FOSTERING SOFTWARE CONCEPTUAL DESIGN VIA THE FUNCTION-BEHAVIOUR-STRUCTURE DESIGN FRAMEWORK

A thesis submitted in partial fulfilment
of the requirements for the degree of

Doctor of Philosophy
by

T.G.Lakshmi
(Roll Number: 154380002)

Supervisor
Prof. Sridhar Iyer

Interdisciplinary Programme in Educational Technology
INDIAN INSTITUTE OF TECHNOLOGY BOMBAY
2020

Dedicated to

my parents-in-law

Amma and Appa, this work is dedicated to your
unconditional and unwavering support.

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources, which have thus not been properly cited, or from whom proper permission has not been taken when needed.

(Signature)
(Name of the student)
(Roll No.)

Date: _____

Approval Sheet

This thesis titled **"Fostering software conceptual design via Function-Behaviour-Structure design framework"** by T.G.Lakshmi is approved for the degree of Doctor of Philosophy.

Examiners

_____

_____

_____

Supervisor (s)

_____

_____

_____

Date: _____ Chairman

Place: _____

Abstract

Engineering design is an ill-structured problem solving and open-ended task (Dym et al., 2005) because design problems have ill-defined goals, states and solution steps. Engineering graduates are expected to design solutions to open-ended real world problems. Due to the complex nature of engineering design, the teaching and learning of this skill is reported to be difficult (Dym et al., 2005). Conceptual design is an important and critical step in design (Pahl et al., 2013). Conceptual design is described as a process in which the functional requirements of the design problem are transformed into descriptions of solution concepts (Chakrabarti & Bligh, 2001). Although all the processes in design are vital for the end result, a strong case can be made for selecting the conceptual design as most critical to the final design (Chakrabarti & Bligh, 2001). The conceptual phase of design thus becomes very significant, as designers tend to develop numerous early ideas and solutions in this phase.

Software design has several common activities with other design domains (Cross et al., 1996). However, the dynamic and intangible nature of software poses unique challenges in software conceptual design (SCD), specifically, components are logical and intangible, and behaviours of such intangible components need to be simulated along with simulation of end-users interactions (Petre et al., 2010). Experts create integrated solutions that fulfil the requirements. Novices find designing software solutions for open-ended problems daunting. There have been previous studies of novice difficulties (Eckerdal et al., 2006), however the underlying mechanism that causes these difficulties has yet to be unearthed. Moreover the ways to alleviate them in the context of SCD have not been reported. Current teaching-learning methods do not explicitly train students to overcome these difficulties (Armarego, 2009). There is a need to understand novices' design processes and explicitly train computer-engineering students in SCD. This is the motivation of this thesis; firstly, to develop an understanding of novices' design processes in SCD and secondly use this understanding to design supports for novices' to create integrated SCD.

We used the function-behaviour-structure (FBS) design framework (Gero & Kannengeiser, 2014) as a lens to analyse novice processes as well as support the

creation of SCD. We followed a design based research methodology (Barab, 2014). We started with understanding novices' design processes, the design strategies and cognitive processes. To identify these, we used protocol analysis (Gero et al., 2011) with novice computer engineering students (Study 1), to collect data, as they create software conceptual design for open-ended problems. We found that novices are fixated to a single view of the software solution and unable to utilize multiple formal representations of UML to model SCD. Additionally the solutions that novices create lack integration.

FBS framework provides an integrated view of design. The individual F/B/S design elements correspond to the UML representations use case, class diagram and sequence diagram respectively. In a software engineering course, students learn about syntax, semantics and processes to create the formal (UML) representations (Medvidovic et al., 2002). The representations are presented as isolated views of the design solution. However to create cohesive solutions that fulfil requirements, i) designers need to utilize the various representations and ii) ensure that the different representations are integrated. Experts use heuristics techniques to do all of them together. Novices need to be explicitly trained to be able to do so.

Linking the FBS design elements correspond to the software design processes of requirement definition, implementation, assessment, analysis, testing (Kruchten, 2005). We propose a FBS based intervention, where the learners are scaffolded to identify and then link FBS design elements from a given software design problem. In the proposed intervention we manifest the FBS framework as a graph, where the F/B/S design elements form the nodes and links connect the FBS nodes. This FBS representation, which we call the FBS graph, is a visualization tool for learners to interact, create and evaluate the SCD of design problems. In the intervention we support the novices towards building the integrated models of the conceptual design via the FBS graph.

In the initial version of the interventions we provided novices with the FBS definitions, worked examples of FBS graphs for design problems and procedural information of creating UML representations from FBS graphs. We conducted lab studies with novices (study 2 & 3) to understand their difficulties. We analysed the FBS graphs and used thematic analysis (Braun & Clarke, 2017) to analyse participants' perceptions of difficulties. We found that novices need support to understand syntax and semantics of FBS graph. They also need scaffolds and prompts

to create a FBS graph. Study 1, 2 and 3 provided us with a set of features, scaffolds and task structure for the creating the FBS graph based intervention. These findings formed the requirements for the next cycle of design-based research (DBR).

The set of features, scaffolds and task structure were used to design a prototype of a technology enhanced learning environment. We employed heuristic evaluation (Nielsen, 1992) to identify usability problems and redesigned the user interface. This led to the learning environment named 'think & link'. 'think & link' incorporates the pedagogy of improvable models (Dasgupta, 2019) and consists of tasks at progressive planes of cognition – doing, evaluation and synthesis. The tasks are sequenced such that the learners are explicitly taken through all planes of cognition. There are three phases in 'think & link'. In the first phase the problem context (mood based music player), a non-editable FBS graph and a series of questions (activity) are provided to the learners to construct the FBS conceptual model. Followed by the second phase where the learners need to edit the FBS graph and evaluate it in the same problem context (mood based music player). The last phase in the system is for the learner to create a FBS graph in the new problem context set by them.

We evaluated 'think & link' in two field studies (study 4 & 5). In these field studies, we captured the pre-post solution artifacts, conceptions, perception of the SCD process and the learner interactions in the learning environment. We evaluated the pre-post solution artifacts based on design criteria by Eckerdal et al. (Eckerdal et al., 2006). Our findings indicate that participants shifted towards creating dynamic representations for SCD. From thematic analysis (Braun & Clarke, 2017) of the participant conceptions and perceptions of the SCD process we saw a shift in understanding of SCD and conceptual change (Vosniadu, 2007; Vosniadu, 2019) in participants' perception of processes of SCD.

The contributions of this thesis include (i) a detailed characterization of the novice software design processes; (ii) a FBS graph-based pedagogy for teaching-learning of integrated SCD; (iii) set of features and scaffolds necessary for teaching-learning of integrated SCD that supports novice design processes; and (iv) a teaching-learning environment for learners which also includes an instructor authoring interface.

Keywords: Software conceptual design, function-behaviour-structure design framework, novice software design processes, technology enhanced learning environment

Table of Contents

List of Figures

List of Tables

Abbreviations

SCD Software Conceptual Design

FBS Function-Behaviour-Structure

UML Unified Modeling Language

TEL Technology Enhanced Learning

TELE Technology Enhanced Learning Environment

DBR Design Based Research

RQ Research Question

DQ Design Question

LQ Literature Question

UI User Interface

Chapter 1

Introduction

1.1 Background and Motivation

One of the fundamental activities of engineering is design. Engineering graduates are expected to design solutions to solve real world problems. Design is central to engineering as a practice. According to IEE90 (IEEE. IEEE Standard Glossary of Software Engineering Terminology. IEE Std 610.12-1990, IEEE, 1990),

"Design is both the process of defining the architecture, components, interfaces, and other characteristics of a system or component and the result of that process."

Among the various phases in design, conceptual design is one of the initial phases. In engineering, conceptual design is defined as a phase in which – "*The functional requirements are elicited and schematic descriptions of solution are generated*" (Chakravarti & Bligh, 2001). Conceptual design is considered to be inherently hard and needs to be supported (Chakravarti & Bligh, 2001).

In, software engineering discipline design is considered as a pivotal activity (Pressman, 2005). Although there are many similarities with other design disciplines, the end product of design, software is abstract. This adds to the challenges to the activities and processes of design. Software conceptual design characteristics include "*description must be implementation- independent; it should be easy to understand; it should be precise enough to support objective analysis; and it should be lightweight, presenting little inertia to the exploration of different points in the design space*" (Jackson, 2013). It is a standard practice to create various representations of unified modeling language (UML) to represent a software conceptual design (Krutchen, 2005).

Professional software designers use various design strategies such as mixed breadth-depth approach (Ball et al., 2010) and cognitive processes like inductive reasoning (Tang et al., 2010). Experts use domain-specific knowledge as well as cognitive and metacognitive skills to solve complex and ill structured software design problems (Sonnentag et al., 2006) (Ball et al., 1997). In science and engineering (NGSS) often the term disciplinary practices is utilized to represent the expert

processes (National Research Council, 2012). The disciplinary practices in software conceptual design (SCD) evident from expert literature involves problem understanding and generating integrated solutions fulfilling all requirements (Ball et al. 2010). Problem understanding refers to the activities such as requirements analysis, defining goals, constraints, and stakeholders. By doing so, the experts extract the functionalities of the software. At the same time they also create integrated models of different views of solution in their mind (Petre, 2009; Hungerford et al., 2004). They are able to integrate the different UML representations. However novice studies in software design indicate, "*majority of graduating students cannot design a software system*" (Eckerdal et al., 2006).

In a software engineering course, students learn about syntax, semantics and processes to create the formal (UML) representations. However when students encounter open-ended real world problems they are unable (Eckerdal et al., 2006) to utilize the formal representations. So what are the novices design strategies and cognitive processes while doing SCD? How to foster disciplinary practices of integrated solution generation for SCD to undergraduate computer engineering students?

These questions are the motivation of this thesis:

To develop understanding of novice difficulties while creating SCD, and foster disciplinary practices of SCD.

1.2 Research Goal

Further in Software Engineering Body of Knowledge (SWEBOK), software design (Tremblay, 2001) is:

"Viewed as a process, software design is the activity, within the software development life cycle, where software requirements are analysed in order to produce a description of the internal structure and organization of the system that will serve as the basis for its construction."

Based on the definitions from design and specifically software design literature definitions, we have synthesized our definition as SCD being the process of analysing the requirements, and creating solution descriptions as representations that fulfil all the requirements. The outcome of SCD is multiple integrated representations

describing different aspects of the solution software. This description needs to aid implementation of the software.

Traditional teaching and learning of UML design is based on a combination of lectures for syntax/semantics and modeling tools (Akayama et al., 2013). However the current teaching and learning methods are unable to address the problems in students such as, inability to create basic UML representations for real world problems. Additionally, the underlying reasons for students unable to create SCD for real world problems is not well understood. We did not find literature relating to design processes and cognitive mechanisms of undergraduate engineering students with respect to SCD. An understanding of the underlying reasons is necessary to support the doing and learning of SCD.

The broad research problem guiding this thesis is -

Developing understanding of novice processes in software conceptual design and using the understanding to design a technology enhanced learning environment to support novices learning of SCD.

1.3 Solution Overview

1.3.1 Theoretical Basis

This work aims at supporting the teaching and learning of novices in creation of integrated software conceptual design. The theoretical foundation of the intervention is function-behaviour-structure (FBS) design framework (Gero & Kannengiesser, 2014). The FBS framework (Gero & Kannengieser, 2014) models designing in terms of three design elements: function (F), behaviour (B) and structure (S). Functions, describe what the design is for; behaviours, describe what it does; and structures, describe what it is. For example in a mood detection based music player, the functionality of mood detection using facial features corresponds to function (F). Emospark web camera, which detects mood based on facial features, corresponds to the structure (S). Extraction of facial features by the web-cam to detect and determine the user's mood corresponds to the behaviour (B). Along with FBS elements the framework has 2 sets of behaviours, expected behaviour (Be) and behaviour derived from structure (Bs). These elements are connected to each other by a set of transformation processes as depicted in Figure 1.1. Figure 1.1 also consists of

examples for each of the FBS design elements for the design problem of mood based music player. The set of processes as labelled include – (1) formulation which transforms functions to expected behaviours, (2) synthesis which maps expected behaviour to the structure, (3) analysis of structures which leads to generation of behaviours of structures, (4) evaluation of expected behaviour and behaviours extracted from structures, (5) documentation which contains the formal design description. There are three types of reformulation – (6) reformulation of structures, (7) reformulation of expected behaviour and (8) reformulation of functions, which are done to evolve the problem and solution together.

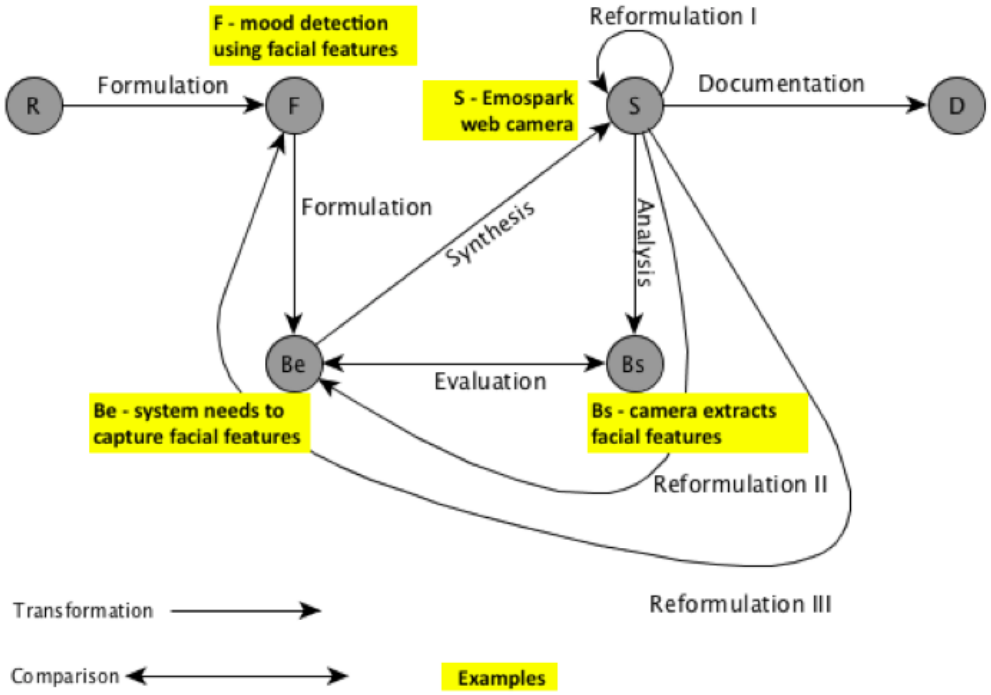


Figure 1.1 FBS design framework with examples from the design problem of 'mood based music player'

1.3.2 Solution Process

As described in the previous section the research goal is to develop understanding of novice processes in software conceptual design and design a technology enhanced learning environment to support integrated solution building for SCD. To achieve this we employed design-based research (Barab, 2014). The goal of DBR (sometimes also referred to as design experiments) is to use the close study of learning as it unfolds within a naturalistic context that contains theoretically inspired innovations, usually

that have passed through multiple iterations, to then develop new theories, artifacts, and practices that can be generalized to other schools and classrooms (Barab, 2014). The goal of DBR aligns with our research goals, as we want to examine novices' process of creating SCD and design /develop learning interventions to support SCD.

We followed a design-based research (DBR) methodology (Barab, 2014). To address the research goals we conducted two iterations of DBR. Our first goal was to unpack novice design strategies and cognitive processes. To identify these, we conducted an exploratory qualitative study (study 1) where novices were required to create conceptual design for a software design problem. The design artifacts and process of design were collated. We used the function-behaviour-structure (FBS) and conceptual design cognition lenses to analyse the design process and cognitive processes respectively. We found that novices were unsuccessful in creating SCD when they fixate to either one or all FBS elements initially. This prompted us to create teaching-learning interventions based on the FBS design framework. The FBS framework manifests as an FBS graph in our learning intervention. Example of a FBS graph for the 'mood based music player' is provided in Figure 1.2. The FBS graph in Figure 1.2 is made up of F/B/S elements as nodes and the connections between the F/B/S elements are established using the links. The FBS graph as a representation allows for creation/traversal from top-down or bottom up and connections made between any pair of dyads. CS undergraduate learners already are familiar with graphs as representation.

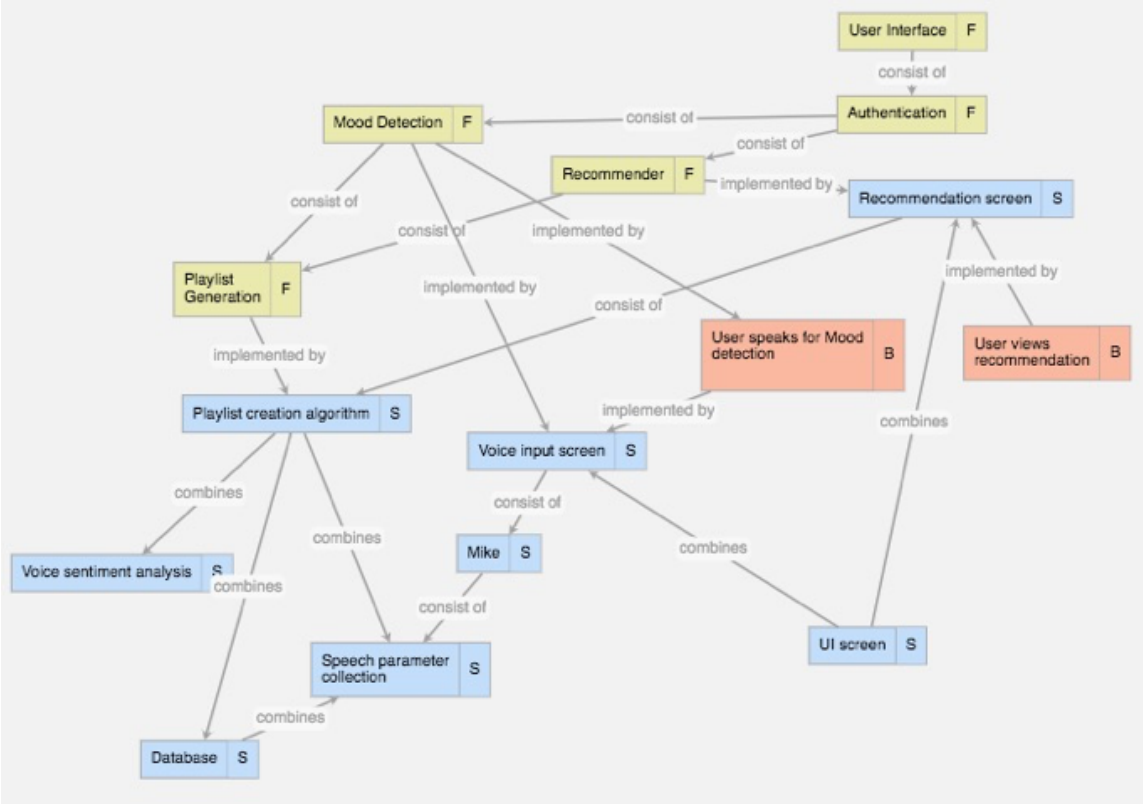


Figure 1.2 Sample function-behaviour-structure (FBS) graph for the design problem 'mood based music player'

In DBR cycle 1, we created preliminary interventions with the FBS graph. Study 2 and 3 were conducted as qualitative laboratory studies to unpack the novices' difficulties while learning using FBS based intervention. Insights from these studies helped us in identifying the features and supports that are required in FBS graph based intervention.

In DBR cycle 2, we revised our designs to come up with 'think & link'. 'think & link' is a web-based, self-learning FBS based learning environment. Workshops for SCD using 'think & link' were conducted in nearby engineering institutes (study 4 & 5) with undergraduate computer and information technology students. Both these studies had the research design of single-group pre-post. We examined participants' pre-post conceptual design using criteria for software design from literature (Eckerdal et al., 2006). We additionally collated and analysed pre-post open-ended answers to capture their perceptions about software conceptual design. 'think & link' records the participants' actions in the learning environment. We analysed these action logs to identify action sequences, which indicate the participants' usage of affordances in the learning environment.

1.3.2 'think & link' Pedagogy

Software design is a complex activity and highly ill-structured. Software designers are required to design solutions for a wide range of problems in diverse domains. The development environment is also highly dynamic and complex as requirements and technologies keep on changing. Rittel and Webber (Rittel & Webber, 1973) suggest that design has characteristics of 'wicked problem', which is that they do not have a well-defined set of potential solutions.

Teaching-learning efforts in software design need to be directed towards students being able to perform ill-structured tasks. Since designing a software system is an ill-structured problem, it requires additional knowledge and strategies apart from basic content knowledge. Students need to know how to apply relevant domain specific knowledge in the problem context in order to come up with effective software design solutions.

Essential characteristics of expertise in software design often include problem comprehension, planning, use of visualizations, and knowledge of strategies (Sonnentag, 1998). Literature suggests visualizations are a helpful cognitive tool in the solution development process (Sonnentag, 1998). Design involves a combination of complex cognitive processes as well as metacognitive strategies. While creating software design, experts are known to utilize the strategy of mixed breadth-depth (Ball et al., 2010). Professional software designers co-evolve the problem and solution implicitly during a design session (Tang et al., 2010).

UML diagrams are created during the SCD task. Each representation in UML corresponds to a view of the solution (Niepostyn & Bluemke, 2012). Experts have the ability to build integrated models of different views of solution in their mind (Petre, 2009; Hungerford et al., 2004). They are able to integrate the different representations. The integrated model building by combining the various representations can be referred to as disciplinary practice.

The FBS based pedagogy requires a representation through which learners can symbolize function, structure, behaviour and establish the relationship between them. The FBS framework manifests as a FBS graph in the intervention. The pedagogy includes creation and manipulation of a FBS graph for a given design problem. Among the various representations, graph was chosen as it allows for – i) creation/traversal from top-down to bottom up and ii) connections to be made between any pair of dyads. As the learners create the nodes and link the dyads the

appropriate design processes (see Figure 1.1) are triggered. In the intervention learners are provided scaffolds to create, modify and evaluate FBS graph. By creating an integrated representation of the FBS graph, learners would be able to create integrated solution designs.

The expert practices with respect to design strategies and integrated model building is incorporated in our pedagogy by using the FBS graph. From our studies (1, 2, & 3) we have identified difficulties of novices in SCD as well as FBS based intervention. These findings inform the design of task structure as well as scaffolds.

