure 6.9 - where the blue bar refers to configuration 4, the red bar refers to configuration 6, and the green bar refers to configuration 8) that adding other visual cues to the set of configurations 4, 6 and 8 did not affect much the accuracy of the depth detection as a highlighting technique. As a comparison of the results between these three configurations, participants' accuracy in the configuration 4 is overall higher than those of the other two configurations. The participants' efficiency in the colored-node's configuration (88.57%) is superior to the other two configurations (83.09% for the shaped-node and 84.52% for the combined color and shape configuration).

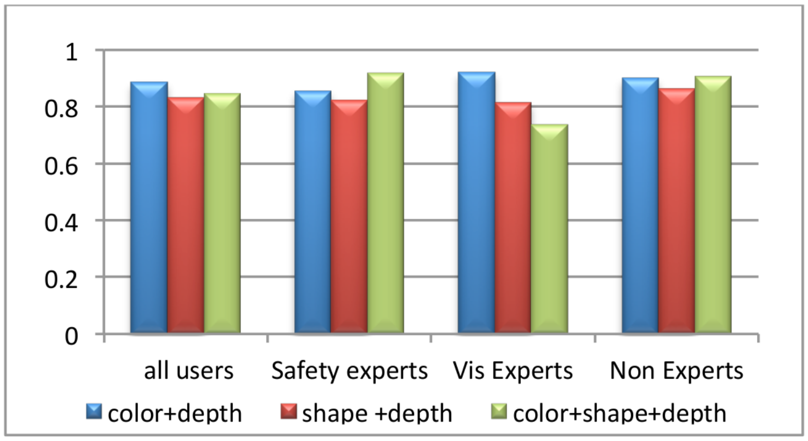


Fig. 6.8: Average accuracy for configurations 4 (color+depth), 6 (shape+depth), and 8 (color+shape+depth) in Task 3 with combined visual cues.

Participants described that the color cue with the depth cue convey the information intuitively. Some participants experienced that colors supported the understanding of the scenario more quickly. Most of the participants answered in the open-ended questionnaires form that different colors or different shapes were easy to detect in each layer, without doing any affect on the depth cue for the node. This finding is an indication of the possibility to combine other visual cues with the depth cue. This gives us the possibility to keep colors or/and shapes to encode other attributes for the underlying data, but with longer required time.

Overall, results (see Figure 6.4 and Figure 6.8) show that experts (either safety or visualization) and non-experts achieved almost the same accuracy, which gives us an indication of the understandability and usability of the stereoscopic highlighting for the public usage. On the other hand, results also indicate that people need some time to get

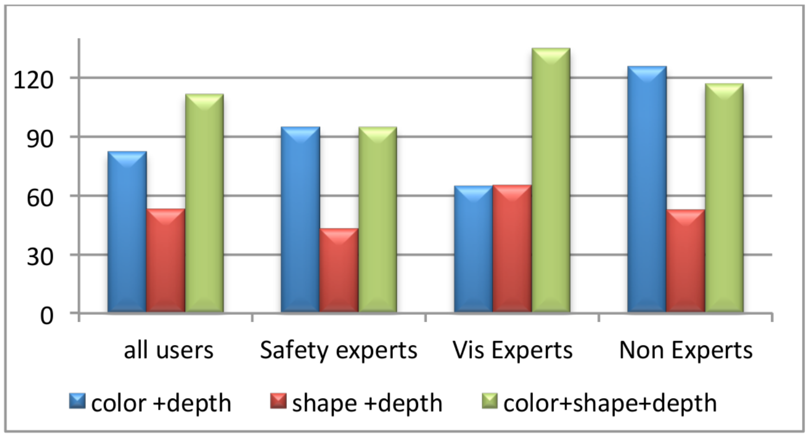


Fig. 6.9: Average time in seconds for configurations 4 (color+depth), 6 (shape+depth), and 8 (color+shape+depth) in Task 3 with combined visual cues.

familiar with the technique and with the stereoscopic devices as well as some time to adjust the sight with the depth cue. Moreover, the standard deviation value of the accuracy for all participants in all configurations is around 0.014, which indicates that participants' performance in all configurations was approximately the same. However, the deviation of the required time for all configurations is around 14.3 seconds. This gives us an approximate vision about the differences in the participants' response time for the depth variations. Some participants needed more time to read the configuration while others reacted faster with the changing process. For example, we observed that females were slower than males especially when dealing with high number of layers.

Although, our data size was not big enough to generalize this set of findings. But, it can be useful to give an impression of the possibility of using such technique in graph visualization applications. We believe that combining this technique with other highlighting techniques can be more effective in solving issues regarding the scalability aspects in 2D graph visualizations. However, this needs to be proved by investigating this technique with different data sizes and with different stereoscopic display devices in different application domains.

6.2 The Reflection Layer Extension

Utilizing the stereoscopic depth as a highlighting technique suffers from many potential drawbacks, e.g., viewers need more time to realize the depth variations in stereoscopic displays compared to when using colors [9], which can be observed instantly. Therefore, the depth cue needs to be optimized in a manner that viewers can read the depth variations faster and more accurately.

In the previous section, we provided details of an evaluation study in which we measured the accuracy of common viewers in detecting the number of depth layers in a stereoscopic display. In that study, viewers were unable to give highly accurate results for the nodes and their specified layers. As mentioned earlier, the average accuracy was 85% that is not sufficient for critical applications, especially in our case where our target application domain was safety critical systems. The reason behind this average accuracy was that many times the viewers were unable to read the target configuration properly.

In [129], a set of approaches has been proposed to enhance users' perceptions in detecting the depth cue. Few examples are: the size of the node for indicating the distance of the node from the viewer, lighting cue for enhancing the depth variation such that the lighter nodes look closer to the viewer while the darker nodes look farther, the shadow for conveying information about the hierarchy of the tree.

To enhance the viewer's perception in detecting the 3D impression, we present the solution by providing an extra layer, called the **Reflection Layer**, at the bottom of the graph to support the viewers in detecting the depth cue faster and more accurately. This reflection layer is achieved by projecting the set of nodes of the 3D node-link diagram onto a 2D plane at the bottom of the rendering space in order to increase the detection speed of the nodes in different layers, as shown in Figure 6.10.

Our enhancement to the stereoscopic highlighting technique is based on adding a reflection layer with additional characteristics such as the *layer line* and the *depth line*. Figure 6.10 shows a prototype that we implemented to evaluate (see the forthcoming section) our proposed solution. Below is the explanation of our extension:

- **The Reflection Layer:** The main extension to the original stereoscopic highlighting technique is the addition of a reflection layer

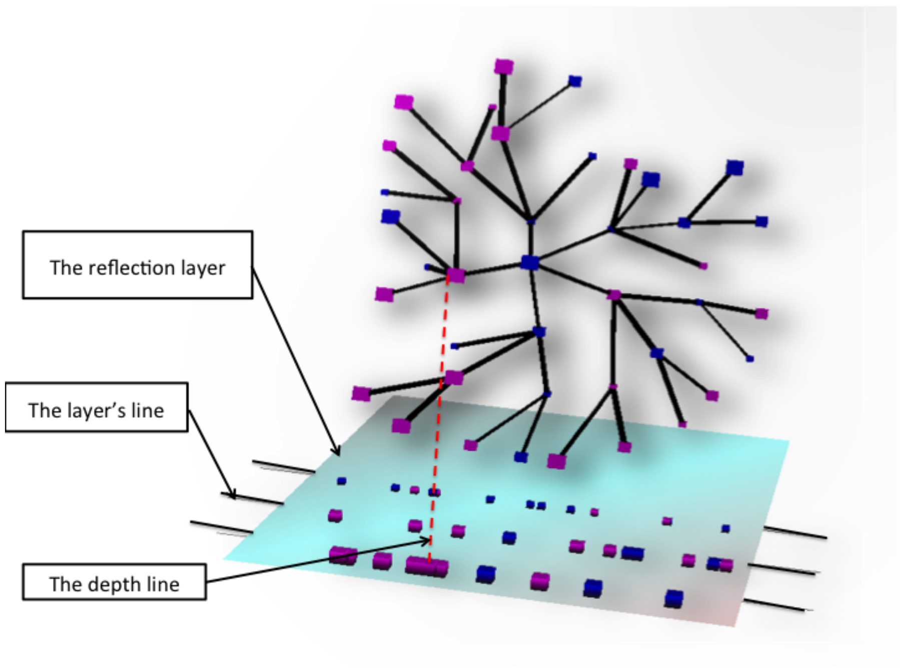


Fig. 6.10: The reflection layer extension.

at the bottom of the rendered graph. The purpose of this layer is to show the projection of nodes from the 3D space into a 2D plane orthogonal to the 3D rendered graph. Each node is placed on the layer according to its depth value. This reflection layer also indicates the number of the different depth layers, as it arranges the nodes according to their depth values.

- **The Layer's Line:** This is illustrated in Figure 6.10, where a horizontal line is drawn on the reflection layer to discern the specified layer.
- **The Depth Line:** This is an on-demand option. Viewers can use it to see the connection between the node in the 3D space and its reflection on the reflection layer. This is drawn as a dashed red line between the node and its projection.

6.3 Evaluating the Reflection Layer Extension

We carried out another evaluation study for evaluating the accuracy of our presented *reflection layer* extension to the stereoscopic highlighting technique. We performed the evaluation study through two main settings in order to check the effectiveness and the feasibility of our proposed solution on viewers' performance. The first setting was the standard one, i.e., without the reflection layer (as mentioned in Section 6.1), while the second setting was with the reflection layer that we presented in the previous section. In this study, participants from different backgrounds evaluated our reflection layer solution in a controlled environment through task-based evaluation tests [57], and then provided an overall feedback through sets of closed-ended and open-ended questionnaires forms. The results of the study provide an indication of an increment in the viewers' abilities in detecting the depth more accurately with less time using our extended technique compared to the standard one. This also indicates the usefulness of our reflection layer solution in reducing the limitations of the stereoscopic highlighting approach.

The 3D display setting during the controlled experiment was same as described earlier in Section 6.1.1. The participants were seated directly in front of the 3D monitor. A pre-session was performed to insure that each participant was able to see the 3D effects accurately. We allowed participants to adjust the height of the seat until they saw the 3D effects properly, with almost the same horizontal distance for all participants.

In the forthcoming subsections, first we provide the details about the study design and execution and then we discuss and analyze the results.

6.3.1 Study Design and Execution

Study Goals

The goals of our second evaluation study were:

- To measure the viewers' abilities in perceiving the variations in depth values with and without the reflection layer.
- To know whether the depth variations can be used to encode ordinal data attributes beside other visual cues, like color or/and shape with and without the reflection layer.
- To judge whether viewers are able to perceive the variations in depth values more accurately and quicker with passage of time after

getting more experience. Moreover, the comparison of time for each task with and without the reflection layer.
- Finally, to investigate the viewers' abilities in interpreting the depth variations into some meaningful information related to the underlying diagram.

Hypotheses

We hypothesized that using our reflection layer solution in the stereoscopic highlighting technique enhances the viewers' abilities of reading the configuration faster and more accurately. Viewers could detect the number of different layers directly using this reflection layer and estimate the number of nodes associated with each layer faster and more accurately. Hence, the viewers' perceptions are enhanced through using this reflection layer extension. Consequently, the viewers' accuracy in detecting the depth variation is also increased.

Here, we present the set of hypotheses that we diagnosed for our this evaluation study. Based on the results of our previous study (described in Section 6.1), we observed that viewers were able to read the variations in depth levels for small data set; so in this study we expected:

- **H1:** The accuracy value is irrelevant to the data size but it requires longer time for viewers to read a configuration with big data size.
- **H2:** The accuracy of detecting the variations in depth values will be significantly increased by adding an extra layer (i.e., the *reflection layer*) at the bottom that reflects the current configuration at the bottom of the 3D world.

User Groups

The participants in this study were from two backgrounds. The first group, the *vis. experts* group, consisted of participants having background in the visualization area. While the second group, the *non-experts* group, consisted of participants from other backgrounds. At the beginning of each test, we asked the participant to provide some personal information including age, gender, sight/perception problems or any form of color blindness in order to analyze the results from different perspectives. We asked 8 participants (2 females, 6 males) to undergo our experiment. Out of these 8 participants, 4 were from the vis. experts group while the remaining 4 belonged to the non-experts group. The participants' age ranged from 24 to 34 years with a mean age of 28.5 years. Four of these participants (two from each group) also participated in the first evaluation study while the remaining four were participating for the first time. Half of these participants wear corrective glasses in their normal routine life.

Experiment Layout and Execution

Our initial 2D visualization showed a tree structure consisting of 41 nodes, as shown in Figure 6.11. In Figure 6.11a, we show the initial appearance of the layout in a 2D representation, while in Figure 6.11b and Figure 6.11c we show the 3rd dimension impression of the layout with different configurations. The layout is rendered using the Vrui package [102]. The color cue was used together with the depth cue, as in the previous study (see Section 6.1) we observed that different visual cues do not affect viewers' abilities in detecting the depth cue.

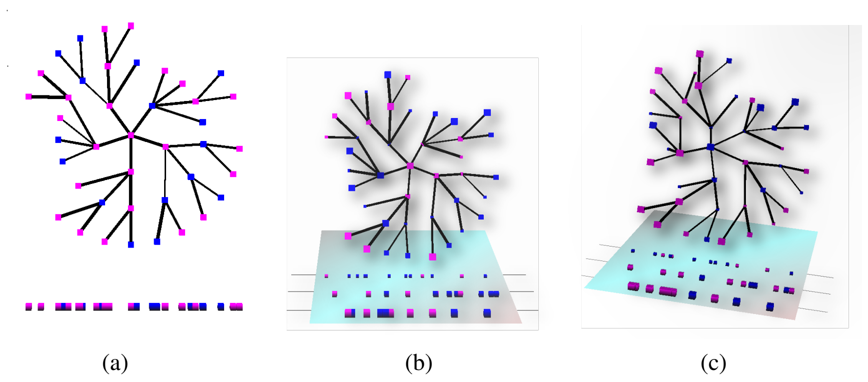


Fig. 6.11: An experiment's sample for the three layers configuration. In *(a)* we show the initial setting, in *(b)* we show the front view of the graph when the depth values being changed, and in *(c)* we show a side view of the same configuration to visualize the 3D effect.

For each configuration we provided two possibilities, the first one was with the reflection layer while the second one was without it. To reduce the learning effect, the appearance of the reflection layer was randomly based from the viewers' perspectives. However, the order of the tasks appearance was carefully settled to ensure viewers' abilities in reading both configurations. We kept the order of tasks same for all participants but the given sequence varied in difficulty, as we wanted to get results from an unbiased perspective. The participants had the option of skipping the task if they feel confused. Table 6.1 provides the tasks configurations in our this controlled experiment.

Table 6.1: The second evaluation study tasks' configurations.

Task ID	**No. of layers**	**The reflection layer**
Task 1	3 layers	No
Task 2	4 layers	Yes
Task 3	4 layers	No
Task 4	5 layers	No
Task 5	3 layers	Yes
Task 6	5 layers	Yes

For the experiment, we added a simple animation during each test to indicate the changing process between the depth values of graph nodes in the set of configurations in real time. Each participant started with a simple 2D node-link layout rendered in a 3D context, as it is shown in Figure 6.11a where all nodes have the same shape but with different colors. Then we changed the depth values for a predefined set of graph nodes to create a 2.5D representation of the configuration, as shown in Figure 6.11b. The participants were able to see only the front view of the layout and were not allowed to rotate the layout. We tried to measure the accuracy of using this technique by giving a rough indication of the number of layers that a viewer can really read and can use to encode some information about the data itself. The results from the first study (see Section 6.1) indicated that most of the participants detected the three layers configuration more accurately than the four or five layers configurations. We were curious whether our reflection layer scheme at the bottom improves significantly participants' accuracy in detecting the number of layers and in picking the right number of nodes at these layers compared to the previous study, as well as the overall reduction in time for performing the tasks.

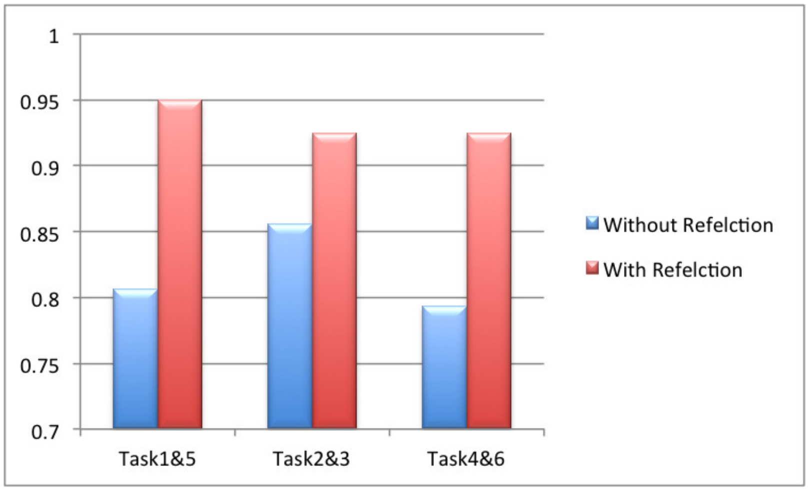
At the end of all task-based tests, we gave participants a closed-ended questionnaires form and an opened-ended questionnaires form in order to get their feedback and impressions regarding the overall environment and the proposed approach, especially related to our reflection layer scheme. The closed-ended questionnaires form consisted of 13 questions. For each question, there were six options (*strongly agree, agree, neutral, disagree, strongly disagree,* and *don't know*) based on the likert scale. Each participant was given a maximum of 45 minutes to complete all the six tasks. This excludes the training time and the answering time to the questionnaires.

6.3.2 Results and Discussion

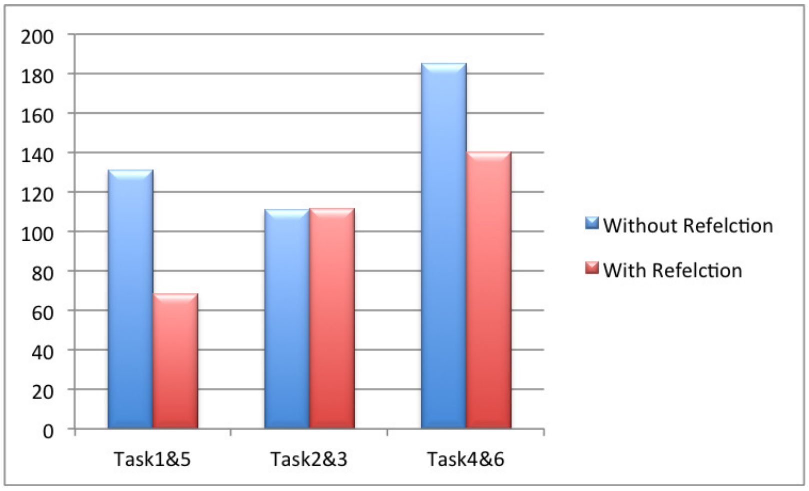
The results of this evaluation study provide an indication that the accuracy using the stereoscopic depth as a highlighting technique in stereoscopy devices is not affected much by the data size, although viewers needed more time to find different depth levels and the number of nodes at each layer. When comparing the results of this study with the results of the previous study (see Section 6.1.2, we observe that the accuracy was 86% for the previous data size (16 nodes) while the accuracy of this study data size (41 nodes) is 81% for the configurations without the reflection layer.

Figure 6.12a shows the results of all participants' accuracy with two options, with and without the reflection layer. When analyzing the results of all configurations, we observed that the accuracy values are higher in the cases of using the reflection layer than the cases of without using it. Overall, the average accuracy value for all those tasks that were supported by the reflection layer (i.e., task 2, task 5, and task 6) is 93% compared to 81% accuracy value for the set of tasks (i.e., task 1, task 3, and task 4) that were not supported by the reflection layer. The results show, compared to the previous study (see Section 6.1.2) where the overall accuracy average for tasks with same color and shape was 86%, that relatively larger nodes number does not influence significantly to the accuracy level. In fact, the added bottom reflection layer increased the accuracy compared to the first study even though with larger data size and also compared to the tasks without bottom reflection layer support in this evaluation study.

The average time (see Figure 6.12b) for all tasks without the reflection layer support is 142.4 seconds (i.e., the combined mean time of task 1, task 3, and task 4), while the average time for tasks with the reflection layer support is 108.8 seconds (i.e., the combined mean time of task 2, task 5, and task 6). This indicates that the our proposed reflection layer solution supports significantly the viewers in judging



(a)



(b)

Fig. 6.12: (a) Average accuracy of all tasks for all user groups. (b) Average time of all tasks for all users groups.

the number of layers, as through this it is obvious for them to understand the depth levels in the underlying configuration. Moreover, it also helps the viewers in understanding and in counting the nodes at each layer faster compared to without the reflection layer support. Few of the participants mentioned in the open-ended questionnaires feedback that they used the our reflection layer support when they were confused about the number of nodes at some particular layer.

The standard deviation value for all participants' accuracy in all tasks without using the reflection layer was 0.018 while using the reflection layer was 0.007. This indicates the homogeneity in participants' answers. However, there is no much difference in respect to the deviation of the accuracy value in both options. However, there is a significant difference in the time deviation value. The standard deviation for the time in the tasks without using the reflection layer support is 16.46, which is relatively large value indicating the heterogeneity in their results. While the standard deviation for the time in the tasks with using the reflection layer support is 4.38. This indicates the high agreement level of participants' performance in all tasks.

By comparing the results of tasks with the same number of layers without and with the reflection layer support, we conclude that our presented reflection layer solution improved significantly the participants' performance both in the accuracy and in the time performance. Overall, the results support our set of hypotheses and we can conclude that the data size does not affect the accuracy significantly. Moreover, over time after getting experienced with our reflection layer support, it helps the viewers in understanding the depth levels and in picking the right component at the right depth level. In the questionnaires feedback, 7 out of 8 participants strongly agreed that our reflection layer scheme helped them in finding the number of depth levels. Further, 6 out of 8 participants agreed that they used this reflection layer in judging the depth levels and for picking the components at different layers. All of them said that they would prefer to work in an environment supported by our reflection layer approach.

6.4 Stereoscopic Filtering Technique

Querying some aspects about large graphs requires a big amount of time for viewers to perceive or to perform. To solve this problem, we present a new technique, derived from the stereoscopic highlighting approach [5], that helps the viewers in detecting and filtering important topological features about the set of the sub-graphs within the large graphs. We call it the **Stereoscopic Filtering Technique** (**StFT**). In this technique, the depth is used as a filtering axis for some data aspects. It performs very well in stereoscopic devices; however, it can be modified easily for working in 2D devices. In this section, we explain the core idea of the StFT and present some initial results.

6.4.1 Topological Feature-Based Layout

The feature-based graph drawing concept, called as the *TopoLayout*, was firstly proposed by Archambault et al. in [25]. It is a multi-level feature based approach for drawing large graphs.

The pipeline of this approach consists of four main steps: The first step is called the *decomposition* phase in which the graph is decomposed into many sub-graphs, based on the topological features for each internal sub-graph. For example, if a set of nodes in one sub-graph is topologically connected with each other as a tree then this set of nodes are grouped together into one node, called a *meta-node*, that represents them. Currently, TopoLayout detects seven topological features including trees, complete graphs, bi-connected components, clusters, and the undefined structure that is called the unknown features [25]. The second step is called the *feature layout* phase in which the meta-nodes or the grouped sub-graphs are layout using one of the layout algorithms tuned with its topological feature. The third phase is called the *crossing reduction* phase, which aims to eliminate the crossings ratio in the resulted layout. Finally, the fourth step, the *overlap elimination* phase, aims to change the node sizes in the final layout to ensure that none of the nodes overlap each other. The final result of TopoLayout is a tree representing the graph hierarchy, in which each node represents a sub-graph in the original graph and each meta-edge represents the relation between two sub-graphs in the original graph.

This layout technique helps in drawing relatively large graphs. Also, it provides the viewers the required details about the internal structure of the graph, which can be useful for extracting more information about the graph itself (see Figure 6.13).

6.4.2 The Filtering Approach

Our **St**ereoscopic **F**iltering **T**echnique (**StFT**) utilizes the depth cue to expand the underlying graph such that the interested feature or the required sub-graph comes to the front plane, closer to the viewer rather than the remaining parts of the graph. This makes the highlighted sub-graph in focus while the rest of the graph is rendered in the background

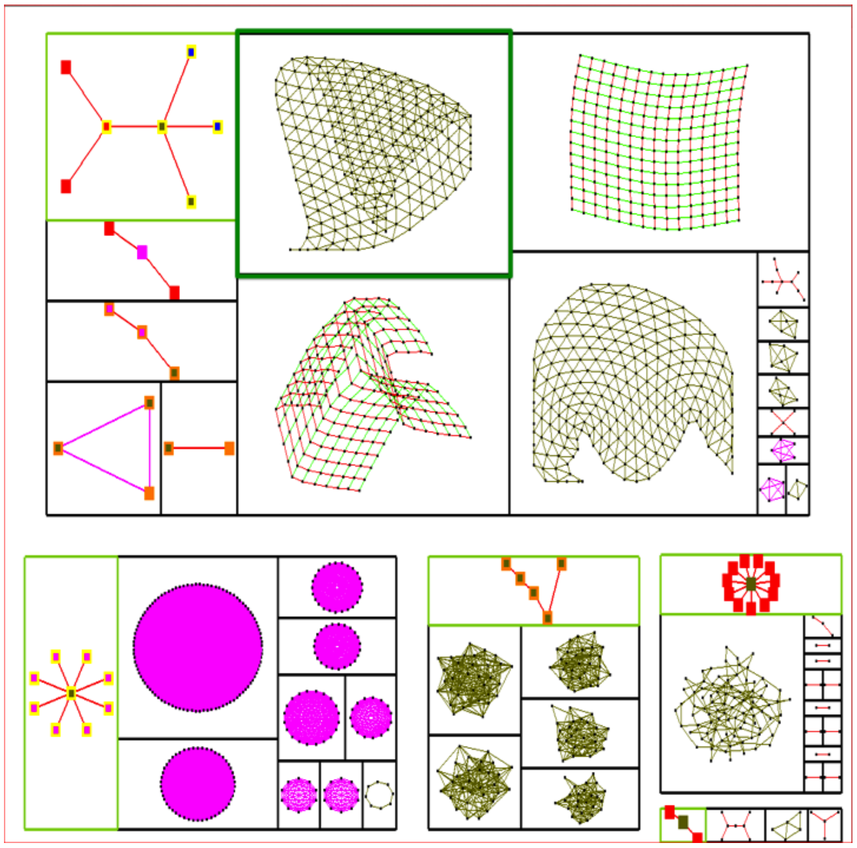


Fig. 6.13: Few examples of layouts using TopoLayout algorithm (images are produced using the libgraph package [78]).

of the scene with a transparent appearance. This helps the viewers in keeping the context of the selected sub-graph.

The input in our technique is the result of the TopoLayout algorithm. In this case, each sub-graph within the large graph is rendered using an algorithm tuned to its topological feature. Further, the sub-graph is also colored according to its topological feature. To query about single sub-graph using the current TopoLayout algorithm, viewers could select one meta-node from the meta-tree and then the borderline of the corresponding sub-graph is colored in green. Although this feature may work for querying about one sub-graph, it does not help viewers in getting more information about the number of a specific topological feature in the current level, e.g., the number of trees. Also, TopoLayout does not give to viewers the ability to explore one specified sub-graph separately.

Our StFT approach solves these issues by introducing the 3rd dimension as a filtering parameter to filter the specified set of sub-graphs together. This is done by bringing the specified set of sub-graphs to a plane closer to the viewer while placing the remaining parts of the graph to the background in a further plane. This gives the viewers the feeling of the “irrelevancy” of the remaining graph and let them focus on the closer plane, which contains the “relevant” information. This technique lends itself to the *focus+context* approach [5] naturally.

The advantage of our technique over the 2D graph filtering techniques is its ability to expand in depth, which gives more space to the filtered sub-graph. This helps in rendering the required sub-graph in a separate plane that gives the viewers the chance to explore, rotate, zoom, and interact with it individually. Viewers can query about the topological features of the current sub-graph, and can also expand it in depth to get another level-of-detail in depth. This allows the viewers to query about the multi-level of details on the graph.

One possible problem is the cluttering issue that might appear in the final layout due to the multi-planes overlapping. However, this can be handled by controlling the transparency level of the irrelevant planes, i.e., the planes in the deeper depth towards the screen (e.g., the overall graph, which is the background of the entire layout). Also, this can be handled easily in stereoscopic devices as they support the depth cue impression. In stereoscopic devices, viewers could distinguish between different depth layers easily because of the depth cue support in such technologies. This has already been proved by Alper et al. in [5], where an evaluation study was conducted to show the feasibility of using the depth cue as a highlighting parameter in stereoscopic devices. In this regard, wee have already discussed our conducted evaluation studies in the earlier sections of this chapter.

6.4.3 Implementation

We implemented a simple prototype of our stereoscopic filtering technique, which offers many possibilities to explore large graphs drawn using the TopoLyout algorithm. The first option was to explore one specific topological feature, e.g., the number of trees in the current level of details. To achieve this, the current prototype offers a menu interaction facility where viewers can select an option to show the required topological feature, , e.g., Trees. As a result of this action, a new plane – closer to the viewer – is created having all the selected topological features. In this case, all the trees will be rendered in the highest plane while the remaining graph will be rendered at further depth in the background of the scene.

In our current implemented prototype, we can change the transparency value of the further planes such that they look more transparent in order to reduce the cluttering ratio between the overlapped planes. To keep the context of the graph exploration task, we color the borderlines of the selected topological feature from the original graph in green. This helps the viewers for investigating the original places of the selected topological features in the initial graph.

The other option, offered by our prototype, is the facility of exploration of only one targeted sub-graph. To achieve this, we offer a 6-degree of freedom mouse interaction facility to interact with the 3D scene. This option allows the viewers to select only one targeted sub-graph. This makes the selected one to pop-up in a closer plane where it is ready to be explored. Also, viewers can select another sub-graph from the last selected one for more expansion. This creates a multi-plane exploration world, as shown in Figure 6.14.

The current prototype also offers the possibility of toggling between the different planes in the scene in real time. We use a slow animation to show the changing process of the different planes in the space. We designed this to help the viewers in following the changing process of the planes in the 3D scene. In the forthcoming subsection, we show some preliminary results of the prototype.

6.4.4 Results

The current prototype was implemented using two packages. This first one is the Libgraph package [78], which supports graph visualization techniques and many graph algorithms. The second one is the Vrui package [102], which is a toolkit to provide a virtual reality (VR) development environment. The Vrui package shields the application developing process from the particular configuration of the VR environment. It works on many different hardware platforms. Therefore, it is possible to render the application in many different environments such as 3D displays, Power-Wall displays, or CAVE systems. This

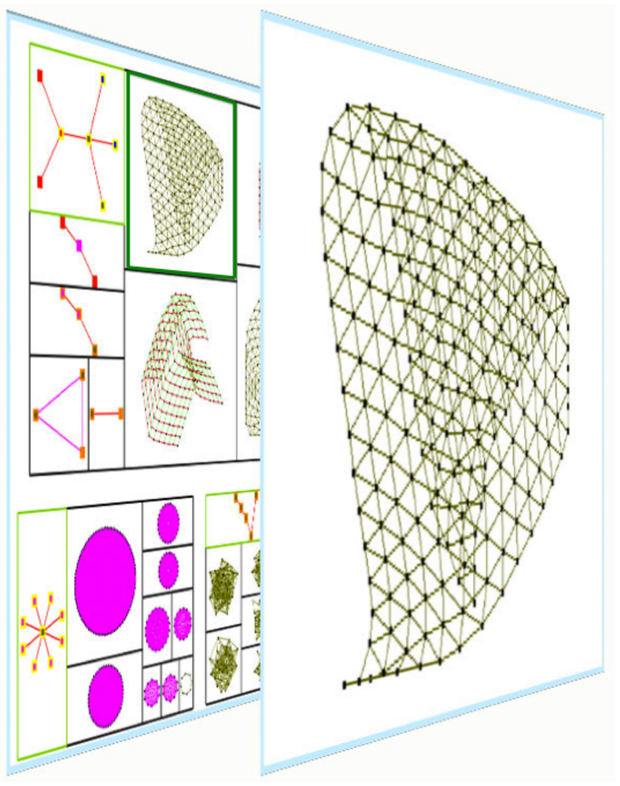


Fig. 6.14: The *Side-View* shows the 2.5D layout of the graph, where the side-view of the whole world is depicted for showing the different planes in the scene.

feature increases the scalability and the portability of our prototype into different environments, which also helps in showing the depth cue in the 3D world more precisely.

In Figure 6.14, the *side-view* of the graph after selecting one specified topological feature is shown. The side-view shows the number of current planes in the scene and the relation between them is revealed using the color. In this example, a mesh-like sub-graph was selected using the mouse interaction technique, which was then popped-up into another plane for showing the other level of details of the selected sub-graph. In Figure 6.15, the *front-view* of the same example is shown. In this view, viewers are able to see the two planes: the front one shows the active plane while the second one is more transparent and showing the background of the scene, which is the initial graph.

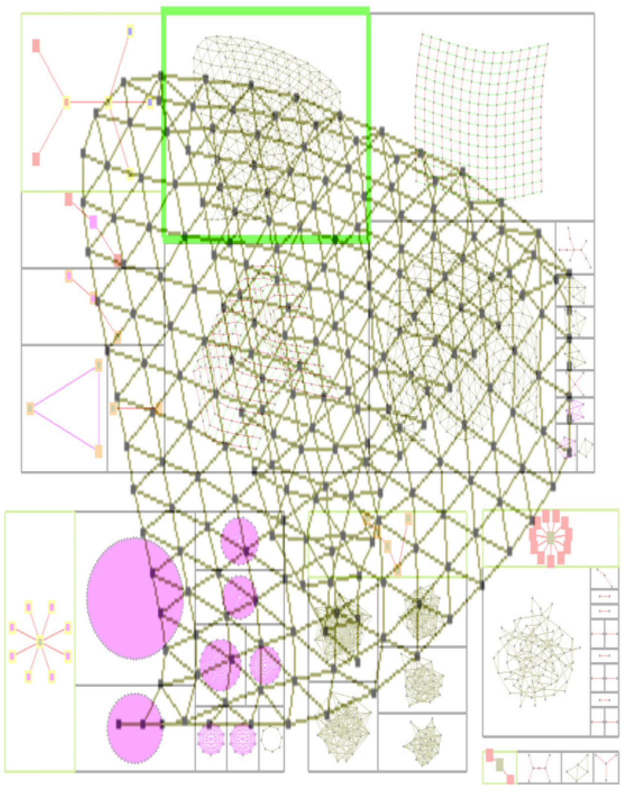


Fig. 6.15: The *Front-View* of the selected green bounded sub-graph.

In Figure 6.16, we show the multi-level front-view of a sub-graph with 3 levels of details. While in Figure 6.17, we show the multi-level side-view of the same query. From this example, we found out the cluttering issue that might appear due to the extra number of planes. However, this problem is less significant in 3D displays due to the natural support of the depth cue.

6.5 ExpanD: A Stereoscopic Depth Expanding Technique for Compound Graphs

In this section, we present the idea of utilizing the stereoscopic depth to encode the structural relations between the nodes of compound graphs. In compound graphs, there are two main relation types: the *structural* relations and the *adjacency* relations. One of the main traditional visu-

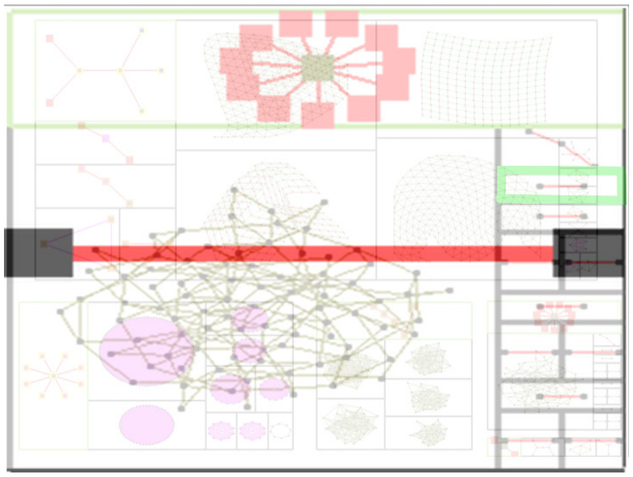


Fig. 6.16: The *Front-View* of the 3 levels of details query.

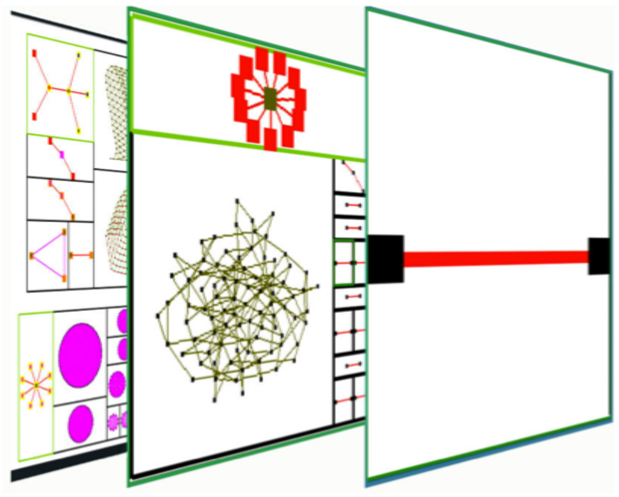


Fig. 6.17: The *Side-View* of the 3 levels of details query.

alization schemes for the compound graphs is visualizing the structural relations using the containment representations. In this case, the compound node includes the internal structure of it. The adjacency relations in this approach are represented using the explicit edges between the nodes.

We leverage the usage of the 3D technology in the graph visualization domain in a way that we allow viewers to expand the compound nodes in the 3D world. For this, we present a new technique, called **ExpanD** (**Expan**d in **D**epth), for expanding the compound nodes in depth. It helps the viewers in selecting a compound node in the 3D world to be expanded. When the viewer selects a compound node in the 3D world, a new plane is created closer to the viewer containing the children of the expanded node. This expanded node is rendered in a layer closer to the viewer (see Figure 6.18) to show the effect of the stereoscopic depth.

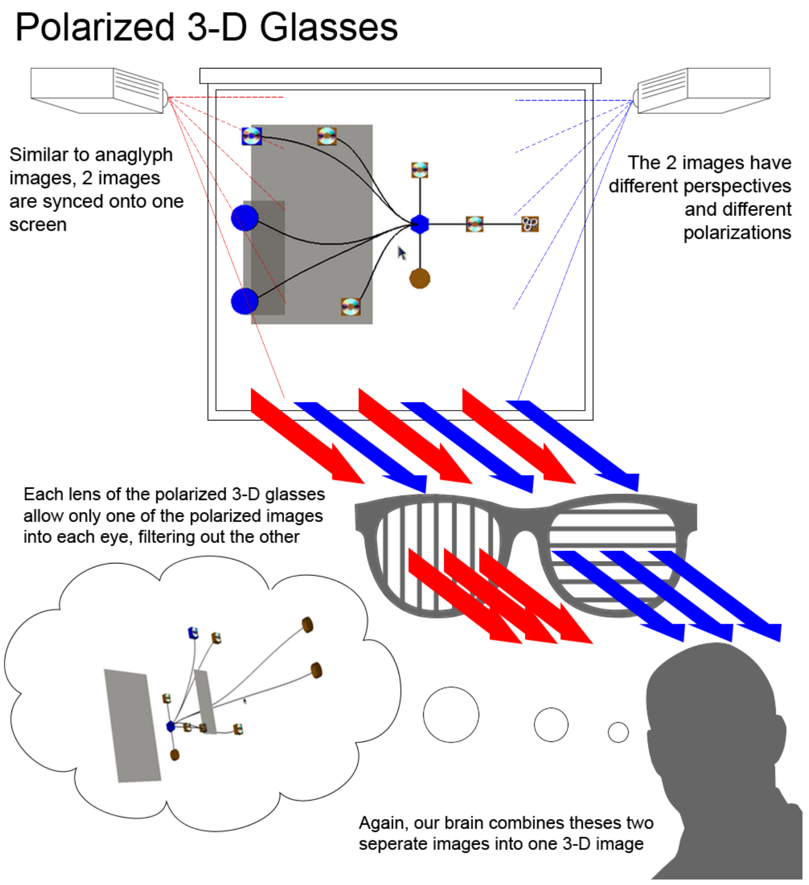

Polarized 3-D Glasses
Similar to anaglyph images, 2 images are synced onto one screen
The 2 images have different perspectives and different polarizations
Each lens of the polarized 3-D glasses allow only one of the polarized images into each eye, filtering out the other
Again, our brain combines theses two seperate images into one 3-D image

Fig. 6.18: An illustration to show the human brain perceiving of the stereoscopic depth using polarized 3D glasses.

We designed and conducted a controlled evaluation study where we invited 30 participants from different backgrounds to evaluate our

ExpanD technique using different configurations. The aim of this study was to understand the viewers' ability in detecting the variations in the stereoscopic depth through showing the structural relations between the compound nodes with different graph sizes and transparency effects. We measured the participants' accuracy level and detection speed during the controlled experiment. Also, we collected their feedback to measure acceptance level from the viewers' perspective. The results of the study show that stereoscopic depth can be used to encode data aspects of large graphs under certain circumstances. Moreover, they showed a high acceptance rate of the technique. On the other side, we also observed some shortcomings of using ExpanD in special circumstances.

6.5.1 The ExpanD Approach

In this subsection, we explain our stereoscopic expanding technique where we leverage the 3D technology in the graph visualization domain such that we allow the viewers to expand the compound nodes in the 3D world. For this, our technique, called **ExpanD** (**Expan**d in **D**epth), provides the facility of expanding the compound nodes using the stereoscopic depth cue in the 3D space.

Our ExpanD technique interactively supports the compound nodes exploration via expanding them in depth. It is the first approach that tries integration between the 2D representations and 3D technologies through exploring structural relations in compound graphs. Our technique utilizes the stereoscopic depth to encode the structural relations between nodes of the underlying compound graph. In compound graphs, there are mainly two relation types: the *structural* relations and the *adjacency* relations. One of the main traditional visualization schemes for the compound graphs is visualizing the structural relations using the containment representations. In this case, the compound nodes include their internal structures while the adjacency relations are represented using the explicit edges between the nodes (see Figure 6.19).

There are two main operations to explore such graphs, i.e., the *expanding* operation and the *contracting* operation. The expanding operation is used for opening the compound node to show its internal structure, while the contracting operation is used to close the expanded node.

Upon expanding the compound node, a new level of details is shown up. This requires changing the layout of the initial graph layout. In the

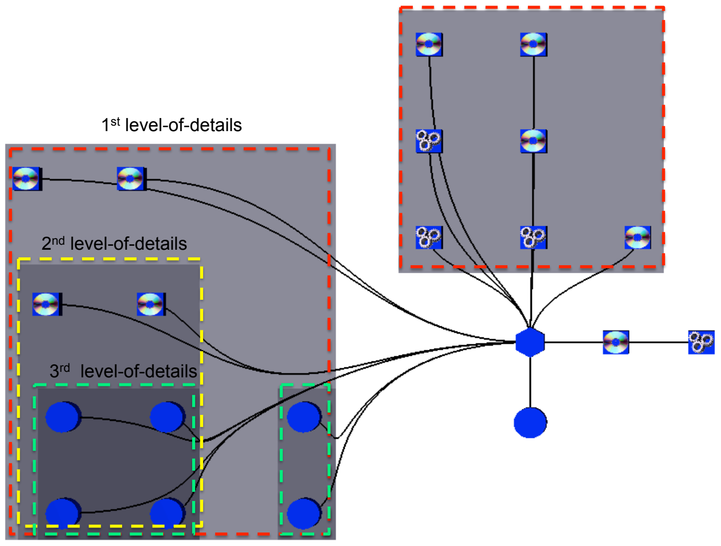


Fig. 6.19: A snapshot for showing the *parent-child* relation in the graph using ExpanD. Here, the depth value is zero for all the nodes.

cases of traditional 2D representations, there is a need to shift all the nodes in all possible horizontal or vertical directions in order to fit the new visual elements into the scene. This leads to change the size of the nodes in a way that the size of the expanded node is scaled up while the non-expanding nodes are scaled down. This process in many cases produces a non-pleasant graph representation, which lowers down the graph readability especially in the case of large graphs.

Through our ExpanD technique, viewers can expand any compound node in the 3D space. This results in opening a new plane closer to the viewer containing the children of the expanded node. In this case, the contracted nodes at the same level-of-details of the expanded node are rendered in a layer further to the viewer.

Our ExpanD technique helps in visualizing interactively the different levels of details in compound graphs, as shown in Figure 6.19. We designed the technique to align the new level of details to a layer that has a distinct depth value closer to the viewer rather than the remaining graph. In our approach, viewers have the ability to change the depth value between any two layers interactively using a slider GUI, e.g., they can increase or decrease the distance between any two depth-layers on demand. Also, viewers can change the other graph settings interactively, like the transparency of the depth layers in which case the further layers look more transparent than the closer layers in the 3D world. Transparency was proposed by [60] to show the hierarchical relations of clustered graphs. This is useful in decreasing the occlusion ratio between the different visual objects at different depth layers. Consequently, this increases naturally the emphasis to the closer objects reside in the closer layers. The layer transparency is assigned to each depth layer using the Equation 6.1:

$$Transparency = 1.0 - \sqrt{\frac{Min(maxD - compLevel, Layers)}{Layers}} \quad (6.1)$$

Here, *maxD* represents the value of the last hierarchy level in the graph, *Layers* is the parameter indicating the number of the current visible layers, which can be controlled by the viewer interactively using a GUI slider (we set 3 as its default value)), and *compLevel* is the level of the component in the hierarchy.

Moreover, we designed our ExpanD technique to be sensitive to the number of children of the expanded compound node. It estimates the size of the children container based on the total number of children. For this, ExpanD uses the CluE algorithm, described earlier in Chapter 4. ExpanD visualizes the parent and its children in different planes with different colors. To distinguish between the expanded nodes and the non-expanded nodes, ExpanD visualizes them using different styles, e.g., it visualizes the expanded node as a 2D gray box while it visualizes the children as blue boxes (see Figure 6.19). To show the variations in the hierarchical levels of nodes, ExpanD changes the hue of the gray color according to its level. For example, the closer planes are shown with darker gray colors than further ones.

One of the main advantages of our ExpanD technique is that it can work with many multi-level layout algorithms. For Example, it works fine with the orthogonal layout algorithm, the Sugiyama algorithm, and the grid layout algorithm (for more details about graph visualization techniques, see [81]). However, we did not test it with the force directed layout algorithm. Moreover, ExpanD uses slow animations to show the changing process of the graph. This helps the viewers to keep the track with the set of changes occurred in the graph. This also helps the viewers in preserving the mental map.

6.5.2 The Evaluation Study

To evaluate our ExpanD technique, we designed an evaluation study to understand the viewers' ability in detecting the variations in the stereoscopic depth through showing the structural relations between the compound nodes with different graph sizes and transparency affects. We measured the participants' accuracy level and detection speed during the controlled experiment. Also, we collected their feedback using a closed questionnaires form to measure the participants' acceptance level.

The aim of this study was to understand the viewers' ability in detecting the graph depth based on two main parameters, i.e., the graph transparency ratio and the graph size. Since we were not able to completely randomize participants and run the study with all the participants in parallel controlling external variables (such as room lightning, temperature, time of the day, etc.), we designed our study with a quasi-experiment design approach [133] .

In the following, first we present our study goals, hypotheses, and metrics. We then describe the design rules used for the controlled experiment and the measurement instruments in our study.

Goals

Here, we present the set of goals of our study. We expect that adding colors and transparency to graph nodes improves the viewers' perception of graphs and allows them to easily recognize different target nodes. According to our expectations, we formulated the goals of our study as follows:

- **G1:** Measure the viewers' ability of detecting the stereoscopic depth variations in compound graphs.
- **G2:** Measure the *influence of graph size* on the viewers' ability of detecting the stereoscopic depth variations in compound graphs.
- **G4:** Measure the *influence of transparency* on the viewers' ability of detecting the stereoscopic depth variations in compound graphs.

Hypotheses

Based on our expectations and study observations, we defined the following hypotheses:

- **H1:** Viewers are able to focus more on the closer nodes (i.e., the expanded section of graph) than the further nodes (i.e., the unexpanded section of the graph).
- **H2:** An increment in the graph size does not affect the viewers' ability of perceiving the depth. However, we expect that hypothesis H2 increases the time of the depth detection.
- **H3:** Enabling the transparency parameter increases the depth detection speed and improves the accuracy level.
- **H4:** Combining more than one visual cue with the depth cue does not affect on the viewers' abilities of detecting the variations in depth.

Metrics

Based on these sets of goals and hypotheses, we defined the following metrics for our study that apply to all hypotheses:

- **Accuracy:** The ratio between the number of correct answers and the number of right answers.
- **Speed:** The time elapsed for each task.
- **Acceptance Level:** We collected the participants' feedback about their acceptance of our approach using a closed-ended questionnaires form with eight questions, where each question offered six different options based on a Likert scale (scaled from 0 to 5 to show the degree of agreement with each question from the participant's perspective; the sixth option was "Don't know"). We defined these acceptance questions based on Technology Acceptance Model [151].

Study Design and Procedure

This evaluation study was designed as a quasi-experiment under laboratory conditions and with a maximum time-frame of 30 minutes per participant. Participation requirement was limited to the viewer's ability of seeing 3D effects properly and to have a minimum experience on graphs. For this reason, we selected participants from students, researchers and professors from the computer science domain studying or working at the University of Kaiserslautern.

We managed to have 30 participants that took part in our experiment. In these participants, 9 were females and 21 were males. The participants' age ranged from 22 to 60 years with a mean age of 32 years. 13 participants were wearing contact lenses or glasses while the remaining 17 participants did not. For each participant, the experiment began with a pre-questionnaires form for collecting profiling information such as age, gender, sight/perception problems or any form of color blindness, and experience in watching 3D movies or using 3D technologies in daily life. All participants are used to 3D technologies and are used to see 3D movies with 3D glasses. Then, the experimenter showed a graph in 3D and asked the participant if she/he is able to recognize the variations in depth.

To minimize the experimenter effects and to promote replicability of the study, we used standardized instructions and material. The study protocol and the accompanying material were developed in a way that no interaction between the participants and the experimenter was necessary after the first setup.

Study Tasks

All tasks were completely automated and participants were asked to answer to a set of questions, based on what they recognize on the 3D monitor. Each task was based on a different configuration. Each graph configuration was based on different compound nodes (Comp) that could be expanded. Depth difference between any two layers was 10 points and nodes were represented with different colors, shapes, and textures.

At the beginning of our experiment, first we show the initial graph as shown in Figure 6.20a. Then we explained to the participants the behavior of the initial graph and showed them how each node of the current graph represents one possible level of details.

In total, there were four configurations:

- **Configuration 1 (Conf1):** Small graphs size with 13 nodes, 9 adjacency edges, and 3 layers. The first layer contained 6 nodes, the second layer contained 4 nodes, and the third layer contained 3 nodes. Depth difference between any two layers was 10 points and the *transparency* was disabled (see Figure 6.20b and Figure 6.20c.
- **Configuration 2 (Conf2):** Same as configuration 1, but with enabled *transparency* based on the Equation 6.1.

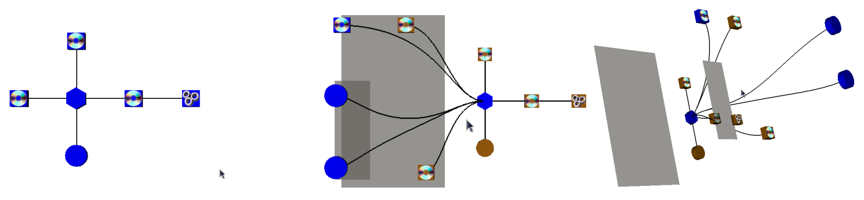


Fig. 6.20: *a)* The initial compound graph; *b)* the front view of the graph after expansion of two nested compound nodes; and *c)* the side view of the graph after expansion of the two compound nodes as it is shown in (b).

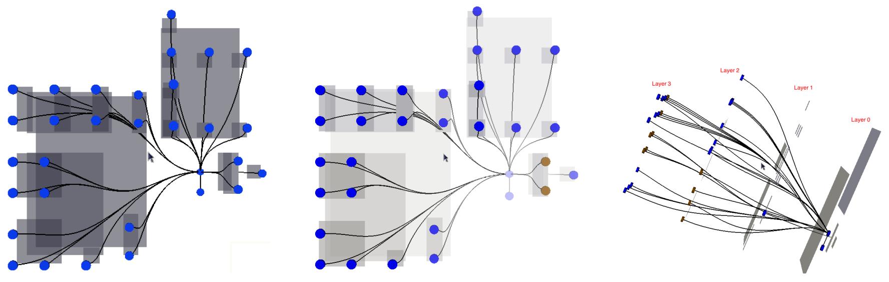


Fig. 6.21: *a)* All nodes are opaque and expanded with different depth values, *b)* all Nodes are expanded using the transparency effect with different depth values to encode different levels of details, and *c)* a side view of the graph where all nodes are expanded with different depth values.

- **Configuration 3 (Conf3):** Large graph size with 56 nodes, 28 adjacency edges, and 4 layers. The first layer contained 6 nodes, the second layer contained 13 nodes, the third layer contained 32 nodes, and the fourth layer contained 16 nodes. The *transparency* was disabled (see Figure 6.21a).
- **Configuration 4 (Conf4):** Same as configuration 3 but with enabled *transparency* (see Figure 6.21b).

We performed the test with the following order: Conf1, Conf2, Conf3, and then Conf4. For each configuration, we ran these four main tasks:

- **Task 1**: Based on a depth scale (where the layer nearest to the viewer was at depth value 0, then the next further was at depth value

1, and so on), after expanding the *CompA* we asked each participant the following questions:

  - What is the depth value of node *CompA* and *CompB*?
  - Do node *CompA* and *CompB* have the same depth values?
  - What is the depth value of node *CompX*?
  - **Task 1 Description:** *CompA* and *CompB* nodes were on the same layer with the same depth value. The only difference was that *CompA* was an expanded node therefore its inner nodes were shown in a closer depth layer (see Figure 6.20b) and was represented in gray color, while *CompB* was not an expanded node therefore it was represented in blue color. The aim of this task was to measure the dominant cue amongst the color cue and the depth cue. The *CompX* node was on the different side of the graph. The aim of this task was to measure the viewers' ability of adapting their sight to the variations in depth in *Conf1*.

- **Task 2:** Find the relations between the CompA, CompC, and CompE nodes.
  - **Task 2 Description:** The *parent-child* relation was represented using the containment representation in a way that the parent node was the one that contained the child node (see Figure 6.19). In this task, the participants were supposed to distinguish between the structural relations and the adjacency relations amongst the nodes. In this case, *CompA* node was the parent of *CompC* node and *CompC* node was the parent of *CompE* node.
- **Task 3:** What is the depth value of nodes CompA, CompB, CompX, and CompY? The participants were asked to give the answer as quick as possible.
  - **Task 3 Description:** All these nodes had different depth values with different shapes and colors. Moreover, they were laid on different locations in the graph. The goal of this task was to measure the viewers' abilities in adapting their sight according to the depth value of the selected node.
- **Task 4:** Contract CompA node, and expand CompY node and one of its child CompZ node. Find the number of depth layers after the changing process.
  - **Task 4 Description:** In this task, we changed the scene of the graph in real-time using the slow animation in order to show the changing process to the participant. Then we asked participants

to follow the changing process and report us with recent changes by giving the number of layers in the scene and the depth value of the selected nodes. In this case, *CompA* node was contracted and *CompC* node was expanded followed by *CompZ* node.

6.5.3 Results and Discussion

In this section, we describe the results of our study and analyze the analyze the collected results for different evaluation tasks.

Accuracy	Task 1		Task 2		Task 3		Task 4		Average	
	Mean	STD	Mean	STD	Mean	STD	Mean	STD	Mean	STD
Conf1	0.95	0.15	0.95	0.15	0.95	0.06	0.95	0.19	0.95	0.14
Conf2	0.50	0.50	0.50	0.46	0.50	0.46	0.50	0.35	0.50	0.44
Conf3	0.94	0.16	0.93	0.22	0.86	0.35	0.84	0.29	0.89	0.25
Conf4	0.53	0.46	0.48	0.45	0.50	0.51	0.71	0.38	0.55	0.45

Table 6.2: The average accuracy for all participants in all the four configurations.

Time in Seconds	Task 1		Task 2		Task 3		Task 4		Average	
	Mean	STD	Mean	STD	Mean	STD	Mean	STD	Mean	STD
Conf1	13.22	21.33	12.07	11.29	6.97	4.00	15.23	11.69	11.87	12.08
Conf2	9.03	5.12	7.90	4.47	7.83	4.04	11.17	5.63	8.98	4.81
Conf3	15.27	21.10	11.33	8.08	9.53	6.68	12.73	9.39	12.22	11.32
Conf4	10.23	4.57	8.17	4.08	5.83	3.88	11.87	8.63	9.03	5.29

Table 6.3: The average detection speed for all participants in all the four configurations.

In Table 6.2 and Table 6.3, we show the results of Conf1 and Conf2 according to the detection accuracy and the detection speed. In both configurations, the same graph size was used and the same visual settings were applied the all sets of nodes. In Conf1 we disabled the transparency parameter, which was then enabled in Conf2. Both tables show the average values and the standard deviations of accuracy and speed metrics.

From the results (see Table 6.2), we observe that the average accuracy of all tasks of Conf1 for the depth detection was 0.95 while the average accuracy for the Conf2 was 0.59. This significant variation between the two values provides an indication that the participants were deceived by the transparency influence. This also indicates that the viewers can be attracted more by the color cue rather than the depth cue in stereoscopic platforms when using the transparency effect. In order to measure the variations in the participants' responses, we also calculated the deviations between the participants' responses (see Table 6.2). From this result, we can easily see that the average of the participants' deviation for all the tasks of Conf1 was 0.14 while it was 0.44 for all the tasks of Conf2, which is an indication of the high difference between the participants' responses for the two tasks.

This led us to find out the reasons behind this significant variation. Therefore, we repeated the same test with large graph size (i.e., with Conf3 and Conf4) where the same tasks were given. This time too we calculated the same two metrics with the same statistical values (i.e., average and standard deviation). This time the average accuracy of all the tasks of Conf3 was 0.89 while the average accuracy of all the tasks of Conf4 was 0.55. The standard deviation average value of all the tasks of Conf3 was 0.25 while it was 0.45 for all the tasks of Conf4. This confirmed our finding of the variation between the two settings, as we also noted the difference in the previous settings.

For finding out the reasons behind this difference, we tried to correlate it with the speed of the detection metric. The average value of the detection speed was 11.87 seconds for all the tasks of Conf1 while it was 8.98 seconds for all the tasks of Conf2. In the case of the speed deviation, it was 12 seconds and 4.81 seconds respectively. This difference in the participants' responses was probably due to the transparency parameter influence, as we observed that when it was disabled then the participants were focusing more on detecting the depth. They were able to detect it more quickly because all the objects in the scene were opaque, so the color cue was not as attractive for them as the depth cue. On the other side when we enabled the transparency in Conf2 and Conf4, it is clear from the results that the participants focused more towards the color cue than the depth cue.

In Figure 6.22, we show the comparison between the participants' accuracy in terms of enabling or disabling transparency. The average time of all participants for this comparison is shown in Figure 6.23

The above observation can be confirmed from the speed of detection, as in the case of depth dominance the average time value was higher compared to when enabled transparency. The same observation can also be found in the case of larger graph sizes, as it was in Conf3 and Conf4. The average detection speed for all the tasks in Conf3 was 12.4 seconds while it was 9 seconds for all the tasks in Conf4. This also indicates that the participants were focusing more towards the color cue when transparency was enabled.

From these results, we can say that the viewers prefer to use the color cue in detecting the different layers when it is available compared to the depth cue. This is because human perceive colors instantly and they do not need much time for colors compared to perceiving the depth.

Regarding the variance value, statistical values in Conf3 indicate a significant deviation (i.e., 11.3 seconds) between the participants while it was less (i.e., 4.8 seconds) in Conf4. This shows the harmony between the participants' responses when enabled the transparency. This also confirmed our earlier conclusion about the dominance of color cue over the depth cue.

In order to study the influence of the graph size over the depth detection speed, we compare (see Table 6.3) between the results of Conf1 vs. Conf3, and Conf2 vs. Conf4. The average accuracy ratio between Conf1 and Conf3 was approximately 1.05, which indicates a slight variation between the participants' performance although the ratio of the graph size in these two configurations was 4.3 (56/13 nodes = 4.3). The detection speed ratio in these two configurations was 1.02, which is not much significant in this case. The variation ratio in these two configurations was 0.98, which indicates homogeneity of variance between the participants' responses. From these analyses, we can conclude that the graph size probably does not affect the accuracy detection. In Figure 6.24 we show a comparison of the participants' accuracy in detecting depth in terms of graph size, while in Figure 6.25 we show the average time for all participants in this regard.

We also compare the results of Conf2 with the results of Conf4, as the two configurations had the same settings. The only difference was the graph size (see Section 6.5.2). The ratio between the average accuracy values of all the tasks in these two configurations (i.e., Conf2 and Conf4) was 1.09 while the ratio in the variations of the participants' responses in these two configurations was 0.99. This confirms our

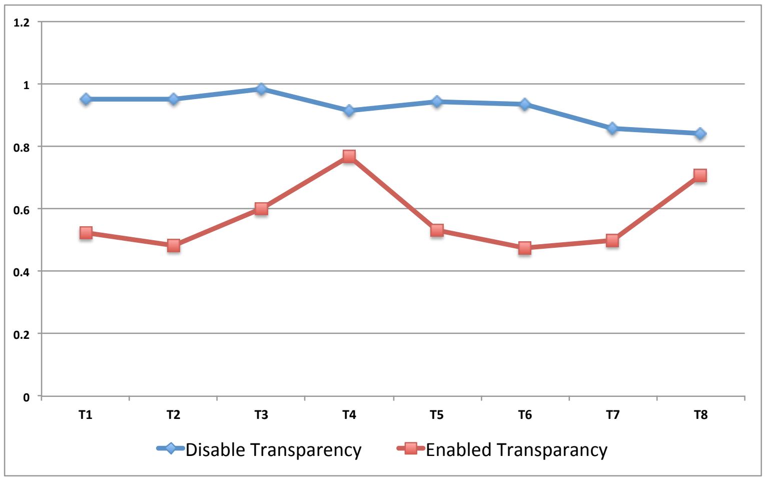


Fig. 6.22: Average accuracy results of all participants in cases of enabled and disabled transparency parameter according to Equation 6.1

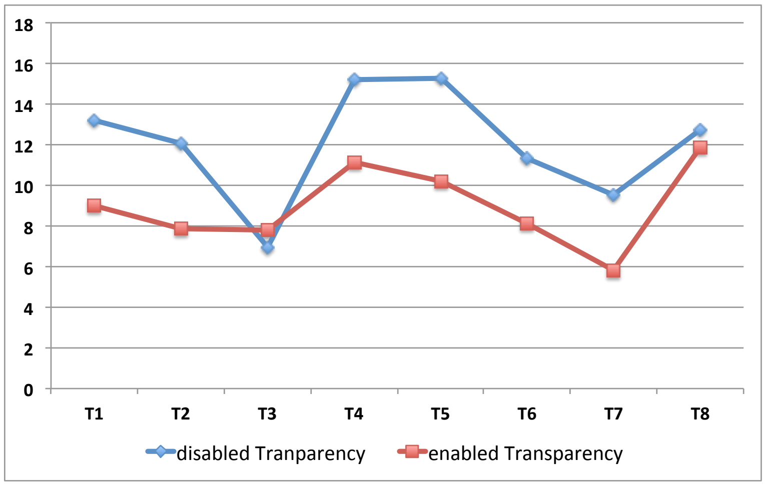


Fig. 6.23: Average detection speed of all participants in cases of enabled and disabled transparency.

hypothesis that the graph size does not affect much on the participants' accuracy level. When we compare the ratio in the detection speed, we also observe that the ratio between the average of the participants' speed in all the tasks of these configurations was 1.0. Also, the ratio between the variances of the detection speed was 0.91. These values do not show significant differences in all the participants' performance although the

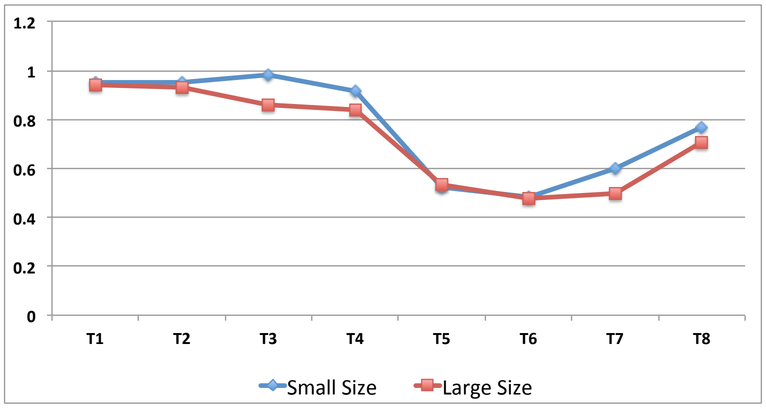


Fig. 6.24: Average accuracy results of all participants in terms of the graph size parameter.

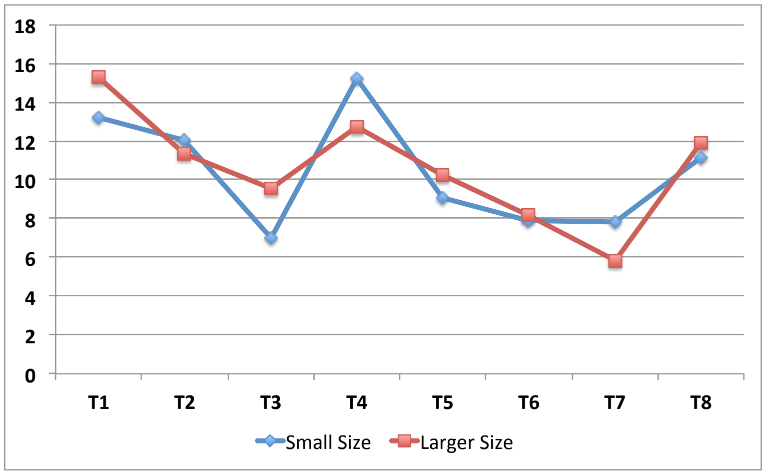


Fig. 6.25: Average time results of all participants in terms of the graph size parameter.

graph size was significantly different in these two cases. From this, we can conclude that changing the graph size in our experiment did not decrease the accuracy of participants' detection. Also, it did not influence on their detection speed.

When comparing the results of the participants who wear glasses or contact lenses in their daily life with the participants who do not, we observe a significant variation in the participants' accuracy values. For example, the average accuracy of all the participants without glasses/contact lenses in all the tasks of all the configurations was 0.84

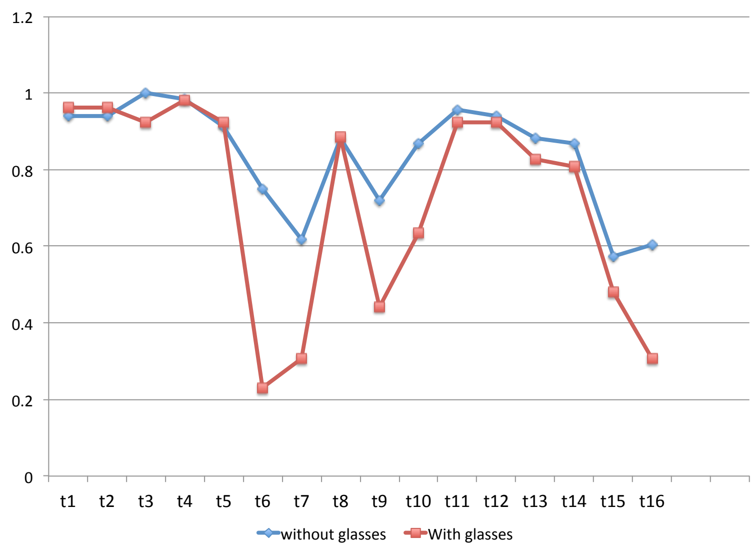


Fig. 6.26: Average accuracy results of all participants in all tasks, grouped together either wearing or not wearing glasses/contact lenses.

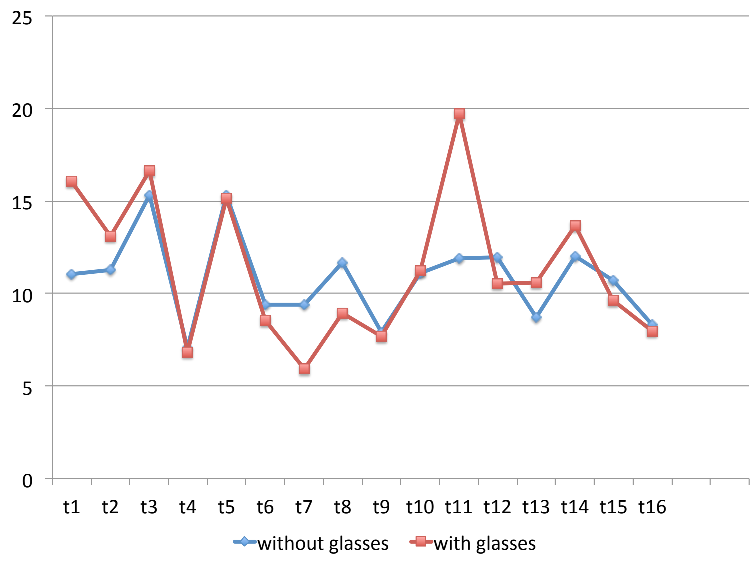


Fig. 6.27: Average detection speed of all participants in all tasks, grouped together either wearing or not wearing glasses/contact lenses.

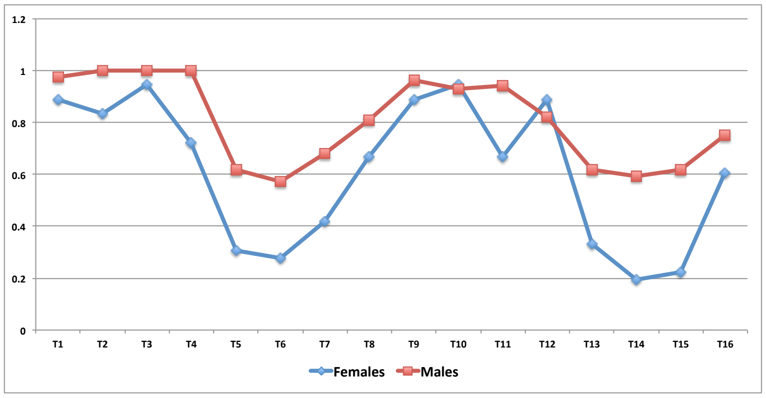


Fig. 6.28: The average accuracy of participants, based on gender, in all tasks for each configuration.