'think & link' (https://thinknlink.tech/) is a web-based self-learning environment for teaching - learning of software conceptual design based on FBS framework. In 'think & link' the FBS graph, is a visualization tool for learners to interact, create and evaluate the software conceptual design of problems. In 'think & link' we support novices towards building the integrated models of the conceptual design via the FBS graph.

'think & link' consists of learning tasks in three phases. In the first phase learners are provided a design problem context, a corresponding FBS graph and a series of questions. The learners are required to answer the questions by interacting with the FBS graph and build their conceptual understanding of FBS. In the second phase, learners edit the FBS graph to create their own version of the design solution. They are also required to evaluate the resulting FBS graph based on predetermined criteria of software conceptual design. The learners critique the example FBS graph and use it as the basis for incorporating their own ideas of design. In the last phase, learners create a FBS graph for a new problem context set by them. The tasks in each phase are interspersed with planning, evaluation and reflection tasks. The Figure 1.3 captures each of the phases along with the learning objective in each phase.

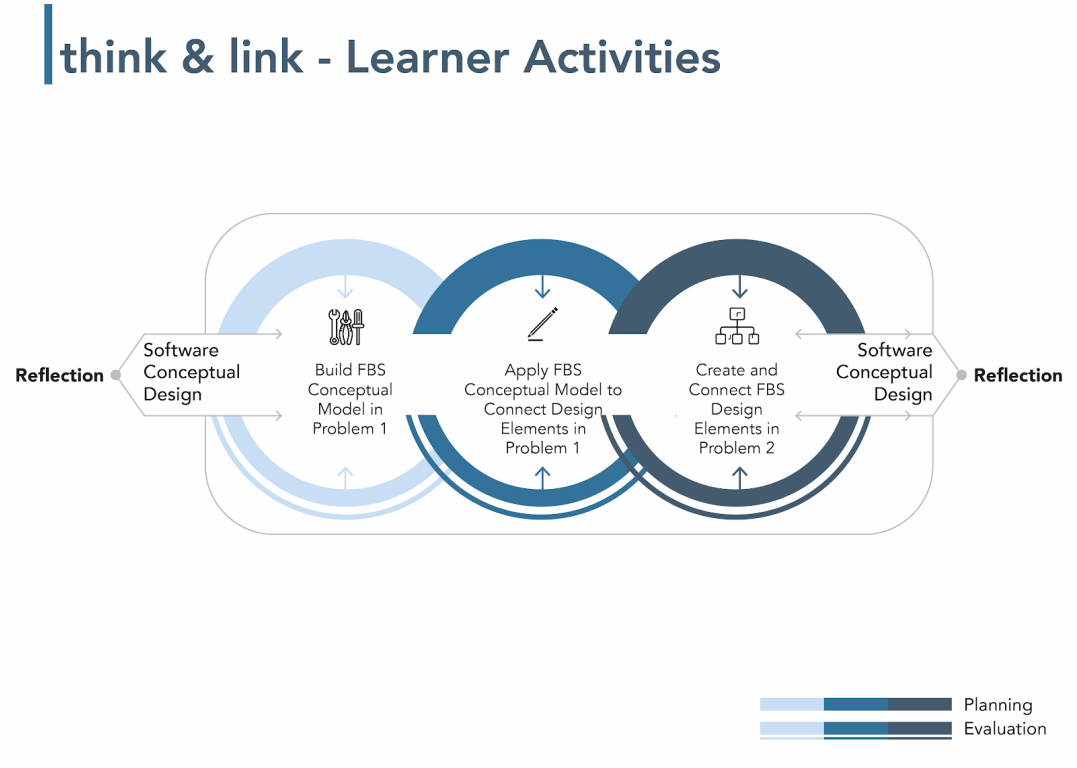


Figure 1.3 Learner activities and tasks in 'think & link'

1.4 Research Questions

Each iteration of DBR starts with requirements, on the basis of which design of an intervention is created. The intervention is then implemented in a learning setting, and data collected to evaluate the intervention. The data collected are then analysed which provide insights about the implementation of the intervention. The findings from the analysis form the basis of requirements for the next iteration of DBR. In this thesis we conducted two iterations of DBR as in Figure below.

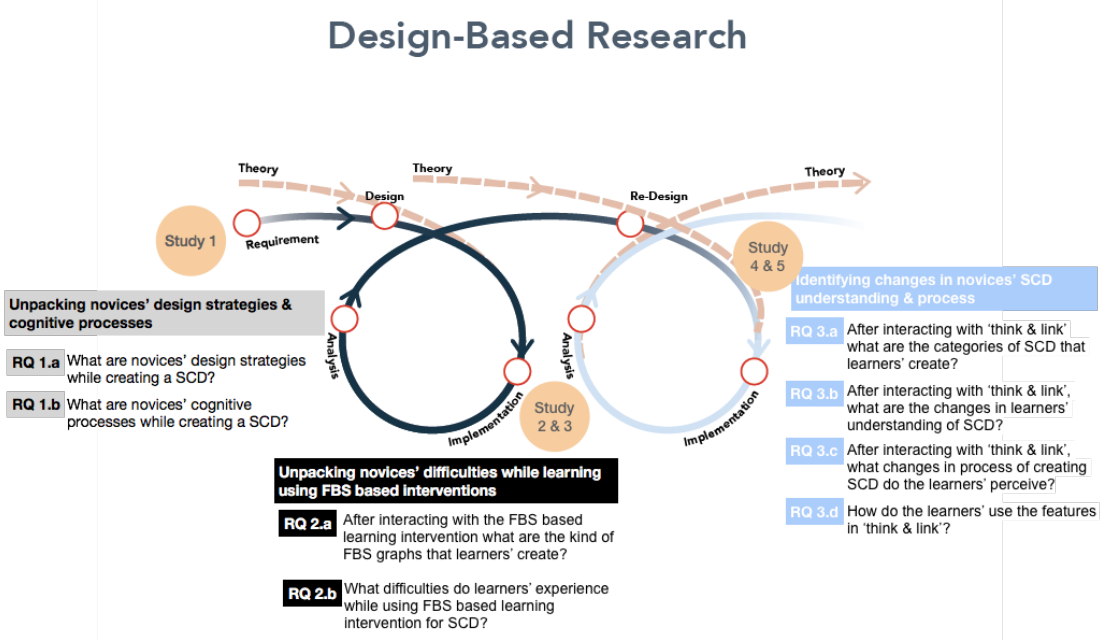


Figure 1.4 Design based research cycles in this thesis

The first iteration starts with the research goal of unpacking novices' design strategies and cognitive processes (Study 1). This led to the design of FBS based learning designs, which were then evaluated using laboratory studies (Study 2 & 3). The goal of these studies was to unpack novices' difficulties while learning using FBS based interventions. The findings of these studies were passed on to the next phase. In iteration 2 of DBR we designed and developed the intervention 'think & link'. This web-based learning environment was taken to novice undergraduate students as SCD workshops (Study 4 & 5). The goal of these studies was to identify the changes in novice understanding as well as design processes after using 'think & link'.

DBR 1- Problem Analysis: Understanding novice design strategies and cognitive processes

1. **Study 1**: Broad RQ - How do novices create software conceptual design?
    a. What are the design strategies that novices follow while creating a software conceptual design?
    b. What cognitive processes do novices use while creating software conceptual design?

We studied the novice processes using protocol analysis (Gero et al., 2011). The data was analysed using linkography (Kan & Gero, 2017) and deductive coding (Hyde, 2000) based on cognitive processes of conceptual design (Hay et al., 2017). The findings from study 1 helped us understand the novice design processes and

difficulties. The findings motivated us to create intervention based on the FBS framework to alleviate the difficulties.

DBR 1- Design and Evaluation: Initial solution designs and evaluation

2. **Study 2 & 3**: Broad RQ - How to support novices' while learning SCD in FBS based intervention?

a. After interacting with the FBS based learning intervention, what are the kinds of FBS graphs that learners create?

b. What difficulties do learners' experience while using FBS based learning designs for SCD?

In this cycle we built preliminary interventions using the FBS design framework. We studied the effect as well as difficulties of novices while using the preliminary interventions in laboratory studies. The FBS graph artifacts were evaluated using the adapted rubric of Lindland et al. (1994). The novice difficulties were extracted from inductive thematic analysis (Braun & Clarke, 2017). The findings from this cycle were used in the design of the tasks and activities in the FBS graph based intervention. These findings also helped us design features in the learning environment.

DBR 2 – Design and Evaluation of 'think & link'

3. **Study 4 & 5:** Broad RQ - What are the changes in novices' after using 'think & link'?

a. After interacting with 'think & link' what are the categories of SCD that learners' create?

b. After interacting with 'think & link', what are the changes in learners' understanding of SCD?

c. After interacting with 'think & link', what changes in the process of creating SCD do the learners' perceive?

d. How do the learners' use the features in 'think & link'?

In this cycle, the findings from the previous cycle were utilized to create a FBS design framework based learning environment, 'think & link'. 'think & link' is a self-learning web page, which consists of learner tasks, and activities to edit, create and evaluate FBS graph. In order to evaluate 'think & link' we conducted workshops at several engineering institutes where the undergraduate computer and information technology students participated in the study and used the learning environment. The research design was a single group pre-post test. We captured participants'

understanding of SCD pre-post via open-ended questionnaires. We also recorded retrospective interviews and focus group interviews of participants during and after the workshop. 'think & link' also logged the participants' actions in the learning environment. The pre-post design solutions were evaluated using criteria for software design (Eckerdal et al., 2006). The responses to open ended questionnaires and interviews were analysed by inductive thematic analysis (Braun & Clarke, 2017). We employed sequence mining to extract the sequence in which participants completed the task in 'think & link'.

1.4 Scope of thesis

Studies about design processes and teaching-learning interventions are intertwined with the context in which they take place. In this section the various aspects of the context in this thesis are described, starting with the design problems, participants, learning conditions and technology.

1.4.1 Design Problems

The four design problems in the Table 1.1 are the ones that have been used in this thesis. The four design problems have been chosen based on the familiarity of software systems usage among the students. For example the systems such as ATM, payment authentication is familiar to participants as they encounter such systems in their day-to-day lives. These four problems were selected, as the students would be familiar in terms of usage, at least partially, to the software systems. In these problems the functional specifications are open-ended, and a part of the problem (ATM, payment systems, recommender system, music player) gives indication for the functional decomposition. The indications for functional decomposition make the design problem tractable for novices. Open-ended in this thesis means that no requirements were provided to the students. Students had to assume the requirements from the problem and solve the problem.

As we have different design problems that the students worked on, it is important to establish similarity among the problems so that we can evaluate the SCD and compare the design strategy among the participants. However, the problems were given to an expert software designer, who is an instructor as well. According to the expert, the problems are equally matched in terms of complexity, time taken to solve, and amount of code that needs to be written. They are in between the innovative and creative design problem category (Brown & Chandrasekaran, 2014).

Table 1.1 Software design problems used in this thesis

Sno #	Design Problems	Study 1	Study 2	Study 3	Study 4
1	Design a fingerprint ATM system	✓	✓	✓	✓
2	Design a mood based automatic music player	✓	✓	✓	✓
3	Design a fingerprint based payment system	✓	-	-	-
4	Design a cooking recipe recommender system	✓	-	-	-

1.4.2 Participants

Participants, learners, and novices are interchangeably used in the thesis. The typical representation characteristics of the novices' are that they are computer or information technology engineering students from any Indian engineering institute. Students in their third year and above would have the appropriate domain knowledge and exposure. This criterion was chosen as such students were exposed to courses such as 'Structured Object Oriented Analysis and Design' (semester 5) and 'Software Engineering' (semester 6) as a part of their engineering curriculum. These two courses cover topics of software design approaches, software-modeling tools, characteristics of software solution etc. As the course contents included such concepts, it was appropriate to consider that they had prerequisite knowledge for the activity. However second year students can also participate, provided they are exposed to UML modeling concepts and tools. All participants volunteered for the research study. Informed consent from all participants was obtained before the beginning of studies. No participation fee was provided, however, certificates were provided for all participating students. The objective was to obtain a typical representation of learners from the age group (19-22) with appropriate domain exposure. The participants are representative of Indian urban engineering students, proficient in English as a medium of instruction and communication.

1.4.3 Learning conditions

The interventions for teaching-learning SCD is designed for self-learning. It is intended as a supplementary learning activity for the course software engineering. As the course commences, 'think & link' activities can be performed during laboratory hours. For final year engineering students, 'think & link' can be part of their final year project activities.

1.4.4 Technology

'think & link' is developed in HTML-CSS, Javascript, and PHP with backend in MySQL. It is currently designed for desktop and laptop users. Users would require internet access to use the learning environment.

1.5 Contributions

With the context as described in section 1.4, we conducted the research studies 1 to 5. The overarching research goals of this thesis are to develop understanding of the novice design process and support the learning of SCD. In this section we highlight the contributions of this thesis

- Theoretical understanding of novice difficulties - Towards the theory of novice software design practices, for the researchers in computing education research, learning scientists and design education this thesis identifies-
  1. novice design strategies and cognitive processes in SCD
  2. novice difficulties while learning from FBS graph based intervention.
  3. the scaffolds required for novices to perform SCD.
- Pedagogy - Towards the pedagogy and learning design for software conceptual design, for the instructional designers and software engineering educators this thesis presents-
  1. pedagogical design of a FBS based learning environment for teaching and learning of software conceptual design
  2. a set of features and scaffolds for novices teaching-learning of FBS based software conceptual design
- Learning environment development - For software engineering students and software engineering educators we have the web-based learning environment 'think & link'. 'think & link' is an instantiation of the FBS based pedagogy. It helps learners to create integrated multiple representations by thinking in terms of FBS for a given design problem context. We have provided a teacher-authoring

tool for different FBS graph contexts. 'think & link' is available online in this link - https://thinknlink.tech/ . To access 'think & link' student interface create a login id or use this credential: user id – Prathiksha, password –seokjin. To access the teacher interface use this credential: user id – etiitb, password – thinknlink2019.

1.6 Structure of Thesis

The Figure 1.5 below summarises the chapters and their structure in this thesis. In this chapter, we presented our primary research objective, motivation, broad research questions and scope of work. The second chapter presents an overview of the related work and background literature. In the third chapter we present the overall research methodology and the research questions roadmap. In the chapters 4, 5, 6 and 7 we present the details of research studies with their corresponding results and findings. Chapter 8 we discuss the results from the lens of conceptual change. Chapter 9 provides the summary of the findings and discusses them with respect to the research goals. Chapter 10 summarizes the contribution of this thesis along with future work.

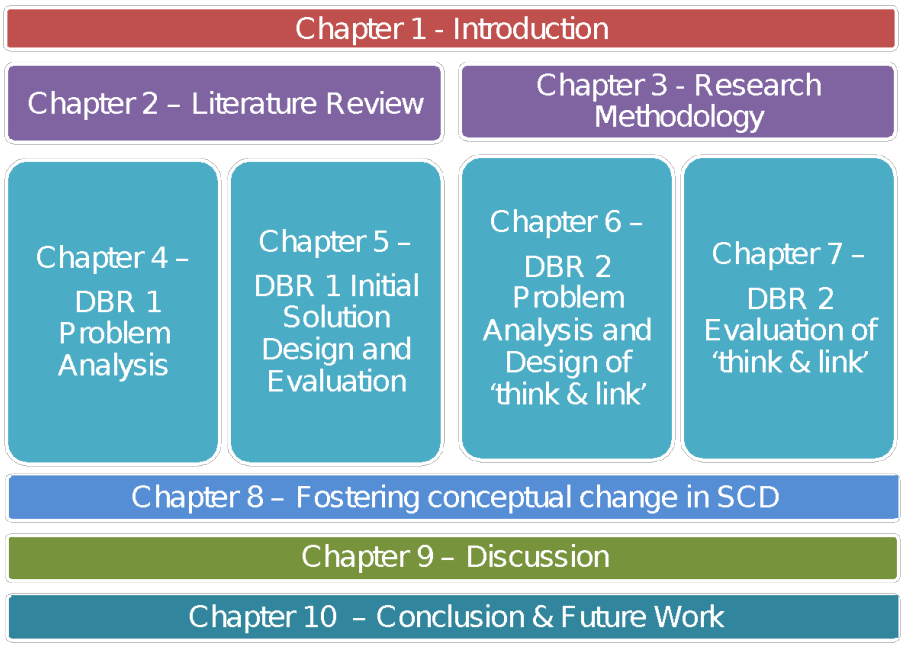


Figure 1.5 Chapters in this thesis

Chapter 2

Review of Literature

2.1 Organization of Literature Review

In this chapter we review the literature related to software conceptual design. To understand about software conceptual design, we have referred to the literature in design, engineering design and software design. The Figure 2.1 captures the parent disciplines in this thesis.

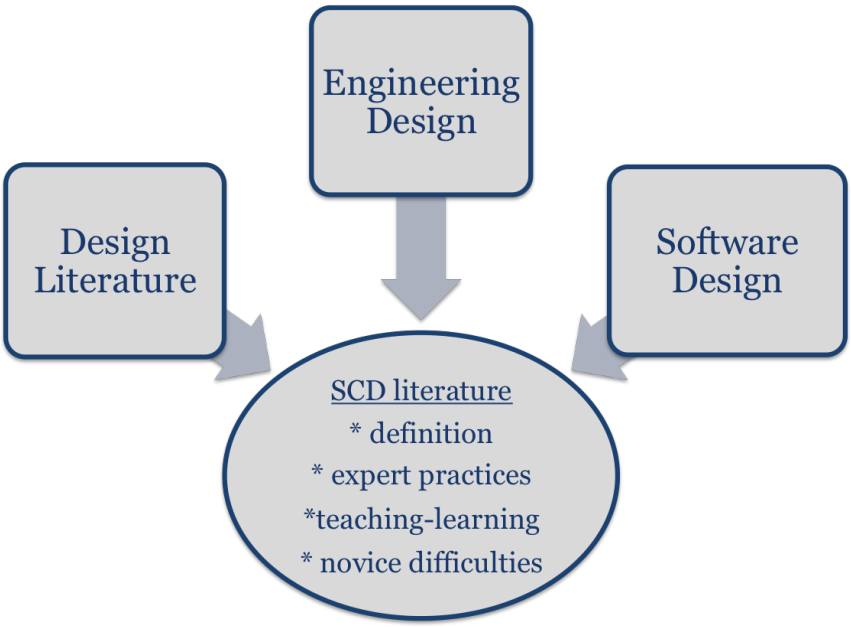


Figure 2.1 Parent disciplines in this thesis

The design literature helps us scope the definition of conceptual design, providing us with gainful insights on cognitive processes involved in conceptual design (Hay et al., 2017) and the various frameworks involved in the conceptual design. This corresponds to section 2.2, 2.3 and 2.6. The software design literature provides us with insights on the quality parameters of SCD, expert design strategies in SCD, teaching and learning of SCD and novice difficulties reported in SCD. They correspond to section 2.2, 2.3, 2.4, and 2.5. The details about how each parent discipline contributes to the sections in this chapter are presented in Figure 2.2.

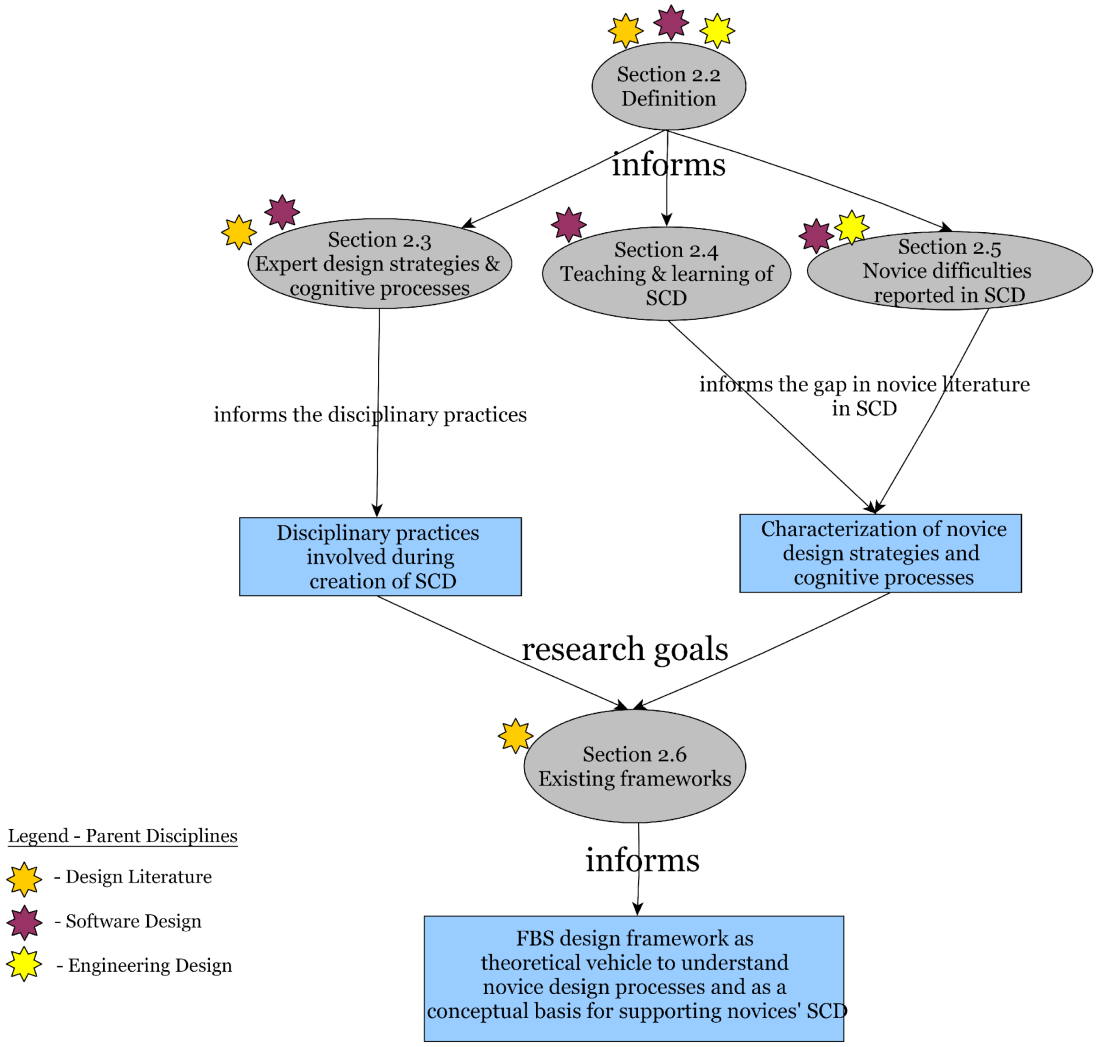


Figure 2.2 Organization of literature review

2.2 What is software conceptual design?

Conceptual design is defined as a phase in which – "The functional requirements are elicited and schematic descriptions of solution are generated" (Chakrabarti & Bligh, 2001). Software conceptual design (Jackson, 2013) has the following characteristics:

- description which is implementation independent
- support analysis
- support exploration of design spaces

Conceptual design is defined in many domains as captured in the Table 2.1 below:

Table 2.1 Definition of conceptual design in various domains

Sno	Domain	Definition
1	Engineering Design	Conceptual design commences with high-level description of requirements and proceeds with a high level description of solution (Mc Niell et al., 1998)
		Conceptual design is a phase in the process of designing, when solution principles are developed to meet the desired functions (Pahl & Beitz, 2013)
		Conceptual design is a creative process (Jill & Benami, 2010)
2	Product Design	In this part of the design process, designers try to understand the underlying design problem and then generate some initial solutions. Conceptual design can also be a very confusing process since there is little concrete information available to designers. (Masur & Salustri, 2007)
		Relatively ambiguous stages of the design process known as conceptual design (Hay et al., 2017)
		How functional requirements of a design problem are transformed into schematic descriptions of design solution concepts (Chakrabarti & Bligh, 2001)
3	Software Engineering	It is a description must be implementation-independent; it should be easy to understand; it should be precise enough to support objective analysis; and it should be lightweight, presenting little inertia to the exploration of different points in the design space. (Jackson, 2013)

From the Table 2.1, we have synthesized the definition as - *A conceptual design conveys the functionality and working of the system*. There are various quality parameters (Lindland et al., 1994) of conceptual design such as complete, consistent, formal, modifiable, testable, traceable, and understandable. Lindland et al. (1994) compiled these properties from existing frameworks in engineering terminology. The parameter complete refers to '*everything that the software is supposed to do is included in the solution*.' Consistency refers to the degree of consonance in the solution details. Formal parameter refers to the degree of domain specific formal language used in the solution description. Modifiable refers to the degree to which changes can be made to the solution description such that the completeness and consistency properties of the solution are not hampered. Traceable and testable refer to the ease of referencing requirements and verifying solution design. The parameter understandable refers to comprehension of the solution by non-computer specialists.