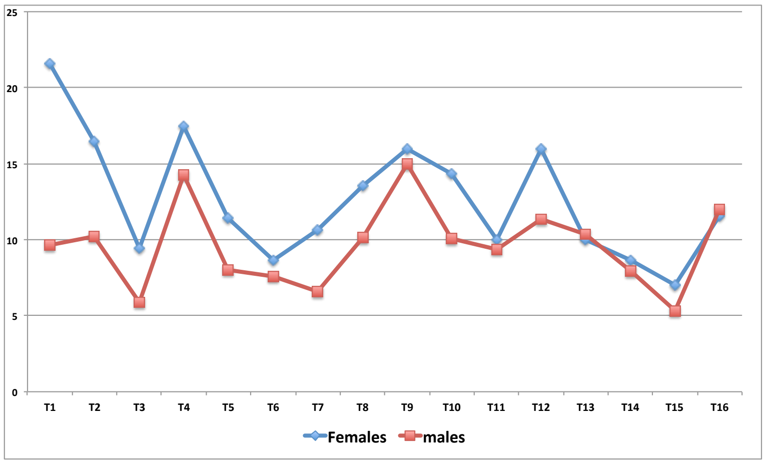


Fig. 6.29: The average detection speed of participants, based on gender, in all tasks for each configuration.

compared to 0.71 for the participants who wear glasses/contact lenses (see Figure 6.26). However, the difference between the average detection speed between these two groups was not significant, as it was 10.8 seconds for the participants without glasses/contact lenses and 11.3 seconds for the participants with glasses/contact lenses (see Figure 6.27). From these results, we can conclude that the reason behind the less accuracy of the participants with glasses/contact lenses is probably due to the fitting of 3D glasses. This is more in the participants who were wearing glasses, as the 3D glasses above their normal glasses probably may not filter the polarization correctly. This indicates that the industry needs to think in providing better designs for 3D glasses so that they can fit more properly on normal glasses.

In our study, the female participants were approximately one third of the total participants (i.e., 9 females out of total 30 participants). Interestingly, we observed a significant difference in performance when grouping participants based on their gender (see Figure 6.28). For example, in the case of the accuracy value we found that the male participants outperformed the female participants mostly in all the tasks for each configuration. Moreover, when we compare these two groups' detection speed we notice that the female participants required more time in completing the same set of tasks compared to the male participants (see Figure 6.29). However in Task 3 of Conf3 and Conf4, we observed an overlapping between the male and female participants' speed curve (i.e., 21 in male participants compared to 9 in female participants). At the same time, the accuracy of female participants for these two configurations was significantly less compared to the male participants.

During the experiment, we also recorded the participants' expressions when they were answering. We noticed that many female participants expressed that they lost the depth cue in the case of inclusion of other cues, as they felt more attracted to the other visual cues like the color or the shape. Therefore, they did not observe the variations in the depth between the different nodes. In few cases, some of the female participants insisted that all nodes are on the same plane and they are able to feel the hierarchical behavior of the final layout only due to the color degradations, not due to the depth variations.

This opens the question that why this occurred most of the times only to the female participants? Is it the case only in our study or it can be generalized in other studies of this domain? We aim to investigate more about this in our future work.

Based on the analysis of results, we can conclude the following: The participants' accuracy did not deviate significantly when different graph sizes were used in the test, which confirmed our hypothesis **H2**. When we enabled the transparency parameter we expected an increment in the detection accuracy. However, results show the opposite behavior of the participants as they were attracted more towards the color cue rather than the depth cue. In the case of all opaque nodes, the participants were able to distinguish between different visual cues, this confirmed **H4**. However, enabling the transparency made the color cue to be the more dominant one. This finding slightly contradicted with our **H3** hypothesis.

Based on the results of the closed questionnaires, we confirmed our hypothesis **H1** as most of the participants gave positive feedback related to our ExpanD approach in general. However, our hypothesis regarding the transparency effect **H3** again was not confirmed. This shows a consistency between participants' performance during the test and their opinion after the test. Again the transparency affect was not helpful as 20 participants out of 30 participants voted negatively for this parameter with the depth parameter. This is summarized in Table 6.4.

Questions	1	2	3	4	5	Don't Know
1 - It was easy to get the depth value for the required node without the transparency effect.	0	1	0	9	20	0
2 - It was easy to get the depth value for the required node with higher depth values and without the transparency effect.	0	1	3	5	21	0
3 - It was easy to get the depth value for the required node with the transparency effect.	7	13	5	3	2	0
4 - It was easy to distinguish between the depth values for the required nodes at different locations in the graph.	0	2	6	11	8	0
5 - Overall, it was effective to use the variation in depth to find the parent-child relationship in the graph.	1	0	3	12	12	1
6 - Overall, I liked the idea of expanding the required component in a new depth layer.	0	3	1	9	14	2
7 - Overall, I found it easy to distinguish between the color cue and the depth cue.	2	3	3	10	10	2
8 - Overall, it was easy to understand the concept of using the depth as an expanding parameter in node-link diagrams.	0	0	2	8	18	0

Table 6.4: The summary result of the closed questionnaires form.

In our study, the male participants slightly outperformed the female participants in detecting the variations in depth. Although, we cannot generalize our finding about the differences between the participants' performance based on gender; however, this shows us some limitations of our technique. It also provides us possible indication on future research. A possible future direction can be to find the reasons behind these differences while working with different data sizes and other visual parameters on larger population size.

6.6 Summary

In this chapter, we presented our contributions towards the stereoscopic highlighting technique that was initially proposed by Alper et al. in [5]. First, we presented the evaluation study that we conducted to judge the accuracy and efficiency of using such technique for highlighting important data aspects in node-link diagrams. The results of this study indicate that the stereoscopic depth approach in stereoscopic devices is relatively an accurate technique if it is applied in a pure stereoscopic environment. Results also indicate that people need some time to get familiar with perceiving the depth and detecting the important graph features. We also noticed that other visual encoding techniques to highlight nodes in graph (e.g., using colors or shapes) do not affect much the depth cue highlighting technique. In fact, the other supported visual cues helped the participants in getting more insight about the data quickly.

However, results of the first study provided an indication for the need of extending the technique in a way that could help the viewers in judging the depth value for the nodes faster and more accurate. Targeting this, we presented our *reflection layer* extension at the bottom of the graph on which the graph nodes are projected. We evaluated the feasibility and the efficiency of this reflection layer by conducting a second evaluation study in a controlled environment. In this case, the data size was not large enough to generalize the results. However, it gives an indication that additional support, like our reflection layer, can increase the accuracy of the stereoscopic highlighting technique in stereoscopic devices. One possible future work in this direction could be to evaluate the possibility of increasing the viewers' accuracy level in immersive environments through utilizing new interactive techniques. For this, one solution could be to build a framework that would provide on demand the depth cue, such that viewer could interact directly with the graph and could request for more depth cues to support the correctness of their judgment in a real time. This would give us an indication of the possibility of using the stereoscopic highlighting technique for critical information visualization applications.

Further, we presented our StFT technique in which we suggested using the depth cue to filter topological features inside the large graphs visualized using the TopoLayout algorithm. Preliminary results of the technique show promising directions in utilizing the depth cue as an approach to show several topological-features inside the large graphs.

Moreover, it was interesting for us to find some possible applications where we can utilize the stereoscopic highlighting technique. In this direction, we presented our **ExpanD** technique that utilizes the stereoscopic depth to highlight structural relations between nodes in the compound graphs. The main contribution of ExpanD is its ability of exploring part of the graph in 2D diagrams by bringing them closer to the viewer. This helps in reserving other highlighting attributes, like color or/and shape, for encoding other data properties. One of the main advantages of using this technique is its ability of achieving the *focus+context* views naturally.

We also presented the details of conducted study in order to evaluate the participants' accuracy and detection speed as well as their acceptance level towards the ExpanD technique. The results of this study show some interesting findings that encourage us to proceed in this direction, as the majority of the participants highly accepted our approach. Moreover, the detection accuracy was high in those configurations where all nodes were opaque. Interestingly, transparency parameter did not help much in increasing the detection accuracy or the speed in finding the required depth value for the selected nodes. These findings also open the doors for us to study the reasons behind this behavior in more details.

7

Visualizing and Integrating the System 3D Model with the Abstract Representations of CFT Models

*This chapter is dedicated to present the 3D aspects of our visual ESSAVis framework. The chapter consists of three parts. The first part of the chapter presents a tool, called **3DintEx**, that allows interactive exploration of the structural and behavioral aspects of a system model in 3D environments. The tool accepts a CAD model of the system and other data sets that describe the system safety behavioral aspects. 3DintEx was built using the Vrui framework [102], which makes it portable and scalable on a range of VR environments (e.g., desktop 3D displays, PowerWall screens, or CAVE systems). 3DintEx extends the Vrui framework in order to provide options that ease up the interactive exploration of the CAD model. Further, it was designed to link the available input devices to 3D interaction facilities for extracting more information from the system 3D model. The second part of the chapter presents the details of the **SceneMan** (**Scene Man**ager) tool inside the ESSAVis that integrates between the system 3D view and the 2D abstract representation of the safety scenario. The final part of this chapter presents the **Collaborative Aspects** of our ESSAVis framework. In this regard, we focus on the collaborative elements and the setup from the application point-of-view rather than the technical implementation.*

In this chapter, we present two 3D tools of our ESSAVis framework: the first one is a 3D visual tool called **3DintEx** (The **3D int**eractive **Ex**plorer), and the second one a views integration tool called the **SceneMan** (**Scene Man**ager). Further, we introduce ESSAVis as a collaborative platform to be used by engineers and safety expert for supporting the maintenance process of embedded systems.

3DintEx works on stereoscopic displays to render 3D models. Moreover, it allows viewers to extract different behavioral aspects information about the rendered model. It also helps viewers in understanding the structural relations between the system components. It provides a set of options through extending the Vrui framework [102] for performing different activities with the system 3D model, such as: a 6 degree of freedom interaction facility, explosion view of the system 3D model, controlling the transparency level of the system 3D model through a slider bar, and selecting any specific component through a 3D selector tool. All these options are provided via a set of functions and interaction facilities in order to ease up the exploring process of such models in a more realistic form.

The SceneMan tool inside ESSAVis is responsible on integrating the system 3D view with the graph representation of the abstract data. It combines between the 3D view (i.e., the system 3D model) with the 2D view (i.e., the graph representation of abstract data in 2D form) to create a 2D*plus*3D environment. This resulting 2D*plus*3D environment is a synchronization between the views in-order to provide the full integration between the different aspects about the underlying system. The SceneMan tool provides an algorithm to support the views alignments to respect each other in a 3D world. Moreover, it gives the viewers the ability to control the positions of the textual information boxes to get more insight about the underlying data. Finally, it offers different interaction techniques in order to interact with each view individually without distorting the other view.

We have successfully applied both tools in the embedded systems domain for exploring the behavioral aspects, such as the safety and reliability, of these systems using the stereoscopic technologies [14].

7.1 The RAVON Robot

In this section, we provide brief description about the RAVON (Robust Autonomous Vehicle for Off-road Navigation) robot that we use as a case study in our ESSAVis framework. RAVON was developed and implemented in the Robotics Research Lab (RRL) of the University of Kaiserslautern under the supervision of the head of lab Prof. Dr.

Karsten Berns. The goal behind this work was to build a platform for autonomous off-road robotics and behavior concepts. The work started on it in 2003 and since then it went through different stages in order to provide enhanced options and facilities. In the stage, the robuCAR TT was developed and implemented (see Figure 7.1) based on the work by the Robssoft company [66, 104, 124]. This provides the base of the current RAVON robot. RobuCAR TT mainly consists of four wheel-drive electric cassis that are connected with a suitable and efficient on-board PC.

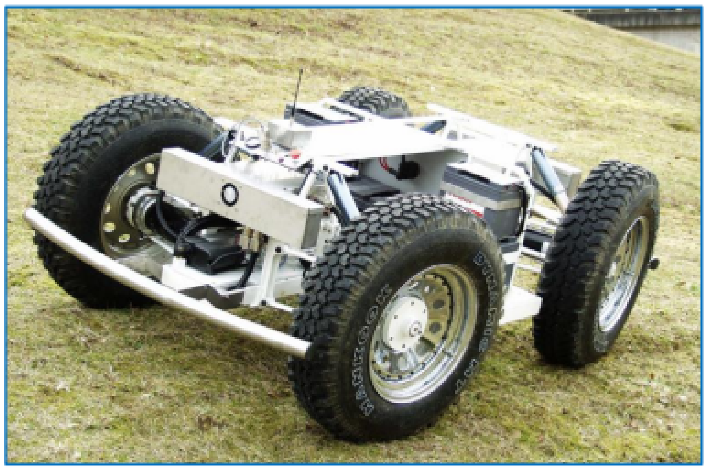


Fig. 7.1: The robuCAR TT that provides the base of RAVON.

After several iterations of developments and improvements from the RRL team, RAVON has been emerged like a small car with many physical features (see Table 7.1) [104, 66].

Table 7.1: RAVON physical measurements.

Measure	Measuring Unit
length	2.35 m
width	1.4 m
height	1.8 m
weight	750 kg

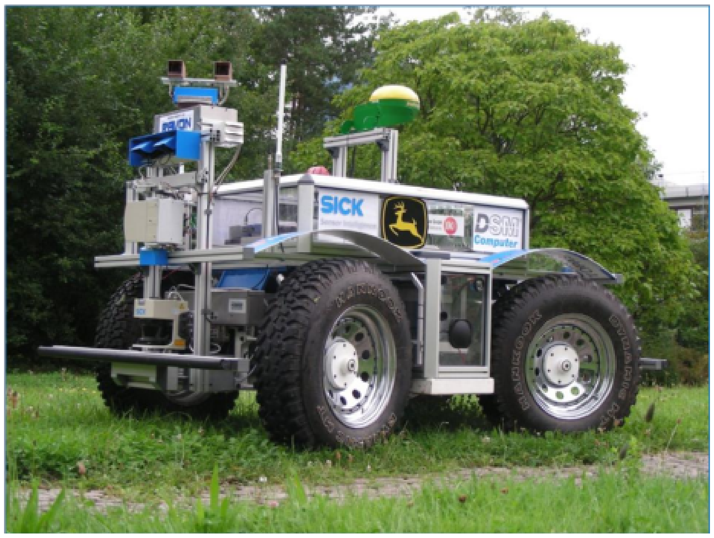


Fig. 7.2: RAVON in the current form (the picture is courtesy of the RRL at the University of Kaiserslautern [104]).

In Figure 7.2, we show the current real appearance of RAVON after the several enhancements. RAVON is considered a reliable and effective robot as it achieves the specified task correctly. For example, it is able to make sharp turns in order to avoid obstacles with changing the robot orientation. This is achieved because RAVON has two axes that are able to steer independently. Regarding its motor, RAVON has four motors with 900 W for each. This enables RAVON to drive at a speed up to 3 m/s. Moreover, it is also equipped with four Hankook off-road to support the climbing tasks. Finally, RAVON operates between 3-4 hours in average depending on the driving speed and the steepness of the terrain [104, 124, 66].

7.2 The 3DintEx Tool

In this section, we provide details about our developed tool 3DintEx, an integral part of the ESSAVis framework, that allows interactive exploration of the structural and behavioral aspects of a system model in 3D environments. We implemented the 3DintEx tool in a way to make it portable with any stereoscopic platform. This was achieved through using the Vrui toolkit, which works over any stereoscopic platform. 3DintEx takes as input a scene-graph representation having the geometrical information of the system 3D model. For explaining different options provided by 3DintEx, we visualize the RAVON robot (see the RAVON in reality in Figure 7.2 and through our framework in Figure 7.3). Below, we explain different functions of the 3DintEX tool that help in analyzing certain system behavioral and structural aspects [14].

The input of our 3DintEx tool is a file in VRML format [33, 41, 146, 52, 32], where the file extension should be as ".wrl". The CAD model in our case can be distributed into many files with the same extension,. For example, in our case study of RAVON robot (see Section 7.1) the 3D geometry data is distributed in 394 VRML files, where all these files are connected with each other in a scene-graph data structure. The root of such scene-graph is a file from where we start parsing or invoking the other files in this model. In our RAVON case study, we start from to the "ravon-complete-asm.wrl" file and send this file to the VRML parser (see Section 7.2.1). The VRML parser invokes other related files using the depth-first search algorithm, until it reaches to leaf nodes or shape nodes in order to start rendering the CAD model recursively.

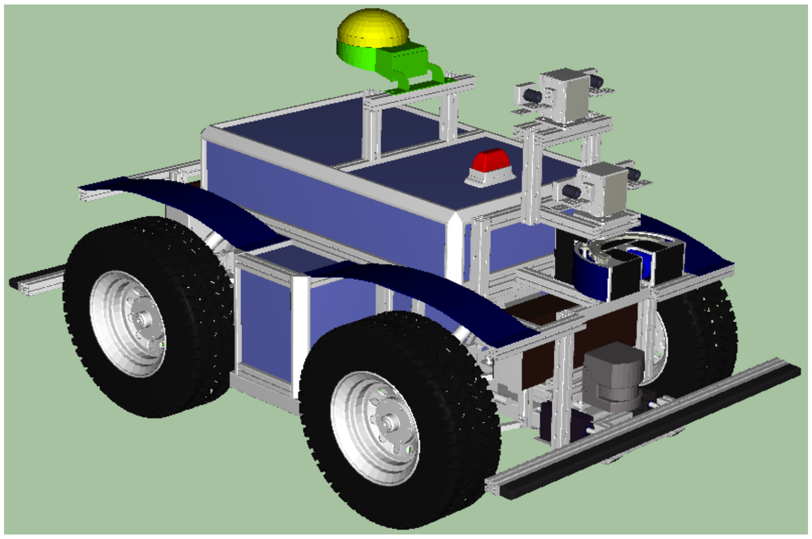


Fig. 7.3: 3DintEX visualization of the RAVON robot in 3D form.

7.2.1 The VRML Parser

For parsing the input VRML files [33, 41, 146, 52, 32] , we start from the root node file. This represents the relations between this file and the other files using the scene-graph data structure [52, 32]. The children of this file are other VRML files that are invoked immediately as the parser encounters them. The invocation process is performed through the depth-first search algorithm. We provide the pseudo code of our VRML parser in Algorithm 2.

```
Procedure VRML Parser
Input: The root file of the scene-graph data structure of the CAD model
Output: GroupNode list, AnchorNodes list, ShapeNode list
Stack = Empty Stack
GroupList = Empty GroupNode list
AnchorList = Empty AnchorNode list
ShapeList = Empty ShapeNode list
Stack.Push(root)
while Stack.Not empty() do
end
*GraphNode=Stack.top();
if GraphNode.type()== groupNode() then
    Add GraphNode to GroupList;
    Stack.pop(GraphNode) Stack.push(GroupNode.children);
end
if GraphNode.type()== AnchoNode() then
    get AnchorNode.description();
    Add AnchorNode to AnchorList;
    Stack.pop(AnchoNode);
    Stack.push(AnchorNode.children);
end
if GraphNode.type()== ShapeNode() then
    Add ShapeNode to ShapeList;
end
```

Algorithm 2: The VRML Parser Pseudo Code

When we build the required data structure representing the CAD model then we can access the required parts that we need to map with the data from the safety information in the CFT model (see Section 7.3.1 for further details). At this stage, we can extract the required information that we need for visualizing the underlying system 3D model. Also, this is the first step we need for providing the interaction facilities with the system 3D model. For example, the exploding view (see the forthcoming subsection) of the underlying system is highly depend on the information that is stored in the group list and the shape nodes list, extracted using the presented Algorithm 2.

Further, this data structure enables us for providing the changing transparency of the CAD model on-demand using the transparency slider (see the forthcoming subsection). Our parser is built over the VRML parser that is provided by the Vrui framework [102].

7.2.2 3DintEx Interaction Options

In this subsection, we provide brief description of few of the options provided by our 3DintEX tool in order to interact with the underlying system 3D model.

The Explosion View

One of the common ways to enhance the comprehensible exploration of 3D models is via providing an exploded view of the underlying system model. This technique has the benefit of providing an opaque view of all parts with full details compared to other techniques (e.g., the clipping plate technique or the transparent effect technique). In the explosion view of 3DintEx, system components in the 3D model diverge from each other in all directions in a sphere (see Figure 7.4).

3DintEx gives viewers the ability of controlling the speed and distance in this explosion using a slider bar interaction option. The value of this slider bar ranges from 0.0 to 4.0, where 0.0 means the distance between neighboring system components is 0.0 (i.e., the system components are glued together as in the original model) while 4.0 means that the distance between any two levels in the scene-graph hierarchy is 4.

When a viewer drags the slider bar, components in the 3D model split away from each other according to the sliding speed with a distance equals to the slider bar value. 3DintEx traverses the model tree of the scene-graph structure to find out the geometry of each component using the depth-first search algorithm. Then it applies a transition process to transformation nodes of anchor nodes. It then transforms these nodes by shifting them in x, y, and z directions based on the slider bar value. The components in the 3D model come closer to each other through decreasing the slider bar value. This changing in the distances does not affect the general layout of the 3D model.

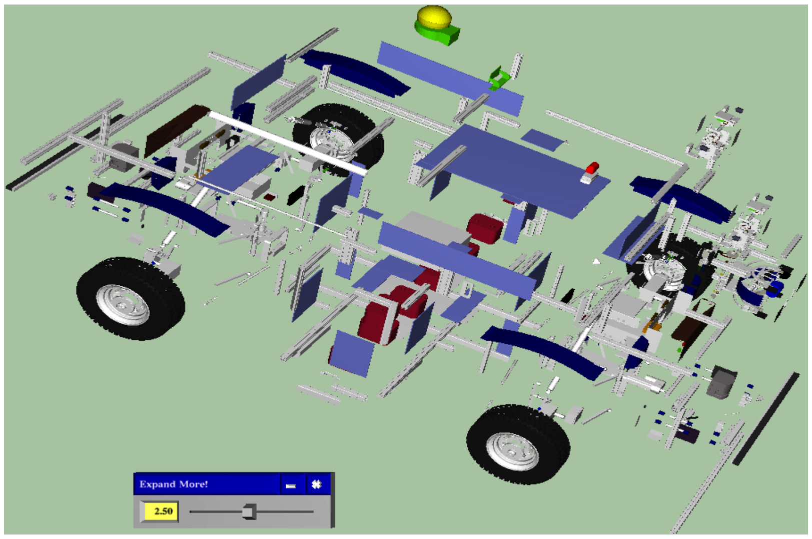


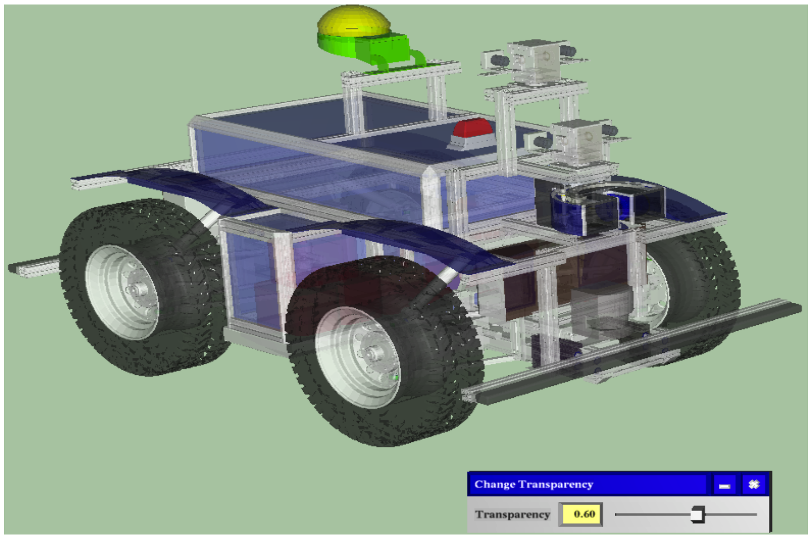
Fig. 7.4: The slider bar interaction facility (shown at the bottom) is used for controlling the explosion view.

The Transparency Controller

3DintEx uses another slider bar for controlling the transparency level of the underlying system 3D model. In this case, the slider bar value ranges from 0.0 to 1.0, where 0.0 means opaque while 1.0 means full transparent (see Figure 7.5 in which transparency is 0.5). This transparency change happens in real time.

The 3D Selector Tool

3DintEx provides an interactive facility for picking any one of the 3D components using a generic selector tool (see Figure 7.6), which can be mapped easily with any pointing device like mouse device in a desktop environment. The generic 3D selector tool is helpful in picking the required 3D component on demand. The selection process is calculated based on finding the first intersection point between the ray and the 3D component geometry polygon. This interaction tool functions well due to the exploded view as it eases up the picking process of the internal components. Upon selecting a component, viewers get the option of showing some information about the selected node on-demand. 3DintEx highlights (in red color) the selected node to distinguish it amongst others.

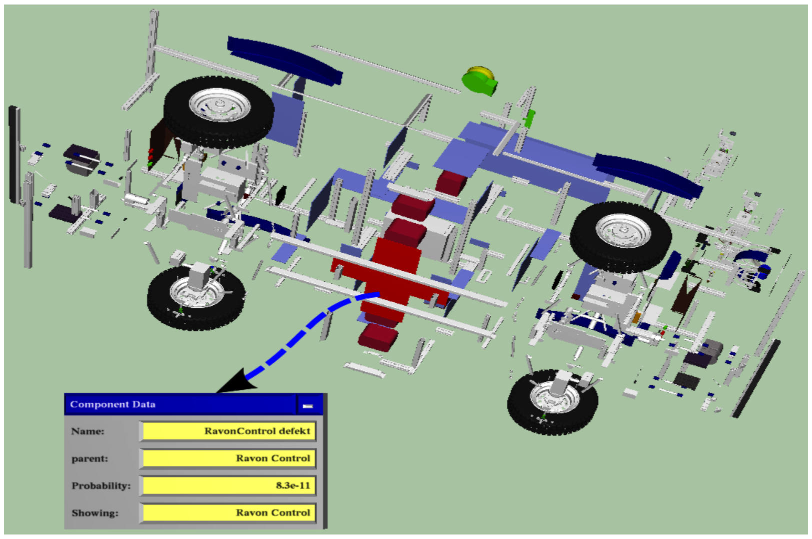


Transparency 0.60

Fig. 7.5: The slider bar (shown at the bottom) interaction technique is used for changing the transparency level on-demand.

The Main Menu

Through this facility, 3DintEx provides a list of all components' names in the underlying system 3D model. In this case, viewers can scroll to



Name:	RavonControl defekt
parent:	Ravon Control
Probability:	8.3e-11
Showing:	Ravon Control

Fig. 7.6: The 3D Selector tool is used for getting further information on-demand about the picked component.

find the interested component, which is then highlighted in red color (as shown in Figure 7.7).

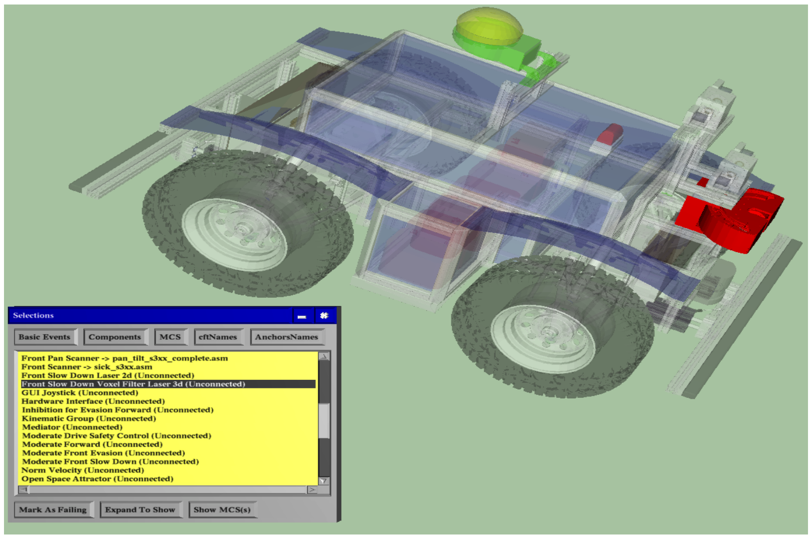

Selections
Basic Events
Components
MCS
cftNames
AnchorsNames
Front Pan Scanner -> pan_tilt_s3xx_complete.asm
Front Scanner -> sick_s3xx.asm
Front Slow Down Laser 2d (Unconnected)
Front Slow Down Voxel Filter Laser 3d (Unconnected)
GUI Joystick (Unconnected)
Hardware Interface (Unconnected)
Inhibition for Evasion Forward (Unconnected)
Kinematic Group (Unconnected)
Mediator (Unconnected)
Moderate Drive Safety Control (Unconnected)
Moderate Forward (Unconnected)
Moderate Front Evasion (Unconnected)
Moderate Front Slow Down (Unconnected)
Norm Velocity (Unconnected)
Open Space Attractor (Unconnected)
Mark As Failing
Expand To Show
Show MCS(s)

Fig. 7.7: A selected component through the Main-Menu is highlighted in red. Other components are shown in transparent mode.

7.3 The Scene Manager (SceneMan) Tool

The collaboration between different engineers in complex systems like the embedded systems faces several challenges in practical situations, e.g., the divergence between perspectives of safety experts and system engineers. This gap between the collaboration happens because safety experts feel more interest towards representing the relations between the participating components in the current scenario based on the failure relations among them, while on the other hand system engineers feel more interest towards understanding the structural relations between the system components. Further, there is a variation in data representations for the both groups, e.g., the hardware model of the underlying embedded system (that is normally used by system engineers) is mostly represented as a 3D data type while the safety scenario (that is normally used by safety experts) is mostly modeled as a fault tree and is considered to be an abstract data type, which can easily be represented as a 2D graph representation.

In order to tackle above mentioned challenge of filling the gap, we present a hybrid solution to visualize the two data types in a 2D*plus*3D world, such that the 2D graph representation is available to visualize the abstract fault tree structure while the 3D part is used to visualize the hardware model of the underlying system. We also propose to visualize this 2D*plus*3D world using the stereoscopic displays in a manner that the stereoscopic depth is utilized to encode some aspects about the nodes in the graph representation. In this context, we use the stereoscopic depth to encode the structural relations in the abstract graph. In our case, the depth is used together with other visual cues to encode or convey some data aspects.

The main contribution of our **SceneMan** (**Scene Man**ager) tool inside the ESSAVis framework is that it accepts different data types and renders them in a 2D*plus*3D space. In this case, both views are visualized based on their nature as we mentioned above. Further, we present some of the interaction techniques provided by the SceneMan tool that allow viewers to get more insight about the system 3D model. Moreover, we utilize the stereoscopic depth to encode nodes attributes in the 2D representation of the fault tree. This is achieved interactively through expanding compound nodes in the graph.

In the forthcoming section, we describe how the SceneMan tool synchronizes between the different perspectives or views of the system. The first view is the 3D model that is resulted from the 3DintEx tool (see Section 7.2) while the second one is the graph representation that is resulted from the LayMan framework (see Chapter 5). Both views represent different perspectives about the safety aspects of the system. The variation in perspectives leads to have different naming styles between the components in the system. This variation came from the differences in the backgrounds of the responsible engineers and safety experts who are responsible to analyze the safety aspects of the system, such that each group describes the safety scenario from their point of view based on their background.

To map between the actual components in the system and the components in the CFT model, we arranged a focus group meeting with different system engineers and safety experts in order to help us in knowing which component in the 3D model is described in the graph representation of the CFT model and which is not. This step was necessary for us to proceeding the implementation of the linkage between the two views. This step was done at the beginning of the project to fix the style of the component names that is used in the safety scenarios of the system. After accomplishing this step, we designed a data structure to hold this mapping between the two views, as it is shown in Figure 7.8.

7.3.1 SceneMan Views Alignment Mechanism

In Figure 7.8, we show the overall idea behind the proposed integration. The SceneMan tool is responsible for integrating the system 3D view with the graph 2D representation of the abstract data, which results in creating a 2D*plus*3D environment. This synchronization between the two views gives the ability to know different aspects about the underlying system at the same time. The tool supports the views' alignment in a way that they respect each other in the 3D world. Further, it gives the viewers the ability to control the positions of each view in order to focus more towards the current interested view.

Moreover, it offers different interaction techniques for interacting with each view individually without distorting the other one. This is achieved as the resulting 2D*plus*3D environment is managed using the *scene-graph* concept. In this case, the root node represents the whole 3D space containing all the views while the children of this root are the 2D abstract view for the graph representation of the component fault tree and the 3D model view of the underlying system.

Our ESSAVis framework provides this above-mentioned 2D*plus*3D environment, in which we consider the overall view of this environment as a 3D world. As mentioned above, this environment is managed using the scene-graph concept where the root of this scene-graph is the 3D world of the environment. Therefore, the elements of this 3D space should be arranged together in order to reduce occlusions; especially, when the size of any of these two views is changed then the size of the other view should be adapted in order to guarantee the proper readability of the scene.

For the clarity reasons, we decided to assign a fixed position for the textual information box in the scene. This box was assigned in the left lower corner of the screen. However, viewers can change the position or re-size the box on-demand using the mouse interaction technique.

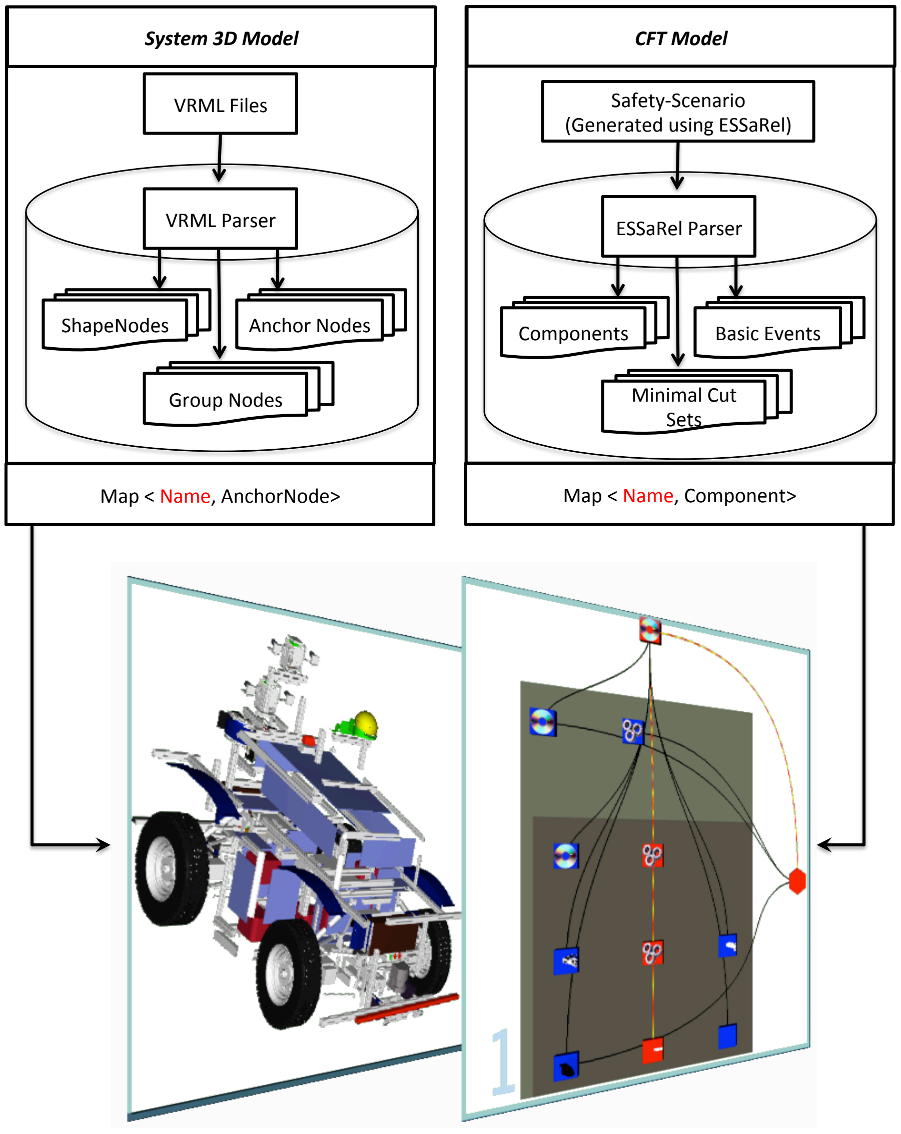


Fig. 7.8: The mapping operation done through the SceneMan unit.

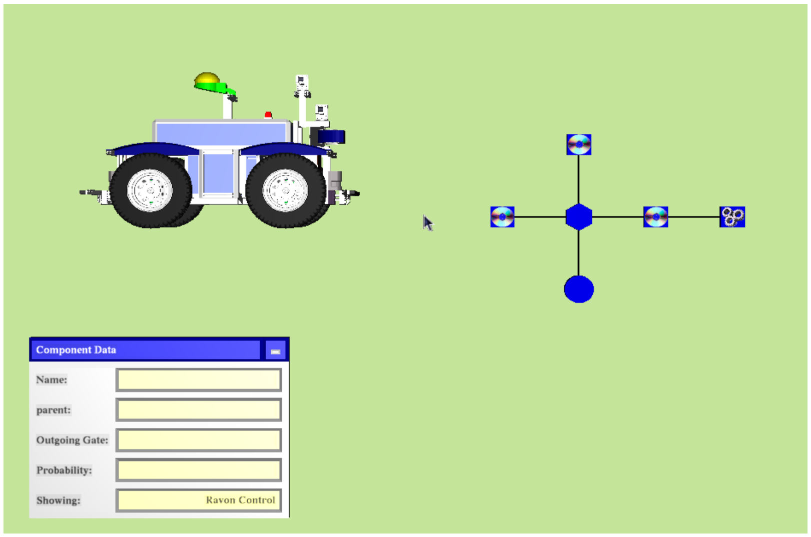
In the following, we explain how we designed the relation between these two main views. At the initialization time, both views have the same size and the fixed positions, as we pre-configured these settings in the configuration file. These settings are passed to the tool as input before the start of rendering. Upon rendering, the two views are shown up side-by-side in the main window where the system 3D model is shown on the left side of the scene while the 2D abstract representation is shown on the right side of the scene, with approximately the same size.

In order to minimize the occlusion effects, elements in the 3D space are arranged together especially when the size of any of the two views is changed. In our case, the size of the view (that is not in current focus) is adapted consequently to guarantee the increasing of the readability of the view that is in current focus. Initially, both views are arranged *side-by-side* in the 3D world, as shown in Figure 7.9. This initial configuration is changed when the viewer starts interacting with the 3D scene. For example, both views can be arranged into a layered view where these views are stacked above each other, as shown in Figure 7.12. In this case, the depth cue is used to show the importance degree of the view. For example, for safety experts the graph representation is the preferable view; therefore, they may prefer to align it on the top of the system 3D model. Here viewers can also get more focus on the top-view by increasing the distance in depth between it and the lower view. Additionally, the size of any view can be adjusted based on the degree of interest of the required view. The distance can also be adjusted using the dragging interaction operation, such that the required view is hold using the mouse and then it is dragged further or closer to the viewer on-demand.

To adjust the two views on-demand, our SceneMan tool provides a number of options with different layouts. Here we brief introduce these options:

- **The Side-by-Side Option:**
  The initial one is the *side-by-side* option, in which both representations are displayed next to each other having the same size and at the same distance from the viewer (see Figure 7.9).

- **The Big-Small Option:**
  In this option, viewers can specify the dominant view on-demand. For example, they can assign the system 3D model to be dominant (i.e., the *Big* view) and the graph representation to be the less dominant (i.e., the *Small* view) and vice versa. This is useful when the current viewers are more interested in exploring one view more than the other one (e.g., system engineers are more interested in the 3D

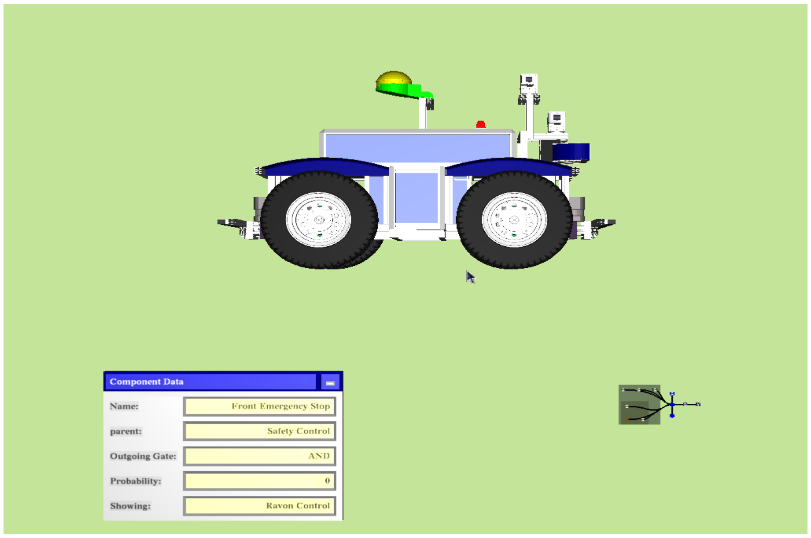


Name:	
parent:	
Outgoing Gate:	
Probability:	
Showing:	Ravon Control

Fig. 7.9: An integrated *side-by-side* view, showing the correlation between the infected components in the hardware model and the abstract representation.

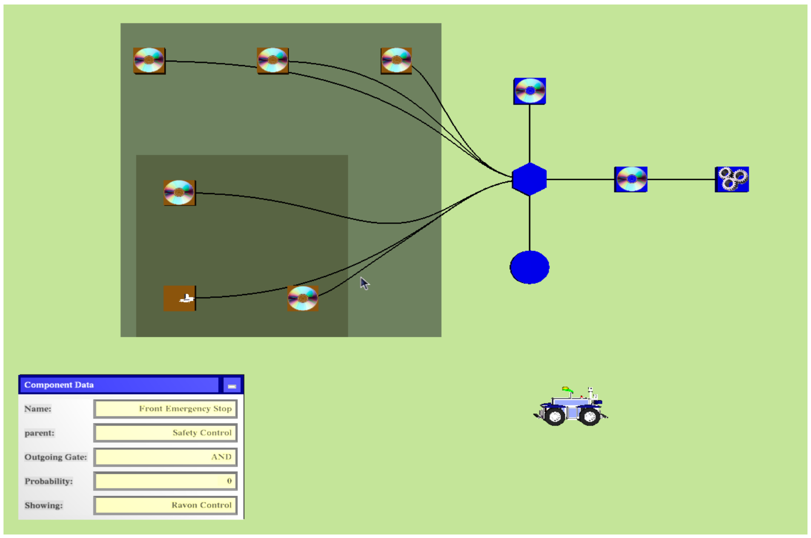
model of the system rather than the abstract graph representation). The two cases for the two views are shown in Figure 7.10 and Figure 7.11 respectively.

- **The Layered-View Option:**
  We use the depth to show the importance degree of the view. Initially, the two views are arranged at the same depth value that is zero in our case (as shown in Figure 7.9). However, this arrangement can be modified by changing the view through the *layered-view* option. For example, if the viewer is interested in the hardware model, then the view of hardware model is popped up to a layer closer to the viewer point and the second view is stacked behind the current active view in the first layer as it is shown in Figure 7.12. In this example case, the second layer represents the abstract representation while the first layer represents the 3D model of RAVON robot respectively. The viewer can toggle between the different views on demand and the interested view dominants the screen by coming on the first layer near to the viewer.


Component Data

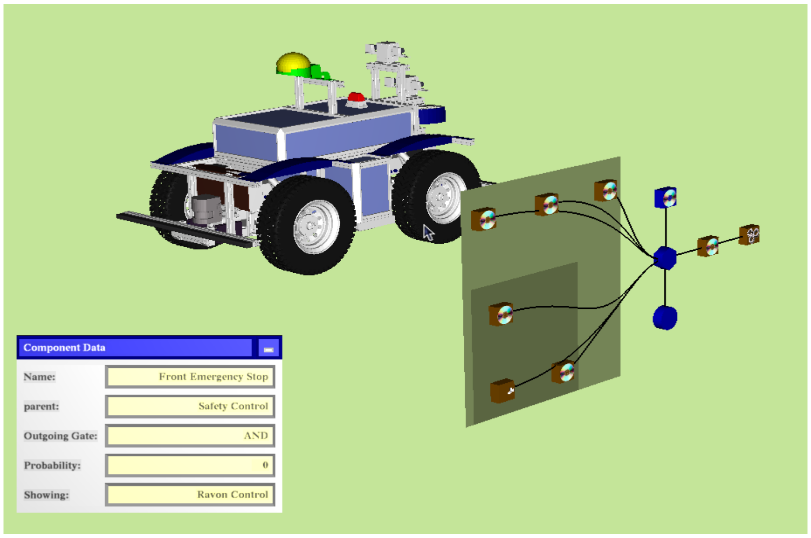
Name:	Front Emergency Stop
parent:	Safety Control
Outgoing Gate:	AND
Probability:	0
Showing:	Ravon Control

Fig. 7.10: The *big-small* option with the system 3D model view as the dominant one while the graph representation view as the less dominant one.


Component Data

Name:	Front Emergency Stop
parent:	Safety Control
Outgoing Gate:	AND
Probability:	0
Showing:	Ravon Control

Fig. 7.11: The *big-small* option with the graph representation view as the dominant one while the system 3D model view as the less dominant one.


Component Data

Name:	Front Emergency Stop
parent:	Safety Control
Outgoing Gate:	AND
Probability:	0
Showing:	Ravon Control

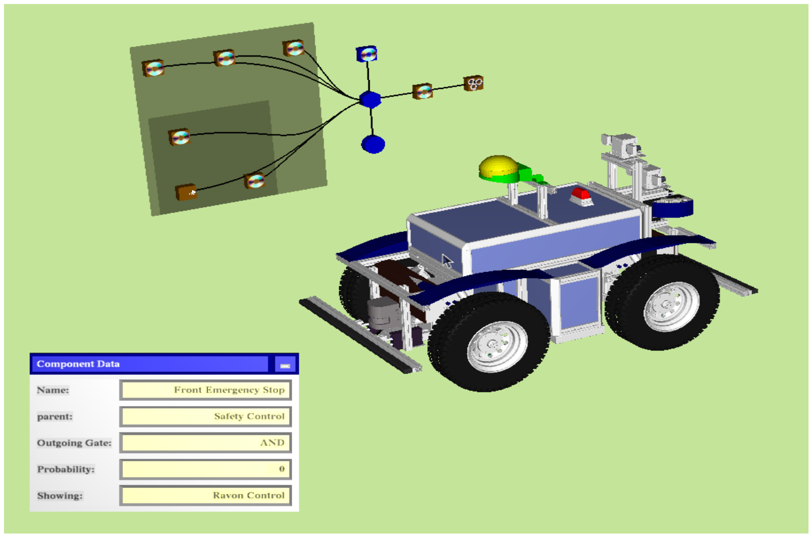
Fig. 7.12: An integrated *layered-view* showing the correlation between the two views using the stack metaphor, such that the most important view (here the system 3D model) is on the top of the stack while the other view (here the graph representation) is laid at the second layer in the stack.

7.4 ESSAVis Interaction Options

Our ESSAVis framework provides different menu in order to give the facility to viewers to interact with the system 3D model and the abstract graph representation, to manage the views laying position, and to get the required safety information about the underlying system. Here we provide brief details of these menus and the options provided by them.

The Main Menu

It is the main interface to interact with our ESSAVis framework. This menu is a pop-up menu that is triggered via clicking on the right mouse button at any free space in the 3D world. It provides the basic options for extracting the required information about the underlying system. For example, viewers can load a new CFT model stored as an xml file. This requires the framework to parse this file and then convert it to the required data structure that is necessary for visualizing. Further, viewers can load another CAD model formatted as VRML files, in which case ESSAVis clears all the data structure elements and then does



Component Data

Name:	Front Emergency Stop
parent:	Safety Control
Outgoing Gate:	AND
Probability:	0
Showing:	Ravon Control

Fig. 7.13: An example of the integrated *layered-view* option where the graph representation is on the top of the stack and the system 3D model is at the second layer in the stack, in contracts to Figure 7.12.

empty them in order to store the new file structure. Then it sends this data to the VRML parser to convert it to the required data structure for preparing it for the visualization step. This Main-Menu has several sub-menus (e.g., the *graph-layout* submenu, the *views layout* submenu, the *setting* submenu, the *safety-information* submenu, etc.) in order for arranging and organizing the scene.

The Graph-Layout Menu

This submenu provides a list of graph layout algorithms suitable for visualizing the underlying CFT model. Viewers can select any of these listed layout algorithms, which results in the adjustment of the graph accordingly. Currently, the framework provides the facility to arrange the graph according to these layout algorithms: the orthogonal layout algorithm [72], the Sugiyama layout algorithm [143], the force-directed layout algorithm [58], and the radial layout algorithm [59]. All these algorithms are integrated with the CluE algorithm presented in Chapter 4, in order to navigate through the different levels of details.

The Views Layout Menu

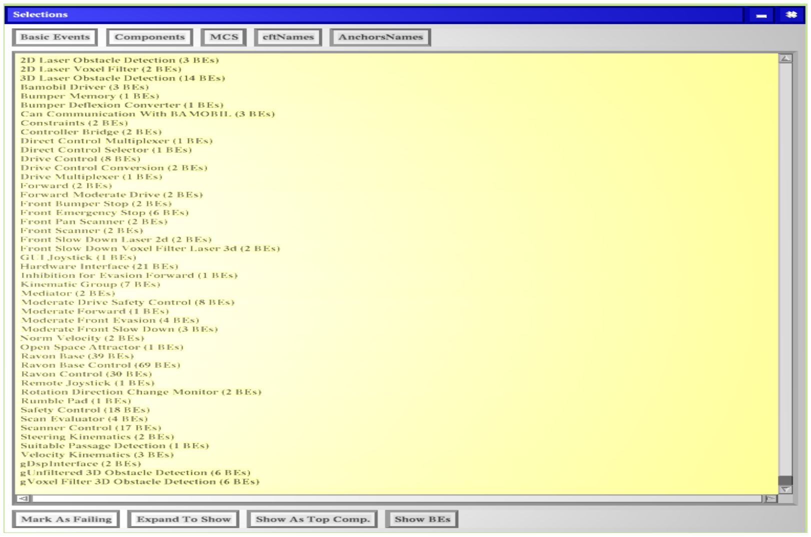
This sub-menu provides different layout options in order to assign the relations between the two main views. Viewers can toggle the relation of the views based on the selected layout option.

The Setting Menu

The sub-menu provides several options in order to allow the viewers to adjust the visual appearance of the platform objects. For example, the *transparency slider* is one of the interaction techniques that is used in the framework to control the transparency of the undelrying system 3D model. This transparency slider is placed in this submenu.

The Safety Information Mneu

This submenu is one of the most important ones as it provides the options to get required information about the safety aspects of the underlying system. This submenu contains five main tabs (see Figure 7.14).



```
2D Laser Obstacle Detection (3 BEs)
2D Laser Voxel Filter (2 BEs)
3D Laser Obstacle Detection (14 BEs)
Bamobil Driver (3 BEs)
Bumper Memory (1 BEs)
Bumper Deflexion Converter (1 BEs)
Can Communication With BAMOBIL (3 BEs)
Constraints (2 BEs)
Controller Bridge (2 BEs)
Direct Control Multiplexer (1 BEs)
Direct Control Selector (1 BEs)
Drive Control (8 BEs)
Drive Control Conversion (2 BEs)
Drive Multiplexer (1 BEs)
Forward (2 BEs)
Forward Moderate Drive (2 BEs)
Front Bumper Stop (2 BEs)
Front Emergency Stop (6 BEs)
Front Pan Scanner (2 BEs)
Front Scanner (2 BEs)
Front Slow Down Laser 2d (2 BEs)
Front Slow Down Voxel Filter Laser 3d (2 BEs)
GUI Joystick (1 BEs)
Hardware Interface (21 BEs)
Inhibition for Evasion Forward (1 BEs)
Kinematic Group (7 BEs)
Mediator (2 BEs)
Moderate Drive Safety Control (8 BEs)
Moderate Forward (1 BEs)
Moderate Front Evasion (4 BEs)
Moderate Front Slow Down (3 BEs)
Norm Velocity (2 BEs)
Open Space Attractor (1 BEs)
Ravon Base (39 BEs)
Ravon Base Control (69 BEs)
Ravon Control (30 BEs)
Remote Joystick (1 BEs)
Rotation Direction Change Monitor (2 BEs)
Rumble Pad (1 BEs)
Safety Control (18 BEs)
Scan Evaluator (4 BEs)
Scanner Control (17 BEs)
Steering Kinematics (2 BEs)
Suitable Passage Detection (1 BEs)
Velocity Kinematics (3 BEs)
gDspInterface (2 BEs)
gUnfiltered 3D Obstacle Detection (6 BEs)
gVoxel Filter 3D Obstacle Detection (6 BEs)
```

Fig. 7.14: The Safety Information Menu

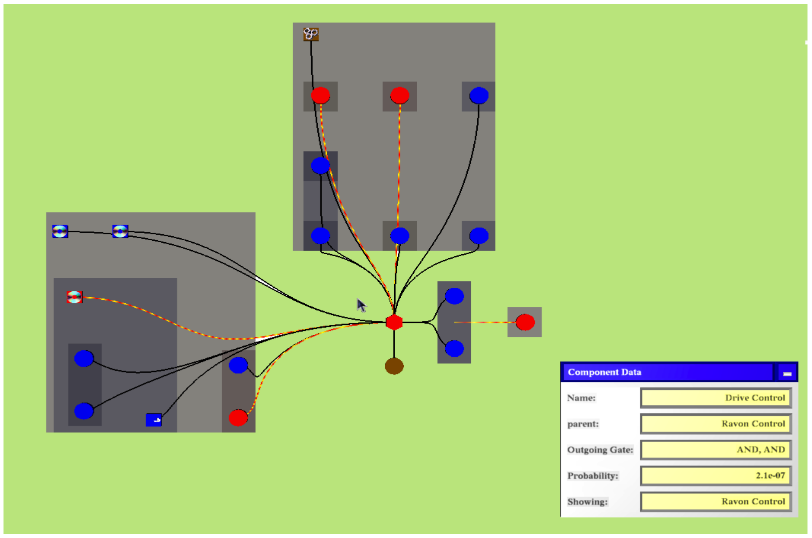
The first one is for showing the list of components in the current CFT model together with their number of basic events. The second one is to get information about the basic events and their probabilities in the current CFT model, and the third tab is to show all the minimal cut sets in the current CFT model with the option to sort them based on their probabilities or their sizes. In each of these three tabs, ESSAVis provides two buttons in order to allow viewers to highlight the selected elements in the list and then expand them to show their place in the visualized CFT model. It is worth mentioning that each basic event has a probability value, which indicates the occurrence probability of such an event. The current version of the framework provides information about the basic event probability via a label containing the value that is attached to the basic event name in the list. Therefore, this submenu provides the required safety information together with the probability of each element. We implemented the required algorithm that calculates the probability-to-fail of each element of the underlying CFT model recursively.

One of the important aspects about ESSAVis is showing the Minimal-Cut-Set (MCS) information. The MCS indicates the least number of basic events that are required to cause a top-event situation in the system. ESSAVis provides a mechanism to show the relations between lists of MCSs in the current scenario with correspondence components in the graph representation of the CFT.

This facility aids safety experts in diagnosing the critical components in the current scenario. For this, ESSAVis gives viewers the ability to interact directly with the list of MCSs and to select one of the entries of the underlying scenario. The framework then highlights the corresponding elements in the graph representation. Due to the viewer's selection, the set of infected components in the current scenario are highlighted in red color (see Figure 7.15).

Viewers can also sort MCSs based on their sizes or their probability to fail. Moreover, ESSAVis calculates the list of components that are influenced by the list of the most critical MCSs. Then it provides viewers the chance to highlight all the components that are related with the list of critical MCSs. This indicates the list of components that are in risk to fail due to the relation with those critical MCSs.

The last two tabs of this submenu provide the mapping between the components in the safety scenario and the components in the 3D model of the system.


Component Data

Name:	Drive Control
parent:	Ravon Control
Outgoing Gate:	AND, AND
Probability:	2.1e-07
Showing:	Ravon Control

Fig. 7.15: Showing the failure path between the selected components using the mouse interaction techniques, in which all the participating nodes in the failure are in red color.

The 3D Interaction Options

To interact with the scene using the input devices like the keyboard or the mouse interaction techniques, ESSAVis was designed to allow viewers to zoom in or out using the mouse wheel. For rotating the whole scene, viewers can hold the mouse clicked together with the 'X' key on the keyboard. Further, it also supports the interaction techniques provided by the underlying Vrui toolkit [102], e.g., viewers can scale up or down the scene using the scale bar option. This option gives an insight about the zooming in or out ratio.

Sometimes, viewers need to interact with one view without affecting the other view. For this, ESSAVis accepts different interaction options [7]:

- *'A' key + holding the left mouse button:*
  Through this option, viewers can interact only with the graph view of the CFT model without affecting the system 3D model. This option enables the viewers to *pan* the graph view over the 3D world.
- *'S' key + holding the left mouse button:*
  Through this option, viewers can interact only with the graph view of the CFT model without affecting the system 3D model. This

option enables the viewers to *rotate* the graph view over the 3D world.

- *'A' + 'S' + holding the left mouse button:*
  Through this option, viewers can interact only with the graph view of the CFT model without affecting the system 3D model. This enables the viewers to *scale* the graph view over the 3D world based on the mouse direction.
- *'Q' + holding the left mouse button:*
  Through this option, viewers can interact only with the system 3D view without affecting the graph view. This option enables the viewers to *pan* the 3D model view over the 3D world.
- *'W' + holding the left mouse button:*
  Through this option, viewers can interact only with the system 3D view without affecting the graph view. This option enables the viewers to *rotate* the 3D model view over the 3D world.
- *'Q' + 'W' + holding the left mouse button:*
  Through this option, viewers can interact only with the system 3D view without affecting the graph view. This option enables the viewers to *scale* the 3D model view over the 3D world where the scaling up or down is selected based on the mouse direction.

7.5 ESSAVis in A Large Visual Collaborative Environment

Maintaining modern embedded systems requires close collaboration between system engineers who designed them and safety experts who analyzed possible failures in such systems in order to avoid any possible hazard. In this section, we show how our ESSAVis framework is used in large visual collaborative environment in order to enhance the collaboration between these two groups (i.e., the system engineers and the safety experts) through supporting the safety analysis process in a collaborative manner. In this case, the large visual collaborative environment consists of one large display and multiple smart devices (i.e., smartphones or smart tablets) through which engineers and safety experts interact with the visual elements on the large display for understanding the failure mechanisms of the underlying system. This whole setup provides an intuitive visual interaction to speed up the communication between the different interested participated groups having different backgrounds [6].

To support such setup, we enhanced our ESSAVis framework in order to support the multi-users interaction; therefore, different related engineers and experts can collaborate easily to maintain the safety and the reliability level of the underlying system. For this, our setup provides an integrated visual collaborative environment that contains two views, the system 3D model and the abstract model of the failure mechanism; same as in the normal desktop settings but here on the big large display while giving the facility to interact with each element on the scene individually through multiple smart devices. Here, we focus on the collaborative aspects of this setup, the design decisions, and the collaborative platform elements [6].

7.5.1 The Collaborative Environment Mode

The maintenance process of embedded systems requires a multi-step collaboration between the system engineers and the safety experts. However, one obstacle that arises often in this collaboration is the difference in the system perspective by these different interested groups and the different sets of used vocabularies.

Mainly, ESSAVis works in two different modes, the *single-user mode* and the *multi-users mode*. The single-user mode is applied in desktop settings where users can interact with ESSAVis with traditional interaction facilities, as described earlier in the previous section. This mode is designed for those users who are interested in analyzing the system individually, especially when dealing with the tasks (e.g. extracting the critical values for the specified components in the safety-scenario) that do not require collaboration with someone else.

The second mode, the multi-users mode, is designed to support multi-users interaction in a collaborative manner with the visual platform of ESSAVis. The configuration of this mode consists of three main elements (see Figure 7.16): the main display (relatively large display, e.g., a TiledWall), two or more smart devices (i.e., smartphones or smart tablets), and the visual elements to be displayed. The smart devices are connected with the large display via a wireless connection. They allow users to interact with the visual elements that are rendered on the main display. Users can query different things through these smart devices, e.g., about the set of safety critical components or the main reasons that might trigger a safety-critical situation. Moreover, users can extract the location of those components in the system 3D model and also in the abstract representation of the failure mechanism. The query results are displayed textually on the smart devices and graphically on the large display [6].

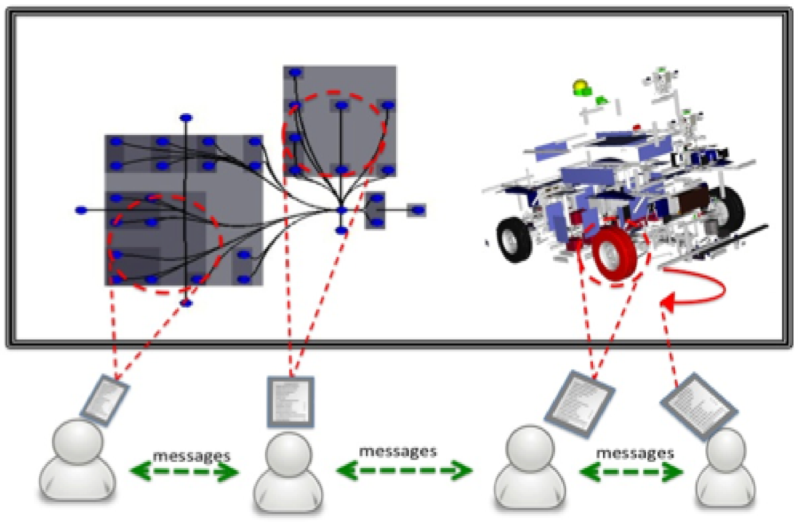


Fig. 7.16: An abstract representation of ESSAVis usage in a large visual collaborative environment, showing different perspective for each user during the failure analysis process.

7.5.2 System Collaboration Aspects

In this subsection, we describe briefly the collaborative aspects of our collaborative setup, which allows users to collaborate through smart devices. Our setup allows different users to interact with different visual views (e.g., with the abstract representation or with the system 3D model) on the large display simultaneously (see Figure 7.17). Moreover, multiple users can perform different operations on the same view at the same time.

The following two scenarios illustrate some possible interaction options provided by our setup:

- **Scenario 1:** ***Querying Safety Aspects Visually***

  Two users (e.g., user-A belongs to safety experts while user-B

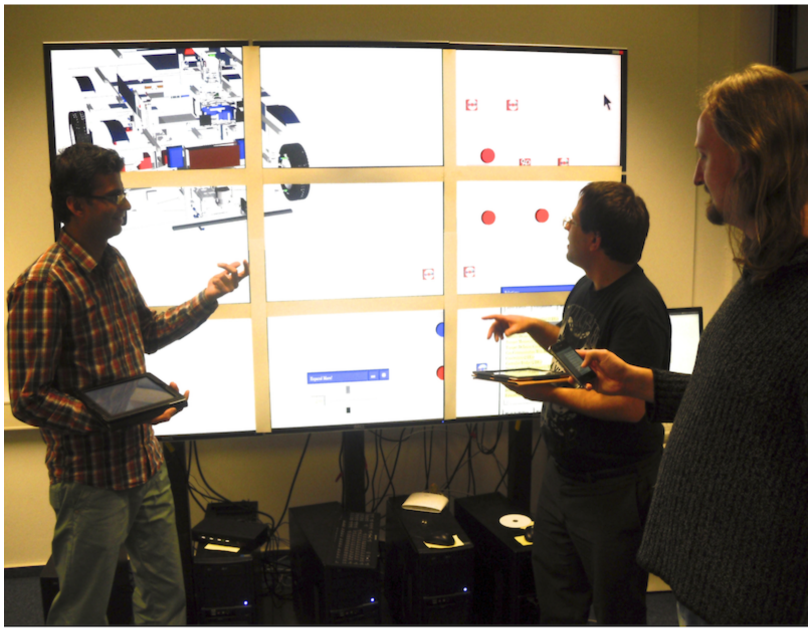


Fig. 7.17: Three users are interacting with the visual collaborative environment using different smart devices and discussing the results.

belongs to system engineers) want to analyze the current safety scenario. User-A is interested in understanding the failure path between two specific components in the current safety scenario, while user-B is interested in finding the location of these two components in the system 3D model. Both users are equipped with tablet devices and are standing in front of the TiledWall display. The tablets are connected with the large display via a wireless connection, and equipped with our developed mobile app (see Figure 7.18) that allows users to interact with the visual elements on the large display screen. For this scenario, user-A scrolls down the components' list in the safety scenario in the mobile app on the tablet device. Then he/she selects the interesting ones to highlight them and expand them if required. The result of this action will be reflected on the large display, i.e., both selected components will be highlighted in red color and an animated path will be drawn between them to show the failure relation in the abstract view of the safety scenario. At the same time, user-B shows the location of these two components in the system 3D model. The platform allows both users to interact simultaneously with the visual platform for extracting the required safety information about the system, which will be helpful for these users to discuss possible solutions.

Fig. 7.18: A user is interacting with the visual elements on big screen big screen through the developed mobile app by Bauer et al. [31] on an Apple iPad..

- **Scenario 2:** ***Changing the Perspective***

  We suppose that user-A is not satisfied with the size of the abstract view of the failure mechanism. User-A now wants to scale down or rotate the view according to the desired level. However, user-B has no problem with the size of the system 3D model. Our setup solves this conflict by providing each user an independent set of interaction facilities in order to allow them *scaling*, *rotating*, *zooming*, and *panning* the required view without distorting the other view.
  In this case, our setup allows each user to extract the relevant safety information from the required view and then synchronize it with the other view, without distorting the other view. Moreover, each user gets the textual description of his/her query on his/her smart device, which facilitates them in communicating and asking further questions to each other about the system condition. The collaboration in

such a visual interactive manner reduces the communication gap; therefore, it speeds up the maintenance process in such scenarios.

7.6 Summary

We presented our developed 3DintEx tool that supports intuitive interactions with the system 3D model. It provides different interaction techniques (e.g., the 3D selection tool, the exploded view option, the transparency controller, etc.) to interact with the underlying system 3D model in order to get more insight about the system on demand. The tool has been successfully applied initially for exploring the safety aspects in embedded systems. A possible future direction of this work is to provide a selective explosion operator over sub-parts of the main system.

Further, we presented the motivation behind our developed SceneMan tool as part of the ESSAVis framework. This tool is a scene manager unit and integrates between the 3D model view of the system and the graph representation of the CFT model. It was also designed to integrate smoothly and seamlessly between the other tools inside ESSAVis framework like the LayMan (see Chapter 5 and the 3DintEx. This tight integration between these different tools provides the users a rich insight about the safety situation of the underlying system.

We also presented the different views alignment options provided by the SceneMan tool that helps the users in controlling the two views on demand. Further, we presented different interaction options provided by ESSAVis that help the users in controlling the two views or getting the required information about the system 3D model or the graph representation or the current safety situation on demand. Moreover, we also listed different interaction options through the mouse and keyboard that are useful for controlling the two views.

Finally, we presented the setup to use the ESSAVis in a large visual collaborative environment [6]. This presented setup supports the failure analysis process of embedded systems in a collaborative fashion with multiple users from different background. This collaborative setup contains a large display (i.e., a TiledWall in our case) and smart devices (currently, we have developed the mobile app for Apple iOS devices), and it allows multiple users to interact individually with the large display and to collaborate with each other for understanding the current safety situation of the underlying system. A possible future work in this direction is to design and execute detailed user evaluation study with expert users from both groups (i.e., the system engineers and the safety experts) to better judge the effectiveness and the feasibility of our setup.

This chapter covered the 3D visualization side of our ESSAVis framework. In the forthcoming chapter (i.e., Chapter 8), we provide the details of an evaluation study that we conducted to compare between ESSAVis and ESSaRel [137] in terms of *accuracy*, *efficiency* and *user acceptance* in the domain of analyzing safety aspects in embedded systems.

8

The ESSAVis Framework Evaluation Study

In this chapter, we describe an empirical study containing a controlled experiment to compare between the two platforms, ESSAVis and ESSaRel [137], in terms of accuracy, efficiency, and user acceptance in analyzing safety aspects of failures occurred in embedded system represented as CFT models. We present the study design and the study goals. Then we provide the list of tasks we included in our comparative experiment. We explain the procedure we used to evaluate the differences between the ESSAVis and the ESSaRel. In the conducted empirical study, we invited 25 participants from different scientific backgrounds to undergo this test. Then we analyze the results using different statistical tests to measure how the results are significantly different based on the used tool. Results of the empirical study show that both tools provide relatively, the same accuracy level. However, ESSAVis outperformed ESSaRel in efficiency because participants were able to analyze tasks in the given safety scenarios in shorter time due to the advanced and intuitive interaction techniques provided by ESSAVis as well as its visual support. Finally, participants accepted ESSAVis better than ESSaRel again due to the visual support and interaction facilities provided by it. Many participants gave the credit to ESSAVis due to its intuitiveness and easiness.

The evaluation study was performed in a controlled environment. We tried to control all the possible parameters that can affect on participants' performance, like the seat position, the test platform, the test atmosphere, and the training session for all participants. As many of the real end users (i.e., the *safety experts* and the *system engineers*) participated in the experiment; therefore, we can call it a controlled user evaluation study.

This reminder of the chapter is divided into four main sections. The forthcoming section (i.e., Section 8.1) provides brief introduction to the ESSaRel tool that we used in the conducted evaluation study to compared with ESSAVis. We will not provide description about our developed ESSAVis platform here, as we have already provided it in Chapter 7. The remaining three sections provide details about the study, discussion about the results and findings. In Section 8.2, we present the experiment designing and execution details; i.e., the goals, the metrics, the hypotheses, the experiment procedure, the set of material required for the test, the participants' information, the result analysis approach, the list of tasks included in the experiment, and the analysis procedure. In Section 8.3, we discuss in details the results and the possible trends in participants' behavior during the experiment. Further, we provide details about the closed-ended-questionnaires and the explain the participants' responses in the open-ended questionnaires. Finally, we discuss the study findings and conclude this chapter in Section 8.4.

8.1 Introduction to ESSaRel

ESSaRel[1] (**E**bedded **S**ystems **Sa**fety and **Rel**iability Analyser) is a tool that was developed to provide a flexible dependability integration framework for embedded systems. It can compose and integrate analysis models with different characteristics, e.g., Fault Tree (FT), Component Fault Tree (CFT), and state/event fault trees [137]. ESSaRel has previously been used in industry (CESAR Project Report 2010 [45], p. 43) for the safety analysis process.

Figure 8.1 shows the main interface view of ESSaRel, The main centered window (i.e., the CFT window) is for editing the safety model. This is achieved by dragging the small graphical shapes and dropping them into this window's drawing area. This area is useful for depicting the participating components and the connections among them. The right window in the figure is used to show the system structure through the hierarchy of the components in the current safety scenario. This helps to detect the structural relations between the parent components and their children. ESSaRel is considered a powerful tool for editing the safety model [137]. However, it lacks the ability to show an overall

[1] http://www.essarel.de/

view of the safety scenario. This makes the process of tracing a failure rather difficult.

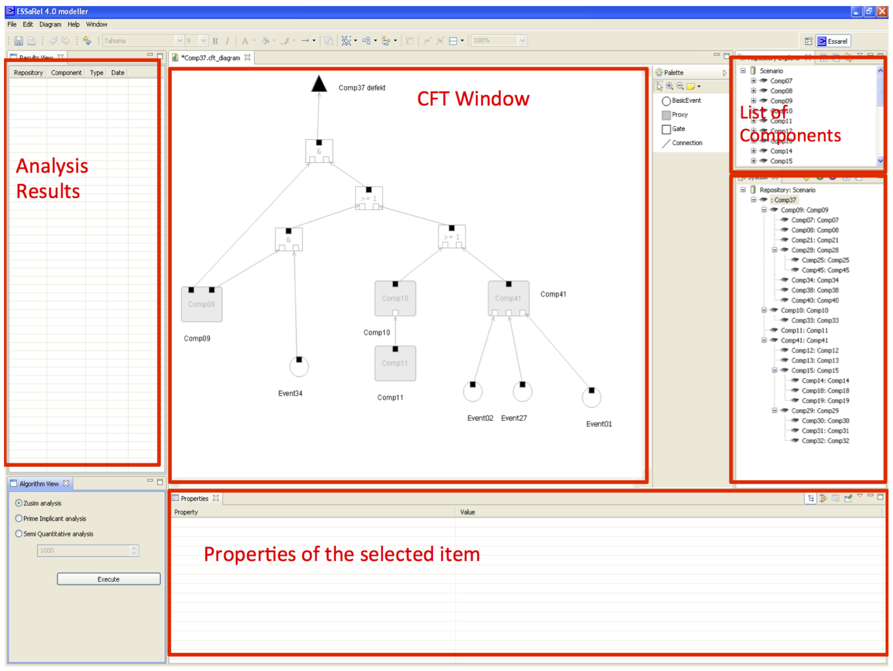


Fig. 8.1: The ESSaRel interface, where red rectangles show different main views.