Conceptual design is a critical step in design (Pahl & Beitz, 2013) and an important phase (Dym et al., 2005; Chakrabarti & Bligh, 2001). Around 60% of the total product cost is fixed at the conceptual design phase (Trends in Concept Design, 2011). This phase is important as: i) the problem as well as the solution domain co-evolve in this phase (Suwa et al., 2000), ii) problem scoping happens at this phase (Jiang & Yen, 2013), iii) different phases of the design process are highly interconnected, as it is the first phase the results of conceptual design affect all the remaining phases (Pahl & Beitz, 2013). Peculiar characteristics of software design, such as dynamicity and intangibility makes this activity more challenging. Conceptual design is inherently hard and needs to be supported (Chakrabarti & Bligh, 2001).

Considering the conceptual design as software product, we could follow the ISO 9126 standard "Software Product Evaluation-Quality Characteristics and guidelines for their use" and its characteristics functionality, efficiency, maintainability, portability, usability and reliability and all their sub characteristics (ISO 2001). However software conceptual design is not a complete software product as the problem and solution are still evolving at this stage. If we consider a conceptual model as a software requirement specification (SRS), we could apply the quality properties defined by Davis et al. (Davis et al., 1993) or the International Standard ISO 830 (IEEE, 1998). Davis et al., (Davis et al., 1993) has defined 24 qualities that SRSs should exhibit (see Table 2.2). From the Table we see that the parameters in

SRS overlap with the parameters discussed above in conceptual design by Lindland et al (1994). The SRS adds the parameters persistent storage, annotation and reusability.

Table 2.2 Software Requirement Specification (SRS) document quality parameters (Davis et al., 1993)

1. Unambiguous	13. Electronically stored
2. Complete	14. Executable/Interpretable
3. Correct	15. Annotated by relative importance
4. Understandable	16. Annotated by relative stability
5. Verifiable	17. Annotated by version
6. Internally consistent	18. Not redundant
7. Externally consistent	19. At right level of detail
8. Achievable	20. Precise
9. Concise	21. Reusable
10. Design independent	22. Traced
11. Traceable	23. Organized
12. Modifiable	24. Cross-referenced

In the domain of software, there are many other proposals for measuring quality in conceptual designs although from different views of software conceptual design. They are presented in Table 2.3.

Table 2.3 Quality parameters in SCD

Literature	Qualities	View of SCD
Moody and Shanks (Mood & Shanks, 1994) and Moody et al. (Moody et al., 1998)	completeness, integrity, flexibility, understand ability, correctness, simplicity, integration, and implement ability	Data Model (Database design, ER models)
Olivé, A. (Olive, 2000)	Completeness, correctness, principle of conceptualization (design independent conceptual schema), syntactically	Conceptual Modeling of Information Systems

	valid, simplicity, ease of understanding, and stability (flexibility, extensibility, modifiability)	

However, many of the quality parameters presented in the above Table 2.3 cannot be adopted, as software conceptual design is not a complete product. At this stage the problem and the solution are co-evolving. However, the goals of the problem, which are the requirements, need to be fulfilled by conceptual design. At the same time, the means by which the goals are fulfilled need to be logically coherent.

A functional requirement specifies a function that a system or system component (i.e., software) must be capable of performing (Brackett, 1990). They can be stated from a static and dynamic perspective. Static perspective describes the functions performed by each component, whereas dynamic perspective describes the internal working of the system. This corresponds to 'complete' in the Lindland et al (1994) framework. The SCD solution description needs to include all the details that address both the static and dynamic functional requirements.

In the solution, there are various components that address different functional requirements. The components need to be compatible, so that the solution is well integrated. This corresponds to consistency in the Lindland et al (1994) framework.

So the parameters that are considered for quality of SCD are – (i) fulfils functional requirements, (ii) logically cohesive solution parts. These are the two parameters that we have scoped in this thesis for SCD. In the next section we discuss the SCD practices of experts reported in the software design literature.

2.3 What are the design strategies and cognitive processes involved in SCD?

Ball et al (Ball et al., 2010) report that several research studies document: i) the characteristics of experts in various design domains and ii) the multiple approaches followed to study expertise in design. As software design experience can be application domain dependent, it is different from other engineering design disciplines where the context of the domain is relatively constant (Tang et al., 2010). For instance, the issues faced by software designers working in the scientific domain are quite different to those working in the transactional financial system domain. Expert

software designers need to learn the domain and create software that addresses the problem. So, software designers have abilities that are domain specific at the same time general problem solving abilities.

Cognitive processes refer to the internal mechanisms or processes that transform or operate on mental representations (Hay et al., 2017). Mental representations refer to concepts or mental entities that stand in relation to physical entities. On the other hand the term strategy in oxford dictionary indicates 'a plan that is intended to achieve a particular purpose'. A design strategy describes the general plan of action and the sequence of particular activities i.e. tactics, design methods to arrive at a design (Gero & Kannengiesser, 2014). A design strategy needs to provide a framework for intended actions to operate and control to adapt the actions according to the responses (Gero & Kannengiesser, 2014). Research in cognitive processes of experts is also undertaken to understand the role of cognitive processes in design (Suwa et al., 2000). The research studies conducted however varied in the way design itself is looked at – linear search process or iterative exploratory process. The shared ontology by Hay et al (Hay et al., 2017) provides a generic classification of cognitive processes involved in conceptual design.

Expert heuristics have underlying cognitive processes, which have become implicit to them. Expert designers first approach a software design problem by coming up with an intermediate solution containing a breadth of features addressing several requirements. The designers then focus on realizing each feature in the intermediate solution. While doing so they seem to shift towards the depth of each feature. Overall findings report a use of mixed breadth -first and depth-first solution development with switch to depth-first strategically (Ball et al., 2010). It is considered that professional software designers co-evolve the problem and solution implicitly during a design session (Tang et al., 2010). Studies also point out that reasoning techniques, such as appropriate contextualization of design problems, explicit communication of design reasoning, explicit design reasoning and the use of inductive reasoning contribute to the effectiveness of software design (Tang et al., 2010). Tang et al. (2010) describe 'design reasoning' as the ability to fulfil the requirements at the same time be cognizant about the consistency of the whole solution. Tang et al. (2010) refer to many inductive reasoning techniques that software designers' use such as analogical reasoning, scenario-based reasoning to name a few.

In engineering education literature, there are many studies that examine the novice-expert engineering design differences. Jonnasen (Jonassen, 2000) points out that solving a design problem requires designers to structure the problem. The experts spend a considerable time in problem analysis and structuring the problem rather than jumping to the solution. There have been various studies on how designers' design and reports of expert –novice (E, N) or senior-junior (S, J) study in design. Many of them report differences in the design processes between the groups. The Table 2.4 collates the various studies and the findings thereof. In Table 2.4 the first column lists the design process that the studies have focussed on. The last column indicates the difference in the amount of time spent between the cohorts. From the Table (2.4) we see that experts spend more time than novices in problem analysis, design iterations, solution evaluation and regulation of cognition.

Table 2.4 Collating the various study findings from expert-novice (E, N) and senior-junior (S, J) in design

Design Process	Study	Difference	
		Group	Time spent in design process
Problem Analysis	(Mathias, 1995)	Expert (E)-Novice (N)	N<E
Information Gathering	(Cross et al., 1994)	Senior (S) – Junior (J)	J < S
Design Iterations (composition and construction)	(Seitamaa-Hakkarainen & Hakkarainen, 2001)	Expert (E)-Novice (N)	N<E
Regulation of Cognition	(Kavali & Gero, 2002)	Expert (E)-Novice (N)	N did not exhibit

Solution Evaluation	(Ahmed et al., 2003)	Expert (E)-Novice (N)	N<E
Problem Scoping and Alternate Solutions Development	(Atman et al., 2005)	Senior (S) – Junior (J)	J<S

2.4 How does teaching and learning of SCD happen?

Literature points to two approaches in teaching and learning of software engineering design – (i) pedagogical and (ii) tool based.

(i) Pedagogical approaches: Some of the teaching and learning approaches in software design are project based learning (Teel et al., 2012), game development, problem based learning (Schlling & Sebern, 2013; Abelson & Greenspun, 2001; Shin et al., 2014), simulation based approach (Oh, 2002), and collaborative games (Monslave et al., 2014). The pedagogical approach can be broadly categorised into three- realism, topical, and simulation (Ellis, 2008).

The realism approach focuses on giving real world context for the students while learning software engineering. It includes industry participation (Beckman et al., 1997; Wohlin & Regnell, 1999; Kornecki et al., 2003), emphasizing non-technical skills such as marketing, project management (Gnatz et al., 2003; Goold & Horan, 2002), and teamwork (Navarro & Van Der Hoek, 2005).

Topical approaches aim to educate students in detail about a topic generally not covered in depth in mainstream textbooks and lectures. These approaches do not focus on specific delivery methods, but instead focus on the mere addition of the topic as a crucial component of an effective and complete education in software engineering. Some examples of such topics are formal methods (Abernathy et al., 2000), real-time software engineering (Kornecki, 2000), and specific software processes such as the rational unified process (Halling et al., 2002).

Finally, simulation approaches are those that have students practice software engineering processes in a (usually) computer-based simulated environment. Within the realm of software engineering simulations, there are three main types: industrial simulations brought to the classroom (Collofello, 2000; Pfahl et al., 2001), game-based simulations (Drappa & Ludewig, 2000; Navarro & Van Der Hoek, 2005), and group process simulations (Nulden & Scheepers, 2000; Stevens, 1989). The

simulation approaches add realism to the learning environment in different ways. Industrial simulations add real project data in the simulation model; game-based simulation adds realistic game scenarios; group process simulation adds characters that behave like real-world participants.

The 'learning context' is the focus of the pedagogical approach. Each of the above approaches the learner is either placed in a real-life context, or real-life like context. Additionally newer practices of the software engineering domain are presented as topics to the learner thereby enriching the context. There is a lack of learner as the focus in the pedagogic approaches discussed above. The next approach we discuss is the tool-based approach.

(ii) Tool- based approaches: In the engineering domain, for conceptual design there exists various tools and formal notations such as Causal Functional Representational Language (Iwasaki et al., 1993), Kritik (Bhatt et al., 1994), Schemebuilder (Bracewell & Sharpe, 1996), FBS modeller (Umeda et al., 1996), Idea Inspire (Chakrabarti et al, 2017). The details about each of these tools are omitted, as they are not central to the theses. However, when we examine the tools we find that they are either deeply rooted in the domain (e.g., bond graph, case based reasoning) or have a very steep learning curve of representation (e.g. CFRL).

In the software domain, hundreds of industry-grade software engineering tools are introduced each year (Pressman, 2005). The most comprehensive tools packages from software engineering environments also known as Integrated Development environments (IDE) that integrate a collection of individual tools around a central database (repository). The tools are focused to manage complexity, process models, and coordinate global teams e.g. GENESIS (a generalized, open-source environment designed to support collaborative software engineering work). The other area in which tools for software engineering are towards specific design approaches like architecture based design, aspect oriented software development or model driven software development. Tools environments will respond to a growing need for communication and collaboration and at the same time integrate domain-specific solutions that may change the nature of current software engineering tasks.

The tools specific in the context of conceptual design for novices that alleviate their difficulties have not been built. Similar to the pedagogical approach the tool-based approach the tool-based approach does not place the learner at the centre.

Additionally, software design teaching and learning approaches has been directed towards software engineering methodologies and processes. In the above teaching and learning and tool based interventions most of them are targeted at requirements analysis, project management, development & design methodology, modularity, documentation, non-technical knowledge, and skills.

Despite the various teaching and learning efforts software design and programming is still challenging for students. The teaching and learning efforts in software engineering and software design need also to be directed towards students being able to perform ill-structured tasks (Moritz et al., 2005). The teaching and learning efforts are also not directed towards finding and alleviating learner difficulties. However teaching and learning tools for specific novice difficulties for software conceptual design is still not available.

2.5 What are the difficulties that novices encounter in SCD?

In this section we bring in the literature about novice design strategies and cognitive processes in general engineering design as well as software design. From the general engineering domain literature it is reported that novices use depth-first (Ahmed et al., 2003; Hokanson, 2001), treat all issues in design equally (Pan et al., 2010), tend to be data gatherers (Vishwanathan & Linsey, 2013; Jansson & Smith, 1991). Novice designers carry out several activities that are classified as a thought or an action rather than a design strategy (Pan et al., 2010). Novices fail to employ specific design strategies and follow a pattern of 'trial and error' (Chrysikou & Weisberg, 2005). It is reported that having no strategy could also be a random search strategy (Hokanson, 2001). Novices have been reported to have difficulty in starting the design and generating ideas/solution concepts (Pan et al., 2010). Similar findings have been reported where student designers reported 'getting ideas and refining them is the hardest part' (Hokanson, 2001).

Design fixation is reported as a major deterrent to the idea generation activity (Vishwanathan & Linsey, 2013). Design fixation entails the blind adherence of designers to their initial ideas or presented examples (Jansson & Smith, 1991). Design fixation can play a counter productive role as it limits the solution space. However as engineering design is an open-ended and ill-structured problem, it requires the creation and evaluation of multiple designs. Design fixation in novices may occur due to incomplete mental models (Vishwanathan & Linsey, 2013). Incomplete mental

models refer to the expertise in a particular design context. Novices tend to have limited expertise and hence the incomplete mental models. The other fixation that occurs with novices is to constrain their ideas to variations of their initial concepts (Kiriyama & Yamamoto, 1998). Literature also points to the difference in memory retrieval strategy (Vishwanathan & Linsey, 2013) between experts and novices. It is said that experts perform better memory retrieval, as they tend to analyse the problem, employ cognitive processes and create representations for solving the design problem. To understand design fixation among novices the actions, strategies and processes that novices follow needs to be studied meticulously.

In the context of software design, the ‘‘Scaffolding’’ experiment, a multinational, multi-institutional project looked at the approach that undergraduate students take to design software (Eckerdal et al., 2006). In this study, the authors had given novice designers the task of designing a super alarm clock. The results pointed out that ~41% designs that the novices created only added an insignificant amount of detail as design and created unimplementable design content. The authors noted that the novices were not able to create designs that had overview of parts and relationship between parts. The study concluded with a broad result that graduating students couldn’t design a software system (Eckerdal et al., 2006).

In the computer-engineering curriculum SCD is taught as courses object oriented analysis and design, software engineering in the third year undergraduate level. In these courses, unified modeling language (UML) notations are a standard representation mechanism (Medvidovic et al., 2002). Students are unable to utilize the formal representation mechanism and connect them to create a software design. In a study conducted by Chren et al. that documented the students’ mistakes in UML diagrams (Chren et al., 2019), it was reported that students had difficulties in diagrams such as state machines, which required them to integrate the different views and details across multiple diagrams.

To support novices in the modelling tasks as well as designing quality software design it is necessary that we start by understanding novices’ design processes. To unpack the difficulties, unearth patterns in their actions and the associated cognitive processes, it is necessary to study novices’ SCD task. A study of this nature would help us identify (i) learner’s prior knowledge and experience, (ii) intuition and sense making resources, which can be recruited, in formal education (Levy & Wilensky, 2008) (iii) skills or competencies that need further development.

Literature suggests that learning is more likely to lead to a change in practice if a needs assessment has been conducted (Marshall & Pennington, 2009). So, novice design studies need to be more focused in understanding their design approach, difficulties and pedagogical needs. This leads to the next question on what theoretical framework and support we need to examine novices' SCD. This question we attempt to answer in the next section.

2.6 What are the theoretical frameworks to examine and guide SCD?

From the SCD definition in section 2.2, we see that doing design often involves formulating the problem, analysing requirements, making decisions that impact the solution design. In this thesis we have the dual goals of understanding how novices create SCD and supporting their process to help them create integrated SCD.

The way we view design itself would affect the process we choose to study novice's design processes and support them. *"We consider design to be creation and manipulation of representations with available external tools to fulfil the set of requirements with the provided criteria."* This view aligns with the distributed cognition theory (Hutchins & Klausen, 1996). The theory of distributed cognition suggests that cognition involves the interaction with the external environment. External representations are integral to the distributed cognition theory as they not only offload cognition but also allow for manipulations in shared representation space (Kirsh, 2010). Software design domain comes with its own set of formal representation mechanisms like UML diagrams (Pressman, 2005). However, Petre (2013) suggests that it is not quite used as a design tool in practice.

Many of the frameworks reported in literature to study novice processes in the context of engineering design and software designing view the design process in terms of ill-structured problem solving steps. For example, to document engineering student design processes, Atman and Bursic (1998) used the scheme of problem definition; gather information, modelling, feasibility analysis, etc. In other studies by Hughes and Parkes (2003), while examining software engineering they have grouped the actions of designers related to the activities such as requirement analysis, design meetings, debugging, re-engineering, maintenance and review.

The need for a common framework, which can be utilized to study and compare the different analyses, emerged. In the design literature the FBS design framework (Gero & Kannengeiser, 2014), has been used in various design domains

including architectural, mechanical, software and business process design (Gero & Kannengeiser, 2014a). This led us to take the FBS design framework as a lens to study novice design processes.

The next goal of the thesis is to support novices to create SCD. In section 2.2, we defined SCD to utilize formal representations. It is a standard practice to create models of solution using the unified modelling language (UML) representations (Medvidovic et al., 2002). The representations in UML belong to a specific view. For example the use case represents the stakeholder view, the class diagram represents the components and the properties of them. The sequence diagram represents the interactions between the components and their properties. The need for an integrated representation (Niepostyn & Bluemke, 2012) was recognized. So to support novices to create integrated SCD, we need to provide them tools to integrate the representations.

The different UML representations such as use-case, class, sequence diagrams depict the functional, structural and behavioural aspects of the software solution. The FBS design framework integrates the various representations in UML. Additionally Galle (2009) points out that the FBS design framework can be utilized to design tools to assist practising designers. So we propose to use the FBS design framework to address both the goals of this thesis. Before we proceed to the solution features, we would need to define the terms function, structure and behaviour (FBS). We use the definition of FBS provided by Gero and Kannengeiser (2014). In the next section, we detail the design framework elaborately.

2.7 Examining and guiding activities of SCD using FBS design framework

The FBS design framework (Gero & Kannengeiser, 2014) models designing in terms of three design elements: function, behaviour and structure. Each of the design elements is defined as below:

- Function is the teleology (purpose) of the design artefact (`what the artefact is for')
- Structure is defined as its components and their relationships in design (`what the artefact consists of ').
- Behaviour is defined as the artefact's attributes that can be derived from its structure (`what the artefact does').

This is based on the idea that all designs can be represented in terms of: functions, which describe what the design is for; behaviours, which describe what it does; and structures, which describe what it is (Gero & Kannengeiser, 2014). In this framework the goal of designing is to transform a set of functions into a set of design descriptions (D). The function (F) of a designed object is defined as the purpose; the behaviour (B) of that object is either derived (Bs) or expected (Be) from the structure, where structure (S) represents the components and their relationships. The Figure 2.3 is repeated here to represent the series of transformations between the FBS elements. Humans construct connections between function, behaviour and structure through experience and through the development of causal models based on interactions with the artefact.