ESSaRel helps safety experts in “modeling” the CFT model. It is an editor that offers basic functionalities required to model safety-critical scenarios in technical systems. It offers a multi-view feature to present the multi-level concept in the CFT model. Therefore, the internal structure of a specified component is hidden on the upper level. Users can select a specific component by using the mouse, which results in opening a new window showing the internal structure of the component.

Although this facility helps safety experts to modify the internal structure of the required component, it makes it difficult to understand the overall system structure because the new view does not preserve the relations between the current level and the upper level.

ESSaRel shows the structural relations among the system’s components in a side-view explorer (see right side of Fig. 8.1), where all the incorporated components in the current scenario are listed hierarchically according to their relations in the system. Again, the system structure shows the list of all collaborating components; however, it does not present the component-subcomponent relations visually. The user needs to open a new view to see the subcomponents' structure.

Therefore, in order to explore the internal structure of each component, the user needs to select the required component to allow displaying the next level of details. As mentioned above, the view opens in a new window without any indication of the previous level, which makes it difficult to investigate the internal structure of each component and identify the main reasons that cause a problem in the upper-level component.

Furthermore, tracing the failures between the multiple levels in the CFT model is also quite a difficult task in this approach. This leads to difficulties in tracing the failure propagation path between the required components.

8.2 Experiment Design and Execution

In this section, we provide details about the experiment designing and execution for our conducted study.

8.2.1 Goals and Metrics

Here we specify the goal and metrics of our conducted controlled experiment. The high level goal is firstly specified using the GQM goal specification template [29].

The Experiment Goal

Analyze ESSAVis and ESSAREL: A comparison between the both tools in terms of accuracy, efficiency and user acceptance level in analyzing fault tree models from one of the real end users' (i.e., safety experts) perspective.

Test Metrics

In order to compare the results for the two platforms, we identified a set of metrics by means of a GQM model [29]:

- *Metrics for Accuracy:* We measure and compare the percentage of correct answers for each task.
- *Metrics for Efficiency:* We measure and compare the time needed for answering each task
- *Metrics for User Acceptance:* We collect the participants' feedbacks and compare them.

Measurements are applied for each tool. Figure 8.2 presents the corresponding GQM Tree.

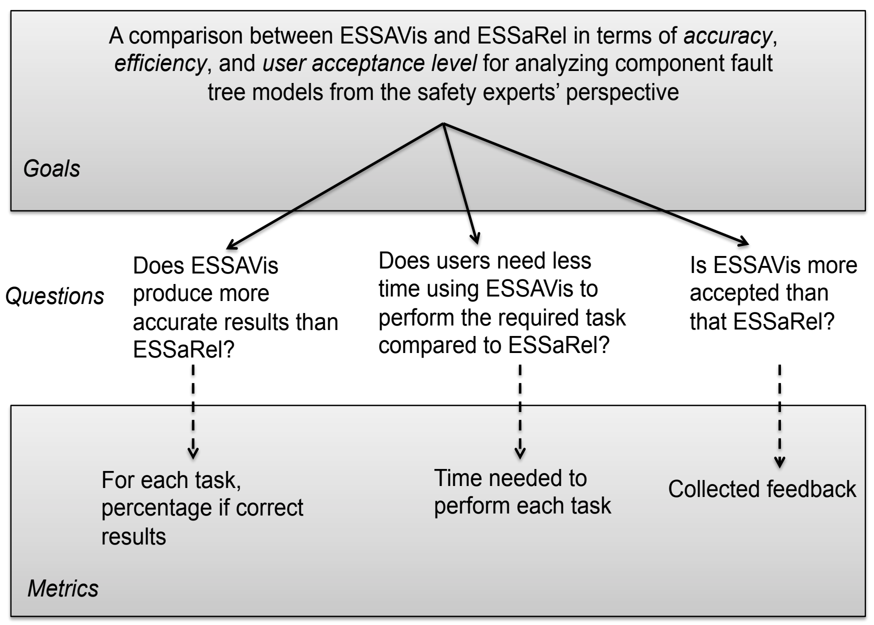

A comparison between ESSAVis and ESSaRel in terms of *accuracy*, *efficiency*, and *user acceptance level* for analyzing component fault tree models from the safety experts' perspective
Goals
Questions
Does ESSAVis produce more accurate results than ESSaRel?
Does users need less time using ESSAVis to perform the required task compared to ESSaRel?
Is ESSAVis more accepted than that ESSaRel?
For each task, percentage if correct results
Time needed to perform each task
Collected feedback
Metrics

Fig. 8.2: GQM Tree Specification for the designed experiment.

8.2.2 Hypotheses

In order to compare the results for both tools, we specified the following hypotheses:

- Hypothesis 1:
  - *Null hypothesis (H1.1):*
    Accuracy (ESSaRel) $>=$ Accuracy (ESSAVis)
  - *Alternative hypotheses (H1.2):*
    Accuracy (ESSaRel) $<$ Accuracy (ESSAVis).

- Hypothesis 2:
  - *Null hypothesis (H2.1):*
    Efficiency (ESSaRel) $>=$ Efficiency (ESSAVis)
  - *Alternative hypotheses (H2.2):*
    Efficiency (ESSaRel) $<$ Efficiency (ESSAVis)
- Hypothesis 3:
  - *Null hypothesis (H3.1):*
    User acceptance (ESSaRel) $>=$ User acceptance (ESSAVis)
  - *Alternative hypotheses (H3.2):*
    User acceptance (ESSaRel) $<$ User acceptance (ESSAVis)

8.2.3 Execution

Based on the identified goals and hypotheses, the user evaluation study was designed as a controlled experiment under laboratory conditions with a maximum timeframe of two hours per participant. To reduce learning effects, we randomized the participants for each sub-group in such a way that half of the participants ran the experiment first on ESSAVis and then on ESSaRel, while the other half did it the opposite way.

8.2.4 Participants

In this conducted experiment, 25 participants from the University of Kaiserslautern having different background participated. We divided the participants into two groups: the *real end users* group and the *other users* group. The real end users group included safety experts and system engineers with at least three years of industry experience from the Fraunhofer IESE institute and the Robotics Research Lab of the University of Kaiserslautern, who deal with embedded systems in their routine work. The other users group included software engineers and visualization experts from the same university. We created these two high-level categories in order to assess the participants' feedback according to their expertise in dealing with embedded systems. Table 8.1 provides the number of participants in each sub-group of the two above-mentioned groups. Out of 25 participants, 21 were male participants and 4 were female participants. The age of the participants ranged from 20 years to 42 years, with a mean of 28.8 years. A questionnaires form was used for knowing the background of each participant.

Group	Sub-Group	Frequency	Percentage
Real end users	Safety expert	6	24 %
	System expert	4	16 %
Other users	SE expert	4	16 %
	Vis. expert	11	44 %
Total		**25**	**100 %**

Table 8.1: Number of participants in each sub-group.

8.2.5 Study Procedure

To assure that the participants had a basic level of knowledge about the two platforms, a 30-minute tutorial on ESSAVis and a 30-minute tutorial on ESSaRel were conducted. Since the participants from the safety expert sub-group already had thorough knowledge of how to use the ESSaRel platform, they were excluded from the ESSaRel training session. We allowed all the participants to ask further questions during the training session and during the test in case of any technical issues. At the end of each test, we asked each participant to fill in two questionnaire forms. The first was a closed questionnaires form with eight questions, where each question offered six different options based on a Likert scale (scaled from 0 to 5 to show the degree of agreement with each question from the participant's perspective; the sixth option was "Don't know"). The second form was an open-ended questionnaires form with three general questions, where one of three options could be selected (i.e., ESSAVis, ESSaRel, and Doesn't Matter). The participants were also asked to give their feedback about the selected options. Each test lasted for one hour per participant, excluding the training time and the time for filling in the questionnaires.

The experiment was performed using a desktop environment, with the participants seated in front of the computer. They interacted with the platform with a normal mouse and a keyboard. We allowed the participants to try the target platform prior to the test and encouraged them to ask for more details about any interaction technique during the test.

During the experiment, only one trainer participated in order to avoid any differences in the given material and the training session to all participants. In Table 8.2, we show the chronological procedure of the experiment.

Step order	**Description**
1	Collecting participants information
2	Training: Providing the safety information background. Train participants on how to use the targeted tool
3	Feedback session on the training material
4	Introduction to the study
5	Presenting the main steps of analyzing the safety scenario using the targeted tool
6	Distribution of the background information material
7	To analyze the system further the Root component is investigated. Calculate the MCSs for the Root component. Order the MCSs by size and find the most critical ones. ($size < 3$), hint: after the first 12 there are no smaller ones. Order the MCSs by probability and find the most critical ones. ($p > 1e-12$), hint: after the first 12 there are no bigger ones.
8	Find the subcomponents of the Root component that are the most critical ones. Find the components that are affected by the most critical MCSs
9	Give the number of basic events in the Root-component?
10	Show the failure propagation for the events in a pre-specified component
11	Find the interaction/connections of Component X and Component Y respecting the system failure
12	Find the events in the root-component that have the highest impact on safety of the system. Find the events that are in the most critical MCSs.
13	Filling the closed-ended questionnaires.
14	Filling the opened-ended questionnaires.

Table 8.2: The test procedure description

During executing the test for both tools with participants; however, the following were the deviations from the test plan:

- Using different rooms for each tool.
- Differences in rooms temperature, as one of the room was with air conditioner while the other room was without. The test was conducted in summer; therefore, we think there might be differences in participants' performance on each tool based on the surrounding atmosphere.
- Participants were responsible on collecting time information by means of their stop watch.

8.2.6 Study Scenario and Tasks Description

Based on the defined system model, an expert in safety analysis from the *Software Engineering Research Group: Dependability* of the University of Kaiserslautern defined a safety scenario and modeled the corresponding safety models.

The defined safety scenario we used in our study describes one of the possible situations where the RAVON robot [124] can hit other solid objects. The assumption of this scenario is that RAVON moves forward in a moderate drive mode. Moreover, there is no water on the way nor are there any unexpected slopes or gradients. The components of this possible hazard comprise 48 compound components in the RAVON structure with 70 basic events that can appear in different locations in the FT model simultaneously. However, due to the test constraints, we decided to include only part of the scenario. The scenario we used in our test consisted of 25 nodes with more than 33 basic events.

It is important to note that the safety scenario was the same in both tests, with different names in order to avoid any learning effects.

In total, six tasks were defined for both tools with the help of the expert in safety analysis:

- **Task 1:** Find the Minimal Cut Set (MCS) lists for the underlying safety scenario. This included three sub-tasks:
  - **Task 1.1:** Find the MCS information.
  - **Task 1.2:** Sort the found MCS according to their size.
  - **Task 1.3:** Sort the found MCS according to their probability values.
- **Task 2:** Find the components that are affected by the most critical MCSs.
- **Task 3:** Find the number of basic events in the root component.
- **Task 4:** Find the failure path relation between the two pre-specified components.
- **Task 5**: Find the interaction/connections of components **CX** and **CY** respecting the system failure.
- **Task 6:** Find the events that have the highest impact on the safety-level of the system. (**Hint:**) find the events that are in the most critical MCSs.

Before performing tests with all participants, we conducted a pilot study with a safety professional from the Fraunhofer Institute for Experimental Software Engineering (Fraunhofer IESE), who was involved throughout this whole evaluation process. The above-described settings were finalized after changes were made in the initial settings according to this pilot study feedback.

8.2.7 Analysis Procedures

During and after the experiment, we collected data to be analyzed for testing our hypotheses. For each task, we computed and reported the accuracy of results (in percentage) and the time needed in seconds.

The answer for each closed-ended questionnaire was given on a scale between 1 to 5:

- 1: The subject strongly disagrees
- 2: The subject disagrees
- 3: The subject neither disagrees nor agrees
- 4: The subject agrees
- 5: The subject strongly agrees

The sixth option “Don’t know” was provided in the case if the participant was unable to give an answer.

For each open-ended questionnaire, participants had to choose between ESSAVis, ESSaRel and “it doesn’t matter”.

Except for the open-ended questionnaires, a descriptive statistic analysis was performed for giving quantitative statistical information about the metric. For comparing the metric scores between both groups, we first tested the data for normality with a Shapiro-Wilk’s test [134]. When scores for both groups were normally distributed, then we performed an Independent T-Test [134] for comparing the means; otherwise, we performed a median test for comparing the medians. For open-ended questionnaires, we compareed frequencies of participants’ choices.

8.3 Results and Discussion

In this section, we describe the results of our study and analyze these collected results for different evaluation tasks. We also report analysis results for each metric of the GQM tree (Figure8.2). For each task, we compute and report the accuracy of the results (in terms of number of *correct answers / number of right answers*) and the time needed to complete the task (in seconds). Except for the open-ended questionnaires, we performed a descriptive statistic analysis to provide quantitative statistical information about the relevant metrics.

To compare the metric scores between the both groups (i.e., *real end users* and *other users*), we first tested the data for normality with a Shapiro-Wilk's test [134]. When scores for both groups were normally distributed, we performed an Independent T-Test [134] to compare the means. Moreover, we also performed a median test to compare the medians. For the open-ended and the closed questionnaires, we compared the frequencies. In most tasks, it was shown by a Shapiro-Wilk test that the data in the two groups are not normally distributed. Therefore, we ran a Wilcoxon Signed-Rank test [134] to determine if there were differences in the time needed for each task on both platforms.

We show the overall overview of the study results in Table 8.3. In this table, we divided the participants into two groups (i.e., *real end users* and *other users*), as mentioned in the previous section. Further, Figure 8.3 and Figure 8.4 shows the results in terms of accuracy (in percentage) and completion time (in seconds) through three categories: the *real end users* group, the *other users* group, and a combined result of all participants. This group categorization helps to understand the behavior of the participants' performance according to their background.

Results Analysis of Task 1.1:

In Task 1.1, the participants were asked to calculate the Minimal Cut Set (MCS) information related to the current safety scenario. This information is very important from the safety experts' perspective in order to assess the safety criticality of the underlying system. MCSs provide information about the safety-critical components in the scenario and the critical safety path between these critical components. That is why both tools (i.e., ESSAVis and ESSaRel) provide this information due to its importance in the analysis process. Expected answers for this task were *Yes* or *No*. This task measured the time that participants needed for calculating the required information and then started analyzing the safety scenario accordingly. All participants answered with *Yes* in both groups. Regarding the needed time, as shown by a Shapiro-Wilk test, data in both groups are not normally distributed.

A Wilcoxon Signed-Rank test was run to determine if there were differences on the time needed for this task when using ESSAVis compared to when using ESSaRel. There was a statistically significant reduction in time needed when using ESSAVis, as 19 participants needed less time with ESSAVis while 5 participants needed less time with ESSaRel. However, the analysis per participant category did not contradict the general result.

In this task (i.e., Task 1.1), there was a statistically significant reduction in time needed when using ESSAVis, as 19 participants spent less time with ESSAVis, whereas 5 participants spent less time with ESSaRel. However, the analysis per participant category did not contradict the general results. ESSAVis provides this information in an intuitive manner as it presents this information in one of its menus, called the *safety information menu*. Users can trigger this menu option simply by interacting with the ESSAVis *Main Menu* by right-clicking the mouse button. They do not need to calculate it manually, as it is calculated automatically for the root component. In the case of ESSaRel, users need to perform a relatively longer process. They get the MCS information via three steps: (1) select the option “MCS Calculation Option” from the “Analyze Menu” in the menu bar; (2) select the required component, which is the root component in our case; (3) ESSaRel shows the MCS information in a separate window through a table representation.

Results Analysis of Task 1.2:

In Task 1.2, the participants were asked to calculate the most critical MCS information in the given safety scenario according to the *size parameter*. The MCS is considered to be critical if it has a lower number of basic events. In our test, an MCS was considered critical if it had less than three basic events.

To achieve this task in ESSAVis, participants were trained to trigger the safety menu and then chose the minimal cut set tab. This shows the list of all minimal cut set from the current CFT model. Moreover, ESSAVis provides the facility for sorting this list according to the size parameter or the probability value. In ESSaRel, the list of minimal cut set is shown together with their size information. However, there is no option to sort this list, which can easily be fixed if the ESSaRel developers consider adding this option in the future.

In this task, 8 participants provided more accurate results with ESSAVis, 5 participants with ESSaRel, while the remaining 11 participants' accuracy was the same in both cases. In both tools, the participants were able to accomplish this task accurately; however, the average time was comparably less in ESSAVis (i.e., 42 sec.) compared to ESSaRel (i.e., 107 sec.). This is because ESSAVis automatically sorts the MCS menu entries, unlike ESSaRel, where this is done manually. The average accuracy of all participants in this task was 98% in ESSAVis and 85% for ESSaRel, see Table 8.3 and Fig. 8.3c. We observed that this difference in performance is not due to the visualization power of ESSAVis; in fact, it is more related to the lack of some options in ESSaRel. The required time of this task for all participants in ESSAVis was 42 seconds while it was 107 seconds for ESSaRel, see Table 8.3 and Fig. 8.4c. This shows a significant difference in time for completing such a simple task in both tools.

Results Analysis of Task 1.3:

Task 1.3 was to order the pieces of the critical MCS information according to the probability value of each one. The MCS is also considered to be critical if it has a high probability value of failing. In our test, we specified $> 1 * 10^{-12}$ as the probability threshold.

Again, ESSAVis provides an option to sort the entries list according to their probability values, while in ESSaRel, users need to calculate this information manually. In this case, the participants were able to calculate this information in ESSAVis easily and with high accuracy (i.e., 94% accuracy, see Table 8.3 and Fig. 8.3c), while in ESSaRel, many participants decided to skip this task due to the required tedious calculations. Moreover, the time required for accomplishing this task in ESSAVis (i.e., 39 sec.) was significantly less than in ESSaRel (i.e., 740 sec.), see Table 8.3 and Fig. 8.4c. Again, this is due to the automated nature of ESSAVis over ESSaRel. However, this can be improved easily if ESSaRel developers would, in the future, consider automating the sorting option and the probability calculation step. At the time we performed the study, ESSaRel did not provide an automated option to accomplish these basic tasks.

Results Analysis of Task 2:

Task 2 was dedicated to find the set of the affected components by the most critical minimal cut sets. This task required participants to investigate both the CFT visualization and the list of minimal cut sets.

This task depended on the results of the Task 1, as participants were supposed to find out the most critical minimal cut sets from in that task.

In ESSAVis, participants were able to achieve this task by selecting the critical minimal cut set from the *safety menu* then click on the *Expand-to-Show* and *Mark-as-Failing* buttons at the bottom of the menu. This opens the compound components until it reaches to the required components and then it highlights them in red color. Users of the ESSAVis can get the names of these components by putting the mouse over the highlighted parts, and then they can achieve this task. However, the situation in ESSaRel is different, as it does not provide this kind of interactions. Participants were supposed to search manually for these components in the CFT representation based on the information from the previous task.

In this task, 13 participants provided more accurate results with ESSAVis, 5 participants provided more accurate results with ESSaRel, and the accuracy results of the remaining 6 participants were equal in both cases. The overall accuracy value for ESSAVis was 0.82%, while the accuracy value for ESSaRel was 0.67%. Moreover, the overall difference in time was also not significantly different in both cases (i.e., 200 sec. for ESSAVis and 188 sec. for ESSaRel).

However, we observed that the *vis. expert* sub-group outperformed all the remaining sub-groups using the ESSAVis tool. This is probably due to their greater experience in doing visual query and navigation over the different data views. ESSAVis provides an interactive way to highlight the components in the abstract representation of the safety scenario. Users can achieve this task quickly and easily by clicking on an option attached to the selected MCS, as ESSAVis then highlights the required components. Users can also expand the view to the level where these components are located in the abstract representation of the failure. Therefore, the participants found the names of the required components easily by hovering the mouse over the highlighted components. Hence, extracting this information using ESSAVis was more intuitive than using ESSaRel because ESSaRel provides this information via an undirected process.

The average accuracy of all participants in this task using ESSAVis was 84% while it was 67% in ESSaRel, see Table 8.3. The required time was 200 seconds in ESSAVis and 187 seconds in ESSaRel, see Table 8.3. We observed that the time difference is not significant as participants were using first time the ESSAVis. We expect that this can be improved if participants use ESSAVis for longer time so they get familiar with the tool.

Results Analysis of Task 3:

In Task 3, the participants were asked to find the number of basic events in the root component of the given safety scenario. The participants' understanding of the task differed: Some of them calculated the number of basic events on the root-component level only, while others calculated this information recursively level by level over the whole tree. Therefore, we obtained different answers depending on the participants' understanding. If they understood it as the number of basic events at the root-component level only, then they calculated the number of cycle symbols in the safety scenario in both platforms. However, if they understood it the other way around, then they summed up the number of basic events at each component level by level. ESSAVis provides this information directly through the *safety-information menu*; whereas in ESSaRel, users need to traverse the entire safety scenario hierarchy and then sum up the number of basic events in each component one by one.

Due to the differences in understanding, we received different responses to this task, which is why we cannot determine accurately which platform performed better. The accuracy value was relatively similar for both platforms (i.e., 0.88% and 0.87%), , see Table 8.3. However, the time required to complete this task using ESSAVis (i.e., 75 sec.) was significantly less than when using ESSaRel (i.e., 193 sec.), , see Table 8.3. This is due to the visual support and the interaction techniques provided by ESSAVis, as it allows expanding and closing the required component without losing the overall context.

Results Analysis of Task 4:

In Task 4, the participants were asked to find the logical relations between the two pre-specified components as well as the failure propagation path between these two components. With regard to finding the path between these two components, four participants were more accurate in ESSAVis, two participants were more accurate in ESSaRel, and the remaining participants' accuracy level was the same in both cases. However, the average time in case of ESSAVis (i.e., 75 sec.)

was significantly less than the average time required in ESSaRel (i.e., 193 sec.), see Table 8.3. This is because ESSAVis provides animations to show the failure relations between different components. Also, as mentioned earlier, ESSAVIS provides better interaction options for expanding and closing the required component in real time.

Many participants also mentioned during the test that ESSAVis is more intuitive compared to ESSaRel because of the interaction options, the visual support, and the different highlighting techniques provided by ESSAVis. Especially the safety experts, who belong to the real end users of ESSAVis and work with ESSaRel in their daily work routines, appreciated that ESSAVis shows the multi-views of the related components in the safety scenario.

Results Analysis of Task 5:

In Task 5, participants were given two components and they were asked to give the logical connections between them. These two components were on different levels without any structural connection between them. The only relation between them was the failure relation.

In order to find the logical relation between the two pre-specified components, the participants were asked to navigate through the failure hierarchy. ESSAVis keeps the gate information but does not provide an explicit representation of it, while ESSaRel provides it explicitly and shows it through symbols. This task can be accomplished in ESSAVis through clicking the two specified components and then noting the color of the Top-Event node. If the top-event node becomes red, this means that both components are in one minimal cut set and have an AND relationship; otherwise (if the color is blue) they are not. The result of this task shows a non-significant difference between the two tools, as 5 participants were more accurate with ESSAVis, 4 participants were more accurate with ESSaRel, and the remaining 15 participants' accuracy level was the same in both cases. Regarding the time required for this task, 15 participants spent less time with ESSAVis, 7 participants spent less time with ESSaRel, and one participant spent the same amount of time in both cases. The average accuracy in this task for all participants using ESSAVis was 68%, see Table 8.3.

In ESSaRel, the participants were expected to navigate through the CFT model manually and keep in mind the logical connections between these two components. However, some of the safety experts used the minimal-cut-set information and found if these two components are in one minimal cut set or not, and if so then they have an *AND* relationship otherwise they do not affect the TopEvent. The average accuracy of this task for all participants in ESSaRel was 62%, see Table 8.3.

There were no significant differences between the two tools in this task regarding the accuracy. However the time of achieving this task in ESSAVis was 100 seconds in compare to 150 seconds for ESSaRel, see Table 8.3.

Results Analysis of Task 6:

In Task 6, participants were asked to find the most influential basic event in the current CFT model. This task can be achieved in two ways, either the basic event with the highest probability to fail or the one that occurred in the most critical minimal cut set has to be found. Therefore, for this task we collected four different answers (two possibilities for both tools). We allowed having this variation in the understanding as the safety expert who helped us in designing the experiment defined the most influential basic event in the system based on these two mentioned definitions. Therefore, for this task we collected four different answers (two possibilities for both tools). In ESSAVis, the information about the probability of each basic event is listed in the basic events tab of the Safety-Menu, while in ESSaRel it is shown in a properties panel. The minimal cut set information was available through a menu in both tools. However, in ESSaRel participants should navigate manually through the CFT to extract the basic-event probability.

In regard of which definition the participants follow, they gave us partially correct answers. The average accuracy was 84% in ESSAVis. In ESSaRel, participants needed to perform the analysis manually to get this answers; therefore, the average accuracy for this task was lower than the ESSAVis (i.e., 76%), see Table 8.3. The time of this task was relatively similar in the two tools as it was 110 seconds in ESSAVis and 103 seconds in ESSaRel, see Table 8.3.

Results Analysis of Closed-Ended Questionnaires:

Table 8.4 and Table 8.5 show all participants' feedback in the closed-ended questionnaires forms that were given them after finishing the test on ESSAVis and ESSaRel respectively. In these figures, the two most frequent answers given in response to each question are highlighted in yellow color. Here we analysis the results of these closed-ended questionnaires that show participants' trend and satisfaction towards the two used platforms.

- **CEQ 1:** *It was easy to get the safety information about the safety scenario (efficiency + accuracy)*

  As shown by a Shapiro-Wilk test, the data in both groups (i.e., the ESSAVis testing group and the ESSaRel testing group) are not normally distributed. Therefore, a Wilcoxon Signed-Rank test was run to determine if there were differences on the answers given by participants. There was a statistical significant difference between both groups in favor of ESSAVis, as 18 participants provided a better feedback for ESSAVis, 1 participant provide for ESSaRel, and 4 participants provided the same answer for the both tools. The analysis per participant category does not contradict the general result.

- **CEQ 2:** *The abstract view in ESSAVis / ESSaRel was helpful in picking the components and the relations between them. (efficiency + accuracy)*

  As shown by a Shapiro-Wilk test, the data in both groups are not normally distributed. Therefore, a Wilcoxon Signed-Rank test was run to determine if there were differences on the answers given by participants. There was a statistical significant difference between both groups in favor of ESSAVis, 18 participants provided a better feedback for ESSAVis, 5 participants provided for ESSaRel, and 1 participant provided the same answer for the both tools. The analysis per participant category does not contradict the general result.

- **CEQ 3:** *The abstract view of the failure mechanism in ESSAVis / ESSaRel provides an overall structure of the failure mechanism. (efficiency)*

  As shown by a Shapiro-Wilk test, the data in both groups are not normally distributed. Therefore, a Wilcoxon Signed-Rank test was run to determine if there were differences on the answers given by participants. There is a statistical significant difference between both groups in favor of the ESSAVis, as 21 participants provided a better feedback for ESSAVis while 2 participants provided for ESSaRel. The analysis per participant category does not contradict the general result.

- **CEQ 4:** *Overall, the interaction facilities of the ESSAVis / ESSaRel were enough to get the required safety information. (efficiency + accuracy)*

  As shown by a Shapiro-Wilk test, the data in both groups are not normally distributed. Therefore, a Wilcoxon Signed-Rank test was run to determine if there were differences on the answers given by participants. There is a statistical significant difference between both groups in favor of ESSAVis, as 21 participants provided a better feedback for ESSAVis, 1 participant provided for ESSaRel, and 1 participants provided the same answer for the both tools. The analysis per participant category does not contradict the general result.

- **CEQ 5:** *Overall, it was effective to use the different interactions styles provided by ESSAVis / ESSaRel to get the required information.*

  As shown by a Shapiro-Wilk test, the data in both groups are not normally distributed. Therefore, a Wilcoxon Signed-Rank test was run to determine if there were differences on the answers given by participants. There is a statistical significant difference between both groups in favor of ESSAVis, as 19 participants provided a better feedback for ESSAVis and 2 participants provided for ESSaRel. The analysis per participant category does not contradict the general result.

- **CEQ 6:** *Overall, how much you liked the idea of expanding the required component in ESSAVis / ESSaRel.*

  As shown by a Shapiro-Wilk test, the data in both groups are not normally distributed. Therefore, a Wilcoxon Signed-Rank test was run to determine if there were differences on the answers given by participants. There is a statistical significant difference between both groups in favor of ESSAVis, as 18 participants provided a better feedback for ESSAVis, 2 participants provided for ESSaRel, and 3 participants provided the same feedback for the both tools. The analysis per participant category does not contradict the general result.

- **CEQ 7:** *Overall, the ESSAVis / ESSaRel tool would increase the performance of people who work in such related environments.*

  As shown by a Shapiro-Wilk test, the data in both groups are not normally distributed. Therefore, a Wilcoxon Signed-Rank test was run to determine if there were differences on the answers given by participants. There is a statistical significant difference between both groups in favor of ESSAVIS, as 16 participants provided a better feedback for ESSAVis and 1 participant provided the same feedback for the both tools. The analysis per particpant category does not contradict the general result.

- **CEQ 8:** *Overall, with the ESSAVis / ESSaRel tool, the safety analyzing process was easy to understand.*

  As shown by a Shapiro-Wilk test, the data in both groups are not normally distributed. Therefore, a Wilcoxon Signed-Rank test was run to determine if there were differences on the answers given by participants. There is a statistical significant difference between both groups in favor of ESSAVis, 20 participants provided a better feedback for ESSAVis, 1 participant provided for ESSaRel, and 2 participants provided the same feedback for the both tools. The analysis per participant category does not contradict the general result.

From the feedback of the closed-ended questionnaires we observed that most of the participants (i.e., 22 participants) gave credit to ESSAVis rather than to ESSaRel (i.e., 3 participants), due to the interaction facilities and the visual support provided by ESSAVis. However, we cannot deny the minority of participants who preferred ESSaRel over ESSAVis. These participants mentioned that the ESSAVis interaction approach is a bit confusing and very new for them, making it difficult to adapt to it. For these participants, the interaction methods provided by ESSaRel were more "intuitive". These participants mainly had a software engineering background, and we observed that the main reason for their preference for ESSaRel was their experience with similar tools as ESSaRel.

We also observed that participants without any previous experience in using visual cues for analyzing exhibited more resistance towards the visual support of ESSAVis. Most of these users were older than 38 years, which somehow shows a rough indication of their commitment to the old habit of using the textual form of analysis rather than modern interaction analysis facilities, which rely more on visualization and interactive interfaces. However, we believe that this resistance can be eliminated through training and practice.

Results Analysis of Open-Ended Questionaires:

In the feedback from the open-ended questionnaires, the majority of the participants chose ESSAVis as the preferred tool for working in the future. Only two out of twenty-five participants voted for the ESSaRel tool. We found that these two participants were more familiar with the kind of interfaces provided by ESSaRel. Table 8.6 shows the overall participants' preference in each open-ended question.

Results Analysis Conclusion

We conclude from the above discussion that all these results' findings do not contradict the hypotheses of the study. We can also conclude that the visual support and the interaction techniques help users to understand the safety scenario more accurately and more swiftly. Moreover, we observed that the platform that automates the background calculations really helps users to trust the analysis result more than the platform where the users need to perform these calculations manually. We can trace this observation from the acceptance ratio of ESSAVis compared to ESSaRel.

8.4 Conclusions and Findings

In this chapter, we provided the results of the conducted comparative evaluation study between our developed ESSAVis tool and the ESSaRel tool that was designed to be an editor of the CFT models. The evaluation study focused on analyzing part of the CFT model. The tests in the experiment consisted of those basic tasks that are necessary to measure the criticality of the Top-Event situation. The goal of this evaluation study was to measure how the visualization can help in speeding up the safety analysis process in embedded systems domain as well as making this process more accurate and effective. We applied this study on analyzing a CFT model representing a safety critical situation in RAVON [124] robot. Results of the study show an increased accuracy level among the participants when they used ESSAVis in compared to ESSaRel with relatively shorter times for all tasks.

The improvements in results not only depends on the visualization capabilities of ESSAVis but also on the automation process of implemented algorithms in it, which are required to calculate the probability of each minimal-cut-set. We also observed that the majority of participants preferred ESSAVis compared to ESSaRel due to its intuitiveness and the different interaction techniques provided by ESSAVis.

Generally speaking, from the results in the previous section we can see that ESSAVis outperformed ESSaRel in most of the tasks, mainly due to its visual support provided during the analysis process. Further, ESSAVis provides institutive facilities to interact with the abstract representation of the safety scenario, which is missing in ESSaRel. However, ESSaRel was mainly designed to support the safety scenario modeling process and partially designed to analyze these generated safety scenarios. Due to this difference, we designed our study to consider only those options that are provided by the both tools in analyzing the safety scenario. Therefore, we excluded some tasks that are provided only by ESSAVis and not by ESSaRel, in order to allow us to objectively compare between the two tools.

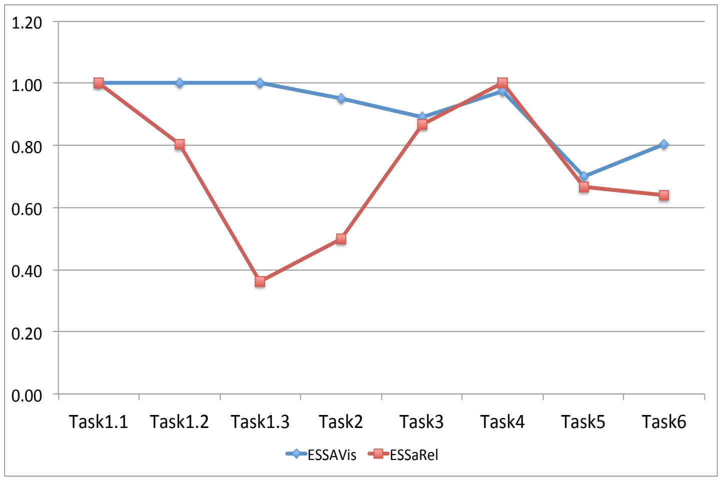
After conducting the experiment, we came up with some conclusions regarding the further steps in supporting the safety analysis process in embedded systems. Generally, we believe that the visual support and the intuitive interaction techniques help the users in understanding the safety scenario faster and more accurately. We also believe that the tool that automates the background calculations really helps the users in trusting the analysis results more than the manual-approach based tools. We can find this from the acceptance ratio of ESSAVis by the participants over the ESSaRel tool.

As in every study, there are some limitations for our study, too. First, the study was designed to include only those tasks that can be performed using either platform. On the one hand, there were tasks that were supported by ESSaRel more than by ESSAVis, like editing the safety scenario or changing the probability values of the safety scenario elements. On the other hand, there were also tasks that were supported only by ESSAVis through its visual options. We avoided evaluating those tasks that cannot be performed by either of the two platforms in order to maintain consistency in the results.

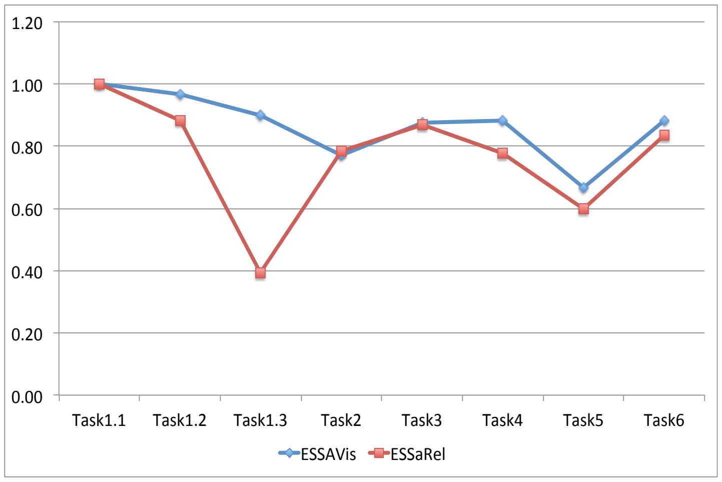
Due to some technical issues, we conducted the two tests in different rooms, with the ESSaRel test being performed in a relatively warm room with the temperature ranging from 25 to 30 degree inside the room. This might have affected the participants' performance, as they may have wanted to finish the test as quickly as possible. Two of the participants explicitly complained about this problem, which is why we believe that those participants skipped some of the tasks that required more time in ESSaRel.

Group (Metric)	**Task 1**						**Task 2**		**Task 3**		**Task 4**		**Task 5**		**Task 6**	
	ESSAVis			ESSAREL			ESSAVis	ESSAREL	ESSAVis	ESSAREL	ESSAVis	ESSAREL	ESSAVis	ESSAREL	ESSAVis	ESSAREL
	Task 1.1.1	Task 1.1.2	Task 1.1.3	Task 1.1.1	Task 1.1.2	Task 1.1.3										
Real End-Users (Accuracy)	1	1	1	1	0.85	0.36	0.95	0.5	0.89	0.87	0.92	1	0.68	0.67	0.88	0.64
Other Users (Accuracy)	1	0.98	0.94	0.9	0.81	0.39	0.77	0.78	0.88	0.87	0.88	0.78	0.7	0.6	0.81	0.83
All Users (Accuracy)	**1**	**0.96**	**0.88**	**0.95**	**0.88**	**0.38**	**0.84**	**0.68**	**0.88**	**0.87**	**0.92**	**0.86**	**0.68**	**0.63**	**0.84**	**0.76**
Real End-Users (Time)	47.66	45.3	28.65	78.7	79.4	413	213.1	171.7	33.95	81.13	33.5	81.125	106.2	138.5	93.4	101.34
Other Users (Time)	39.71	39.72	46.12	70.6	124.1	1002	192	187.4	103.2	253.6	103.2	253.6	98	159.7	124	85.11
All Users (Time)	**42.89**	**42**	**39.13**	**73.7**	**107.4**	**740**	**200.4**	**181.5**	**75.48**	**193.7**	**75.48**	**193.6**	**101.2**	**151.3**	**112.2**	**111.5**
Real End-Users (Efficiency)	2.10%	2.21%	3.49%	1.27%	1.07%	0.09%	0.45%	0.29%	2.62%	1.07%	2.75%	1.23%	0.64%	0.48%	0.94%	0.63%
Other Users (Efficiency)	2.52%	2.47%	2.04%	1.27%	0.65%	0.04%	0.40%	0.42%	0.85%	0.34%	0.85%	0.31%	0.71%	0.38%	0.65%	0.98%
All Users (Efficiency)	**2.33%**	**2.29%**	**2.25%**	**1.29%**	**0.82%**	**0.05%**	**0.42%**	**0.37%**	**1.17%**	**0.45%**	**1.22%**	**0.44%**	**0.67%**	**0.42%**	**0.75%**	**0.68%**

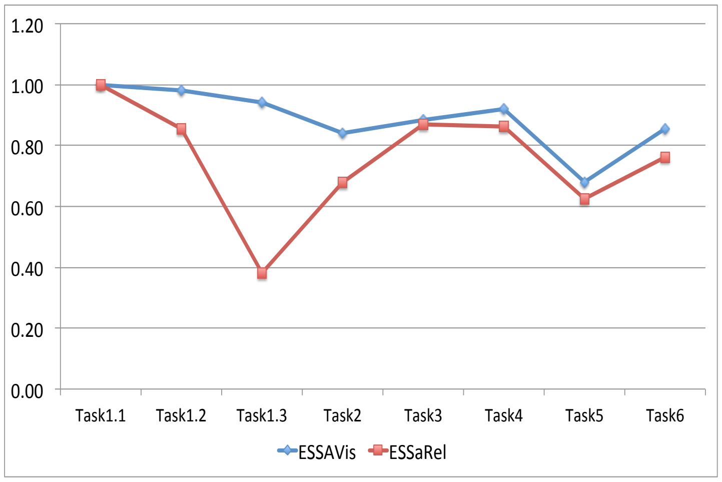
Table 8.3: All statistical values for each tool in the experiment.



(a) Results of the *real end users* group (i.e., safety experts and system engineers)

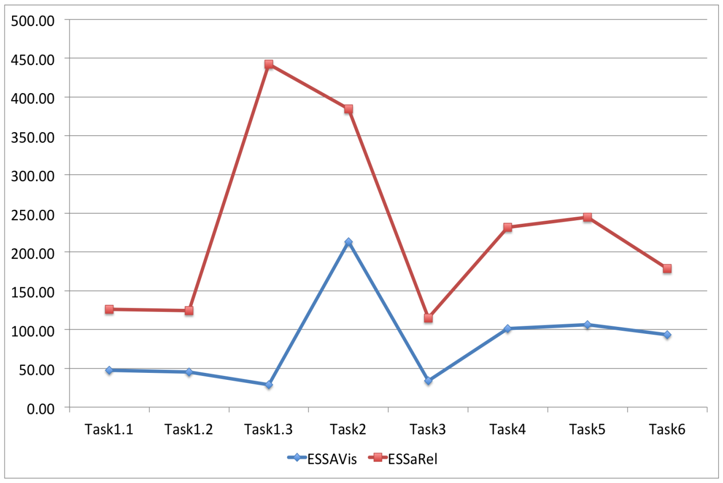


(b) Results of the *other users* group (i.e., SE experts and Visualization Experts)

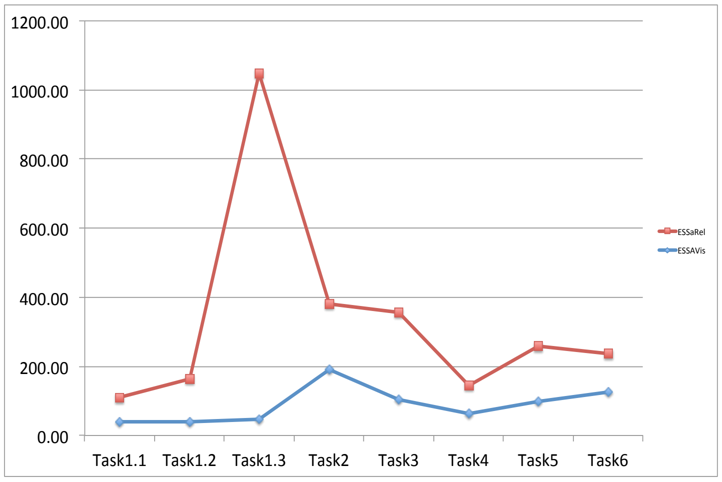


(c) Results of all participants

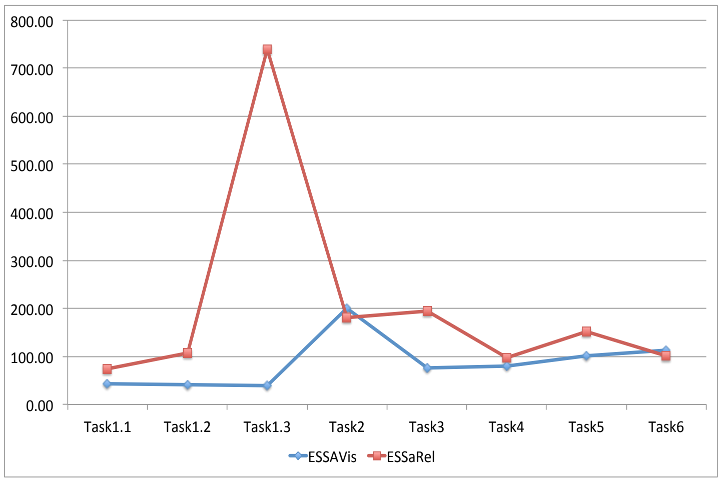
Fig. 8.3: Participants' average accuracy in percentage for completing the tasks in the experiment.



(a) Results of the *real end users* group (i.e., safety experts and system engineers)



(b) Results of the *other users* group (i.e., SE experts and Visualization Experts)



(c) Results of all participants

Fig. 8.4: Participants' average time in seconds for completing the tasks in the experiment.

1 = Strongly Disagree 2 = Disagree 3 = Neutral 4 = Agree 5 = Strong Agree

Questions	1	2	3	4	5	Don't Know
It was easy to get the safety information about the safety scenario	1	1	2	12	6	1
The abstract view was helpful in picking the components and the relations between them.	1	2	1	11	9	0
The abstract view of the failure mechanism provides an overall structure of the failure mechanism.	0	1	1	10	11	1
Overall, the interaction facilities in ESSAVis were enough to get the required safety information.	0	2	3	11	7	0
Overall, it was effective to use the different interactions styles provided by ESSAVis to get the required information.	1	1	1	13	8	0
Overall, how much you liked the idea of expanding the required component and show the internal structure of it with the overview structure of the safety scenario.	1	1	2	3	17	0
Overall, ESSAVis would increase the performance of people who work in such related environments.	0	0	1	11	8	3
Overall, with the ESSAVis system, the safety analysing process was easy to understand	0	1	5	8	10	0

Table 8.4: All participants' feedback in the closed-ended questionnaires form after they finished the **ESSAVis** test.

1 = Strongly Disagree 2 = Disagree 3 = Neutral 4 = Agree 5 = Strong Agree

Questions	1	2	3	4	5	Don't Know
It was easy to get the safety information about the safety scenario	1	9	6	1	2	0
The abstract view was helpful in picking the components and the relations between them.	2	4	10	3	3	0
The abstract view of the failure mechanism provides an overall structure of the failure mechanism	6	6	6	3	2	0
Overall, the interaction facilities in ESSaReL were enough to get the required safety information.	11	4	2	3	0	1
Overall, it was effective to use the different interactions styles provided by ESSaReL to get the required information.	8	6	3	0	2	3
Overall, how much you liked the idea of expanding the required component in a new window	5	5	3	5	3	1
Overall, ESSaReL would increase the performance of people who work in such related environments.	2	7	4	2	1	4
Overall, with the ESSaRel tool, the safety analysing process was easy to understand	5	5	6	2	1	1

Table 8.5: All participants' feedback in the closed-ended questionnaires form after they finished the **ESSaRel** test.

Question:	Which tool was more useful (faster and accurate) in detecting the critical components from the safety scenario?
Answer:	• ESSaRel 1 vote • ESSAVis 21 votes • Doesn't matter 3 votes
Question:	Which tool was providing more insight about the overall structure of the failure?
Answer:	• ESSaRel 2 votes • ESSAVis 21 votes • Doesn't matter 2 votes
Question:	Which tool was providing more interaction facilities?
Answer:	• ESSaRel 2 votes • ESSAVis 18 votes • Doesn't matter 5 votes
Question:	Which tool would you like to use in work in future?
Answer:	• ESSaRel 4 votes • ESSAVis 16 votes • Doesn't matter 5 votes
Question:	In general, any suggestions for improvement in design
Answer:	

Table 8.6: Participants' feedback in the open-ended questionnaires forms after they finished the test on both tools.

9

ESSAVis Usage as A Visual Training Tool in Education

In this chapter, we present a two-perspective visualization approach to show how the visualization power can be utilized in explaining some computer science concepts during the lecture. Mainly, explaining an abstract concept in theoretical computer science requires a lot of imagination of the entities relations in the underlying algorithm or in the underlying data flow. Lecturers need the help of some of visualization techniques in conveying these complex concepts. Our approach helps lecturers in conveying abstract concepts in computer science through producing visual representations from either ***static perspective*** *or* ***dynamic perspective****, depending on the nature of the concept. We suggest an algorithm for the designers to help them in deciding which perspective might be used to visualize the underlying concept. Moreover, we performed a brief evaluation study with some of the embedded-systems engineering students, where we examined our approach in explaining the safety analysis process of embedded systems. We also used our ESSAVis framework as a visual environment for helping the students in understanding the underlying concept in the study. The results of the study show the advantages of using the visualization during the lecture, as most of the participated students in the study commented that our approach and the ESSAVis visual environment helped them in realizing the concept faster. Moreover, it was easy for them in understanding the unseen details of the underlying concept.*

Briefly, trying to explain a theoretical computer science concept using only a textual description would not be sufficient most of the time for students to understand it. For example, explaining what is the data flow in a state diagram without providing the diagram itself creates a big problem in students' comprehension level, because in this case the lecturer relies only on the students' imagination to realize the relations between the diagram components [46]. While providing a simple visualized form of the diagram could solve a lot of illusion from the students' minds. Therefore, visualization for such abstract concepts can be helpful in avoiding many problems as it supports a common language between the students and their teachers, because a standard visual representation for a concept conveys the same meaning to all.

First, let us express the visualization process itself. The visualization is a graphical representation of information, providing a set of visual means for interpreting the required information, to increase the comprehension level of the underlying concept [46]. To achieve this purpose effectively, the visual cues in the representation should be built upon a knowledge acquired by the visualization users. This is necessary because it speeds up the understanding process of the underlying concept [36].

In Figure 9.1, we show a simple graphical representation of one programming language statement into its meaning. In this example, the visualization helps students in realizing the meaning of the statement, by showing its influence of the statement in the computer memory. Through this, visualization can be used effectively to accelerate the process of explaining complex computer science concepts. In this regard, we propose an approach for explaining theoretical computer science concepts visually based on two visualization perspectives. The first one focuses on using the *static visualization* for the underlying concept, while the second one focuses on using the *dynamic visualization* for the underlying concept.

Describing a computer science concept through visualizing from both perspectives (i.e., the *static* perspective and the *dynamic* perspective) helps the lecturers in the class in exploring the concept more effectively. Moreover, this approach helps in showing the unseen relations between the concept components [46], which affects the students' speed in comprehending complex concepts or algorithms.

We conducted a brief evaluation study to know the effect of our visualization approach on the participated students' performance in learning the given computer science concept. The experiment test consisted of

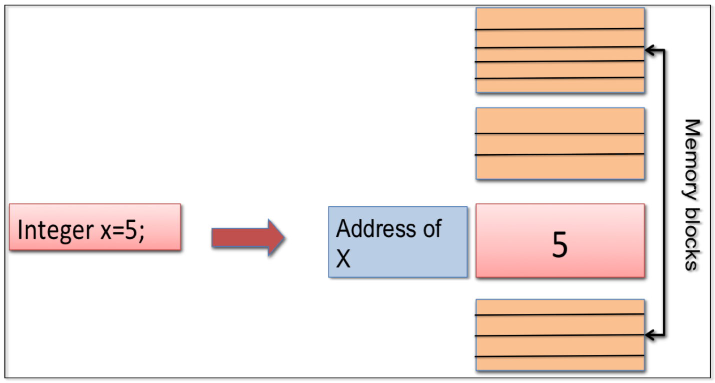


Fig. 9.1: A static visualization represents the variable storage representation in the computer memory during the program execution.

two main phases. The first phase targeted at the static visualization perspective in which we took the fault tree model as a case study. The second phase targeted the evaluation of the dynamic visualization perspective. During this phase, we used our ESSAVis framework, described in earlier chapters of this thesis. The study took place when the earlier version of ESSAVis was ready. In this earlier version, ESSAVis had a simple representation of the fault tree integrated with the system 3D model (see Figure 9.10 in later part of the chapter). Also, at that time ESSAVis had only limited number of interaction facilities. However, the results of this study show a high acceptance level of the visual support provided by ESSAVis (even with this early stage of platform) compared to the static visualization of the fault tree. The participated students gave the credits to the ESSAVis due to the intuitiveness of the visualization and the smoothness of the provided interaction techniques.

9.1 Related Work

Many techniques have been proposed in the literature to visualize computer science applications. For examples, visualization has been used to teach data structures, sorting algorithms, and the graph theory. Moreover, many tools have also been designed to visualize formal languages and the automata theory. Few examples of such tools are Zues in [42], AACE [75] and tango [139].

Bilska et al. in [36] presented a tool, called JFLAP, for creating and simulating the finite automata, the push-down automata, and the Turing machines. They have shown that using such tools in the lecture to demonstrate some theoretical concepts could help both, the lecturer and the students, to understand the complex concepts. Moreover, they also proved that using this kind of tools could solve students' problem of getting a direct feedback about the concept during the lecture; because students can interact with them by changing the parameters values to obtain the result directly. Moreover, students can also use them in labs or in assignments at home for reproducing the examples from the lectures. Additionally, these tools can help those students who want more practice to create their own examples and like to investigate more about the concept.

However, the classification of visualization schemes that was used in education was firstly presented by Segenchuk [140]. The set of recommendations in that was related to using the visualization in education in general. The basic idea was to increase the comprehension level of learners. However, the methodology beneath the idea or the possible applications, which might get benefits from the proposed schemes, was not explained comprehensively.

In [46], Chuda described the possibility of using visualization in the theoretical computer science education. The author developed some animations to visualize the model of the finite automata, push down automata, and Turing machines. The results indicate that students comprehend the theory of computing faster using these visualized models. Also, these models were used to assess the students' knowledge about the course contents. In [119], a framework was presented for the experimental studies of visualization effectiveness in computer science education. The framework suggested a set of metrics for assessing the learning outcomes according to the well-recognized "Grounded in Bloom" taxonomy of understanding. While in [87], the authors hypothesized that the students who use their own visualizations are better in learning algorithms than the students who use computer-based visualizations. They conducted an experiment to test their hypothesis and found that there is no significant difference between the two approaches.

Moreover, new technologies have also been utilized in visualizing concepts in education in general. For example, in [88], the author reviewed the possibilities of using Virtual Reality (VR) in education and discussed how students can get benefit from it. They provided a user study showing the feasibility and the applicability of VR in education by indicating an increasing in the students' comprehension level. In [54], a virtual reality tool was used to model physical behaviors of particles according to Newtonian mechanics, electrostatics, molecular structure, and dynamics. Three virtual worlds were designed and implemented to investigate the effect of immersive and computer-generated experiences on learning the topic in the science. Their results indicate that virtual reality can improve the students' understandability about the mastery of abstract concepts.

From the above brief discussion about the existing tools for visualizing computer science concepts, we found out that most of the current tools are tailored for one specific concept. Moreover, most of the existing tools are designed according to a set of requirements related with the concept's domain. However, there is no framework to describe a general recipe in designing such tools. Compared to them, we present a new approach to decide the visualization scheme that would be interesting and useful for both, the lecturer and the students. In our approach, the selection of the visualization type is based on the nature of the underling computer science concept. In the forthcoming two sections, we explain in detail the two perspectives of our approach.

9.2 The Static Visualization Perspective

The static visualization perspective is related to producing a visual form that does not change over time. Hence, there is no need to provide any interaction technique to interact with the underlying visual form. Although this perspective has some limitations, but it is helpful during the lecture to model those computer science concepts that consist of a set of predefined steps and that do not change over time [140].

This perspective is useful in showing the sequence of events that happen together. For example, it can be used to depict the algorithm steps and the order of these steps. Typically, this can be achieved by showing the flow chart of the algorithm. This flow chart is produced using a set of simple shapes like rectangles, diamonds and arrows (e.g., see Figure 9.2). These simple shapes convey meanings to the students so they can understand the algorithm faster. However, the lecturer needs to invest some time in explaining the meaning of each shape and how these shapes connect with each other to convey the precise meaning of the algorithm.

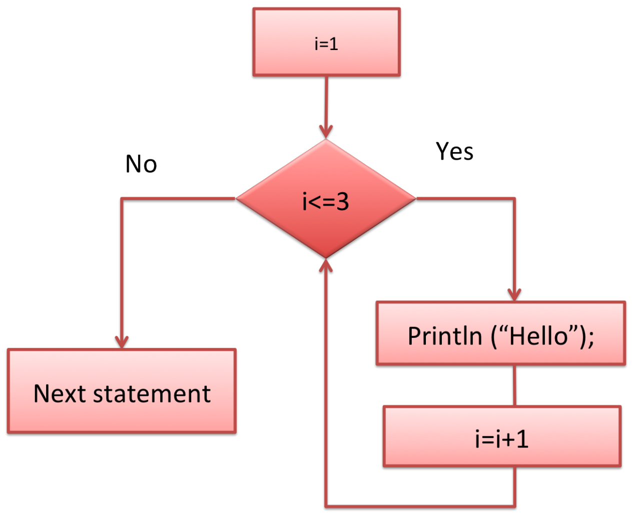


Fig. 9.2: A simple example from the *static visualization* perspective in which a simple flow chart represents the *for-loop*.

We show through an example the usage of the static visualization perspective for modeling the concepts that consist of a set of predefined steps and do not change overtime. The example in this case is the representation of the *three-handshake* process [121] that is used by the Transportation Control Protocol (TCP) in networks. Trying to explain the main steps for establishing a connection using the TCP, the lecturer needs to show the different stages of establishing such a connection.

Describing this process textually can be achieved successfully but then students need more time and effort to understand the several steps of the TCP connection establishment process. The static visualization perspective can be helpful in this regard by providing a simple static visual representation of the concept, as shown in Figure 9.3. Although, students cannot interact directly with this static visual representation; however, they can detect easily the main steps of the connection establishment in the TCP. Moreover, they can relate this simple visual representation with the process name that it represents. This helps them to retrieve the required information faster than using only the textual description of the specified concept. Beside this, producing such visualization does not need much time and effort. Lecturers can produce it directly during the lecture using the writing board.

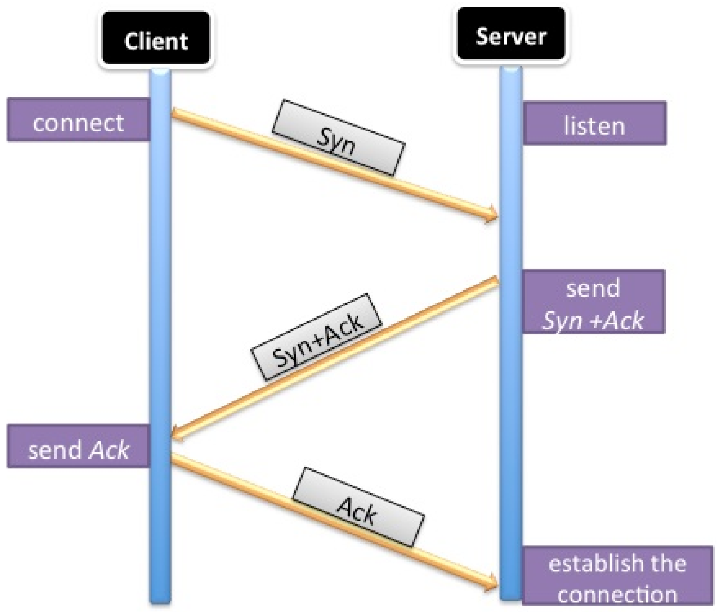


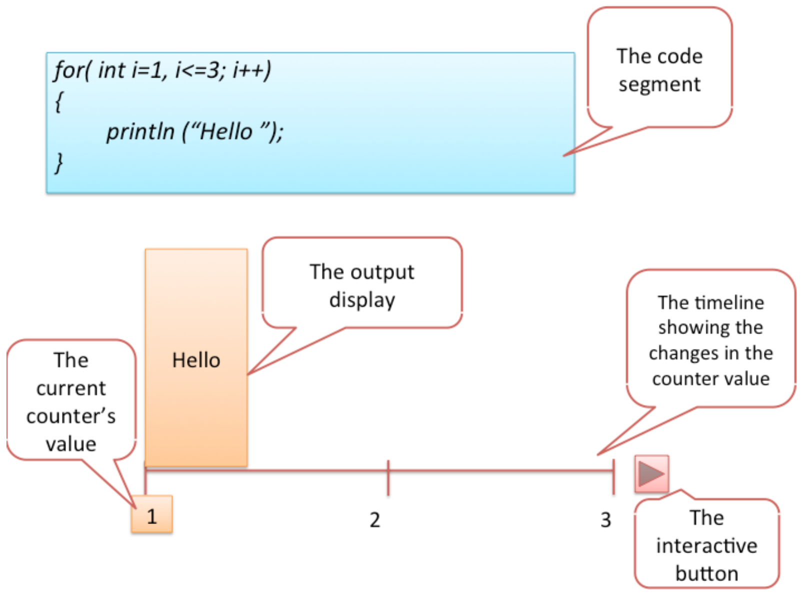
Fig. 9.3: A static visualization representing the *three-handshake* process in TCP connection establishment process.

9.3 The Dynamic Visualization Perspective

The dynamic visualization perspective is related to producing a visual representation that can be changed over time via interacting with it. This kind of visual representations can be utilized in simulating the behavior of the underlying concept, while introducing the concept during the lecture. The mentioned simulation requires both, calculating the parameters of the concept and mapping the calculated parameters to visual cues understood from the concept domain [140]. The resulted visual representation can be used to convey some facts about the underlying concept and the relations between the different parameters that influence each other in the specified topic.

Let us start with a simple example where this kind of visualization can be useful. For example, we assume that the lecturer needs to convey to beginner students the working of the *for-loop* in a program segment. Figure 9.4 illustrates the execution of this program segment through the help of dynamic visualization perspective. The *for-loop* in this program segment consists of three counters, from 1 to 3. Each time the counter changes its value; the program prints the statement "Hello" and leaves an empty line after it. In this case, the visualization interface consists of the following elements:

- *The code view:* This shows the underlying code written in a specified language.
- *The counter axis:* The counter values of the *if-loop*, which in this case ranges from 1 to 3.
- *The output display:* This shows the possible output of the program segment, which in this case is printing of the "Hello" word each time.
- *The connector buttons:* This is a button through which students can interact with the visualization. Each time a student interacts with it, the visualization reacts accordingly.



```
for( int i=1, i<=3; i++)
{
        println ("Hello ");
}
```

The code segment
The output display
Hello
The timeline showing the changes in the counter value
The current counter's value
1
2
3
The interactive button

Fig. 9.4: At initial stage, when the counter is 1, the output is "Hello".

In this visual representation, orange color is used to show the value of the current counter in order to differentiate it in the later part of the configuration from the previous counters. Using such a simple visualization helps not only the students to understand the code faster, but also helps them in keeping the evolution of the code over several steps. This is because they can go forward or backward and then can see the result accordingly.

This kind of visualizations is useful for explaining the output of loops in general, e.g., the *if-statements* or the *switch-statements*. However, they fail in showing the internal representation of variables or the statement interpretation in the code.

For showing its implication in a more complicated case, we take the case of explaining the structure of a local network consisted of 100 nodes over the time. In addition to the set of factors, which could affect the network's performance, the lecturer requires a lot of imagination power and sometimes more efforts to help the students in understanding the status of the network at a certain time. For such problems, the dynamic visualization perspective can offer a solution by providing a virtual representation of the underlying concept or application, like in this case the virtual network (as shown in Figure 9.5 and Figure 9.6 respectively).

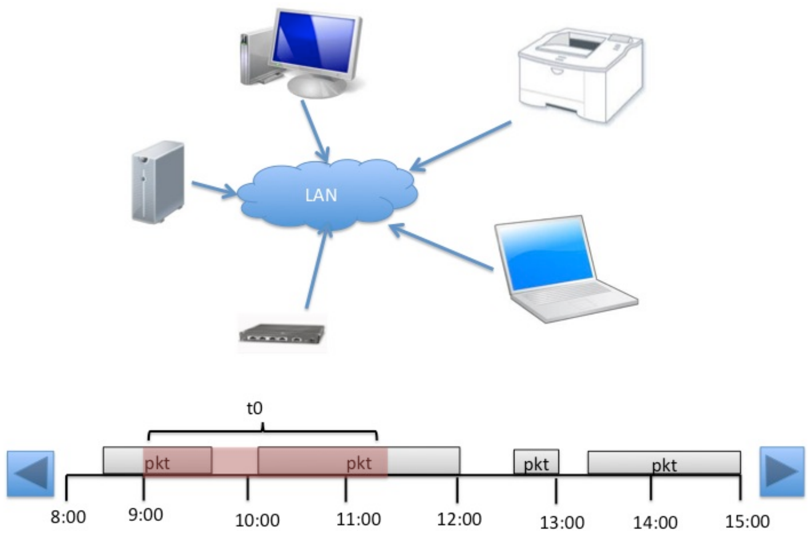


Fig. 9.5: A dynamic visualization of the virtual networks at time *t0*.

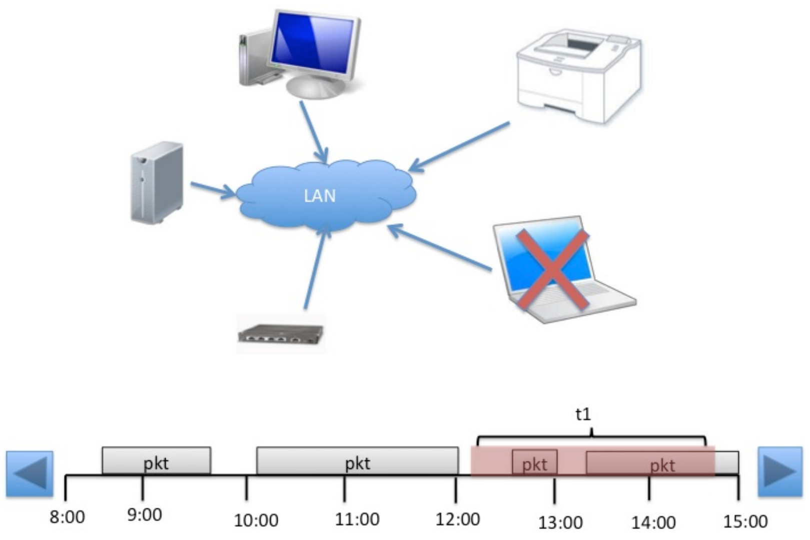


Fig. 9.6: A dynamic visualization of the virtual network status at time *t1*.

Our proposed solution helps the students in understanding the relations between the different parameters that can affect the network's performance. Moreover, students can modify the parameters' values during the lecture and then can investigate the resulting changes happen in network status. In this case, the lecturer gets benefit from this representation by saving the time and the effort in explaining the different set of the parameters, the nodes types, and the topological structure of the network. All these features are displayed by the visualization, so students perceive them visually.

9.4 The Visualization Pipeline

In this section, we present a pipeline for showing the steps that are required to generate the visualization from either of the two above mentioned perspectives (i.e., the *static* and the *dynamic*). The goal is to help the lecturer and the students in deciding the type of visualization that can be modeled for the underlying computer science concept. We model a pipeline, as shown in Figure 9.7, using the flowchart schema, which is already used in the computer science education to show the algorithm steps.

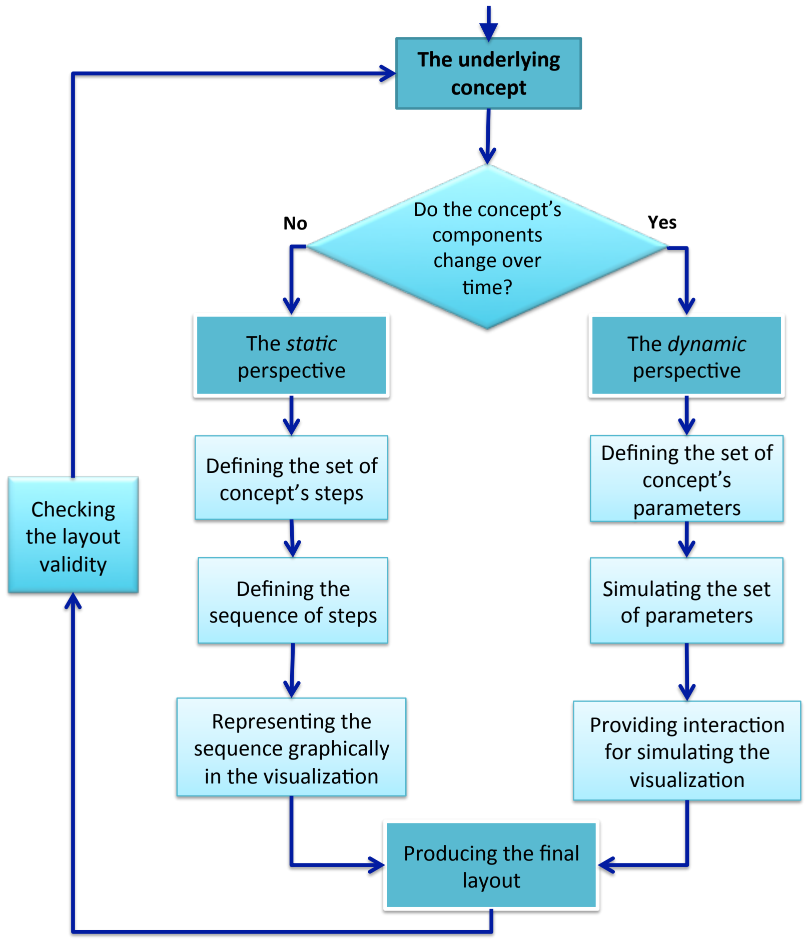

The underlying concept
Do the concept's components change over time?
No
Yes
The *static* perspective
The *dynamic* perspective
Defining the set of concept's steps
Defining the set of concept's parameters
Defining the sequence of steps
Simulating the set of parameters
Representing the sequence graphically in the visualization
Providing interaction for simulating the visualization
Producing the final layout
Checking the layout validity

Fig. 9.7: The pipeline to generate the visualization of a particular computer science concept.

The presented pipeline shows the main steps for deciding the required visualization perspective. The first step is the most important one as based on it further steps are decided. It requires the knowledge about the underlying concept's nature. Moreover, it also depends on the lecturer's aims of conveying the message to students. Therefore, if the lecturer is interested in showing only the main stages of the concept and the relations between these stages, then the static visualization will be used. Otherwise, if the main interest of the lecturer is in demonstrating the behavior of the concept over the time or in showing the set factors within the concept that affect each other, then the dynamic visualization will be used.

In the case of static visualization, first the lecturer defines the main steps of the underlying concept and the set of relations between these steps. The next stage is to define the order of these steps. This is important because this stage gives insightful meaning about the underlying concept. Then in the next stage, a mapping of each step with a geometrical shape is done in order to represent it graphically. This mapping is very critical as both, the lecturer and the students, should be able to understand it

On the other side, if the lecturer is interested in showing the behavior of the underlying concept over a period of time then the dynamic visualization is performed. This requires defining the set of parameters that are needed in the explanation. Once the lecturer finalizes the parameters and the relations between these parameters, then it is the time to start the simulation process. It requires not only computing the values of these parameters but also representing them graphically.

In both cases, the final stage is to produce the final layout according to the previous stages. Then the lecturer evaluates it together with students. In the presented pipeline this step is called the *validation* step, because in this step a brief evaluation is performed to check the applicability of the proposed visualization. The next step is taken based on the evaluation results. If the results are satisfactory then the underlying visualization is ready to be used during the lecture; otherwise, the whole process will be performed again in order to refine the previous solution.

9.5 The Challenges

Our presented approach suggests using the static visualization or the dynamic visualization according to the underlying concept. However, in practice applying this approach is not an easy task. For example, designing a visual representation from the static perspective for an algorithm requires a lot of efforts, because the resulting visualized form should guarantee a full representation of the concept. This requires an agreement between the lecturer and the students about the set of symbols used in the final representation and a mapping between each symbol and its meaning.

Furthe, as the static perspective is not much flexible; students may have the problem of reproducing the visualization at home or in the lab. Moreover, as they do not get a direct feedback from the visual representation so they may also feel unexcited to trace the relations between the representation components. However, this issue can be eliminated if the lecturer asks the students to choose the style and colors of the layout components and provides a key map to define the meaning of their selections.