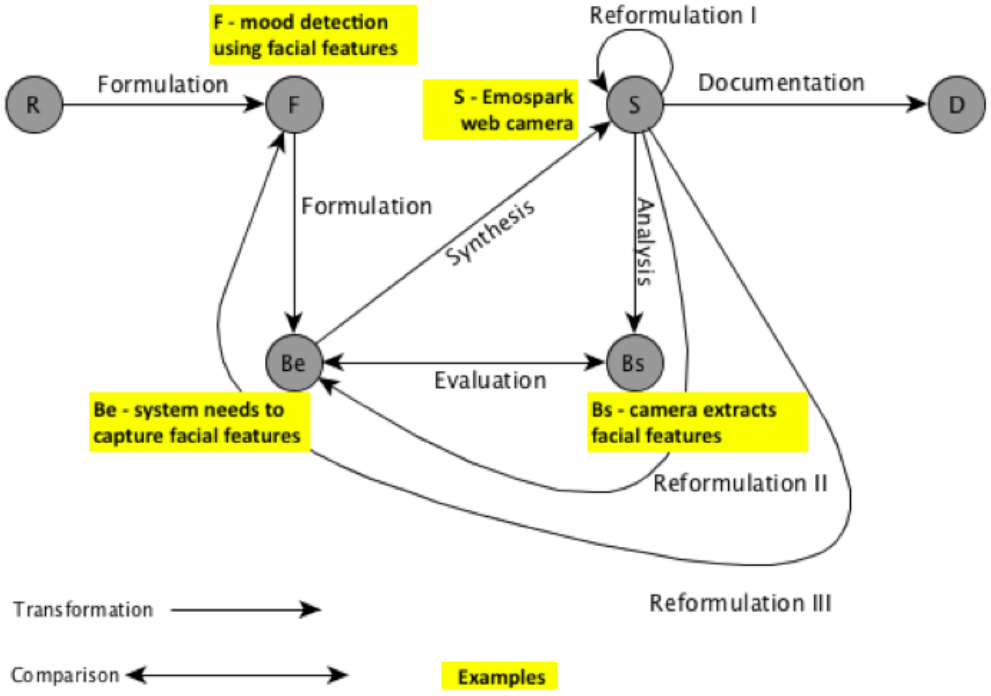


Figure 2.3 Function-behaviour-structure design framework with examples from the design problem of 'mood based music player'

A design description is never transformed directly from the function but undergoes a series of processes among the FBS design elements. These processes include: formulation which transform functions into a set of expected behaviours; synthesis, wherein a structure is proposed to fulfil the expected behaviours; an analysis of the structure produces derived behaviour; an evaluation process acts between the expected behaviour and the behaviour derived from structure; and

documentation, which produces the design description. As seen in Figure 2.3, there are three types of reformulation: reformulation of structure, reformulation of expected behaviour and reformulation of function. The structure, behaviour and function all are part of the reformulations. Reformulation of function changes or redefines the design problem. The reformulation processes expand the problem space and solution space. The Table 2.4 details some examples of function, behaviour and structure for different design problems.

Table 2.5 Examples of elements of FBS design framework

Domain	Problem	Function	Structure	Behaviour
Software Design	Design a fingerprint ATM system	FP scanning	• FP scanner (H/w) • FP scanning modules to capture & store	• Finger placed • Scanner begins • Scan stored
	Design a mood based music player	Facial scanning	• Camera (hardware) • Facial recognition algorithm • Database to store facial features	• User stands near the camera • Presses the scan button • System records the face • Facial features / points are extracted • Classification of the feature to the mood based on algorithm

Biology	Design artificial lungs	Breathing	• Lungs • Windpipe • Diaphragm • Nose • Mouth	• Diaphragm raises • Increases chest capacity • Lungs expand • Air is sucked in through your nose or mouth • Air travels through windpipe • Oxygen is extracted in the lungs

Utility of FBS in SCD - Researchers have developed numerous process models to understand, improve, and support the design and development process considering its particular characteristics. However, the complexity is such that no single model can address all the issues. Furthermore, the many models that have been developed are diverse in focus and formulation. The FBS framework is applicable to any engineering discipline, for reasoning about and explaining the nature and process of design (Krutchen, 2005).

- Universal Framework - The FBS model of designing has been applied to different contexts with different purposes. One may distinguish two main fields of application, *"as a theoretical vehicle for understanding design, and as a conceptual basis for computerized tools intended to support practicing designers"* (Galle, 2009). This corresponds to our two research goals. So we can utilize this framework for both our research goals.
- Supports Abstraction -Software engineers grapple with abstraction at virtually every step in the software engineering process (Pressman, 2005). As design commences, architectural and component-level abstractions are represented and assessed. The FBS design framework is categorized as an abstract micro model that can represent design as elementary abstract processes (Wynn &

Clarkson, 2018). Due to the abstract nature of the processes and the design elements it has the ability to be applied to any application domain in software.

- Integrated View - Typically and most commonly used tool for used in software design is Unified Modeling Language (UML). However most of the designs created using UML describe systems in different notations from different points of view and at different levels of abstraction. The need for a unified and integrated view, which allows for evaluation of the consistency and completeness of the design, was identified (Niepostyn & Bluemke, 2012).

The need for a unified and integrated view in software design, which satisfies the consistency and completeness of the design solution, was identified (Niepostyn & Bluemke, 2012). The FBS framework integrates the different representations of UML. In the conceptual design phase the Function-Behaviour-Structure (FBS) framework allows the designer to create/identify FBS elements and establish their relationships from an open design problem. This framework allows effective reasoning about the functional and causal roles played by structural elements in a system by describing a system's subcomponents, their purpose in the system, and the mechanisms that enable their functions (Gero & Kannengieser, 2014). The framework allows for exploration of problem (function) as well as solution space (structure/behaviour) during the conceptual design task. The FBS framework captures expert-novice differences in design, complex system understanding and may have implications for instruction (Hmelo-Silver et al., 2000). Due to these reasons, we believe that FBS is an appropriate framework to alleviate novices' difficulties of fixation and lack of integration.

2.8 Chapter Summary

- Conceptual design is an important phase in engineering design as well as software design. Around 60% of the total product cost is fixed at the conceptual design phase. This phase is important as: i) problem scoping happens at this phase, ii) the problem as well as the solution domain co-evolve in this phase, iii) different phases of the design process are highly interconnected, as it is the first phase the results of conceptual design affect all the remaining phases. Peculiar characteristics of software design, such as dynamicity and intangibility makes this activity more challenging. Conceptual design is inherently hard and needs to be supported.

- The outcome of software conceptual design is a design solution description that is implementation independent, supports analysis, exploration and communication. The SCD needs to fulfil the requirement at the same time to be a logically coherent solution. These are the quality characteristics of SCD that we focus on in this thesis.
- Experts use domain-specific knowledge as well as cognitive and metacognitive skills to solve complex and ill structured software design problems. Experts are able to utilize strategies such as mixed-depth breadth approach and problem-solution coevolution. They are also able to combine the various UML representations to create integrated design solutions. The cognitive processes that expert software designers' employ are memory retrieval, problem structuring, mental simulation, design reasoning and abstraction to name a few. All these practices of experts constitute disciplinary practices in the domain of SCD.
- Existing teaching and learning approaches focus on the learning context or the content of the software engineering processes and methodologies. Novices have difficulties in creating SCD, however their nature of design processes is still unknown. Additionally the tools are built to support practising designers but not to alleviate novice difficulties. Existing tools do not have scaffolds to support novices in creation of SCD and need for a teaching and learning pedagogy that supports novices to create SCD.
- In software design usually UML is used to model the software solution design. However UML consists of different diagrams that represent the solution in a separate view. For example, the use-case represents the functional view of the solution, class diagram represents the structure and sequence diagram represents the behaviour. There is a need for a unified representation so that novices can utilize them to understand the semantic relationship between the representations.
- The FBS design framework has been used as a basis for modelling designs and design processes in a number of design disciplines, including engineering design, architecture, construction and software design. It has also been used as a lens to understand expert designers' processes. So this framework is suitable for our two research goals that are - understanding novices' SCD processes and creating an environment to support novices' learning of SCD.

Chapter 3

Research Methodology

In chapter 1 we have explained our research goals; firstly, to understand the novice design processes involved in SCD and then to support novices' in creating software conceptual design (SCD). In chapter 2, we have identified gaps in the literature and chosen the function-behaviour-structure (FBS) design framework to examine as well as scaffold the novices in creating SCD. In this chapter, we describe how we chose a research method, to align with the research goals and the details of our research process.

3.1 Choosing a research methodology

Our research goals can be further divided into sub-goals as below:

1. Understand the novices' design strategies and the cognitive processes underlying it. This will bring out the difficulties that novices' face while creating SCD.
2. Create a FBS design framework based pedagogy and an associated learning environment to alleviate novice difficulties and support their SCD creation. Evaluate the effects of novices' learning with the FBS based environment.

The first goal that focuses on understanding novice design processes requires us to look at the actions that novices undertake while creating SCD. This requires us to interpret the actions in a context. From chapter 2, we have argued for the FBS design framework as the lens to look at the novices' actions. With this lens, we look at the design strategies.

The second goal is to alleviate the difficulties and support novices' creation of SCD by designing FBS design framework based pedagogy and an associated learning environment. We do so by systematically studying the novices' difficulties in using the learning environment. The insights from these studies give us the requirements for refining the learning environment features. We also iteratively study the effects of the learners' outcome and understanding of SCD after they have used the learning environment. By studying the learner difficulties and designing a learning

environment to address those we intend to strengthen the software engineering design practice at the undergraduate computer engineering level.

To be able to achieve our research goals we need a research methodology that is systematic, yet allows flexibility of methods based on the context and underlying research question. Our goals align with the design research family of research approaches. It is often referred to as 'educational design research' (Van den Akker et al., 2006). There are many other labels with which this is referred to in literature such as design studies, design experiments, developmental research, and engineering research. Though the labels may differ this family of research methods have the following characteristics (Van den Akker et al., 2006):

- aims at designing intervention to be used in the real world, so practicality of the intervention is measured
- design of intervention is based on theory and the testing of the design contributes to theory building
- focus is on understanding and improving interventions
- incorporates a cyclic approach of design, evaluation and revision

As we understand the characteristics, it is also important to note that the design research methods do not emphasize on isolated variables (Van den Akker et al., 2006). Educational design research (EDR) includes (Kopcha et al., 2015) but not limited to design-based-research (DBR) (Barab & Squire, 2004), design and development research (DDR) (Richey & Klien, 2014), and design-based implementation research (DBIR) (Penuel et al., 2011).

In EDR the three phases central are: analysis and exploration, design and construction, and evaluation and reflection (Figure 3.1 below). All the three phases exist in the methods discussed above- DBR, DDR, and DBIR. Each of the phases interacts with the practice, which increases as the project matures. However, the different methods such as DBR, DDR, and DBIR have focus on specific phases. For example, DDR focuses on "*systematic study of designing, developing and evaluating instructional programs, processes and products that must meet the criteria of internal consistency and effectiveness"* (Richey & Klien, 2014). This corresponds to the second phase. DBIR focuses on *"research on the implementation of reforms and drives iterative improvements"* (Richey & Klien, 2014). This corresponds to the last end of the spectrum. DBR focuses on *"producing useful products (e.g., educational materials) and accompanying insights into how these products can be used in*

education" (Baker & Van Eerde, 2015). So, DBR lies on the left of the continuum. Our research goals are also in the similar spectrum of EDR. As DBR aligns with the research sub-goals that we listed in the beginning of this section we choose DBR for this research work.

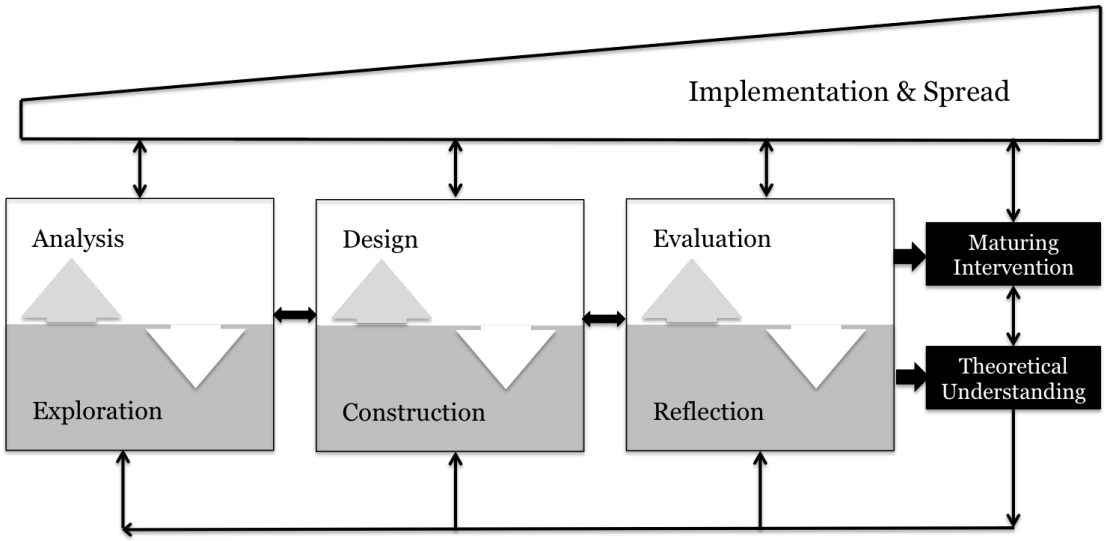


Figure 3.1 McKenney and Reeves (2012; p.159) generic model of Educational Design Research (EDR)

3.2 Design-Based Research Iterations

Design Based Research (DBR) is defined as *"a systematic but flexible methodology aimed to improve educational practices through iterative analysis, design, development, and implementation, based on collaboration among researchers and practitioners in real-world settings, and leading to contextually-sensitive design principles and theories"* (Wang & Hannafin, 2005). The goal of DBR (sometimes also referred to as design experiments) is to use the close study of learning as it unfolds within a naturalistic context that contains theoretically inspired innovations, usually that have passed through multiple iterations, to then develop new theories, artifacts, and practices that can be generalized to other schools and classrooms (Barab, 2014). Conducting DBR requires posing significant questions that can be investigated empirically, linking research to theory, providing a coherent and explicit chain of reasoning, demonstrating impact, disclosing research data and methods to enable and encourage professional scrutiny and critique, and employing methodological practices that are deemed credible and trustworthy and that result in useful claims (Shavelson et al., 2003).

The characteristics of DBR collated by many researchers can be found in Baker and Verde (2015). According to them DBR has the following characteristics:

- developing theories about learning and the means that are designed to support that learning
- implementation of the hypothesized learning is combined with the observation of that happens during actual learning
- developing and evaluating interventions situated in the real-world context
- it is *cyclic* in nature and consists of phases as depicted in Figure 3.1 namely – analysis/exploration, design/construction and evaluation/reflection.

In DBR the research begins with a detailed analysis of the problem, context and participants. Analysis of existing literature on the problem, the associated solutions designed, existing solutions in the given context or different contexts are also done. The studies in this phase often are pilot studies and/or ethnographic studies to understand the requirements of the learners (Figure 3.2). With the requirements, the designers and researchers draw from related theoretical and empirical work to create preliminary learning environment designs. The preliminary designs are then evaluated using various methods to understand the difficulties, learning processes and expected outcome. The reflection on the learning outcome and processes leads to refinement of design and local learning theories (Cobb et al., 2003). Local learning theory refers to a context specific theory about how learning happens in a specific context of the learning environment (Cobb et al., 2003).

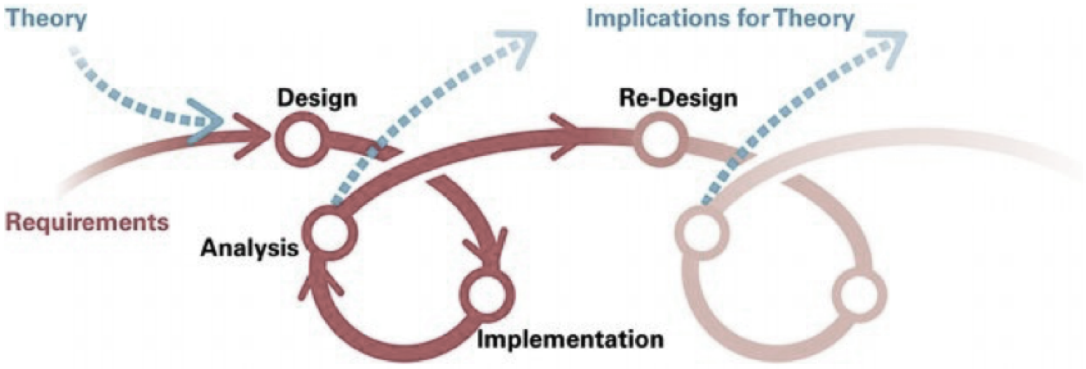


Figure 3.2 Typical DBR based research project phases (Barab, 2014)

We employed DBR methodology to examine novice design processes in SCD and to study learning in environments that are designed by researchers. The research plan is presented in the Figure 3.3. It consists of three stages – (i) learner needs analysis, (ii) iteration 1 and (iii) iteration 2.

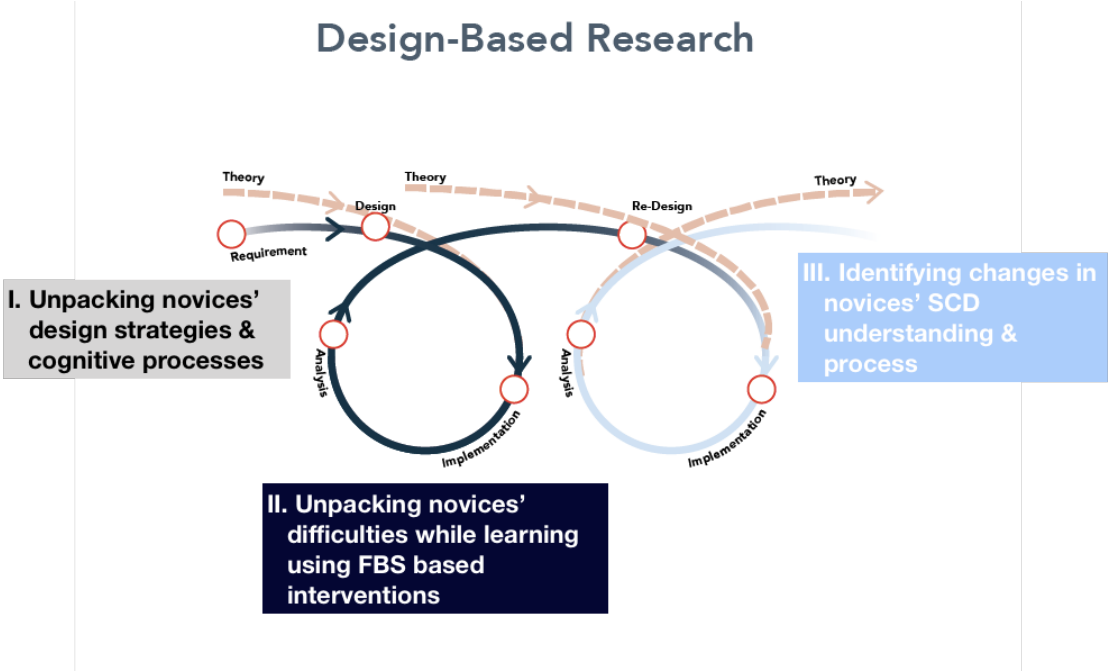


Figure 3.3 DBR cycles and the goals in this thesis

3.2.1 Learner needs analysis

The goal of this phase was to unpack the novices' design strategies and cognitive processes. This corresponds to goal 1 in section 3.1. As discussed in chapter 2, from literature we determined that the FBS design framework was a useful theoretical lens for unpacking novice design processes while performing the task of software conceptual design. We also synthesized findings from literature about – i) novice difficulties in engineering design [Chapter 2, section 2.5] and ii) expert design strategies and cognitive processes [Chapter 2, Section 2.3] in software conceptual design. As part of the problem analysis phase, to understand the novices' design processes in SCD we studied [Study 1] the novice strategies, processes and their difficulties in software conceptual design [RQ 1.1 & RQ 1.2]. The RQs of this iteration and its associated methods are described below.

Study 1: Broad RQ – How do novices create SCD?

1.a. What are the design strategies that novices follow while creating a software conceptual design?

1.b. What cognitive processes do novices use while creating software conceptual design?

We used the FBS design framework (Gero & Kannengeiser, 2014) as the theoretical framework to understand novice processes. We wanted to understand what

novices do in their mind as well as with resources available to create a SCD. We also examined their outcome to understand how these processes lead to the outcome.

"*Protocol analysis is a methodology for eliciting verbal and action reports of thought sequences as a valid source of data on thinking*" (Gero et al., 2011). It has been used extensively in design research to assist in the development of the understanding of the cognitive behaviour of designers (Gero et al., 2011). In the protocol analysis method we have performed the following activities: coding development, capturing video of participants on task, transcription of video, segmentation and coding, analysis of coded protocols, generation of linkograph and analysis of linkograph. In qualitative research, coding is "*how you define what the data you are analysing are about"* (Gibbs, 2007). As identified in Chapter 2, section 2.6, we used the FBS design framework as our coding framework. After coding we established the relationship between the codes using the linkography process. Linkography is a method (Goldschmidt, 2014) in which the relationships between the codes are analysed by creating a visualizing named linkograph. Goldschmidt (2014) also went on to describe methods and metrics to analyse a linkograph. The complete process is described in chapter 4 section 4.1.5.

3.2.2 Design Based Research Cycle -1

The learner difficulties emerging from study 1 provided us with requirements to design pedagogy to support novices' creation of SCD. The pedagogy is based on the FBS design framework (Chapter 2, section 2.6 and 2.7). The FBS design framework manifests as a FBS graph in the pedagogy. The literature points to usage of external representation (Dym et al., 2005). So the pedagogy involves creating, editing and evaluating the FBS graph. This led to preliminary FBS graph based learning environments. Our goals to design these learning environments were that we alleviate the novice difficulties found from study 1 and at the same support novices in creating SCD that satisfy the criteria of (i) fulfilling requirements and (ii) integrated solution design [Chapter 2, Section 2.2]. So for the evaluation [Study 2 & 3] of the learning environments we looked at the participants FBS graph created during and after the intervention. We also captured the participants' reflection on the difficulties they faced with the intervention. These were captured so that we can redesign the FBS graph based pedagogy and create a learning environment that is useful and usable. The RQs and the associated methods are described below:

Study 2 & 3: Broad RQ – What are the difficulties that learners have with FBS design framework based interventions?

2.a. After interacting with the FBS based learning intervention, what are the kinds of FBS graphs that learners create?

2.b. What difficulties do learners experience while using FBS based learning designs for SCD?

The study method for in 2 and 3 was a post-test and semi-structured reflective interview after the participants completed interacting with the learning environment. To evaluate the SCD that learners create after the intervention in the studies [Study 2 & 3], we used the conceptual model quality (Lindland et al., 1994) as the basis for analysing FBS graphs. The conceptual model quality (Lindland et al., 1994), attempts to define quality as it relates to conceptual models. We adapted this framework for FBS graphs. We evaluated the FBS graphs with five participants in the lab studies [Study 2 & 3]. The rubric is available in more detail in Chapter 5, section 5.4, Table 5.3.