On the other hand, the dynamic visualization perspective solves many problems related with the direct feedback and the interaction facilities. This is because, as mentioned earlier, it shows the relations between different parameters visually. However, the main problem of this kind of visualization is the difficulty level of producing such a visual representation. In this case, the designing process is an important step because it defines those main concept parameters that are required to be displayed. This decision is very critical as it eliminates the functionalities of the final visualization toolkit. This also eliminates the lecturer from using the toolkit for different cases during explaining the concept; because of the available visualization toolkit abilities, i.e., if the underlying toolkit does not provide some features then lecturers are not able to explain the missing features. Moreover, often using too many visual cues to represent the required parameters may distract students' attention. Therefore, both the lecturer and the students need to evaluate the final toolkit before using it in the real lecture. Hence, the training session is necessary for both before using a particular toolkit. This can take some time, as it needs to be evaluated on all students' types.

In addition to all of the mentioned challenges, both the static and the dynamic visualization perspectives need to provide such a flexible layout that can be reused and adapted for many concepts in computer science. This also requires much effort in designing and implementing the visualization toolkit correctly. Based on this, we suggest using an iterative designing process in which both, the lecturer and the students, take part to come up with a better visualization of the underlying concept.

9.6 The Evaluation Study

To prove our presented approach, we carried out a brief evaluation study where we asked students from the Computer Engineering Department at the University of Kaiserslautern to participate in the experiment test. This evaluation study was performed on studying and understanding the safety analysis process in the embedded systems domain. The goal of the study was to test the visualization power on understanding the analysis process of safety aspects in such complex systems. Normally, the safety analysis process is based on a method, called Fault Trees (FT) [93], which allows to describe the conditions that have to be fulfilled leading to undesired system failures. The hazardous system-level failure, called *Top Event*, is represented as the root of the tree [93]. While the basic influence factors or failure conditions, called *basic events*, are represented as the tree leafs. These elements are logically related via gates (e.g., AND-gate or OR-gate). Normally, safety experts, who spent quit long time in learning this model, are responsible to produce the FT models for a system. Therefore, trying to explain this concept to the beginner students can be a difficult task, because students may relay on their imagination about the final shape of the resulted FT model. Moreover, these models also contain many other information that also affect the safety of the system, such as the set of the basic events related with each component in the system or the failure path from a specific component to the top-event level.

The main task for the participated students was to extract the failure relations between the safety scenario components. Moreover, they had to extract the structural relations between the scenario elements if there exist any. We hypothesized that involving students in producing the FT model through visualization would increase the comprehension level of such concepts. Therefore, we asked the students from the computer engineering master degree program at the University of Kaiserslautern to participate in our experiment test, which was performed in two stages. The goal of the first stage was to examine the benefits of the static visualization perspective, while the goal of the second stage was to examine the power of the dynamic visualization. In the forthcoming subsections, we present the test in both stages and then we show the results of these two stages.

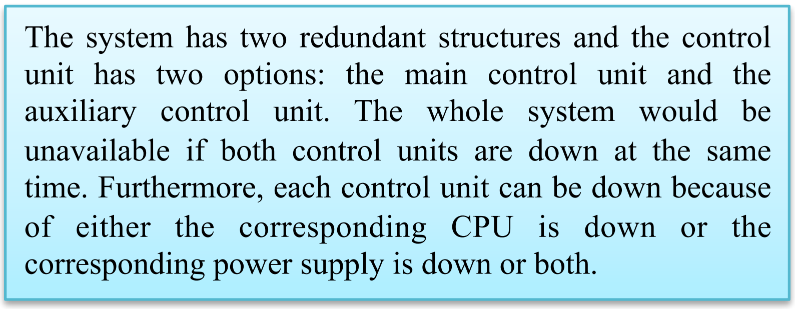
9.6.1 The Test Settings

We asked five computer-engineering students to undergo our test. All of them had a good background in embedded systems. However, they did not have any previous experience in the safety analysis process of such systems. Also, they did not have any background about the fault tree model, the component fault tree concept, and the difference between these two. We intended this in order to eliminate the students experience with such concepts, as then we could judge their understandability of the FT concept after trying our approach.

At the beginning of the test, we gave a full description of the FT concept, which showed them that how this model can be used to analyze the safety aspects of complex embedded systems. We asked them to give their feedback through questions and comments. This session lasted for one hour, which is almost equal to the time of a normal lecture. After this introductory session, we gave them a textual-based safety scenario, shown in Figure 9.8, in order to extract some important safety information out of this scenario. The participated students were able to discuss the scenario with each other and successfully classified the scenario elements into different groups according to their types, e.g., the list of the components, the list of the possible failures in each component, etc. However, it was difficult for them to extract the relations between the collaborated components in the fault tree using only the textual description of the given safety scenario. We then suggested them to use the visualization approach to get more insight about the underlying safety scenario. We started with the static visualization scheme and then we proceeded with the dynamic visualization scheme. The forthcoming two subsections provide these details respectively.

9.6.2 The Static Visualization Perspective Test

To help the participated students in visualizing the safety scenario, we gave them the graphical representation for each element in the scenario. We designed the symbols meaning in this phase according to the original work of [93]. This was done to help the students in keeping the relations between the theoretical meaning of the concept and its visual representation. Table 9.1 lists the symbols and their meanings that were used during the test for visualizing the underlying safety scenario.



The system has two redundant structures and the control unit has two options: the main control unit and the auxiliary control unit. The whole system would be unavailable if both control units are down at the same time. Furthermore, each control unit can be down because of either the corresponding CPU is down or the corresponding power supply is down or both.

Fig. 9.8: The given safety scenario to be visualized, taken from [93].

Table 9.1: The graphical representations for safety scenario elements

Symbol	Description
$\boxed{>=}$	The OR gate
$\boxed{\&}$	The AND gate
●	The Basic Event
➡	The Failure relation

After we explained the meaning of each symbol, we asked the participated students to find out the logical relations between the infected components of the underlying safety scenario. We also asked them to detect the main reasons of each failure in the underlying safety scenario by producing a simple visual representation of this safety scenario by using the symbols set, described in Table 9.1.

At this stage, we observed that the participated students started thinking loudly and talking more about the details of each component in the safety scenario. They also asked some detailed questions to understand more about how to model the required safety scenario. Beside this, they were also able to draw some other conclusions about the system situation under this safety scenario. This happened because they were producing the model by themselves. All of them finished the task successfully in a short time, almost 5 minutes in average. The result of this session was a static representation of the safety scenario, shown in Figure 9.9.

At the end of this stage of the experiment, we conducted an open session to discuss the task of this stage. All of them were comfortable in using the static visualization approach for producing a model of the concept. They also confirmed their abilities in understanding the concept more clearly and the increment in realizing some hidden details that did not appear well in the textual description of the concept (e.g., the relations between the different components). They also appreciated the overall approach and commented that this kind of approach is useful for students to interact with the teacher during the lecture, because normally students need some answers during the modeling phase of the concept.

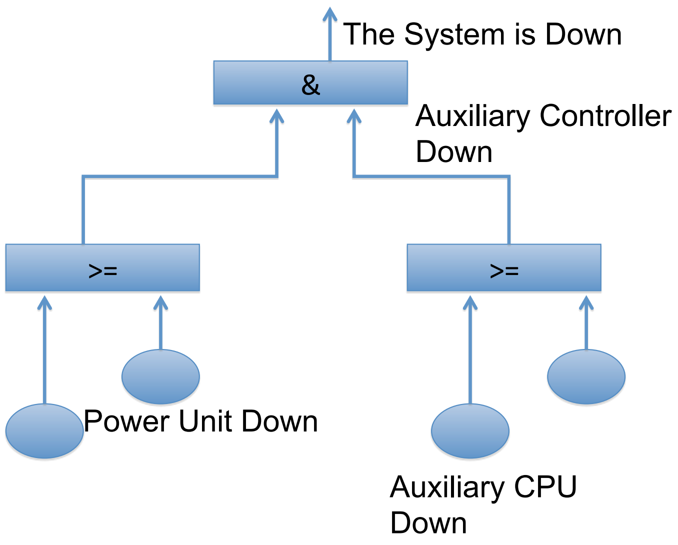


Fig. 9.9: The graphical representation of the safety scenario, textually described in Figure 9.8 [93].

9.6.3 The Dynamic Visualization Perspective Test

During the second stage of the experiment, we asked the participated students to use our ESSAVis framework, described in details in the earlier chapters of this thesis, that is dedicated to visualize the safety scenarios in embedded systems using the two integrated views, the *abstract representation* view and the *system 3D model* view, as shown in Figure 9.10.

ESSAVis provides a 2D*plus*3D environment to visualize the safety aspects of embedded systems [12]. The 2D refers to the first view of

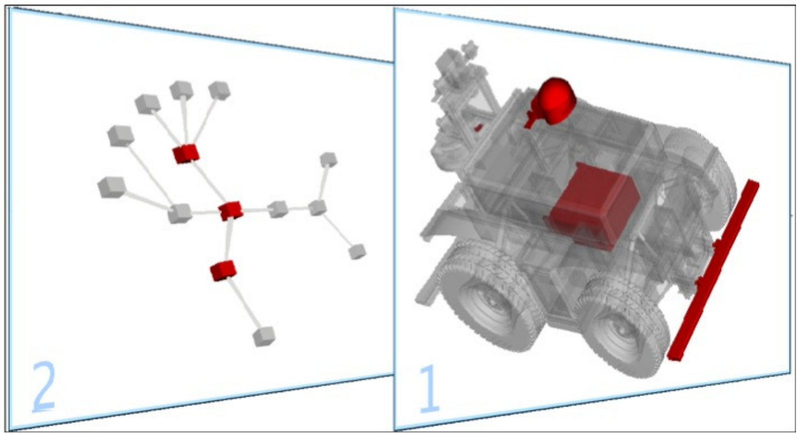


Fig. 9.10: The ESSAVis framework visualizes the safety aspects of the underlying embedded system using the FT model. This figure shows when ESSAVis was in its early stages, as it was used in the study during this stage.

the ESSAVis environment, i.e., an abstract representation of the failure mechanism at a certain time. This is used to show the failure relations between the systems components at that time. Moreover, it can be used to show the type (hardware or software) of each component in the scenario. In this view, viewers can also expand any component to explore the internal structure of the component. The 3D view in the ESSAVis environment refers to the second view, i.e., the 3D model of the underlying embedded system. In our case, we used the RAVON 3D model [124], which is shown at the right side of Figure 9.10.

For the task at this stage, we asked the participated students to use ESSAVis environment for exploring the FT concept using the two different views provided by the framework. The users could interact with each view in the environment individually using the mouse or the keyboard directly.

The 3D model of the underlying embedded system benefited the participated students in mapping the abstract meaning of the concept into a real application. In this case, they could map the infected components with their real physical locations. Students confirmed that this view helped them in filling the gap between the meaning of the concept and the influence of the concept on real applications. They appreciated very highly this view as it helped them in getting more accurate impression about the meaning of the embedded system in general and the failure relations between the components in such systems particular.

9.6.4 Discussions

After we finished the both stages of our experiment test, we had an open session with the participated students to discuss the both stages. The students expressed their experience of using our approach in explaining the given concept. They agreed that using the static visualization perspective for producing the model of the target concept by themselves helped them in understanding the concept faster, because it motivated them to go into more depth about the concept details. For example, modeling the concept in the experiment test helped them in finding the drawback of using the tree structure to model the failure mechanism as it expands in one direction, which makes it not scalable for large models. This helped us in introducing a more advanced concept in failure modeling, the Component Fault Tree (CFT) concept, which is descendant to the FT concept and in which the failure is represented as a DAG (Directed Acyclic Graph) rather than trees [93].

The dynamic visualization perspective was very powerful in filling the gap between the theoretical meaning of the concepts and its real applications. The participated students were able to find extra details about the underlying concept. They were able to interact with the provided toolkit and queried about the system status through changing the related parameters' values. They realized the meaning of embedded systems using our approach and were successfully able to find out the relations between the hardware and the software components in the underlying system. Furthermore, they were also able to realize the difference between the failure relations and the structural relations. Therefore, they found out that the system components have not only the structural dependencies but also the failure dependencies where the failure dependencies do not require the components to be structurally related. It was difficult to convey this critical concept to participated students without visualizing it through the provided ESSAVis environment. Moreover, as we used an animation to show the failure propagation path between the infected components in the scenario, it was easy and quicker for the students to trace the failure path.

Despite of all the advantages of using the visualization to explore some complex computer science concepts, it is clear that it cannot be used alone in the lecture. Lecturers need to explain the concept meaning as well as to train the students for using the visualization tools in order to explore the concept in more depth. We also found out that few students have some problem in understanding the ESSAVis environment itself. They commented that understanding the ESSAVis environment added extra burden on them, as it required extra time and efforts.

9.7 Summary

In this chapter, we presented a visualization approach with two visual perspectives, the *static* and the *dynamic*, as a powerful technique for conveying some computer science concepts during the lecture. The proposed approach helps in understanding the abstract concepts by producing the visual representations, either from the static perspective or from the dynamic perspective, for exploring the complex relations among the concepts' components.

The results of the conducted evaluation study proved the feasibility of our approach, as the feedbacks from the participated students were positive and promising. However, we cannot generalize the findings because the number of participants in this evaluation study was limited, beside that no teacher was involved in it. The study conductor acted as a teacher during the study time, which also affects on the validity of the results. The results also show few barriers of using such approach in education, because the designing process of visualization needs extra effort and time from both, the lecturer and the students. Therefore, it is necessary to design the target visualization carefully to help students mapping the visual cues with some parameters in the underlying concept correctly.

The decision of making the visualization perspective *static* or *dynamic* heavily depends on the concept domain itself, and the set of goals that the lecturer wants to achieve from the target visual representation. Moreover, producing the final visualization requires plenty of practice, because it must be designed according to a direct feedback from the lecturer and the students. Also, the training step during the lecture is necessary for both, the lecturer and the students, before using the visualization. This helps them in evaluating the final layout more accurately.

Overall, the goal of this chapter was to show another aspect of ESSAVis usability. We aimed to investigate its ability to be used as an educational visual platform teaching and training about embedded systems.

10

Conclusions and Future Directions

10.1 Conclusions

In this thesis, we utilized the visualization power as a mean to enhance the understanding process of complex data sets of embedded systems. In this regard, we tried to visualize the hidden behaviors of these data sets from different perspectives. The main focus of this work was to enhance the comprehension level of our end users, which in most cases were the *system engineers* and the *safety experts*, in understanding the inside behavior of these data sets effectively and efficiently.

Generally, we use the Information Visualization (InfoVis) field for several purposes, for example:

- *To represent the structural and behavioral aspects of complex data sets* [98, 82],
- *To synchronize between multiple perspectives of the underlying data sets* [131],
- *To provide different interaction mechanisms for the data extraction process* [131],
- *To provide novel interaction techniques in order to get insight about the multi-level representations of the data* [131, 82], and
- *To produce aesthetics representations of the data* [125].

This thesis aimed at providing contributions in these mentioned directions of InfoVis field. Therefore, the thesis focused on visualizing the *structural* and the *behavioral* aspects of embedded systems. More specifically, we tried to visualize the failure mechanisms of these systems from different perspectives at different abstraction levels, in order to help the end users (i.e., the *system engineers* and the *safety experts*) in understanding accurately and efficiently the reasons behind these failures to avoid those undesired situations that might threaten the system life.

We intensively worked with different graph visualization techniques for visualizing these failure mechanisms. We provided different interaction techniques to help and facilitate the information extraction process. Further, we visualized the system hardware model using 3D visualization techniques. Also, we provided several interaction techniques to extract the required information from the system 3D model on demand.

All contributions were combined and implemented in a visual platform, called the **ESSAVis** (**E**mbedded **S**ystems **S**afety **A**spects **Vis**ualizer) framework. ESSAVis was designed to combine the two required views (i.e., the abstract graph representation view of the safety scenario and the system 3D model view) in one platform. One of its novel contributions was the way it integrates seamlessly and smoothly between these two views in a 2D*plus*3D fashion. ESSAVis provides different interaction options and GUIs in order to allow users intuitively interacting with any of the two views to get the required information on demand. Furthermore, the framework automates the safety analysis process of CFT models, which resulted in speeding up the overall maintenance process of the underlying embedded system.

In order to check the feasibility and the usability of the overall framework and the different contributions towards the graph visualizations and interactions mechanism, we conducted many evaluation studies in which the real end users as well as people from other backgrounds participated in order to evaluate these contributions from the both perspectives, domain specific and the general one. In an intensive evaluation study, we compared ESSAVis with ESSaRel tool in terms of *effectiveness*, *efficiency*, and *user-satisfaction level* regarding analyses of the safety aspects of failures occurring in an embedded system. Results of the study show increases in the participants' efficiency and accuracy levels using ESSAVis compared to ESSaReL as well as a high acceptance rate among the participants towards ESSAVis due to its intuitive visual support.

Overall, the thesis provided the answers to the following research questions at different levels:

- *How to model the failure mechanisms of embedded systems into a graph representation that could reflect the structural and the failure relations between the system components? How to visualize and interact with the resulting 2D graph model?*

  - We used different graph visualization techniques to help our end users (i.e., the system engineers and the safety experts) in understanding the architectural and behavioral aspects of embedded systems. For example, we mapped the failure mechanisms in embedded systems into a compound graph representation and designed the solution based on our end users requirements.
  - Once the solution was implemented, we presented a new interaction technique, called **CluE** (**Clu**ster **E**xpander). CluE provided an intuitive interaction facility to show the internal structure of the clustered nodes. One of its main features is a generic algorithm for expanding or contracting the compound nodes in terms of the used layout algorithm.

- *Does the 3D visualization of the system hardware parts help in understanding the structural relations and the behavioral aspects of the underlying embedded system? What are the suitable interaction techniques that can help the users in getting insight about the system quickly and accurately?*

  - We provided the answer of this question through our developed **3DintEx** (**3D int**eractive **Ex**plorer) unit inside the ESSAVis framework. This tool allows interactive exploration of the structural relations of a system model in 3D environments. The tool accepts a CAD model of the system and other data sets that describe the system safety-behavioral aspects. 3DintEx was built using the Vrui framework [102], which makes it compatible and scalable on a range of VR environments (e.g., desktop 3D displays, PowerWall displays, or CAVE systems).

- *How to combine in one scene the 3D view of the system and the 2D representation of the failure mechanisms of the same system?*
  - In order to answer this question, we presented our developed **SceneMan** (**Scene Man**ager) unit inside the ESSAVis framework, which seamlessly integrates between the different perspectives of the system in a 2D*plus*3D environment.
- *How to measure the usability of the resulting ESSAVis framework from the perspectives of real end users as well as from general users?*
  - To answer this question, we designed and conducted a controlled evaluation study to compare between ESSAVis and ESSaRel [137] in terms of *accuracy*, *efficiency* and *user acceptance level* in analyzing safety aspects of failures occurred in embedded system represented as CFT models. Overall, the results of this study show increases in the participants' efficiency and accuracy levels using ESSAVis compared to ESSaReL, mainly due to the visual support and the interaction facilities provided. We also observed a high acceptance rate among the participants towards ESSAVis due to its intuitive visual support. This opens the doors of thinking of new possible directions how to enrich the safety analysis process of complex systems using interactive visual environments.
- *How to integrate between the 2D representations of the node-link diagrams into the 3D world?*
  - In this regard, we rendered both views (i.e., the 2D graph representation view and the system 3D model view) in a 3D world. This helps the viewers in understanding the system intuitively, however, we designed the system to cast the 2D representations into 3D representations in some situations. For these situations, we used the stereoscopic depth to highlight some aspects about the nodes in the 2D node-link diagrams. This technique is called the *stereoscopic highlighting* technique in which the graph is rendered in a 3D stereoscopic platform and the stereoscopic depth is used to encode the important nodes, such that these

nodes come closer to the viewer than the remaining nodes in the graph. We explored the possibility of using the stereoscopic depth is this domain in several directions. Initially, we evaluated the depth cue in stereoscopic platforms to the traditional visual cues (i.e., color, shape, and size) using different settings. Based on the findings, we optimized it using the *reflection layer* extension. Finally, we applied it on encoding structural relations between the compound nodes in clustered graphs using the our *ExpanD* (*Expan*ding in *D*epth) algorithm. In all cases, we conducted evaluation studies with participants from different backgrounds in order to measure the feasibility and the usability of using the depth cue to highlight aspects of node-link diagrams.

Additionally, we proposed to use the visualization power as a tool for teaching complex concepts in the computer science field. In this regard, we presented our two visual perspectives (i.e., the *static* and the *dynamic*) approach that helps in creating the required visual representation of the underlying abstract concepts to be taught in the lecture. We performed an evaluation study with students having background in computer engineering where they tried to understand the fault detection of the failure scenario concept using the CFT representations produced by our ESSAVis framework. We compared this dynamic visual representation of the concept using ESSAVis with the static representation of concept using the Fault Tree approach. Results of the comparison show high acceptance rate from students to use the dynamic visualization of the concept to understand the failure scenarios. This opens the direction to investigate in this field to answers many questions regarding the design of the visualizations for education purposes. Chapter 9 provided details about our contributions in this direction.

10.2 Future Directions

We used different visualization techniques like 2D visualizations, 3D visualizations, and a combination of both representations in a 2D*plus*3D environment. We also presented the evaluation studies to judge the feasibility and effectiveness of these techniques. However, there is room for optimizing the resulted design and techniques in several possible ways. Here, we summarize the possible future directions in each of our contribution:

Towards the optimization of the 2D graph representation:

- Adding the gate information in the CFT model during the visualization phase of the graph structure.
- Optimization of the layout produced by the LayMan framework, in order to preserve the mental map of compound graphs after expanding/contracting the compound nodes. Currently, we are trying to optimize this issue by controlling the changes in the related nodes postilions as local as possible after expanding or contracting the compound nodes.
- Evaluation the LayMan framework from the time complexity perspective.
- Evaluation of the framework over different graph sizes in different application domains. The presented *LayMan* algorithm in chapter 5 needs to be tested on large compound graphs with different number of levels inside their hierarchies.
- Optimization of the edge bundling handler used by the LayMan framework, in order to visualize the adjacency relations or the cross hierarchy edges between the different levels of compound graphs.

Towards the optimization of the system 3D model representation:

- Optimization of the expansion interaction tool for the system 3D model. In this regard, a possible direction is to optimize the technique in a way that viewers would be able to selectively pick one of the geometry of the hardware node, which results in expansion of this node in order to explore the structure of this node individually.
- Optimization of the transparency effect in order to selectively applied it based on the viewer's request.
- Testing the 3DintEx tool over different 3D technologies like the 3D desktop displays, Power-Wall displays and the CAVE systems in order to compare between the different 3D environments and settings.
- Evaluation of the 2D*plus*3D environment from the viewers' perspectives, using different 3D technologies.

Towards the Optimization of the Stereoscopic highlighting technique:

- Addition of an eye tracker to the 3D stereoscopic system. This would be useful for getting an indication of how to optimize the system in a way to re-render the scene based on the viewer-eyes' angles. This enhancement could help the viewers to detect the depth variations faster and more accurately.
- Application of this technique in different 3D systems (e.g., 3D desktop displays, Power-Wall displays, and CAVE systems) and then evaluating it according to the used technology to measure which technology offers the more accurate results.
- Detailed empirical studies with different configurations and various settings in order to provide solid evidences regarding the feasibility and effectiveness of this technique in general.

Towards the optimization of ESSAVis as an education tool:

- Evaluation study using the latest version of ESSAVis and comparing the results with the evaluation study done with the initial version of ESSAVis, which has been described in Chapter 9.
- Enhancements in the ESSAVis framework in order to make it more appropriate to be used as a visual educational tool for teaching concepts related to graphs as well concepts related to system structural and behavioral aspects.
- Conducting detailed evaluation studies in real classes with real teachers and larger number of students.

Possible future research is to find some general directions that would help in designing "good" visualization for heterogeneous systems, as it was in our case the embedded systems domain. Further possible direction could be to integrate between the several perspectives of such systems in a way that could enhance the viewers' comprehension level and understandability.

References

1. ViERforES, Virtual and Augmented Reality for Highest Safety and Reliability of Embedded Systems, March 2012. "`http://www.vivera.org/ViERforES/`".
2. James Abello and Frank Van Ham. Matrix zoom: A visual interface to semi-external graphs. pages 183–190, 2004.
3. James Abello, Frank van Ham, and Neeraj Krishnan. Ask-graphview: A large scale graph visualization system. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):669–676, 2006.
4. Maneesh Agrawala, Doantam Phan, Julie Heiser, John Haymaker, Jeff Klingner, Pat Hanrahan, and Barbara Tversky. Designing effective step-by-step assembly instructions. *ACM Trans. Graph.*, 22(3):828–837, July 2003.
5. Basak Alper, Tobias Höllerer, JoAnn Kuchera-Morin, and Angus Forbes. Stereoscopic highlighting: 2d graph visualization on stereo displays. *IEEE Trans. Vis. Comput. Graph.*, 17(12):2325–2333, 2011.
6. Ragaad AlTarawneh, Jens Bauer, and Achim Ebert. A visual interactive environment for enhancing collaboration between engineers for the safety analysis mechanisms in embedded systems. In *Proceedings of the Companion Publication of the 17th ACM Conference on Computer Supported Cooperative Work 38; Social Computing*, CSCW Companion '14, pages 125–128, New York, NY, USA, 2014. ACM.
7. Ragaad AlTarawneh, Jens Bauer, Shah Rukh Humayoun, Achim Ebert, and Peter Liggesmeyer. Enhancing understanding of safety aspects in embedded systems through an

interactive visual tool. In *Proceedings of the Companion Publication of the 19th International Conference on Intelligent User Interfaces*, IUI Companion '14, pages 9–12, New York, NY, USA, 2014. ACM.

8. Ragaad AlTarawneh, Jens Bauer, Shah Rukh Humayoun, Achim Ebert, and Peter Liggesmeyer. Essavis++: an interactive 2d plus 3d visual environment to help engineers in understanding the safety aspects of embedded systems. In *Proceedings of the 2014 ACM SIGCHI symposium on Engineering interactive computing systems*, pages 201–204. ACM, 2014.
9. Ragaad AlTarawneh, Jens Bauer, Shah Rukh Humayoun, Patric Keller, and Achim Ebert. The extended stereoscopic highlighting technique for node-link diagrams: An empirical study. In *Computer Graphics and Imaging: IASTED Conference 2013*, 2013.
10. Ragaad AlTarawneh, Jens Bauer, ShahRukh Humayoun, Patric Keller, and Achim Ebert. The reflection layer extension to the stereoscopic highlight technique for node-link diagrams: An empirical study. In *Advances in Visual Computing*, volume 8034 of *Lecture Notes in Computer Science*, pages 1–12. Springer Berlin Heidelberg, 2013.
11. Ragaad AlTarawneh, Jens Bauer, Patric Keller, and Achim Ebert. Essavis: A 2dplus3d visual platform for speeding up the maintenance process of embedded systems. In *Proceedings of the 27th International BCS Human Computer Interaction Conference*, BCS-HCI '13, pages 43:1–43:6, Swinton, UK, UK, 2013. British Computer Society.
12. Ragaad AlTarawneh, Jens Bauer, Patric Keller, Achim Ebert, and Peter Liggesmeyer. Essavis: A framework to visualize safety aspects in embedded systems. In *Proceedings of SIGRAD*, pages 59–66, 2012.
13. Ragaad AlTarawneh, Achim Ebert, and Eduard Kosel. Stft-stereoscopic filtering technique. In *Proceedings of the IEEE VisWeek 2013*, 2013.
14. Ragaad AlTarawneh, Andreas Griesser, Bauer Jens, Shah Rukh Humayoun, and Achim Ebert. 3dintex: A tool to explore interactively the structural and behavioral aspects of system models in 3d environments. In *In IEEE 9th Symposium on 3D User Interfaces (3DUI 2014), March 29-30, 2014, Minneapolis (MN), USA.*, 2014.
15. Ragaad AlTarawneh and Shah Rukh Humayoun. Evolving mobilen prototypen gegen die am besten geeignete design-und interaktionsschema unter verwendung des genetischen

algorithmus. *Zeitschrift für Mobil Human Computer Interaction (IJMHCI), 6(2), pages 1–9*, 2013.

16. Ragaad Altarawneh and Shah Rukh Humayoun. A two-perspective visualization approach for utilizing visualization power in computer science education. In *Proceedings of the 3rd Computer Science Education Research Conference on Computer Science Education Research*, CSERC '13, pages 8:85–8:91, Open Univ., Heerlen, The Netherlands, The Netherlands, 2013. Open Universiteit, Heerlen.
17. Ragaad AlTarawneh, Shah Rukh Humayoun, and Achim Ebert. A hybrid approach for visualizing spatial and non-spatial data types of embedded systems. *Proceedings of J. Abascal et al. (Eds.): INTERACT 2015, Part IV, LNCS 9299, 2015.*
18. Ragaad AlTarawneh, Shah Rukh Humayoun, and Achim Ebert. Expand: A stereoscopic expanding technique for compound graphs. In *Proceedings of the 20th ACM Symposium on Virtual Reality Software Technology (VRST 2014), Edinburg, United Kingdom, November 11–13*, 2014.
19. Ragaad AlTarawneh, Shah Rukh Humayoun, and Achim Ebert. Utilization of variation in stereoscopic depth for encoding aspects of non-spatial data. In *Proceedings of the IEEE 10th Symposium on 3D User Interfaces (3DUI 2015), March 23-24, 2014, Arles, France.*, 2014.
20. Ragaad AlTarawneh, Shah Rukh Humayoun, Johannes Schultz, Achim Ebert, and Peter Liggesmeyer. Layman: A visual interactive tool to support failure analysis in embedded systems. *Proceedings of the 9th European Conference on Software Architecture (ECSA 2015), Dubrovnik-Cavtat, Croatia, September 2015.*
21. Ragaad AlTarawneh, Schultz Johannes, and Shah Rukh Humayoun. Clue: An algorithm for expanding clustered graphs. In *Proceedings of the IEEE 7th Pacific Visualization Symposium (PacificVis 2014), Yokohama, Japan*, 2014.
22. Ragaad AlTarawneh, Patric Keller, and Achim Ebert. A general introduction to graph visualization techniques. In *IRTG 1131 Workshop 2011, OASICS*, volume 27, June 2011.
23. Ragaad AlTarawneh, Max Langbein, Shah Rukh Humayoun, and Hans Hagen. Topolayout-dg: A topological feature-based framework for visualizing inside behav-

ior of large directed graphs. In *Proceedings of the 13th SIGRAD 2014 Conference of the Swedish Eurographics Chapter, Gothenburg, Sweden, 2014*.

24. Ragaad AlTarawneh, Max Steiner, Davide Taibi, Shah Rukh Humayoun, and Peter Liggesmeyer. Does visualization speed up the safety analysis process? In *A. Bondavalli et al. (Eds.): SAFECOMP 2014 Workshops, LNCS 8696*, pages 431–443. Springer International Publishing Switzerland (2014), 2014.
25. Daniel Archambault, Tamara Munzner, and David. Auber. Topolayout: Multilevel graph layout by topological features. *IEEE Transactions on Visualization and Computer Graphics*, 13:305–317, 2007.
26. Daniel Archambault, Tamara Munzner, and David Auber. Grouseflocks: Steerable exploration of graph hierarchy space. *IEEE Transactions on Visualization and Computer Graphics*, 14(4):900–913, July 2008.
27. David Auber, Yves Chiricota, Fabien Jourdan, and Guy Melançon. Multiscale visualization of small world networks. In *Proceedings of the Ninth Annual IEEE Conference on Information Visualization*, INFOVIS'03, pages 75–81, Washington, DC, USA, 2003. IEEE Computer Society.
28. Algirdas Avienis, Jean-Claude Laprie, Brian Randell, and Carl Landwehr. Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing*, 1(1):11–33, 2004.
29. Victor R. Basili, Gianluigi Caldiera, and H. Dieter Rombach. The goal question metric approach. In *Encyclopedia of Software Engineering*. Wiley, 1994.
30. Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, 1998.
31. J. Bauer, Sebastian Thelen, and Achim Ebert. Evaluation of large display interaction using smart phones. *IEEE Conference on Visual Analytics Science and Technology, VisWeek 2011*, 2011.
32. Johannes Behr and Marc Alexa. Volume visualization in vrml, 1993.
33. G Bell, R Carey, and C Marrin. The virtual reality modeling language specification v.2, 1996.

34. P. Bieber, C. Bougnol, C. Castel, J. p. Heckmann, C. Kehren, and C. Seguin. Safety assessment with altarica - lessons learnt based on two aircraft system studies. In *18th IFIP World Computer Congress, Topical Day on New Methods for Avionics Certification*, page 26, 2004.

35. Eric A. Bier, Maureen C. Stone, Kenneth A. Pier, Kenneth P. Fishkin, Thomas Baudel, Matthew Conway, William Buxton, and Tony DeRose. Toolglass and magic lenses: the see-through interface. In Catherine Plaisant, editor, *CHI Conference Companion*, pages 445–446. ACM, 1994.

36. Anna O. Bilska, Kenneth H. Leider, Magdalena Procopiuc, Octavian Procopiuc, Susan H. Rodger, Jason R. Salemme, and Edwin Tsang. A collection of tools for making automata theory and formal languages come alive. In *ACM SIGCSE Bulletin*, pages 15–19, 1997.

37. Katy Börner, Chaomei Chen, and Kevin W. Boyack. Visualizing knowledge domains. *Annual Review of Information Science and Technology*, 37(1):179–255, 2003.

38. Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, December 2011.

39. Marco Bozzano and Adolfo Villafiorita. *Design and Safety Assessment of Critical Systems*. CRC Press (Taylor and Francis), an Auerbach Book, 2010.

40. Ulrik Brandes, Tim Dwyer, and Falk Schreiber. Visualizing related metabolic pathways in two and a half dimensions. 2912:111–122, 2004.

41. Wolfgang Broll and Tanja Koop. Vrml: Today and tomorrow. 1996.

42. Marc Brown. Zeus: A system for algorithm animation and multi-view editing. In *IEEE Workshop on Visual Languages*, pages 4–9, 1992.

43. Stefan Bruckner and Meister Eduard Gröller. Exploded views for volume data. *IEEE Tran. on Vis. and Com. Grap.*, pages 1077–1084, 2006.

44. Jeromy CarriÃÍre and Rick Kazman. Interacting with Huge Hierarchies: Beyond Cone Trees. In *Proc IEEE Information Visualization, IEEE Computer Press, Los Alamitos, CA*, pages 74–81. IEEE, 1995.

45. CESAR Project. "cesar project report"., May 2010. "`www.cesarproject.eu`".

46. Daniela Chudá. Visualization in education of theoretical computer science. In *Proceedings of the 2007 international conference on Computer systems and technologies*, CompSysTech '07, pages 84:1–84:6, New York, NY, USA, 2007. ACM.

47. Andy Cockburn and Bruce McKenzie. An evaluation of cone trees. In Sharon McDonald, Yvonne Waern, and Gilbert Cockton, editors, *People and Computers XIV – Usability or Else!*, pages 425–436. Springer London, 2000.

48. Andy Cockburn and Bruce McKenzie. Evaluating the effectiveness of spatial memory in 2d and 3d physical and virtual environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '02, pages 203–210, New York, NY, USA, 2002. ACM.

49. Christopher Collins and Sheelagh Carpendale. Vislink: Revealing relationships amongst visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1192–1199, 2007.

50. Graph Drawing Community, November 2011. "`http://www.graphdrawing.org/`".

51. Isabel F. Cruz and Joseph P. Twarog. 3d graph drawing with simulated annealing. pages 162–165. Springer-Verlag, 1996.

52. Steve Cunningham and Michael J. Bailey. Lessons from scene graphs: Using scene graphs to teach hierarchical modeling. In *Computers and Graphics, 2001*, 2001.

53. Hubert de Fraysseix, Jänos Pach, and Richard Pollack. How to draw a planar graph on a grid. *Combinatorica*, 10(1):41–51, 1990.