To understand the learner difficulties we utilized the semi-structured interview responses from the learners. We first transcribed the interview responses. Then we employed thematic analysis. Thematic analysis is defined as a qualitative method *"for systematically identifying, organizing, and offering insights into patterns of meaning (themes) across a data set"* (Braun & Clarke, 2017). It is used for organizing, describing in detail and potentially providing interpretations regarding various aspects of the research goal, which are grounded in data (Braun & Clarke, 2017). Our goal in RQ 2.b is to identify difficulties that learners have during the intervention. The semi-structured interview questions are framed according only to elicit responses and the thematic analysis is to identify patterns relevant to this RQ. The steps, unit of analysis, reliability of the coding process are discussed in detail in sections 5.4.2 and 5.4.4 in chapter 5. For the sake of this chapter we are reproducing the DBR iteration Figure along with the studies and the associated RQ in Figure 3.4 below.

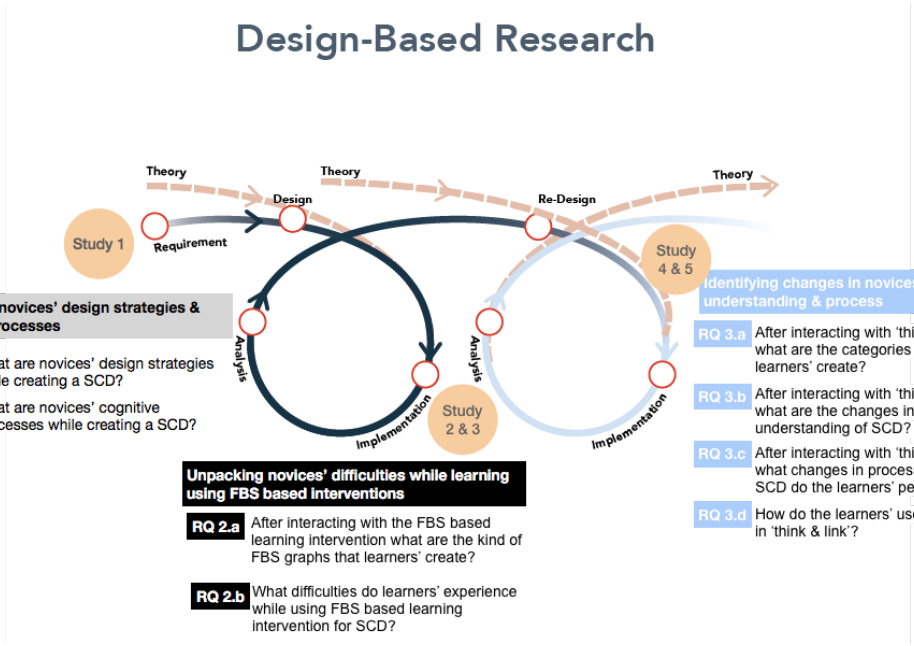


Figure 3.4 DBR iterations, studies and the associated RQs

3.2.3 Design Based Research Cycle -2

The goal of this iteration is to identify the changes in novices' SCD outcome, understanding and process. The results from iteration1 and the findings of RQ1.a, 1.b, 2.a and 2.b guide this iteration. From study 1 [RQ 1.a & 1.b] we unpacked the novices' design processes which helped us identify difficulties that learners face. Next we surveyed literature to design the FBS graph based pedagogy and implemented it as learning environments. The study 2 and 3 [RQ 2.a & 2.b] brought out the difficulties that novices' face in a FBS graph based pedagogy and learning environments. The results form the requirements for the redesign of the learning environment. In this iteration we refined the design of the learning environment to support the –i) creation of integrated software conceptual design and ii) learning of design strategies in software conceptual design. We performed heuristic evaluation to weed out the usability related issues. We evaluate the re-designed learning environment, named 'think & link', with study 4 and 5 to examine the effects of the learning environment. We examined the pre-post difference in -i) outcome of SCD [RQ 3.1], ii) learner understanding and perception of processes in SCD [RQ 3.2 & RQ 3.3]. We also analysed the learners' actions in the learning environment and compared them with their post-test performance. This helped us understand how the features support the learners. The RQs and the associated methods are described below:

Study 4 & 5: Broad RQ – What are the changes in novices' SCD understanding and processes after interacting with 'think & link'?

After interacting with 'think & link'

3.a. What are the categories of SCD that learners create?

3.b. What are the changes in learners' understanding of SCD?

3.c. What are the changes in the process of creating SCD that the learners perceive?

3.d. How do the learners use the features in the learning environment?

The study method in 4 and 5 was a single group pre-post test, with open-ended questionnaires both pre and post. For RQ 3.a. we used the categories of semantic categories of software designs (Eckerdal et al., 2006; Eckerdal et al., 2006(a)). For RQ 3.b and 3.c we used thematic analysis (Braun & Clarke, 2017). To examine the usage of 'think & link' features, we examined the logs in the system. Logs refer to the system action of recording user clicks on a menu/feature button. Each click is recorded as an event that includes information about the user, timestamp, context of the action and the action details. There are two kinds of events that are logged. One is a system-generated event, e.g. phase completion, worksheet saved. The other one is a user-generated event, e.g. click. We used the TraMineR package (Bürgin & Ritschard, 2014) in R to observe and extract the sequences. We also compared the sub-sequences (Ritschard et al., 2014) that are prominent in the post-test categories. This indicated the difference in strategy among the different performance levels.

3.3 Ethical Considerations

The following issues were taken into consideration while finalizing the research methods and data analysis techniques:

- Preparing documentation for taking informed consent from the participants: Participants were given a consent form before every research study detailing the objective and the procedure of the study. In the consent form the data collected for the study were explicitly mentioned. We offered clarification in case they had any queries. Once participants had clarity regarding the above points, they were asked for their consent. They had the option to discontinue the study at any point of time. Additionally, they were assured that participation in the study would have no bearing on their grades and academic performance.

- The anonymity of all the participants was maintained throughout, and all the data was collected, pre-processed, and stored for this appropriately. No one apart from the primary and secondary researchers on the project had access to the computer data and written artefacts of the participants.
- Most of the research studies in this thesis were conducted as lab studies and workshops. As this involved undergraduate engineering students, it is necessary to synchronize it with their calendars. Additionally all studies required that the participants already have undergone courses in UML modeling and software design. This brought the constraints to recruit participants in their second to final year of engineering. Various details of the studies were discussed with the instructors of the courses. To conduct workshops in the institutions necessary permission and consent from the concerned college/institution authorities was obtained. Student participation was voluntary, and they were provided with (workshop) participation certificates for attending the sessions.

3.4 Summary

In this chapter, we explained our choice of DBR as an overarching research methodology and described the details of the two iterations of DBR undertaken in this thesis, DBR1 and DBR2. We also described the studies done as part of each of these iterations and their research methods. DBR1 is elaborated in chapters 4 and 5, while DBR 2 is elaborated in chapters 6, 7 and 8. In the next chapter, we begin by describing the problem analysis phase of DBR 1, study 1.

Chapter 4

DBR 1 Problem Analysis: Understanding Novice Design Strategies and Difficulties

Most of the literature we have on novice design approaches report broad difficulties, compare expert-novice approaches or evaluate and categorize novice-generated artifacts. The section 2.3 brings out the design strategies and cognitive processes involved in software conceptual design. From section 2.5, we see that novices have difficulties while creating software conceptual design (SCD). The novice processes and the difficulties they faced in the SCD tasks have not yet been minutely explored. Our goal is to unpack novices design processes and assist novices in creating integrated SCD that fulfils the requirements of the design problem.

The novices do have intuition and sense making resources, which can be recruited, in formal education (Levy & Wilensky, 2008). Learners come with their own perspective based on their own unique past experience. A learning needs assessment can identify – (i) learner's prior knowledge and experience, (ii) skills or competencies that need further development, (iii) outcomes in particular that the learner wishes to achieve. Literature suggests that learning is more likely to lead to a change in practice if a needs assessment has been conducted (Fry et al., 2008). Novice design studies need to be more focused in understanding their design approach and needs.

This provided us with the motivation to conduct a 'learner needs analysis' study. In Section 2.6, we argued that the FBS design framework is an appropriate *"theoretical vehicle for understanding design, and as a conceptual basis for computerized tools intended to support practicing designers"* (Galle, 2009). *'FBS design framework is the appropriate theoretical lens for software conceptual design*.'

In this chapter we address the first goal of this thesis, which is to understand novice design processes in SCD. By examining their design processes we would come to know of their difficulties that they face in the SCD task. We capture their task process and their artifacts, which are then studied using the FBS design framework. The research method is described in section 4.1.

4.1. Method

The broad research question guiding this study was, 'How do novices create SCD?' By explicating novices' design strategies and cognitive processes, we will be able to compare them to expert SCD design processes. This will bring out the difficulties that novices encounter while creating SCD from a design problem. So, the research questions guiding this study are:

1. How do novices create software conceptual design?
    a. What are the design strategies that novices follow while creating a conceptual design?
    b. What cognitive processes do novices use while creating conceptual design?

'Design strategies' refers to a sequence of particular activities i.e. tactics, design methods to arrive at a design (Mathias, 1995). 'Design strategies' in study 1 corresponds to the sequence of FBS transformation processes (Gero & Kannengieser, 2014) as discussed in section 2.7. Cognitive processes refers to the internal mechanisms, however in study 1 we have operationalized it based on cognitive processes mentioned in conceptual design cognition (Hay et al., 2017).

4.1.1 Participants

For the study we had five participants (male=4, female=1). In this chapter the participants are labelled as par1, par2, par3, par4, and par5. They were under graduates and had completed their third year in computer engineering course from an engineering college near the affiliated institute of the authors. The participants were exposed to courses such as 'Structured Object Oriented Analysis and Design' (semester 5) and 'Software Engineering' (semester 6) as a part of their engineering curriculum. These two courses cover topics of software design approaches, software-modeling tools, characteristics of software solution etc. In these courses participants use design tools such as rational rose, and online platforms such as staruml.io, draw.io, creately etc. As the course contents included such concepts and experience with design tools, it was appropriate to consider that they had prerequisite knowledge for the design activity. The study was conducted over two days, with 3 participants on day one and 2 participants on day two. All the five participants volunteered to be a part of this study.

The objective was to obtain a typical representation of learners from the age group (19–22) with appropriate domain exposure. The participants' curriculum, activities, exposure and knowledge regarding software design are similar to many of the urban Indian engineering students. So it would be safe to say that the participants are representative of Indian urban engineering students.

4.1.2 Design Problems

The four design problems in Table 4.1 below were given to the participants. We came up with the four design problems based on the familiarity of software systems usage among the students. For example the systems such as ATM, payment authentication is familiar to participants as they encounter such systems in their day-to-day lives. In these problems the functional specifications are open-ended, and a part of the problem (ATM, payment systems, recommender system, music player) gives indication for the functional decomposition. By open-ended we mean that no requirements were provided to the students. Participants had to assume the requirements from the problem and solve the problem. Functional decomposition is describing the system based on the input, output and the transformation(s) that occur between input and output. For example due to the usage of ATM, the students know that the ATM requires an authentication/identification for a bank account as input and provides cash as output. The usage familiarity provides the students with indications for functional decomposition. The indications for functional decomposition make the design problem tractable for novices.

We have different design problems that the students worked on. We believed that by providing choice and working on a design problem of their choice the participant would feel motivated. Based on our software development and teaching experience the four different problems are equally matched in terms of complexity, time taken to solve, and amount of code that needs to be written. They are in between the innovative and creative design problem category (Brown & Chandrasekaran, 2014).

Table 4.1 Software design problems given to participants in Study 1

Problem	Day 1	Day 2

1	Design a fingerprint ATM system (par2)	Design a fingerprint based payment system (par4 & par5)
2	Design a mood based automatic music player (par1 & par3)	Design a cooking recipe recommender system

4.1.3 Study Procedure

Each participant was asked to select one of the two problems and create a conceptual design. Participant 1, 2 and 3 (par1, par2, par3) were present on day one. Par1 and par3 chose problem 2 (mood based automatic music player) whereas par2 chose problem 1 (fingerprint based ATM system). Participant4 and participant5 (par4, par5) were present on day 2. Both of them chose the problem 1 (finger print based payment system). We had offered a laptop to the participants in case they wanted to search any information or use any design tool. The participants were informed that at any point during the activity they could search for information, install any software, ask questions and think aloud.

Among the two of us, I, as an observer and researcher was present during the activity. Before the start of the task the verbal instructions on the task requirements were delivered. Participants were encouraged to ask questions about the requirements of the task, which were answered. During the task, participants were encouraged to think aloud. Observations were made while the participants completed the task. A video camera recorded the participant's actions while they went about the task. After every 30 mins participant's progress was evaluated and if deemed necessary prompts were provided to help the participant. The prompts were based on the scaffolding framework for ill-structured problem solving (Xun & Land, 2004) adapted to the context of SCD.

After completion of the task all participants were interviewed. The interview was semi-structured where the questions were framed predominantly to get the participants to reflect on their task. Some participants were forthcoming with their responses as self-reports, while some others had to be prodded more. Questions such as 'Can you elaborate why did you do this?' 'What made you pause on this point,

anything you can reflect on?' were asked so that participants can recollect and explain their process of design.

4.1.4 Data Source

Each participant was required to select one of the problems (Table 4.1) and create a conceptual design. We consider design as a distributed activity, which happens across the tools, environment and artifacts. Additionally design tasks in the workplace also happen in a similar way. So we provided the participants a work like environment, a large Table with pen, paper and a laptop. The artifacts that the participants created were collated. While the participants were on the task, video recording of the activity and screen capture of their interaction with the laptop was captured. Post task the participant was also interviewed. The interview questions were such that we asked the participant to reflect on the process. For example questions such as "Can you elaborate why did you do this? What made you pause on this point, anything you can reflect on?" were posed so that participants can recollect how they arrived at the design elements. For RQ1.a and RQ 1,b, the data sources included the video recording of the activity, screen captures, participant generated artifacts, and the interview transcripts.

4.1.5 Data Analysis

For both the research questions we utilized the three sources of data – task video recording, screen capture and artifacts. The artifacts were analysed based on the categories by Eckerdal et al. (2006). The task video recordings were analysed for the FBS design framework elements. The unit of analysis for function (F) was words/sentences, for structure (S) words and for behaviour (B) it was sentences. We utilized the linkography (Goldschmidt, 2014) method, which is widely used to understand the design process. In our analysis, we represented the FBS codes as a linkograph, which represents not only the codes as autonomous entities, but as design moves with inter-related links. The interrelated links of the FBS codes gave us the opportunity to analyse the design strategies. To unpack the cognitive processes that novices used to create and link the design moves we used the conceptual design cognition framework (Hay et al., 2017).

We began with analysis of the artifacts. The results of this analysis are utilized in both the RQs 1.a and 1.b. In the next sub-section we describe the analysis of the artifacts.

Participant generated artifacts - We analysed the artifacts based on the categories of software designs (Eckerdal et al., 2006; Eckerdal et al., 2006(a)). The categories are presented in the Table 4.2 below. These categories are results of a phenomenographic analysis (Eckerdal et al., 2006(a)) of the phenomenon 'produce a design'. Eckerdal et al. (2006) aimed to gather the understanding of this phenomenon by giving the final year project students a design problem and collecting the design artifacts. Using the artifacts they came up with various categories in which students understood the phenomena of SCD. We utilized these categories of student designs to classify the final artifacts of the participants.

Table 4.2 Categories of software design (Eckerdal et al., 2006)

Category #	Category	Content (Indicators)	Representation (indicators)
0	Nothing	Little or unintelligible content	Single labelled diagram Informal design
1	Restatement	• Restate requirements from task description • No design content other than stated in the description	List or Bulleted items Informal design
2	Skumtomte* (Named after a Swedish marshmallow treat, these designs add a small amount to restating the task)	• Add a small amount to restating task • Unimportant implementation details • No overall system view and any work on modules	Simple GUI
3	First step	Some significant work beyond restatement	Formal notation representing structure Design of one of the system's components

			like GUI or Database
4	Partial design	• Understandable description of parts and overview • Description of parts may be incomplete or superficial • Communication between parts may not be completely described	Formal notation representing behaviour Illustration of relationship between the parts
5	Complete Design	• Well developed solution • Understandable overview • Solution parts description includes explicit communication between them • Formal representations as well as text	Multiple formal notations such as Use case, Class diagram, component diagram

4.1.5.1 Analysis for RQ 1.a. - What are the design strategies that novices follow while creating a conceptual design?

Activity coding – For the RQ 1.a we utilized the three sources of data – task video recording, screen capture and artifacts. We performed three cycles of coding as shown in Figure 4.1 below. In the first cycle we performed activity coding (Bogdan & Biklen, 2007). In activity coding we looked at the participant's observable and distinguishable actions from the three sources of data. We first started by looking at the activity video and coded the actions. In the video whenever the participant wrote in the paper, we paused the video and referred to the artifacts. Similarly whenever the participant would use the laptop we paused the video and referred to the screen capture.

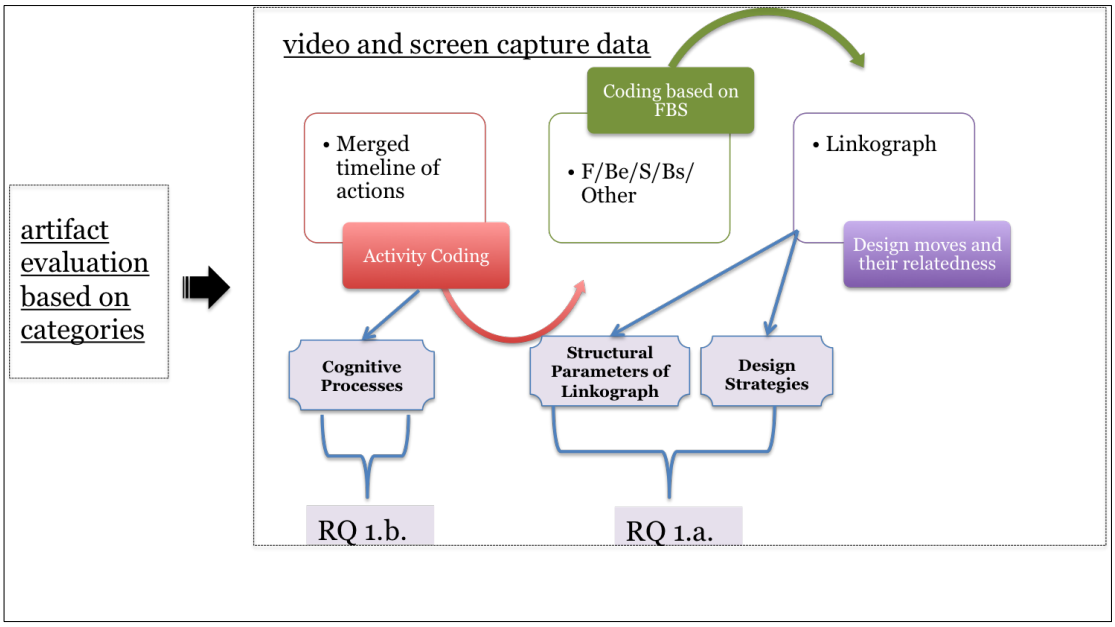


Figure 4.1 Data Analysis for research questions

The source of the actions and the unit for logging each action is presented in Table 4.3 below. This coding did not involve making any inferences and resulted in a merged timeline of actions. This merged timeline of actions consisted of participant's actions including explicit actions like internet search, sketching/writing on the paper. We noted the utterances, gestures and conversations in the episode from the video.

Table 4.3 Activity coding

Action	Logging Unit	Source
writing	line	*Video *Artifact
drawing	each object and annotation	
editing	line /each object/annotation	
deleting	line /each object/annotation	
staring	time spent stare and next observable action	Video
talking	line	

researcher talking	line	
reading	spends some time in the page (> 5 seconds)	*Video *Screen capture
scanning	evident from scroll action	
searching	search query in a search engine	
opening	clicks on a hyperlink	
returning	return to a previous web page	

The output of activity coding is a merged timeline of actions that the participants performed to complete the SCD task. This merged timeline is the input for the next step of analysis where we applied the FBS design framework lens.

Segmentation based on FBS codes- On the merged timeline of actions we looked for the codes: function (F), expected behaviour (Be), structure (S), behaviour of structures (Bs), Other (O). The FBS design ontology (Gero & Kannengieser, 2014) has a set of 6 codes - Function (F), Expected Behaviour (Be), Structure (S), Behaviour of Structures (Bs), Documents (D), Requirements (R). Since we are looking at the process we utilized all the codes except for Requirements(R) and Documents (D). The four codes (F/Be/S/Bs) were adapted to the context of software design. The Table 4.4 below captures the classification indicators in the context of the problem 'mood based automatic music player'.

An action could be coded as F/Be/S/Bs and if none of the four codes are applicable we coded it as other (O). This formed the second cycle of coding. The FBS based coding scheme converts the merged timeline into a series of segments where each segment is tagged with a code, F/Be/S/Bs/O.

Table 4.4 FBS codes on the merged timeline

Design Element	Code	Classification Indicator	Example Design Problem (Mood based automatic

			music player)
Function	F	activity performed by the software system	Mood detection
Expected Behaviour	Be	expected behaviour of the system extracted from the functions	Voice Based Mood Detection (F) - System needs to capture the voice
Structure	S	the solution concepts and components (hardware and software) required to achieve the function	Camera, software to detect mood
Structural Behaviour	Bs	behaviour of the structure, extracted from structures	Camera (S) - Facial features/points are extracted

To establish reliability in the coding process, a fellow researcher and I coded a participant's merged timeline. The coding process was repeated with parts of the participant's data until both the coders converged on the same codes for each of the actions. To identify F and S, the unit of analysis was words or at most two words. Then we examined the context in which these words are used. The classification indicator for function and structure is as given in the Table 4.4. For behaviours (Be and Bs) we utilize sentences. We examine if the behaviour is extracted from the function, then we classify it as Be and if the behaviour comes out of analysis of structures it is classified as Bs. The excerpts of the protocols recorded and the code applied are presented in Table 4.5 below. For example, one of the actions that a participant performs in the SCD task is writing ' identity fraud less likely' (row 3) on the paper. This corresponds to extracting expected behaviour from the system, and is marked with the code Be (expected behaviour).

Table 4.5 Excerpts of protocols with the coded segments

Timestamp	Action	Predicate	Details	Codes
00:12:10 - 00:12:34	reading	the paper in front of him	From the web page information listed goes back to the listing in the paper and looks back at it.	O
00:12:39	opening	the link	Disadvantages of Automatic Teller Machines (www.sapling.com/6119408/disadvantages-automatic-teller-machines)	O
00:12:56	writing	Identity fraud less likely		Be
00:12:57- 00:14:12	reading	the article	Disadvantages of Automatic Teller Machines (www.sapling.com/6119408/disadvantages-automatic-teller-machines)	O
00:14:30	writing	Added layer of security		Be
00:14:31 - 00:15:37	reading	advantages and disadvantages that the participant has written	participant seems to be reviewing the list of advantages and disadvantages that he has written	O
00:15:42	writing	current system		F

00:15:58 - 00:16:23	drawing	*block of ATM Card *connects to landing screen block *connects the previous block and then draws PIN block *connects the previous block and then draws Amount block *connects the previous block and then draws cash block		*S *S *S *S *S *Bs
00:16:34	writing	Proposed System		F
00:16:56	writing	Scenario 1: Card is present with user		Be

The analysis of coded protocols for a design task would need to be more than just the sequence of codes. The design task consists of interdependencies, back –forth and iterations. The method named linkography provides ways to establish links between the codes. The next section describes the method of linkography and the visualization created, namely linkograph.

Generation of linkograph – In the third cycle of analysis we use, linkography. Linkography is a technique used in protocol analysis to study designer's actions (Goldschmidt, 2014). This technique produces a representation of the design process, namely linkograph. Linkograph has been utilized to analyse protocol codes of designers and now is an established method for studying design cognition (Kan & Gero, 2017; Hatcher et al., 2018; Jiang & yen, 2013). In order to generate a linkograph, the protocol codes are listed in a chronological order called design moves (Goldschmidt, 2014). A design move is defined as *" a step, an act, and operation, which transforms the design situation relative to the state in which it was prior to that move"* (Goldschmidt, 2014). After the chronological arrangement of the coded protocols, the links between the design moves are determined by answering the

question – is the design move N related to any of the previous N-1 moves? Based on the answer to this the related design moves for each move is noted down and a diagram similar to the Figure 4.2 below is constructed.

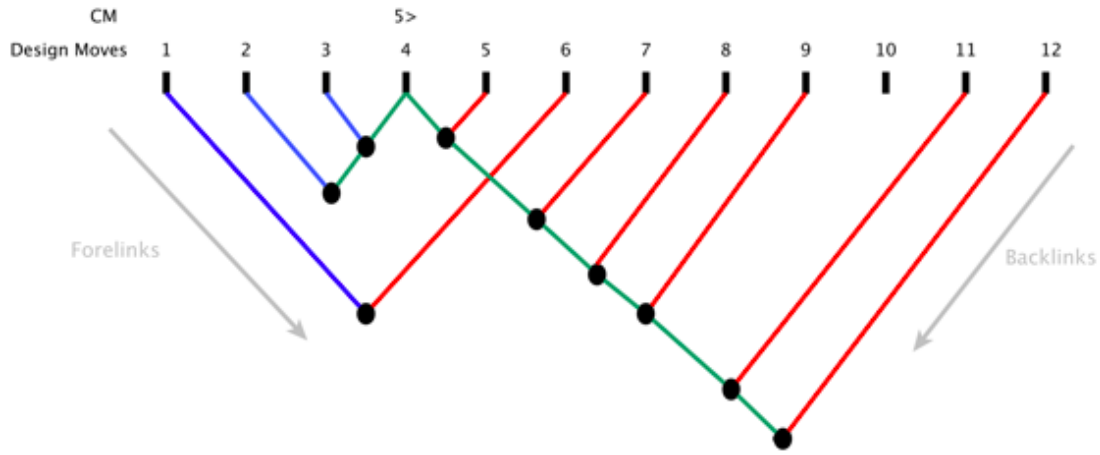


Figure 4.2 Various moves and types of links in a linkograph

In a linkograph, there are 4 different types of moves – orphan (move 10), unidirectional (moves 1–3, 5–9,11, 12), bidirectional (move 4) and critical (move 4). There are two types of links: fore link (blue) and back link (red). The green link is emphasized here to bring out the fore links and backlinks associated with move 4. The forelink indicates a design move utilized ahead in other design moves. The back link indicates the design moves utilized to arrive at a design move. The analysis parameters utilized are – link index and critical move. Link index refers to the ratio between the number of design moves and links. The critical moves are moves that are rich in links to other moves (move 4). A critical move which has higher fore links is indicated as a CM with the symbol > along with the superscript indicating the number of links. For example move 4 is indicated as $CM_5>$. A critical move with 't' back links is indicated as $<CM_t$.

After coding and categorizing for each action (F/Be/S/Bs/O), we assigned numbers to actions (1, 2,..Ni, ….N), which are coded as F/Be/S/Bs in increasing order of timeline. . The actions now are termed as design moves indicating that each of the moves made by the participant is a step, an act, an operation, that transforms the design situation somewhat relative to the state it was in before that move (Goldschmidt, 2014). The goal was to generate a linkograph, which establishes relationships between the actions. Each action Ni was examined with a question – 'if an action Ni is related to any action Ni-1 to 1'. If the action Ni is related to any action

between Ni-1 to 1 the actions are noted down. To establish reliability in the relatedness between actions, the first author and another researcher checked all the actions' relatedness of a participant. The process stopped only when both the researchers converged on their relatedness codes. After this process, a linkograph is created (Figure 4.2). The linkograph is analysed based on its analysis parameters – link index and critical moves. Link index refers to the ratio between the number of design moves and links. The critical moves are moves that are rich in fore/back links to other moves. For each participant we additionally chunked design moves based on spread of critical moves and the logical interlinks between the design moves. A set of moves that are visibly grouped together to create a set is termed as chunk (Goldschmidt, 2014). To analyse the links between design moves, the actual links between the design moves were considered and not the chronological sequence of design moves. This is referred to as semantic analysis of the links. We observed links between the design moves, and tagged it based on design transformation (Gero & Kannengeiser, 2014) like formulation, synthesis, analysis, evaluation, and reformulation. For e.g. a link between F and Be design move is tagged as design strategy formulation.

The above processes of categorizing, coding for F/Be/S/Bs codes in actions and finding the relatedness between these actions lead to the generation of a file as shown in Table 4.6. The excel files consisting of design moves and their related design moves (links) were created for all participants and provided as input to the tool LINKODER (Pourmohamadi & Gero, 2011). The tool provided general statistics on the number of actions, links between the actions, category of links for each participant. The tool also provided a linkograph for each participant. In the linkograph of each participant we additionally chunked design moves based on spread of critical moves and the logical interlinks between the design moves. We analysed each chunk of the participant based on the structural parameters of linkograph and design strategies.

Table 4.6 Template of participant actions provided to LINKODER to generate linkograph

#	Utterance	Code	Links				
1	PIN	S					
2	4 Digit	Bs	1				

3	Large Sum transaction (dual authentication	F					
4	memory	S					
5	visually impaired (convenience)	F					
6	Security or emergency fingerprint	F					
7	person may be forced into withdrawing cash	Be					
8	fingerprint can be replicated	Be					
9	Cardless withdrawal	F					
10	Identity fraud less likely	Be					
11	Added layer of security	Be					
12	block of ATM Card	S					
13	landing screen block	S					
14	PIN block	S	2	1			
15	Amount block	S					
16	Cash block	S					
17	Connecting the blocks of ATM card, landing screen, PIN, amount & cash	Bs	16	15	14	13	12
18	Scenario 1: Card is present with user	Be					
19	Card	S	12	18			
20	landing screen	S	13				
21	FP Authentication	F					
22	FP Authentication and connects 'landing screen' to it	Be	21	20			
23	Amount	S	15				
24	Check against user set cap and connects 'Amount' to it	Be	23				

Summary of analysis process for RQ 1.a – To answer RQ 1.a, we utilized the three data sources of task video recording, screen capture and artifacts. In the first cycle of analysis, we looked at the participant's observable and distinguishable actions from the three sources of data and applied activity coding to create a merged timeline of actions. In the second cycle of analysis, we segmented the timeline of actions to the FBS design framework elements (see Table 4.6). This resulted in a sequence of F/Be/S/Bs/O codes. In the third cycle of analysis, we determined the semantic links

and interdependencies between the codes and created a linkograph. The linkograph is a visual representation of the design codes in terms of the FBS design elements and their links. Using the standard analysis metrics (link index and critical moves) as well as processes (chunking) we analysed each participants' linkograph to explicate the design strategies. We also compared the design strategies of participants according to their final artefact evaluation using Eckerdal et al.'s (2006) categories.

4.1.5.2 Analysis for RQ 1.b. - What cognitive processes do novices use while creating conceptual design?

Hay et al.'s (2017) conceptual design cognition is the lens utilized to explicate the cognitive processes in the participants. The conceptual design cognition presents a generic classification of cognitive processes used in conceptual design from a meta-analysis of 47 protocol studies published over the past 30 years in conceptual design cognition (Hay et al., 2017). Analysing the protocol studies, the authors mapped specific descriptions of cognitive processes in the sample to more generic, established definitions provided in the cognitive psychology literature. It covered various domains such as architectural design, engineering design and product design engineering. The generic classification resulted in 6 categories of cognitive processes. However, not all the categories of cognitive processes are utilized during SCD. In Table 4.7 we present the categories and their details that were applicable only during this study.

Table 4.7 Conceptual design cognition in SCD (Hay et al., 2017)

Cognitive process	Definition	Role in design
Episodic retrieval	retrieval of previous experience	Retrieving experiences or representations
Semantic retrieval	retrieval of type of product and function during concept generation	
Analogical Reasoning	process of using information about known semantic concepts to understand newly presented concepts	Producing & combining concepts
Concept generation	the process of generating ideas for solutions/partial solutions to design problems	

Mental simulation	process of imagining and mental rehearsal of actions to produce or combine solution concepts	
Developing a solution	based on the outcomes of actions taken to structure/restructure the problem during co-evolutionary design	
Problem structuring and analysis	setting up goals and defining constraints	Planning, monitoring & selecting
Evaluating concepts	process of assessing concepts against design requirements, constraints, and other criteria	
Reasoning	process of developing a rationale for design decisions	

For the RQ 1.b we utilized the four sources of data – task video recording, screen captures, artifacts and interview transcripts. We utilized the merged timeline of action, which is generated after activity coding (see Figure 4.1). To this merged timeline of actions, we tagged the self-reports and participants responses to interview questions. We used the conceptual design cognition (Hay et al., 2017) as the lens to find the cognitive processes. The cognitive processes from conceptual design cognition (see Table 4.7) were utilized to code. We took each sentence from the interview transcript, tagged it with a cognitive process from Table 4.7. We coded for the cognitive processes until convergence in codes was achieved. A snapshot of this process is depicted in the Figure 4.3. For example if the participant arrived at a design move by recollecting prior experience we tagged it as cognitive process retrieval. The participant would have uttered the recollection of prior experience.

Participant utterances	Cognitive process	Why?
I usually do this before a project, I will design what all options I need, like this is the current system and I will see how this is better than this	Problem Structuring	Structuring/restructuring a problem based on the outcomes of actions taken to develop and evaluate a solution during co-evolutionary design
I take a current system closest to the problem, then I will see which all parts I can change to achieve solution to problem	Retrieval	Retrieval of type of product and function during concept generation
It is part of my previous experience, where I have developed those projects that have been deployed in my department	Retreival	Retrieval of previous experience
I think of myself as the user, and being a totally naive user, I am just coming into the store, what should I expect it to be?	Concept generation via simulating as end user	Concept generation, i.e. the process of generating ideas for solutions/partial solutions to design problems
I usually generate test casesfor testing the flow of process in my system based on the users.	Problem Analysis	Setting up goals and defining constraints
I look at similar systems like McDonalds, and Apple pay where you just remove your phone and it is paid, so similarly for a finger based system keep my finger on it and the payment is done	Analogical reasoning	Process of using information about known semantic concepts to understand newly presented concepts
When I want to express about the architecture of the entire software I look at the activity diagram based on which I create ER or class diagram, these actually help.	Synthesis	Creative output production - producing and combining concepts

Figure 4.3 Coding of participant interview transcripts for cognitive processes

4.2. Results

The goals of this chapter are to explore the novice design strategies (RQ1.a.) and the cognitive processes (RQ1.b.) that they utilize during the task of SCD. To interpret the results, we grouped the participants according to their design performance. We utilized the categories to evaluate the final artifact of SCD created by novices as formulated by Eckerdal et al. (2006) present in Table 4.2. We evaluated the final artifacts of the participants based on these categories (refer Table 4.8). We see that there are two broad groups of participants.

Table 4.8 Participants' artifact evaluation using categories by Eckerdal et al. (2006)

Participant	Problem	Conceptual design category
par1	Design a mood based automatic music player	Skumtomte
par2	Design a fingerprint ATM system	Partial design
par3	Design a mood based automatic music player	Skumtomte
par4	Design a fingerprint based payment system	Complete design
par5		Skumtomte

Par1, par3 and par5 are in the group ‘Skumtomte’ as the designs lacked ‘overall system view’. Par4’s conceptual design was well developed with integrated formal

representations like component diagram, use case and behaviour diagram. Par2's conceptual design had descriptions of each part of the design however the parts lacked integration between them. Par4's conceptual design falls into the category of 'complete design' and par2's 'partial design'. Par2 and par4's conceptual designs had understandable overview and well-developed solutions. Par4 had explicit interlinking between the parts of the solution which par2 did not. Both par4 and par2 used formal notations of use case, component diagram, class diagram and sequence diagram. Par4 due to the interlinking between the parts had a complete design that can be picked up by the development team for coding. To present the results for both the RQs we have placed the participants into two buckets, successful and unsuccessful. Based on the final artifacts evaluation, par2 and par4 are in the successful group of participants, whereas par1, par3 and par 5 in the unsuccessful group.

4.2.1 RQ 1.a. What are the design strategies that novices follow while creating a conceptual design?

We present the result in two pieces. The first piece of result comes from all the participants' linkograph general statistics. In the section named 'linkograph analysis', we report the analysis of participants' linkograph. We compare and contrast the number of design moves and link index with the two groups of participants. (Table 4.9) We also examine the kind of links, which indicate the design transformation (Table 4.10) in the groups of participants. In the section named 'Initial approach to solution' we detail out the design strategies that participants utilize in the beginning of the task of SCD creation. In the section named, 'solution generation design strategies' the design strategies that participants utilize to create design solutions are elaborated. We zoom into the design move blocks of each participant's linkograph, to extract the design strategies. The design moves blocks are based on the critical moves and patterns in a linkograph. We separated the sections because it was important to unpack novices' initial approach and solution generation strategies. Finally, section 'summary of design strategies' summarizes the design strategies and provides answers to RQ 1.a.

Linkograph Analysis

The linkograph comes with its own set of analysis parameters. As discussed in section 4.1.5.1, evaluation parameters of design move and link index is presented in Table 4.9. We further extended the link index to capture the semantically valid links and

called it Link index'. The Table 4.9 below captures the details of design moves, total links and the link index.

Table 4.9 Participants' design moves, links and link index

Participant	Design Moves	Total Links	Semantically Valid Links	Link index'
par1	77	200	111	1.44
par2	137	176	134	0.98
par3	45	51	17	0.38
par4	143	162	109	0.76
par5	53	44	33	0.62

In a linkograph consisting of a large number of moves there is a larger potential of links. From the Table 4.9 we see that par1 had the highest link index. A high link index may be the result of repetitions or many attempts to achieve synthesis. Par1 even with a high link index is in the category of unsuccessful participants. Among the successful participants par2 & par4, par4 when compared with par2 has lower number of links as well as link index. The link index value and the design quality may not have correlation.

The linkograph is a network of transformation of design processes. In the design process of the participants we observed only 4 of the codes (F, Be, S, Bs) and 7 design processes. Links provide indications of the design processes based on the FBS design ontology. The Table 4.10 shows the FBS related processes.

Table 4.10 Participant wise comparison of design strategies

Par #	Formulation F->Be (%)	Synthesis Be->S (%)	Analysis S->Bs (%)	Evaluation Be<->Bs (%)	Reformulation S->S (%)	Reformulation S->Be (%)	Reformulation S->F (%)
par 1	16.2	37.8	13.5	11.7	11.7	9	0
par 2	6.7	6	31	4	54	27	3
par	94.1	0	5.9	0	0	0	0

3							
par 4	4.6	4.6	49.5	6.4	31.2	0.9	2.8
par 5	27.3	0	18.2	0	30.3	21.2	3

Par3 had the highest % of links pertaining to formulation; par1 had the highest % of links pertaining to synthesis. Par5 did not have links of the category synthesis and evaluation, however had almost all the reformulation links. Par4 had the highest % of links in analysis of structures and par2 had the highest % of links in reformulation of structures and behaviours.

Initial approach to solution

Successful participants. Par2 and par4 anchored their solutions with familiar systems and structures. Par2, who chose the fingerprint ATM problem (see Table 3), at the start anchors to a similar system (card & pin based ATM) and goes on to elaborate the proposed solution based on this system. The initial approach of Par2 was to retrieve working of the card & pin based ATM system from experience and then focus on the main difference between the current and proposed system i.e. authentication mechanism. Par2's complete design moves are presented in Figure 4.4 below. The next couple of paragraphs we explain in detail par2's initial approach.

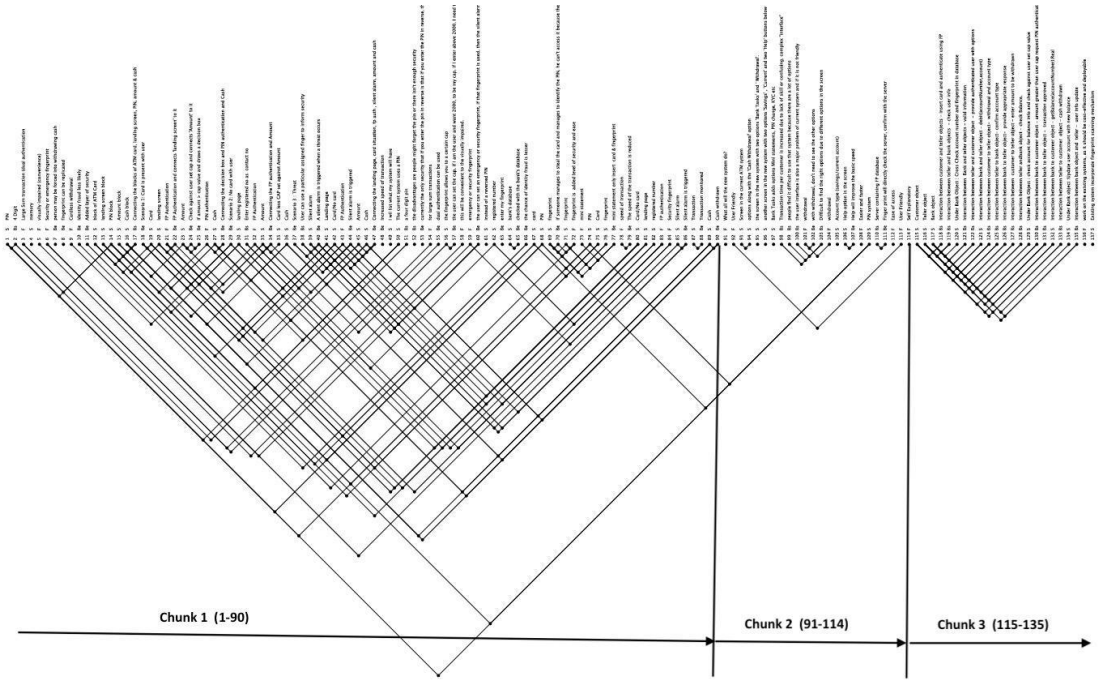


Figure 4.4 Linkograph of par2

The design problem that par2 had chosen to solve is 'fingerprint ATM system'. Par2 started with a similar system (card & pin based ATM) and gradually expanded his proposed solution by: (i) finding key differences between existing systems and proposed solutions, (ii) turning the disadvantages of existing systems to features in proposed solutions. After listing the functions, par2 moved to screen designs, which are also generated by utilizing the existing system (see Figure 4.4). Par2 then created the dynamic behaviour of the proposed system. Par2 started by retrieving experiences of a current existing system, card & pin based ATM. This led to generation of structure (move 1- PIN), which then was analysed to derive the behaviour (S->Bs). Then par2 focused on the main difference between current and proposed mechanisms indicating the cognitive process of analogical reasoning. By this process par2 arrived at the function of dual authentication mechanism (move 3).

Par2 had seeked information on the 'atm system drawbacks' by utilizing the search engine. Par2 looked at the drawback of the current ATM system and incorporated information from the search into the solution as features of the proposed system. The identification of features of the solution indicated the process of defining goals and problem structuring. Par2 also generated failure cases in the proposed system e.g. fingerprint can be replicated, person may be forced or coerced into with drawing cash. These failure cases led to the generation of expected behaviour in the

moves 7 and 8. The generation of failure cases indicated the process of constraint identification and problem analysis. Par2 then continued seeking information about the disadvantages of the current ATM system to incorporate in the proposed solution. To understand more about the current ATM system, par2 constructed an informal representation (block diagram) expressing the use case of 'cash withdrawal'. This move of constructing informal representation concluded at design move 17 ($CM^5>$).

The design problem that par4 chose to solve is ‘Design a fingerprint payment system’. Par4 at the start anchored to a known structure (aadhar API). Par4 then hooked the structure of aadhar API to create the solution for a fingerprint payment system. So the initial strategy of par4 was to retrieve a known structure and then adapt it as a solution for the design problem at hand. Par4’s linkograph consisting of the complete design moves are presented in Figure 4.5.