54. Chris Dede, Marilyn C. Salzman, and R. Bowen Loftin. Sciencespace: Virtual realities for learning complex and abstract scientific concepts. In *Proceedings of IEEE VRAIS*, pages 246–252, 1996.

55. Josh Dehlinger and Robyn R. Lutz. Plfaultcat: A product-line software fault tree analysis tool. *Automated Software Engineering*, 13(1):169–193, 2006.

56. Matthias Deller, Achim Ebert, Stefan Agne, and Daniel Steffen. Guiding attention in information-rich virtual environments. In JJ Villanueva, editor, *IASTED International Conference on Visualization Imaging and Image Processing. Spain*, pages 310–315. International Association of Science and Technology for Development (IASTED), ACTA Press, 9 2008.

57. Alan Dix, Janet E. Finlay, Gregory D. Abowd, and Russell Beale. *Human-Computer Interaction (3rd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2003.

58. Peter. Eades. A heuristic for graph drawing. In *Congressus Numerantium*, volume 42, pages 149–160, 1984.

59. Peter Eades. Drawing free trees. *Bulletin of the Institute for Combinatorics and its Applications, pp. 10 -36*, 1992.

60. Peter Eades and Qing-Wen Feng. Multilevel Visualization of Clustered Graphs. In *Proceedings Graph Drawing, GD*, number 1190, pages 101–112, Berlin, Germany, Jan–Feb 1996. Springer-Verlag.

61. Peter Eades and Mao Lin Huang. Navigating clustered graphs using force-directed methods. *Journal of Graph Algorithms and Applications*, 4:157–181, 2000.

62. Peter Eades and Sue Whitesides. Drawing graphs in two layers. *Theoretical Computer Science*, 131:361–374, 1994.

63. Qing-Wen Feng, Robert F. Cohen, and Peter Eades. Planarity for clustered graphs. In *ESA*, pages 213–226, 1995.

64. Tom Ferrell and Uma Ferrell. Assuring avionics – updating the approach for the 21st century. In Andrea Bondavalli, Andrea Ceccarelli, and Frank Ortmeier, editors, *Computer Safety, Reliability, and Security*, volume 8696 of *Lecture Notes in Computer Science*, pages 375–383. Springer International Publishing, 2014.

65. Benjamin Finkel and Roberto Tamassia. Curvilinear graph drawing using the force-directed method. pages 448–453, 2004.

66. Tobias Föhst, Michael Arndt, Karsten Berns, Christiano Couto Gava, and Raquel Frizera Vassallo. Off-road place recognition using fused image features. In *ISR/ROBOTIK*, pages 1–6. VDE Verlag, 2010.

67. Thomas M. J. Fruchterman and Edward M. Reingold. Graph drawing by force-directed placement. *Software–Practice & Experience*, 21(11):1129–1164, 1991.

68. Insa Marie-Ann Fuhrmann. Layout of compound graphs. *Diploma thesis from Department of Computer Science Real-Time and Embedded Systems Group, Christian-Albrechts-UniversitÃd't zu Kiel*, 2012.

69. George W. Furnas and Benjamin B. Bederson. Space-scale diagrams: Understanding multiscale interfaces. pages 234–241, 1995.

70. Pawel Gajer and Stephen G. Kobourov. GRIP: Graph Drawing with Intelligent Placement. *Journal of Graph Algorithms and Applications*, 6, 2002.

71. M. R. Garey and D. S. Johnson. Crossing Number is NP-Complete. *SIAM Journal on Algebraic and Discrete Methods*, 4(3):312–316, 1983.

72. Natasha Gelfand and Roberto Tamassia. Algorithmic patterns for orthogonal graph drawing. In Sue Whitesides, editor, *Graph Drawing*, volume 1547 of *Lecture Notes in Computer Science*, pages 138–152. Springer, 1998.

73. Andreas Gerasch, Michael Kaufmann, and Oliver Kohlbacher. Rebuilding KEGG maps: Algorithms and benefits. In *IEEE Pacific Visualization Symposium, PacificVis 2014, Yokohama, Japan, March 4-7, 2014*, pages 97–104, 2014.

74. Nahum D. Gershon, Stuart K. Card, and Stephen G. Eick. Information visualization tutorial. In Alan Edwards and Steven Pemberton, editors, *CHI Extended Abstracts*, pages 196–197. ACM, 1997.

75. Peter Gloor. Aace-algorithm animation for computer science education. In *Visual Languages, 1992. Proceedings., 1992 IEEE Workshop on*, pages 25–31, 1992.

76. Tovi Grossman and Ravin Balakrishnan. An evaluation of depth perception on volumetric displays. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '06, pages 193–200, New York, NY, USA, 2006. ACM.

77. Lars Harms-Ringdahl. *Guide to safety analysis for accident prevention*. IRS Riskhantering AB, Bergsprängargränd 2A, SE-11635 Stockholm, Sweden, version number 1.0.1 edition, 2013.

78. Ch. Heine and Universitiy of Leipzig, Novemebr 2011. "`http://www.informatik.uni-leipzig.de/ hg/libgraph/`".

79. Carsten Görg Helen C. Purchase, Eve Hoggan. *How Important Is the Mental Map? An Empirical Investigation of a Dynamic Graph Layout Algorithm*, volume 4372/2007, pages 184–195. Springer Berlin / Heidelberg, 2007.

80. Nathalie Henry and Jean-Daniel Fekete. Matrixexplorer: A dual-representation system to explore social networks. *IEEE Transactions on Visualization and Computer Graphics*,

12(5):677–684, September 2006.

81. Ivan Herman., Guy MelanÃğon, and M. Scott Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE Transaction on Visualization and Computer Graphics*, 6(1):24–43, 2000.

82. Daneil Holten. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE Trans. Vis. Comput. Graph.*, 12(5):741–748, 2006.

83. Mao Lin Huang and Peter Eades. A fully animated interactive system for clustering and navigating huge graphs, 1998. 10.1007/3-540-37623-2_29.

84. Shah Rukh Humayoun, Ragaad AlTarawneh, and Yael Dubinsky. Formalizing user interaction requirements of mobile applications. *Proceedings of the 28th British HCI Conference (BCS-HCI 2014), Southport, September 9-12, 2014*.

85. Shah Rukh Humayoun, Ragaad AlTarawneh, Achim Ebert, and Yael Dubinsky. Automate the decision on best-suited UI design for mobile apps. In *Proceedings of the 1st International Conference on Mobile Software Engineering and Systems, MOBILESoft 2014, Hyderabad, India, June 2-3, 2014*, pages 66–68, 2014.

86. David Hume. *Graph Drawing Algorithms*, pages 400–401. Information Visualization Beyond the Horizon, Springer London, second edition edition, 2006.

87. Christopher Hundhausen and Sarah Douglas. Using visualizations to learn algorithms: should students construct their own, or view an expert's? In *Proceedings of IEEE International Symposium on Visual Languages.*, pages 21–28, 2000.

88. Giti Javidi. Virtual reality and education. In *the 24th Annual Conference on International Education, Monterrey, Mexico*, 2001.

89. Brian Johnson and Ben Shneiderman. Tree-maps: A space-filling approach to the visualization of hierarchical information structures. In *Proceedings of the 2nd Conference on Visualization '91*, pages 284–291, Los Alamitos, CA, USA, 1991. IEEE Computer Society Press.

90. Huzefa H. Kagdi and Jonathan I. Maletic. Onion graphs for focus+context views of uml class diagrams. In *VISSOFT*, pages 80–87, 2007.

91. Huzefa H. Kagdi and Jonathan I. Maletic. Onion graphs for focus+context views of uml class diagrams. In Jonathan I. Maletic, Alexandru Telea, and Andrian Marcus, editors,

VISSOFT, pages 80–87. IEEE Computer Society, 2007.

92. Bernhard Kaiser and Catharina Gramlich. State event fault trees–a safety analysis model for software-controlled systems. *Reliability Engineering System Safety*, 92(11):1521–1537, 2007.
93. Bernhard Kaiser, Peter Liggesmeyer, and Oliver Mäckel. A new component concept for fault trees. *Reproduction*, 33:37–46, 2003.
94. Tomihisa Kamada and Satoru Kawai. An algorithm for drawing general undirected graphs. *Inf. Process. Lett.*, 31(1):7–15, 1989.
95. Goos Kant. Drawing planar graphs using the canonical ordering. *ALGORITHMICA*, 16:4–32, 1996.
96. Richard Karp. Reducibility among combinatorial problems. In R. Miller and J. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.
97. Jana Katreniaková. Mental map preserving cluster expansion. In *SOFSEM (2)*, pages 58–69, 2008.
98. Michael Kaufmann and Dorothea Wagner, editors. *Drawing Graphs: Methods and Models*. Springer-Verlag, London, UK, UK, 2001.
99. Taimur Khan, Henning Barthel, Achim Ebert, and Peter Liggesmeyer. eCITY: A tool to track software structural changes using an evolving city. In *ICSM*, pages 492–495, 2013. to appear.
100. Doug Kimelman, Bruce Leban, Tova Roth, and Dror Zernik. Reduction of visual complexity in dynamic graphs, 1995.
101. Ernst Kleiberg, Huub van de Wetering, and Jarke J. van Wijk. Botanical visualization of huge hierarchies. In Keith Andrews, Steven F. Roth, and Pak Chung Wong, editors, *INFOVIS*, pages 87–94. IEEE Computer Society, 2001.
102. Oliver Kreylos. Vrui virtual reality toolkit, July 2014. "`http://idav.ucdavis.edu/∼okreylos/ResDev/Vrui/index.html`".
103. Pushpa Kumar, Nary Subramanian, and Kang Zhang. Savit: Technique for visualization of digital home safety. *ACIS International Conference on Computer and Information Science*, pages 1120–1125, 2009.

104. Robotics Lab and Kaiserslauetrn University. Ravon:robust autonomous vehicle for off-road navigation, July 2014. "https://agrosy.informatik.uni-kl.de/robotergalerie/ravon/ravon-archiv/".

105. John Lamping, Ramana Rao, and Peter Pirolli. A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies. In *proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 401–408, 1995.

106. John Lamping, Ramana Rao, and Peter Pirolli. A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. pages 401–408, 1995.

107. Martín Larrea, Sergio Martig, and Silvia Castro. Spherical layout for 3D tree visualization. In *Proceedings of the IADIS Computer Graphics, Visualization, Computer Vision and Image Processing*, pages 91–98, 2007.

108. Edward A. Lee and Sanjit A. Seshia. *Introduction to Embedded Systems - A Cyber-Physical Systems Approach*. Lee and Seshia, 1 edition, 2010.

109. Wilmot Li, Maneesh Agrawala, Brian Curless, and David Salesin. Automated generation of interactive 3d exploded view diagrams. *ACM Transition Graphs*, 2008.

110. Wilmot Li, Maneesh Agrawala, and David Salesin. Interactive image-based exploded view diagrams. In *GI '04*, pages 203–212, 2004.

111. Henry Lieberman. Powers of ten thousand: Navigating in large information spaces. In *ACM Symposium on User Interface Software and Technology*, pages 15–16, 1994.

112. Peter Liggesmeyer. *Qualitätssicherung softwareintensiver technischer Systeme*. Spektrum-Verlag, Heidelberg, Germany, 2000.

113. David Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. Henry Holt and Co., Inc. New York, NY, USA, 1982.

114. Peter Marwedel. *Embedded system design*, volume 1. Springer, 2006.

115. Tamara Munzner. H3: laying out large directed graphs in 3d hyperbolic space. In *Proceedings of the 1997 IEEE Symposium on Information Visualization (InfoVis '97)*, pages 2–, Washington, DC, USA, 1997. IEEE Computer Society.

116. Tamara Munzner. Drawing large graphs with h3viewer and site manager (system demonstration). In *In Proceedings of Graph Drawing'98, number 1547 in Lecture Notes in Computer Science*, pages 384–393. Springer-Verlag, 1998.

117. Tamara Munzner and Paul Burchard. Visualizing the structure of the world wide web in 3d hyperbolic space. In *VRML*, pages 33–38, 1995.

118. Ken Nakayama and Gerald H. Silverman. Serial and Parallel Processing of Visual Feature Conjunctions. *Nature*, 320(Mar):264–265, 1986.

119. Thomas L. Naps, Guido R össsling, Vicki Almstrum, Wanda Dann, Rudolf Fleischer, Chris Hundhausen, Ari Korhonen, Lauri Malmi, Myles Mcnally, Susan Rodger, and äsngel. Exploring the role of visualization and engagement in computer science education. In *ITiCSE-WGR 02: Working group reports from ITiCSE on Innovation and technology in computer science education*, pages 131–152, New York, NY, USA, 2002. ACM.

120. Quang Vinh Nguyen and Mao Lin Huang. Space-optimized tree: A connection+enclosure approach for the visualization of large hierarchies. *Information Visualization*, 2(1):3–15, March 2003.

121. Larry L. Peterson and Bruce S. Davie. *Computer Networks: A Systems Approach, 3rd Edition*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

122. Stephen D. Peterson, Magnus Axholt, and Stephen R. Ellis. Technical section: Objective and subjective assessment of stereoscopically separated labels in augmented reality. *Comput. Graph.*, 33:23–33, February 2009.

123. Doantam Phan, Ling Xiao, Ron Yeh, Pat Hanrahan, and Terry Winograd. Flow map layout. In *Proceedings of the Proceedings of the 2005 IEEE Symposium on Information Visualization*, INFOVIS '05, pages 29–, Washington, DC, USA, 2005. IEEE Computer Society.

124. Martin Proetzsch, Tobias Luksch, and Karsten Berns. *Development of Complex Robotic Systems Using the Behavior-based Control Architecture iB2C*, volume 58. North-Holland Publishing Co., Amsterdam, The Netherlands, The Netherlands, January 2010.

125. Helen C. Purchase. Which aesthetic has the greatest effect on human understanding? In *Proceedings of the 5th International Symposium on Graph Drawing*, GD '97, pages 248–261, London, UK, UK, 1997. Springer-Verlag.

126. Marcus Raitner. Visual navigation of compound graphs. In Jänos Pach, editor, *Graph Drawing*, volume 3383 of *Lecture Notes in Computer Science*, pages 403–413. Springer Berlin Heidelberg, 2005.

127. Khairi Reda, Alessandro Febretti, Aaron Knoll, Jillian Aurisano, Jason Leigh, Andrew Johnson, Michael E. Papka, and Mark Hereld. Visualizing large, heterogeneous data in hybrid-reality environments. *IEEE Computer Graphics and Applications*, 33(4):38–48, 2013.

128. Edward M. Reingold, John, and S. Tilford. Tidier drawing of trees. *IEEE Trans. Software Eng*, 1981.

129. George G. Robertson, Jock D. Mackinlay, and Stuart K. Card. Cone trees: Animated 3d visualizations of hierarchical information. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '91, pages 189–194, New York, NY, USA, 1991. ACM.

130. Tom Roxborough and Arunabha Sen. Graph clustering using multiway ratio cut. In *Proceedings of Graph drawing, volume 1353 of Lecture Notes in computer Science*, pages 291–296. Springer-Verlag, 1997.

131. Manojit Sarkar and Marc H. Brown. Graphical fisheye views of graphs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '92, pages 83–91, New York, NY, USA, 1992. ACM.

132. Walter Schnyder. Embedding planar graphs on the grid. In *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '90, pages 138–148, Philadelphia, PA, USA, 1990. Society for Industrial and Applied Mathematics.

133. William R. Shadish, Thomas D. Cook, and Donald T. Campbell. *Experimental and Quasi-Experimental Designs for Generalized Causal Inference*. Houghton Mifflin, 2 edition, 2001.

134. S. S. Shapiro and M. B. Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4):591–611, Dec. 1965.

135. Yossi Shiloach. *Arrangements of Planar Graphs on Planar Lattices*. PhD thesis, Weizmann Institute of Science, Rehovot, Israel, 1976.

136. Guttorm Sindre, Björn Gulla, and Häkon G. Jokstad. Onion graphs: aesthetics and layout. In *VL*, pages 287–291. IEEE Computer Society, 1993.

137. Software Engineering Research Group: Dependability Kaiserslautern University, Essarel Tool. Embedded systems sadety and reliability analyer, May 2014.

"http://essarel.de/index.php?site=backgroundtext".

138. John Stasko and Eugene Zhang. Focus+context display and navigation techniques for enhancing radial, space-filling hierarchy visualizations. In *Proceedings of the IEEE Symposium on Information Vizualization 2000*, INFOVIS '00, pages 57–, Washington, DC, USA, 2000. IEEE Computer Society.

139. John T. Stasko. Tango: a framework and system for algorithm animation. *Computer*, 23(9):27–39, 1990.

140. Steven Segenchuk. The role of visualization in education, March 2014. "http://web.cs.wpi.edu/ att/courses/cs563/talks/education/index.html".

141. Kozo Sugiyama and Kazuo Misue. Visualization of structural information: automatic drawing of compound digraphs. *Systems, Man and Cybernetics, IEEE Transactions on*, 21(4):876–892, 1991.

142. Kozo Sugiyama and Kazuo Misue. A generic compound graph visualizer/manipulator: D-ABDUCTOR. In *Graph Drawing, Symposium on Graph Drawing, GD '95, Passau, Germany, September 20-22, 1995, Proceedings*, pages 500–503, 1995.

143. Kozo Sugiyama, Shojir Tagawa, and Mitsuhiko Toda. Methods for visual understanding of hierarchical system structures. *Ieee Transactions On Systems Man And Cybernetics*, 11(2):109–125, 1981.

144. Yoichi Tanaka, Yoshihiro Okada, and Koichi Niijima. Treecube: Visualization tool for browsing 3d multimedia data. In Ebad Banissi, Katy Börner, Chaomei Chen, Gordon Clapworthy, Carsten Maple, Amy Lobben, Christopher J. Moore, Jonathan C. Roberts, Anna Ursyn, and Jian Zhang 0006, editors, *IV*, pages 427–432. IEEE Computer Society, 2003.

145. Markus Tatzgern, Denis Kalkofen, and Dieter Schmalstieg. Compact explosion diagrams. In *NPAR*, pages 17–26, 2010.

146. Gabriel Taubin, W.P. Horn, F. Lazarus, and Jarek Rossignac. Geometry coding and vrml, Jun 1998.

147. CAIDA The Cooperative Association for Internet Data Analysis, Novemebr 2011. "http://www.caida.org/tools/visualization/walrus/".

148. TU Dortmund and Osnabrück University and University Sidney and University of Cologne and oreas GmbH. Open graph drawing framework, June 2014. "`http://www.ogdf.net/doku.php`".

149. Frank van Ham and Jarke J. van Wijk. Interactive visualization of small world graphs. In *Proceedings of the IEEE Symposium on Information Visualization*, INFOVIS '04, pages 199–206, Washington, DC, USA, 2004. IEEE Computer Society.

150. Jarke J. Van Wijk and Huub van de Wetering. Cushion treemaps: Visualization of hierarchical information. page 73, 1999.

151. Viswanath Venkatesh, Michael G. Morris, Gordon B. Davis, and Fred D. Davis. User acceptance of information technology: Toward a unified view. *MIS Q.*, 27(3):425–478, September 2003.

152. W. E. Vesely, F. F. Goldberg, Norman H. Roberts, and David F. Haasl. *Fault Tree Handbook*. U.S. Nuclear Regulatory Commission, Washington, DC, 1981.

153. Chris Walshaw. A multilevel algorithm for force-directed graph drawing. *Journal of Graph Algorithms and Applications*, pages 253–285, 2003.

154. Colin Ware. *Information Visualization: Perception for Design (Interactive Technologies)*. Morgan Kaufmann, 1st edition, Feb 2000.

155. Colin Ware and Robert Bobrow. Supporting visual queries on medium-sized node–link diagrams. *Information Visualization*, 4(1):49–58, 2005.

156. Colin Ware and Glenn Franck. Evaluating stereo and motion cues for visualizing information nets in three dimensions. *ACM Trans. Graph.*, 15:121–140, April 1996.

157. Martin Wattenberg. Visual exploration of multivariate graphs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 811–819, New York, NY, USA, 2006. ACM.

158. Markus Weber. A survey of semantic annotations for knowledge management. *DFKI GmbH*, page 1, 2008.

159. Cui Weiwei. A Survey on Graph Visualization. *Hong Kong University of Science and Technology Clear Water Bay, Kowloon, Hong Kong*.

160. Elecia White. *Making Embedded systems*, volume 1. O'Reilly Media, 2012.

161. Yi Yang, Patric Keller, Yarden Livnat, Peter Liggesmeyer, and Hans Hagen. Improving safety-critical systems by visual analysis. *Dagstuhl Follow-Up series*, 2012.

162. Ka-Ping Yee, Danyel Fisher, Rachna Dhamija, and Marti Hearst. Animated exploration of dynamic graphs with radial layout. In *Proceedings of the IEEE Symposium on Information Visualization 2001*, pages 43–, Washington, DC, USA, 2001. IEEE Computer Society.

163. Ji Soo Yi, Youn ah Kang, John Stasko, and Julie Jacko. Toward a deeper understanding of the role of interaction in information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1224–1231, 2007.

Ragaad AlTarawneh

EMAIL: tarawneh@cs.uni-kl.de
WEB: http://cg.cs.uni-kl.de/~tarawneh/

ACADEMIC QUALIFICATION:

04/2010 – 07/2015	**PhD Student in Computer Engineering** Computer Graphics and HCI Group in Kaiserslautern University (Link: http://www-hagen.informatik.uni-kl.de/) Supervisor: Prof. Achim Ebert Co-Supervisor: Prof. Hans Hagen, Prof. Peter Liggesmeyer Research Topics in PhD: Human-Computer Interaction (HCI), Information Visualization, Graph Visualization, Software Visualizations, Stereoscopic Visualization, 3D Displays, Usability Engineering, Embedded Systems, and Computer Science Education.
10/2005 – 07/2007	**MSc in Computer Science** [with Honor Degree, 3.88 out of 4.00 GDP] Department of Computer Science, University of Jordan, Jordan Thesis: " USING GENETIC SOLUTION FOR MANET MOBILITY BASED CLUSTERING ALGORITHMS" Supervisor: Dr. Wesam Al Mobaideen Topics: Networks, Networks Management, Object-Oriented Design, Database Management, Distributed Operating Systems, Formal Specifications, Computer Architectures, Natural Languages Processing.
10/2001 – 09/2004	**Bachelor Degree in Computer Science** Mutah University, Jordan [Finished the Degree of 4 years in 3 years only, (82.7 out 100 GDP)]

RESEARCH INTERESTS:

Information Visualization	- Graphs Visualization	- Spatial Data Visualization
	- Visual User Interfaces	- Non-Spatial Data Visualization
	- Virtual Reality	- Compound Graphs Visualization
	- Visual Design	- Visualization for Large Displays
	- Software Visualization	- 3D Visualization
	- Interaction and Navigation	-Stereoscopic Depth-based visualization
Human-Computer Interaction	- User Evaluation studies	- Human factors engineering
	- User-Centred Design (UCD)	- Collaborative interactive environments
Software Engineering	- Safety Analysis	- Software Systems Analysis
	- Software Metrics Analysis	

Working Experience

04/07/2015 – To date

- **Research Assistant**
  Computer Graphics and HCI Group
  University of Kaiserslautern, Germany
  Scientific In-charge: Prof. Achim Ebert & Prof. Hans Hagen

20/04/2015 – 10/07/2015

- **Intern at Intel Labs (Data Visualization Group)**
  Intel Corporation, Hillsborough Portland, Oregon, USA

Research Areas:

Parallel computing and high performance technology for visualizing big data.

10/2010 – 31/03/2015

- **Research Assistant**
  Computer Graphics and HCI Group
  University of Kaiserslautern, Germany
  Scientific In-charge: Prof. Achim Ebert & Prof. Hans Hagen

Main responsibilities:

Coordinating in courses teaching, *working on projects, supervising with master and bachelor students for their seminar and final theses.*

Projects:

1. **"VIER4ES2 - Virtual and Augmented Reality for Maximum Safety and Reliability of Embedded Systems"**. Involvement in this project was at different stages especially for performing system and usability evaluation. Our main collaborations were with Fraunhofer IESE, Robotics Research Lab at University of Kaiserslautern, and Software Engineering: Dependability Group at University of Kaiserslautern.
2. "**UID4Mobile - Scalable User Interface Design for Mobile Applications**" This project was in collaboration with Fraunhofer IESE. The theme of this project was to investigate and build approaches, techniques, frameworks, and tools to help the interaction designers in designing and building scalable mobile apps user interfaces. My involvement was in the evolution of mobile prototypes through genetic algorithms.

Teaching Activates:

1) **Introduction to Information Visualization and Visual Analytics** (Summer Semester 2014, 2015)
   - Giving parts of lectures and conducting exercises.
2) **Human-Computer Interaction** Winter semester 2014/2015
   - Conducting exercises
3) **Information Visualization, Visual Analytics, and Human-Computer Interaction** (Summer School)
   - 2-days summer school at Universidad de los Andes, Bogota, Colombia, July 02-03, 2014
   - 2-days summer school at Universidad Industrial de Santander, Bucaramanga Colombia, July 07-08, 2014

10/2007 – 07/2009

- **Lecturer and Research Assistant**
  Department of Computer and Systems Engineering, Mutah University, Jordan.

11/2006 – 10/2007

- **Programmer**
  Ministry of Agriculture, Government of Jordan.

10/2004 – 01/2005

- **Teacher**
  Ministry of Education, Government of Jordan Government.

Achievements

- Best Paper award in SIG-RAD 2014 conference
- Best Paper award in PID-MAD 2013 workshop

- Jordan Military scholarship for Distinct Students in Jordan (Bachelor study)
- King Abdullah II Scholarship for Distinct Students (Master Study cost)
- Mutah University Scholarship (To support partially PhD studies)
- DFG scholarships from Germany (To support partially PhD studies)

Memberships

- Student member of the Association for Computer Machinery (ACM)
- Student member of the British Computer Society (BCS)

Languages

- English
- Arabic
- German

Research Activities:

Organizing Workshops/Conferences

- In the Program Committee of the IFIP WG 13.7 Workshop on Designing Interaction and Visualization for Mobile Applications (DIViM 2015) (http://hciv.de/divim15/), September 14, 2014, Bamberg, Germany.
- In the Program Committee of the IEEE and ifip 2nd International Workshop on Usability and Accessibility focused Requirements Engineering [**UsARE 2014]** (http://hciv.de/usare14/), August 26, 2014, Karlskrona, Sweden.
-
- Reviewing
  1) ACM IDC 2015
  2) ACM EICS 2015
  3) ACM CHI 2015
  4) ACM IUI 2015
  5) ACM MOBILESoft 2015
  6) DIViM 2015
  7) IEEE UsARE 2014
  8) PID-MAD 2013 workshop at MobileHCI 2013
- Student Volunteer
  1) IEEE VR 2015
  2) IEEE VIS 2014
  3) IEEE RE 2014
  4) ACM IUI 2014
  5) Meeting C++ 2014
  6) IEEE VIS 2013

Research Publications

Journal Publications (peer-reviewed)

1. **Ragaad AlTarawneh** and Shah Rukh Humayoun. *"Evolving Mobile Prototypes Towards The Best-suited Design and Interaction Schema Using The Genetic Algorithm".* In International Journal of Mobile Human Computer Interaction (IJMHCI), 6(2), pp. 1-10, 2014.

Book Chapters (peer-reviewed)

1. **Ragaad AlTarawneh**, Max Steiner, Davide Taibi, Shah Rukh Humayoun, and Peter Liggesmeyer. *"Does Visualization Speed up the Safety Analysis Process?".* Computer Safety, Reliability, and Security Lecture Notes in Computer Science Volume 8696, Springer, pp. 431-443, 2014. Paper presented in the 3rd Workshop on Next Generation of System Assurance Approaches for Safety-Critical Systems (SASSUR 2014) held in conjunction with the 33rd International Conference on Computer Safety, Reliability and Security (SafeComp 2014) on September 10-12, Florence, Italy, 2014.
2. **Ragaad AlTarawneh**, Jens Bauer, Shah Rukh Humayoun, Patric Keller, Achim Ebert. "*The Reflection Layer Extension to the Stereoscopic Highlight Technique for Node-Link Diagrams: An Empirical Study*". Presented at ISVC '13, Advances in Visual Computing, Lecture Notes in Computer Science, Springer, v. 8034, pp 1-12, 2013.

3. **Ragaad AlTarawneh**, Patric Keller, and Achim Ebert. "*A General Introduction To Graph Visualization Techniques*". In Proceedings of the IRTG 1131 Workshop 2011, OASICS, Vol. 27, VLUDS 2011, June 10-11, 2011, Kaiserslautern, Germany.

Conferences (peer-reviewed)

1. **Ragaad AlTarawneh**, Shah Rukh Humayoun, Johannes Schultz, Achim Ebert, and Peter Liggesmeyer. *"LayMan: A Visual Interactive Tool to Support Failure Analysis in Embedded Systems*". In the proceedings of the 9th European Conference on Software Architecture (ECSA 2015), Dubrovnik-Cavtat, Croatia, September 2015. *(To be published)*
2. **Ragaad AlTarawneh,** Shah Rukh Humayoun, and Achim Ebert. *"A Hybrid Approach for Visualizing Spatial and Non-spatial Data Types of Embedded Systems".* In the proceedings of J. Abascal et al. (Eds.): INTERACT 2015, Part IV, LNCS 9299, 2015. *(To be published)*
3. Ragaad AlTarawneh, Shah Rukh Humayoun, and Achim Ebert. *"Utilization of Variation in Stereoscopic Depth for Encoding Aspects of Non-spatial Data*". In the proceedings of the IEEE 10th Symposium on 3D User Interfaces (3DUI 2015), March 23-24, 2014, Arles, France.
4. **Ragaad AlTarawneh**, Shah Rukh Humayoun, and Achim Ebert. *"ExpanD: A Stereoscopic Expanding Technique for Compound Graphs"*. In 20th ACM Symposium on Virtual Reality Software Technology (VRST 2014), ACM, pp. 223--224, Edinburg, United Kingdom, November 11-13, 2014.
5. Shah Rukh Humayoun, **Ragaad AlTarawneh,** and Yael Dubinsky. *"Formalizing User Interaction Requirements of Mobile Applications"*. In Proceedings of the 28th British HCI Conference (BCS-HCI 2014), Southport, September 9-12, 2014.
6. **Ragaad AlTarawneh**, Max Langbein, Shah Rukh Humayoun, and Hans Hagen. "*TopoLayout-DG: A Topological Feature-Based Framework for Visualizing Inside Behavior of Large Directed Graphs."* In Proceedings of the 13th SIGRAD 2014 Conference of the Swedish Eurographics Chapter, Gothenburg, Sweden, 2014. **(Best paper Award)**
7. **Ragaad AlTarawneh**, Jens Bauer, Shah Rukh Humayoun, and Achim Ebert. "*ESSAVis++: an interactive 2Dplus3D visual environment to help engineers in understanding the safety aspects of embedded system*". In Proceedings of the 2014 ACM SIGCHI symposium on Engineering interactive computing systems (EICS 2014), pp. 201-204, ACM, 2014.
8. Shah Rukh Humayoun, **Ragaad AlTarawneh**, Achim Ebert, and Yael Dubinsky. "*Automate the Decision on Best-Suited UI Design for Mobile Apps*". In Proceedings of the 1st International Conference on Mobile Software Engineering and Systems (MOBILESoft 2014) [in conjunction with 36th IEEE ICSE 2014], ACM, pp. 66-68, 2014.
9. **Ragaad AlTarawneh**, Andreas Griesser, Jens Bauer, Shah Rukh Humayoun, and Achim Ebert. "*Poster: 3DintEx -- A Tool to Explore Interactively the Structural and Behavioral Aspects of System Models in 3D Environments*". In IEEE 9th Symposium on 3D User Interfaces (3DUI 2014), pp. 141-142, March 29-30, 2014, Minneapolis (MN), USA.
10. **Ragaad AlTarawneh**, Jens Bauer, Achim Ebert. *A Visual Interactive Environment for Enhancing Collaboration between Engineers for the Safety Analysis Mechanisms in Embedded Systems*. In CSCW Companion '14. ACM, New York, NY, USA, 125-128, 2014.
11. **Ragaad AlTarawneh**, Johannes Schultz, and Shah Rukh Humayoun. "*CluE: An Algorithm for Expanding Clustered Graphs*". In Proceedings of the 7th IEEE Pacific Visualization Symposium (PacificVis) 2014, IEEE, pp. 233-37, Yokohama, Japan, March 4-7, 2014.
12. **Ragaad AlTarawneh**, Achim Ebert, and Eduard Kosel. "*The Stereoscopic Filtering Technique*". Poster in IEEE VIS 2013, Atlanta, Georgia, USA, 13-18 Oct. 2013.
13. **Ragaad AlTarawneh**, Jens Bauer, Patric Keller, and Achim Ebert. "*ESSAVis: A 2Dplus3D Visual Platform for Speeding Up the Maintenance Process of Embedded Systems*". In the 27th Int. British Computer Society Human Computer Interaction Conference (HCI 2013), Brunel University, UK, 9 - 13 Sep 2013.
14. **Ragaad AlTarawneh**, Shah Rukh Humayoun. "A Two-Perspective Visualization Approach for Utilizing Visualization Power in Computer

Science Education". CSERC '13, ACM, 4-5 April, 2013, Arnhem/Nijmegen, the Netherlands.

15. **Ragaad AlTarawneh**, Jens Bauer, Shah Rukh Humayoun, Patric Keller, Achim Ebert. "*The Extended Stereoscopic Highlighting Technique for Node-Link Diagrams: An Empirical Study*". The 14th IASTED International Conference on Computer Graphics and Imaging (CGIM 2013), February 12 – 14, 2013.
16. **Ragaad AlTarawneh**, Jens Bauer, Patric Keller, Achim Ebert, and Peter Liggesmeyer. "*ESSAVis: A Framework to Visualize Safety Aspects in Embedded Systems*". Proceedings of SIGRAD 2012, Interactive Visual Analysis of Data, November 29–30, 2012, Växjö, Sweden.

Workshops (peer-reviewed)

1. **Ragaad AlTarawneh**, Shah Rukh Humayoun, and Abdel-karim Al-Jaafreh. *"Towards Optimizing the Sunburst Visualization for Smart Mobile Devices"*. IFIP WG 13.7 Workshop on Designing Interaction and Visualization for Mobile Applications (DIViM 2015), In conjunction with INTERACT 2015, September 14, 2015
2. Ragaad AlTarawneh, Carl S. Marshall, Selvakumar Panneer, and Cindy K. Chung. *"3DIM: An Interactive 3D Map to Visualize Geo-Spatial Data in Mobile Devices"*. IFIP WG 13.7 Workshop on Designing Interaction and Visualization for Mobile Applications (DIViM 2015), In conjunction with INTERACT 2015, September 14, 2015
3. Ragaad AlTarawneh, Jens Bauer, Nicole Menck, Shah Rukh Humayoun, and Achim Ebert. *"assistME: A Collaborative Mobile Environment to Help Engineers in Maintaining the Factory Pipeline"*. MobilENG 2014 Workshop, In conjunction with 5th international conference on Complex Systems Design & Management (CSD&M) 2014, Paris, France, November 14, 2014.
4. **Ragaad AlTarawneh**, Jens Bauer, Nicole Menck, Shah Rukh Humayoun, and Achim Ebert. " *assistME: A Collaborative Mobile Environment to Help Engineers in Maintaining the Factory Pipeline* ". MobilENG 2014 Workshop, in conjunction with 5th international conference on Complex Systems Design & Management (CSD&M) 2014, Paris, France, November 14, 2014. *(To be appear)*
5. **Ragaad AlTarawneh,** Shah Rukh Humayoun. "*Evolving Prototypes Towards The Best-suited Design and Interaction Schema Through Using The Genetic Algorithm*". MobileHCI '13 Workshop: PID-MAD 2013, Munich, Germany, August 27, 2013. **[Best Paper Award]**