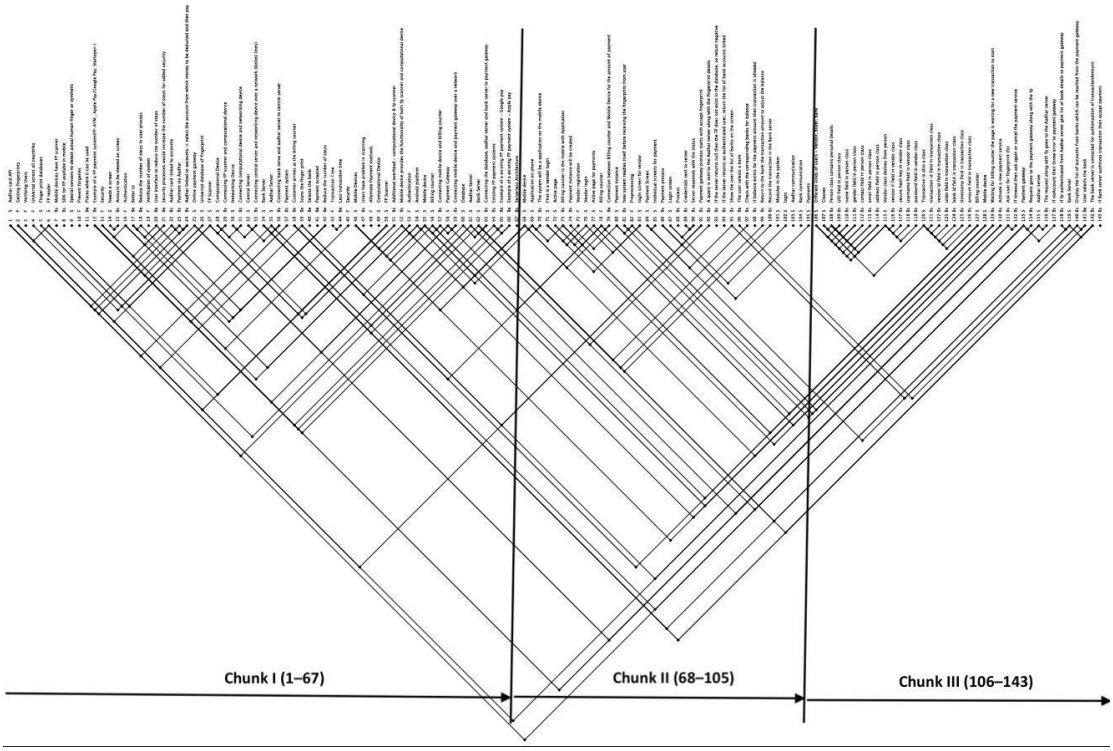


Figure 4.5 Linkograph of par4

Par4 started the design process by immediately associating a structure (aadhar API). Based on this structure par4 generated functions (getting and verifying the fingerprints). Par4 elaborated the solution design by generating structures. For example identifying that there exists SDK for fingerprint readers (SDK for FP). The other process that par4 utilized was to create failure scenarios and generate structure to combat it, e.g., special sensors to detect human or synthetic fingerprints. Par4 also

generated solution concepts by simulating scenarios from a system perspective, e.g., explored the connection mechanism between a fingerprint reader and a laptop. Additionally, par4 identified potential users of the system.

Summarizing, successful participants (par 2 and par4) started with structures of known systems. After retrieving known structures they adapted them to create solutions for the design problems at hand. They extracted behaviours from the structure (Bs). They also created functions (F) from the structures. These correspond to the strategies of analysis and reformulation with respect to the FBS design framework. Par2 and par4 also generated expected behaviour (Be) of the solution from the function, which corresponded to formulation strategy.

Unsuccessful participants. Par1, par3 and par5 are the participants whose conceptual design solutions fall into the category of 'Skumtomte'. Par1 and par 3 chose the problem of 'mood based automatic music player' while par5 chose the problem of 'fingerprint payment system'. All the three participants had similar processes to start the solution for design: searching for solutions/similar systems and solution concepts. The participants were unable to utilize the information from search results in the problem context.

The design problem that par1 chose to solve was the 'mood based automatic music player'. Par1 identified features pertained to addressing a part of the problem (mood detection) e.g. capturing mood, capturing mental thinking. Par1 then spent time on the internet seeking information. Some of the search queries by par1 are *'how can I check a person's mood'*; *'AI to identify a person's mood'*. These search results resulted in many solutions to the problem of mood detection. The participant then quickly added the search results as structures (e.g. heart rate, fingerprint, voice scanners) to the proposed solution. The participant didn't evaluate the integration of such structures. Par1 failed to identify constraints, determine feasibility of such structures and directly added them to the proposed solution. The participant had altered the problem from 'mood based automatic music player' to 'mood detection via different methods'.

When we examined par1's linkograph (see Figure 4.6) we observe that it has the most number of links among all the participants. Par1 has 200 links in the linkograph among which 56% of them are semantically valid links. With 77 design moves par1 has the highest link index 1.44. The high link index in this participant's

case is a repetition of a set of design moves. The design moves are also highly interlinked. The repetition and the high interlinking indicate fixation.

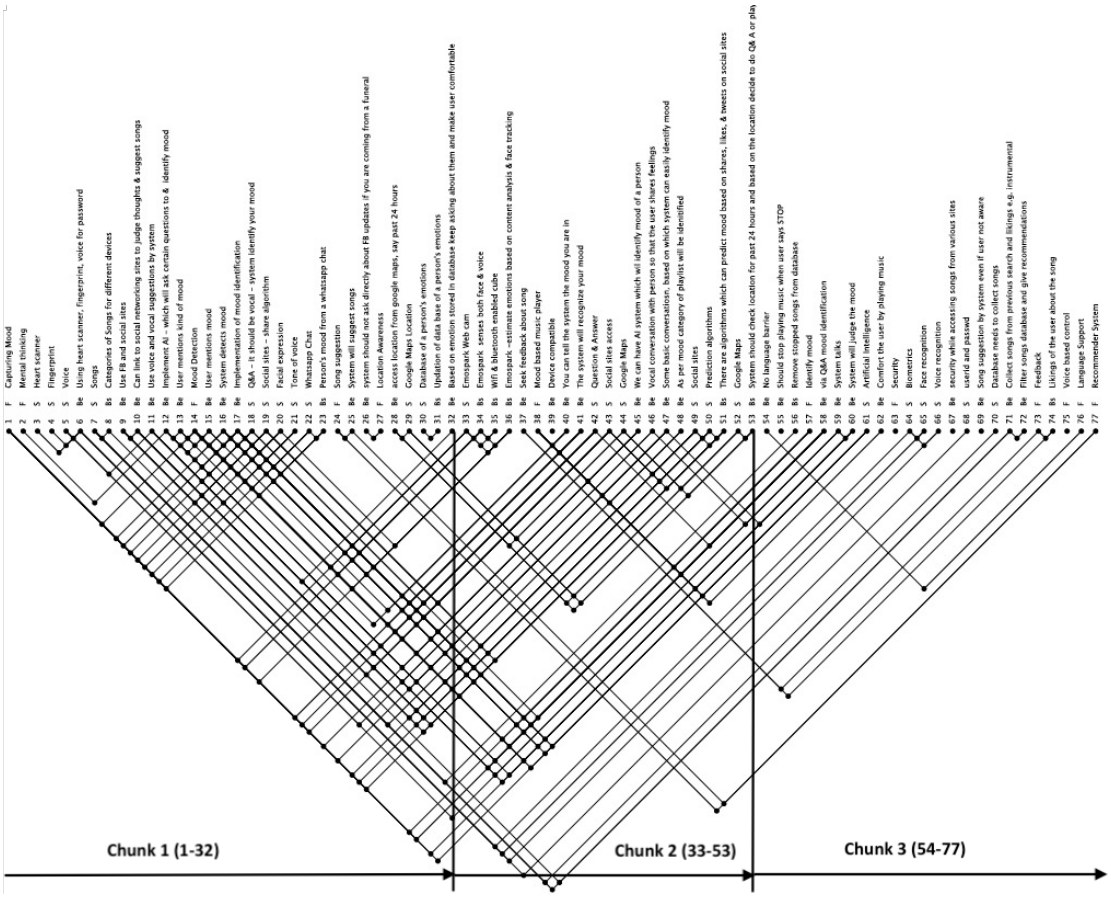


Figure 4.6 Linkograph of par1

The design problem that par5 chose to solve was – 'finger print based payment system'. Par5 started with the feature of security. This feature is generic and is applicable to most systems. Par5 from previous experience retrieved the function and associated structures. Par5 continued doing the same for other features. Par5 then recalled the concept of 'Block chain'. Par5 then proposed to implement a Block chain for secure payments. Throughout the task phase the participant moved between this feature and structure pair. The lack of experience with the concept and similar systems made par5 fixated to this solution concept. Like par1, par5 altered the problem '*design a fingerprint payment system*' to '*block chain for secure payments*'.

Par5's linkograph has 53 design moves and 44 links (see Figure 4.7). Among the links around 755 of links are semantically valid. There are 8 critical moves: 2 ($CM^4<$), 11($CM^4<$), 12($CM^3<$), 15($CM^7<$), 20($CM^3<$), 25($CM^4>$), 29($CM^3>$), 48 ($CM^4>$).

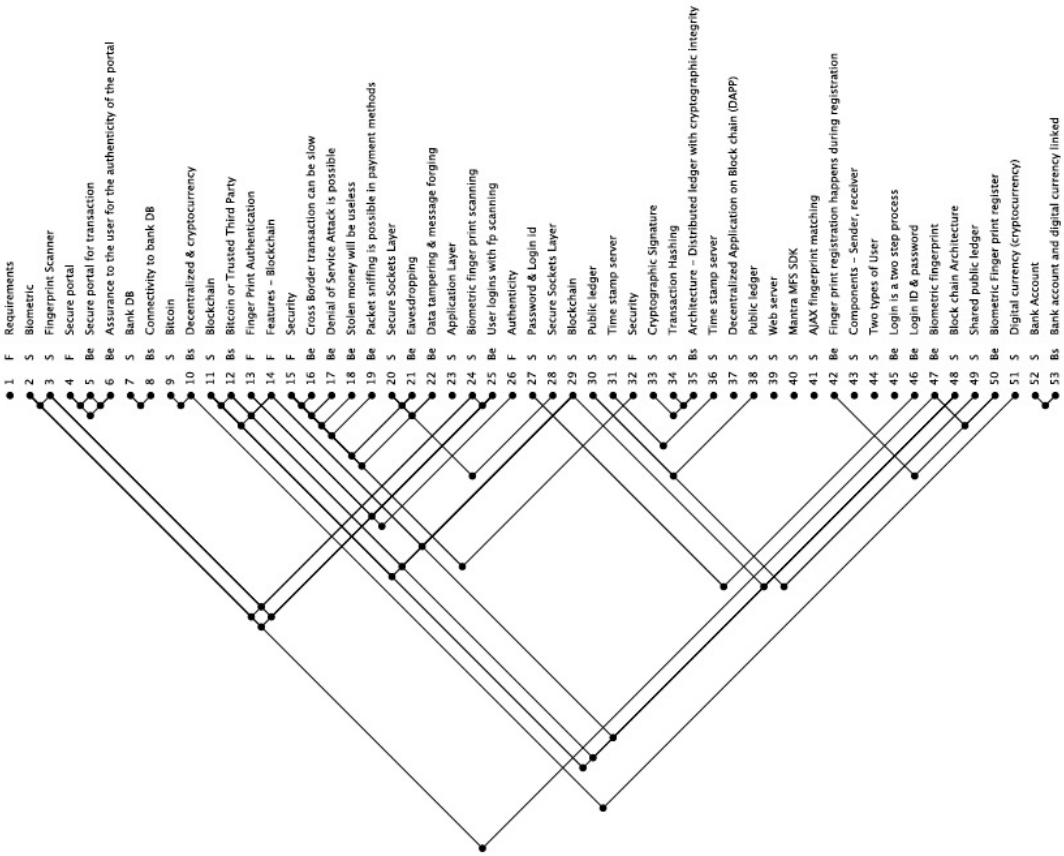


Figure 4.7 Linkograph of par5

Par3 started the creation of a solution by formulating the expected behaviour of the solution. Design moves 1–6 reflect the generation of expected behaviour (see Figure 4.8). At design move 8 par3 generated a function and then formulated the expected behaviour. Design moves 8–11 reflect that design strategy. Par3 continues to formulate the functions and expected behaviour in the similar way from steps 13–18. These design moves correspond to the playlist e.g., placement of song in a playlist, movement of songs in the playlist. From move 19–21 focus on mood detection. Par3 was still formulating functions and expected behaviour for mood detection in the design moves 26–30. Par3 spent a lot of time formulating functions to expected behaviours. Similarly, par1 and par5 start from the prescribed process of problem formulation by performing searches in the search engine on the internet.

The unsuccessful participants started with solution features (F) and then created expected behaviour of the solution (Be). So, we see that unsuccessful participants start with the prescribed process of problem formulation strategy (F->Be).

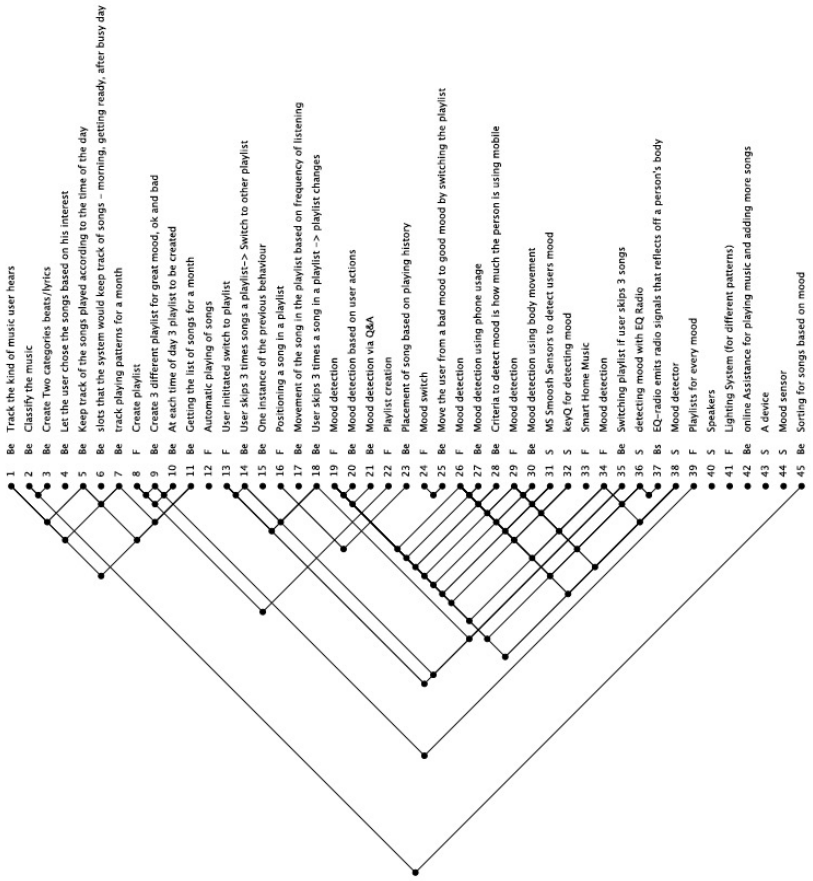


Figure 4.8 Linkograph of par3

Solution generation strategies

Successful participants. After retrieving the working of the card & pin based ATM system from experience, par2 focused on the main difference between the current and proposed system i.e. authentication mechanism. Par2 had seeked information on the drawbacks of the existing system. Combining the results of the search and the earlier identified difference, par2 generated features for the proposed system e.g., dual authentication, visually impaired, emergency fingerprint. For the generated features par2 then went on to generate failure cases. Par2 created a block diagram of the card & pin ATM system's workflow in order to explicate the understanding of the workflow of the system. To generate workflows of the proposed system par2 generated scenarios from the end user perspective, e.g., card present with user. While creating the workflow the participant identified components (landing screen, checks against amount entered in the screen with user cap) and features (i.e. functions such as FP authentication, PIN authentication). Par2 kept anchoring around the similar system (card & pin based ATM) and seeked information in order to extend from the similar

system to the system to be designed as indicated by web searches. Par2 went back to the search query to type in a new search (increased speed of transaction).

Par2 then generated screens for the proposed solution again anchoring it to the similar system (card & pin based ATM). Imagination of users utilizing the system served as the major cognitive process used by Par2 in this part of the design task. For instance, Par2 mapped certain screen elements (separation of cash withdrawal from the other bank tasks) to the feature of 'transaction time'. To do so, when designing the specific task of 'Withdrawal' par2 imagined a user utilizing this feature, leading to the purpose of reduced transaction time. The participant continued doing this for every screen element in the proposed solution. Par2 then created a dynamic representation of the proposed system, a sequence diagram. However he created the diagram for the specific case of withdrawal but was unable to connect back the parts of the representation to previous design moves.

Par4 after starting with structure as the anchor he mentally simulated failure scenarios, which helped him to identify problem constraints, e.g., the system may fail to authorize foreign citizens. He created two versions of the solution (laptop/mobile based). At each step he generated solution concepts by mentally simulating from the system perspective, for example 'System can use the Aadhar API for verifying the fingerprints'. He then imagined the connecting mechanism of the fingerprint reader in the two proposed solutions (laptop/mobile based). Par4 then reflected on the cost and feasibility of both the systems (constraints).

To expand both the solutions par4 draws on similar systems from experience (FP-ATM, Apple Pay/Google Pay, Startopen) and found the differentiating factor between similar systems and the current solution. These actions helped the participant to generate functions, structures and behaviours to each solution. Par4 then seeked information to evaluate each solution using the pros-cons and identifies potential users of the system. Similar to par2, par4 also utilized the orientation strategy of drawing a diagram. However before elaborating the system par4 seeked information from the internet to detail the internals and the solution. Par4 then detailed the mobile-based solution using a block diagram, which closely resembled a component diagram. The solution structures were then synthesized to features of the proposed system and end user behaviours.

We utilized the linkograph to zoom into the design strategies that par2 and par4 employed. Par2 and par4 all along the three chunks started with design strategies

relating to structures, which are analysis and reformulation of structures. Par2 and par4 employed the synthesis and evaluation design strategy, which connects the problem space and solution space. Par4 had a complete design due to the connection between the different parts of the solution details.

Par4 was the only successful participant with a complete design in the final artefact. So to understand par4's solution generation strategies (Figure 4.5) we zoom into the linkograph to capture the design strategies. Utilizing the Figure 4.9 we present par4's design strategies in depth below.

- design strategies around structures: Par4 retrieved and then examined the closest similar system encountered, which happens to be the FP-ATM. Finding the difference between FP ATM and the proposed system, lead to the identification of the structure in the proposed system (payment gateway, design move 25). As seen in Figure 4.5 (chunk I), the design process of analysis occurs first. The process of retrieval and evaluation, lead to generation of solution concepts in the proposed system. The participant began by expanding the solution details by adding structures in the design solution, evident from the moves 27 onwards. To be able to expand the solution and add details, par4 seeked information by querying the search engine. Par4 after identifying and listing structures continued to analyse them to generate behaviours from structures evident in the duos between S & Bs from the links between design moves 27–37. The design moves from 34–42 indicate that Par4 simulated the steps that a user (customer) of the system would undertake as they interact with the system. Par4 generated new structures in chunk II by simulating the deployment of the solution in a real life scenario. This simulation yields newer details of the solution from the moves 71–77. Reformulation of structures and analysis are the strategies that occur highly in chunk II as well as chunk1.
- reflection about implementation feasibility: Par4 reflected on the implementation feasibility and cost of the proposed system. This process of reflection indicates the constraints identification via problem analysis. This process helped par4 to eliminate a structure (computational device) from the previous design and replace it with another structure (mobile device). This analysis strategy lead to the extraction of behaviour of structures and links between moves 46, 51 & 52 (chunk I). The evaluation strategy in all chunks, also indicate the reflection that par4 utilized throughout the task.

- connecting problem and solution phase: In chunk I par4 utilized the synthesis and evaluation design strategy to connect the problem phases. In chunk II and chunk III the reformulation of function and evaluation kept connecting the problem and solution phase. Par4 utilized the strategy of reformulation of structure and analysis of the structure to derive the behaviour of the structure.

The design strategies utilized by par4 in each of the chunks is encapsulated in the Figure 4.9 below. In Figure 4.9, on the left the design strategies in chunk I are presented. The design strategy of analysis (46%) is used first, after which the participant proceeds to formulation (10%). This is followed by reformulation of structures (27%). This strategy is followed by synthesis (7%) and evaluation (10%). In every chunk the sequence of design strategy and the frequency of it are captured in the Figure 4.9. From Figure 4.9 we observe that in par4, as the design moved ahead, the formulation strategy is less employed. However, the strategies related to structure feature in every chunk.

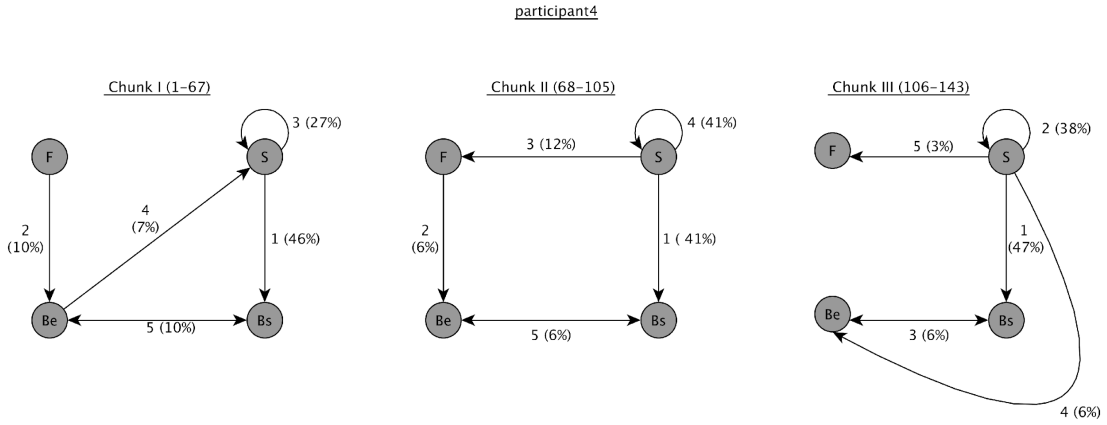


Figure 4.9 Par4 design strategies across chunks

Unsuccessful participants - The conceptual design solution of Par1 only had details for the problem 'mood detection'. Par1 created multiple solutions for this problem alone and couldn't complete the overall design for the stated problem. Par3 on the other hand generated a set of system and end user actions. But couldn't bind them to structures. As indicated in the Table 4.11, par3 spent a lot of time framing the problem however couldn't employ the other strategies. Par5 started by generating features and behaviours of the proposed solution. Par5 utilized the solution concepts of bit coin and crypto currency. In Figure 4.10, we present the design strategies of par3 and par5.

Par3 spent a lot of time formulating functions to expected behaviours. Both par 3 & par5 lacked the strategy of synthesis and evaluation. Par1 & par3 generated structures but for the same sub-problem of mood detection. When we look at the design strategies from Figure 8, we see that par3 and par5 have a high occurrence of the design strategy of formulation of behaviour. However par3 and par5 only kept repeating the strategy of formulation and reformulation of structures. Par1 attempted synthesis and generated structures however it was for the same sub-problem of 'mood detection'. There are indications of fixation across the design moves and strategy of formulation of expected behaviours (F->Be), analysis of structures (S->Bs) and reformulation of structures (S->S). All the participants couldn't generate failure scenarios or add test cases, which could have led to generation of unique features/structures and elimination of conflicting structures. The participants (par1, par3 & par5) did not utilize any informal/formal sketching mechanism and instead used statements to describe the solution.

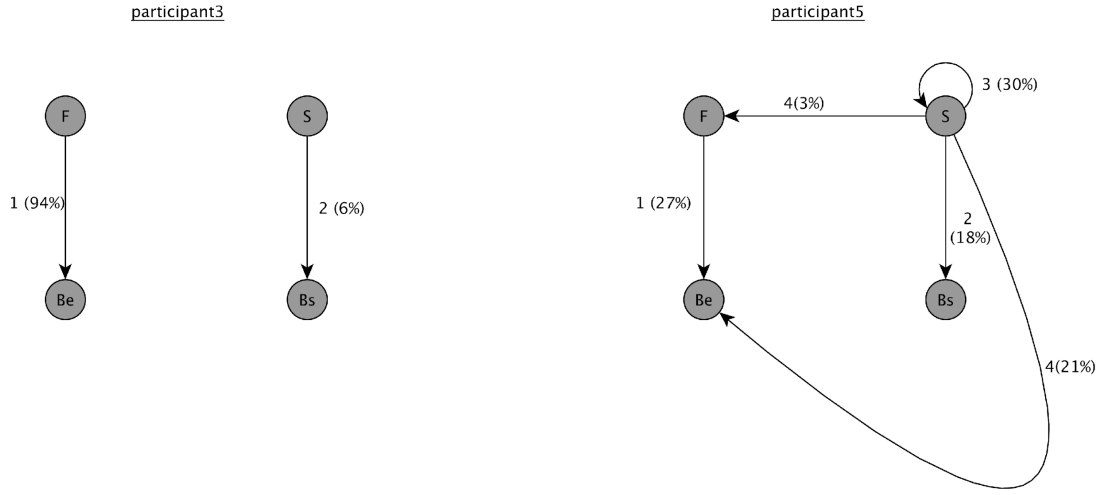


Figure 4.10 Par3 and par5 design strategies across design moves

Summary of design strategies

Design strategies refer to employing the sequence of processes from the FBS design framework. We observed that the participants employ different sequences of processes during the conceptual design task. These processes differ from the prescribed FBS design framework. The diagram 4.11 below captures the design strategies employed by the participants. In the diagram the arrows annotated with numbers indicate the sequence of the design strategies. The thickness of the arrows indicates the frequency of particular design strategies.

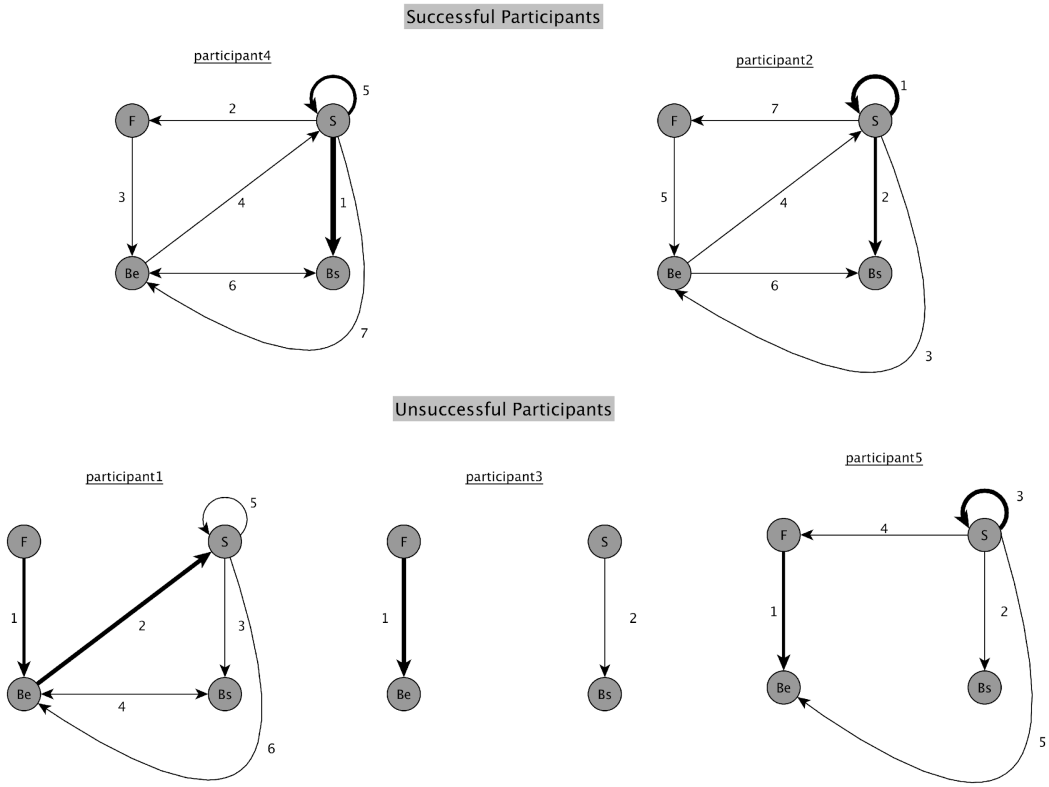


Figure 4.11 Comparison of design strategies across all participants

We have summarized the design strategies below:

Successful Participants

Both par2 and par4 had the final artefact that can be placed in the category of partial design and complete design (see Table 4.8). These two participants are considered as successful participants in the SCD task.

- Start with known and familiar structures: Par2 and par4 anchored their solutions with familiar systems and structures. As depicted in the Figure 4.11, we see that both par2 and par4 have design strategies starting with structures.
- Design strategies start around structures: Par2 and par4 all along the three chunks start with design strategy relating to structures which were analysis and reformulation of structures (see Figure 4.11).
- Employ all design strategies throughout the design solution: Par2 and par4 employed all design strategies.
- Design strategies connecting problem and solution space: Par2 and par4 employed the synthesis and evaluation design strategy, which connected the problem space and solution space. Par4 had a complete design due to the connection between the different parts of the solution details. Par4 chose the path of reformulation of functions whereas par2 chose the path of

reformulation of behaviours. However both of them employed the synthesis and evaluation, which connected the problem space and solution space. Par4 had a complete design due to the connection between the different parts of the solution details.

Unsuccessful Participants

Limited knowledge of the solution concept and experience of such systems inhibited the participant's progress. Par1, par3 and par5 repeatedly utilized the structures and repeated a set of design moves throughout the conceptual design process. Among the three participants we observed that the SCDs lacked overall view and contain some details, however not unified (see Table 4.8). These characteristics correspond to the category of 'Skumtomte'.

- Start from the prescribed process of problem formulation: We see that the unsuccessful participants started from the prescribed process of problem formulation. Par3 spent most of the time in problem formulation. Par1 started with formulating expected behaviour and synthesizing them to structures.
- Employ either problem formulation or solution-based strategies: The mapping between problem and solution process did not at all occur to par3. Par5 employed the reformulation of structures, expected behaviour and function. Par 5 chose a solution concept and spent a lot of time seeking information regarding the concept. Both par3 and par5 lacked the strategy of synthesis and evaluation.
- Repetition of structures/functions and behaviours: Par1 repeated the problem formulation throughout the design and solely focused on the solution to a specific sub-problem. Par1 fixated to a specific sub-problem and was unable to create overall view as well as complete parts. Par1 and par3 generated structures but for the same sub-problem of mood detection. Par5 proposed a solution concept, however couldn't create solution design with overall view as well as complete parts. The focus on the same solution concept was visible in the strategy of par5 of reformulation of structures.

4.2.2 RQ 1.b. What cognitive processes do novices use while creating software conceptual design?

Cognitive processes employed by successful and unsuccessful participants differ from each other. We observed similar cognitive processes between the successful

participants. The Table 4.11 below presents the cognitive process as defined in the conceptual design cognition and maps it to the design strategy where it is employed. The Table 4.11 also provides excerpts/examples from successful participants' actions. The first column in Table 4.11 is the cognitive process (refer Table 4.8). The second column in Table 4.11 is the category of cognitive process. Hay et al. (2017) came up with six categories of cognitive processes. We refer to the category that the cognitive process in column 1 belongs to. The design strategy for which the specific cognitive process was applicable is presented in the third column of Table 4.11. The last two columns give examples from participant's specific artefacts/actions that they referred to while indicating the cognitive process.

Table 4.11 Cognitive processes in successful participants

Cognitive Process	Category of Cognitive Process	Design Strategy	Example	
			par2	par4
Retrieval	Long-term memory	generation of structures, functions	● exploring existing system (card & pin based ATM) and its working first ● use case of the existing system is depicted	● Aadhar API ● similar systems (FP-ATM , Apple Pay/Google Pay, Startopen)
Analogical reasoning	Creative output production	generation of structures, functions, expected behaviours	exploring PIN and the characteristics of PIN as a 4 digit number	● FP-ATM: payment gateway ● Apple Pay/Google Pay: location of fingerprint authorization/FP storage ● Startopen: authentication steps

Problem Structuring (defining goals)	Executive Function	generation of functions	speed of transaction	security
Problem Analysis (constraints identification)	Executive Function	● generation of functions, structures ● reformulating expected behaviour and structures	fingerprint can be replicated, person may be forced or coerced into with drawing cash	failure cases (Aadhar FP collection, FP scanner faulty)
Generating Concepts (via mental simulation)	Creative output production	generation of expected behaviour	card present with the user, no card with the user, threat (end user perspective)	connecting mechanism between laptop and FP reader (system perspective)
Synthesis	Creative output production	● linking expected behaviour to structure ● reformulation of structure, expected behaviour, function	-	integrating component diagram to modules, classes and activity diagram

In the beginning successful participants retrieved experiences and generated structures. For example, Par2 reflects on the working of the current ATM system and *retrieves* the structure of the current ATM system. Par2 additionally generates failure cases. This indicates par2 setting problem goals and identifying constraints.

Par4 retrieved the prior experience of similar systems and associated the structure from those systems. Identifying components from existing structures generates the structures, which indicates retrieval from similar experiences. The other process that par4 utilizes is to create failure scenarios and generate structure to

combat the test case. This indicates that structures and solution concepts are generated via constraints identification and problem analysis. Par4 goes on to identify potential users of the system, which indicates the cognitive process of characterizing and structuring the problem by defining requirements.

Among the below cognitive processes, unsuccessful participants only generated solution concepts. The generation of solution concepts came from the result of information seeking. All the three participants (par1, par3 and par5) had similar processes to start the solution for design: searching for solutions/similar systems and solution concepts. The participants were unable to utilize the information from search results in the problem context. All the participants couldn't generate failure scenarios or add test cases, which could have led to generation of unique features/structures and elimination of conflicting structures. The participants (par1, par3 & par5) did not utilize any informal/formal sketching mechanism and instead used statements to describe the solution.

4.3. Discussion

RQ1.a. - What are the design strategies that novices follow while creating a conceptual design?

We observe that the novices in study 1employed strategy related to structures (Figure 4.11 & Table 4.10). This strategy is similar to the case drive approach (Jiang & Yen, 2013) as reported in the study by Jiang and Yen, in conceptual design. The case drive approach entails making a sequence of adaptations to a detailed concept (Gasparini, 2015). Both successful participants anchored to detailed structures of existing systems and adapted them to work in the newer situations. In the successful participants strategies we observe are evaluation and reformulation. Both par2 and par4 created test cases and failure scenarios to evaluate structures before integrating in the conceptual design. This strategy resembles the strategy of expert engineers who make evaluations of uncertain decisions before incorporating in design (Pan et al., 2010). Successful participants created informal as well as formal representations, which are similar to expert designer's utilization of visualization techniques (Hay et al., 2017). We see that in our study during the SCD task, successful participants exhibited certain strategies and cognitive processes of expert designers.

It is known that experts spend more time formulating, structuring and analysing the problem (Gilhooly, 1997; Atman et al., 2005; Atman et al., 2007; Dinar

et al., 2015). The unsuccessful participants in this study initially formulated the problem, by generating functions and expected behaviour. In literature also we see evidence of such behaviour (Hokanson, 2001), where it has been reported that *"novices who spent a large proportion of their time defining the problem did not produce quality designs"*. In all the participants' linkograph we see evidence of divergence (fore links) and convergence (backlinks), however the high interlinking between design moves in (par1) linkograph indicates design fixation (Goldschmidt, 2017). The design fixation could be due to incomplete models of solution concepts like artificial intelligence, bit coin etc. The unsuccessful participants (par1 and par5) anchored towards abstract concepts. This is referred to as a schema-driven approach (Jiang & Yen, 2013), which entails refinement of a highly abstract concept to a detailed description. Some unsuccessful participants (par1 and par5) took this approach and weren't able to convert them to detailed solutions. Most of the strategies of unsuccessful participants pertained to generation of functions, expected behaviours and structures. The participants repeated the same design artifacts and failed to create a comprehensive (fulfilling all the functional requirements) solution.

In answer to RQ1.b., we observe that the unsuccessful participants were unable to trigger cognitive processes to bind FBS design elements together. Unsuccessful participants employed information seeking cognitive processes. Information seeking is necessarily not an unproductive task. In the case of unsuccessful participants, they selected an unknown solution concept. For example, par5 chose block chain as a solution concept to design the fingerprint based payment system software. This concept was 'unknown' to the participant, but still par5 went ahead and spent most of the time reading and seeking information about the concept. As the unsuccessful participants chose an unknown solution concept for information seeking, they were not able to associate the concepts to the design problem/solution.

Evidence of the following cognitive processes are found in the data analysis for successful participants -

- adapt and integrate the results from the search engine into the solution of the design problem at hand
- synthesize expected behaviours into structures and components in the proposed solution
- generate failure scenarios and test cases to evaluate solution

- generate formal/informal sketches of solution to view and recognize possible interrelations between the different design elements

When compared with the successful participants, the unsuccessful participants were unable to trigger the cognitive processes such as simulation, association, analogical reasoning and synthesis.

4.4. Implications for teaching and learning SCD

These results have implications for teaching and learning of SCD. Some of the implications from results are listed below:

- Explicitly think and link FBS design elements: The learners need to be explicitly made aware of the design elements FBS while creating a SCD. In some of the software modeling tools and languages, like UML modeling, these elements of FBS are present but they are separated. Successful participants in study are able to link the FBS design elements, however unsuccessful participants are able to create the integrated view of solution design. This integrated view (Rosenman, 1991) of the FBS made explicit could help the learners build comprehensive and cohesive SCDs. By explicating the FBS design elements and their relationship this difficulty could be alleviated.
- Sketching: Instructors need to encourage learners to sketch ideas. Both the successful participants sketched informal as well as formal representations during the task. Unsuccessful participants instead wrote statements. Sketching allows for interpretation as well reinterpretation of the design elements (Atman et al., 1999). "*Sketching is a critical part of the early stages of the design process, facilitating ideation and the exploration of conceptual designs*" (Karimi et al., 2018). UML in software design has a variety of formal representation diagrams. As seen in this study and reported in literature novices are unable to utilize the formal representation mechanism in software design (Thomas et al., 2014). Additionally the UML formal representation is limited as it has diagrams which provide a single and limited view of the design i.e. either function, structure or behaviour (Niepostyn, 2015). Teaching and learning of SCD would need to include the integration of UML diagrams (Niepostyn, 2015).
- Anchor an existing solution: When given a software design problem, the learners need to be taught to think about existing systems. Both the successful participants started with an existing system or a known structure. Starting with a known

existing system or a structure provides an anchor to the learner, which can then be utilized. This strategy is similar to case based reasoning (Jiang & Yen, 2013), however in the case of SCD we could teach the learner to retrieve specific design elements like expected behaviour or structures. This needs to be followed up by analysis of structures or synthesis of behaviours. This strategy of mapping functions to structures is referred to as 'fast thinking' (Kannengieser & Gero, 2019). This strategy of fast thinking (Kannengieser & Gero, 2019) could also lead to design fixation, however if explicitly the learner uses this strategy generation of design elements would not be a deterrent.

- Appropriate information seeking: The information seeking behaviour during design tasks is recommended; however the purpose of the information seeking needs careful consideration. The unsuccessful participants searched the internet for similar systems and solutions, however they failed to fit it in the context of the design problem. Novices have been known to gather lots of information, and substituted this activity for doing any actual design work (Cross et al., 1994). Direct search of similar systems and solutions needs a follow up task of contextualizing the results in current problem. The successful participants' had other purposes for information seeking such as – disadvantages of existing systems and cost feasibility evaluation. Goal-directed information (Brand-Gruwel et al., 2005) seeking needs to be followed up by utilizing the result of the search in the context of the problems.
- Developing failure scenarios and test cases: The learners make design decisions by choosing certain structures and building on their behaviours to achieve functions. It is known that experts provide and keep track of rationale for their design decisions (Cross, 2004). To communicate and bring out the design rationale learners need to learn to create test cases to evaluate the design decisions (Cross, 2004). This process of evaluation leads them to construct rationale. However if failure scenarios or test cases are not created the evaluation process would not take place.
- Perspective taking or diverse point of view: When we look closely at the concept generation activities we notice that successful participants performed that task by taking a perspective. Par2 had an end user perspective (e.g., end user utilizes the assigned finger to inform security and alarm is triggered) while par4 had a system perspective (e.g. system has trouble in scanning fingerprint then need to switch to

the card mode). Perspective taking is considered as a designer way of thinking (Kannengieser & Gero, 2019). Explicitly taking perspective of either user or system and switching perspectives while generating solution concepts could result in generation of end user behaviour and behaviour of structures.

4.5. Limitations of study 1

First limitation of this study is the utilization of the lens of cognitive processes in conceptual design. A generic ontology of cognitive processes for conceptual design of different design problems includes cognitive processes that happen inside the mind. Using this lens to look at participant's actions requires a significant amount of inference. The inferences were a result of discussions between us. We do not have confirmations from the participant if the inference was consistent with their mental processes. Second, we did not include any physical behaviour such as gestures and motor actions to infer the cognitive processes. Third, the interpretation to generate a linkograph has subjectivity at different levels: coding for F/Be/S/Bs, determining the links between moves and interpreting the resultant linkograph. We have tried minimizing the subjectivity by undertaking interrator reliability checks as most protocol studies. As both the interrators were co-located we resorted to several iterations and final agreement, so didn't report any kappa statistics. Fourth, all the participants did not work on the same design problem. We wanted the participant to choose a design problem to make them feel motivated and work at the solution for their chosen design problem. Finally, another key limitation is the small sample size. However the goal of this analysis was to provide deeper insight to novices' cognitive processes and difficulties in the context of SCD.

4.6. Reflections and Summary

The result from study 1 offers insight to novices' processes and difficulties during the software conceptual design task. The results from this study about the novice design strategies and cognitive processes could be extended to engineering students having similar characteristics as Indian urban engineering students.

In study 1, we used the software conceptual design categories to evaluate all the participants' artefacts (Eckerdal et al., 2006). Among the five participants two of them (par2 & par4) completed the task successfully. They both generated artefacts ranging from most basic formal representations like component diagram, use case diagram to the complex formal representations like sequence diagram and process

diagram. The design strategies of these participants indicate the presence of all the design strategies of problem as well as solution space (Formulation, Synthesis, Analysis, Evaluation and Reformulation).

The unsuccessful candidates (par1, par3, par5) were fixated to one of the FBS elements during conceptual design. The FBS framework helped to identify that fixation can happen at any of the design elements or a combination of linked design elements. All three unsuccessful participants were unable to utilize formal representation mechanisms naturally even though they had the pre-requisite knowledge.

The important takeaways from this chapter and study 1 is as follows:

- Novice difficulties include inability to defixate and trigger SCD cognitive processes
- Novices need prompts and scaffolds to get them to generate integrated SCD solution
- Implications for teaching and learning—
    - Explicitly think and link FBS design elements - FBS as a logical unit, provide flexibility to start from anywhere
    - Scaffolds/prompt to link problem & solution based strategies via evaluation tasks
    - Encourage sketching
    - Anchor an existing solution - repurpose known solutions
    - Appropriate information seeking, developing failure scenarios and test cases, perspective taking or diverse point of view

Chapter 5

DBR 1 Design and Evaluation: Initial Solution Designs and Evaluation

In the context of software conceptual design (SCD) it is a standard practice to create various representations of unified modeling language (UML) to represent the solution design (Medvidovic et al, 2002). However most of the designs created using UML describe systems in different notations from different points of view and at different levels of abstraction. The SCD is described by integrating the various UML representations. In the formal curricula of computer engineering and information technology, students learn about syntax, semantics and tools to create the formal (UML) representations. However when students encounter open-ended real world problems they are unable to utilize the formal representations or create meaningful SCD (Eckerdal et al., 2006). Novices are unable to utilize multiple integrated representations in UML for a given design problem (Thomas et al, 2014; Lakshmi & Iyer, 2018).

Prior studies on novice difficulties in SCD (Eckerdal et al., 2006; Thomas et al, 2014; Chren et al., 2019) indicate that novices (i) only rewrite problem statements during the design phase, (ii) are unable to utilize formal representations of UML to model SCD and (iii) are unable to utilize multiple UML diagrams for integrated view of solution. In the teaching and learning of UML the novices are not provided with learning opportunities to integrate the different representations. While examining the nature of difficulties Chren et al. (Chren et al., 2019) have reported that novice difficulties in UML modeling include both syntactic and semantic. In contrast expert software designers create integrated solutions that fulfil all requirements (Petre, 2009). Creating integrated solution designs involves– i) utilization of multiple UML representations that are linked and ii) addressing both problem and solution aspects of the given design problem.

From the previous chapter we found the design strategies and cognitive processes that novices employ during SCD task. Novices face specific difficulties of fixation and lack of integration during the SCD task (Lakshmi & Iyer, 2018). For example, in the paper (Lakshmi & Iyer, 2018) for the design problem of mood based

music players it was found that mostly novices only focused on mood detection. In many cases novices' design solution was scattered and fragmented. Novices tend to generate solutions pertaining to fixed views of function, structure or behaviour, due to which they create representations of a single view. Representations of a single view do not embody the solution completely.

5.1. Integrated model building

Software conceptual design (SCD) involves model building by creating various representations, usually UML diagrams. However most of the models created using UML describe systems in different notations from different points of view and at different levels of abstraction (Medvidovic et al, 2002). The different representations together describe the solution design. Connection needs to be made between the various representations to check for the comprehensiveness and cohesiveness of the solution design (Niepostyn & Bluemke, 2012). Experts are able to create comprehensive and cohesive solution designs. They are able to make implicit connections between the various representations (Hungerford et al., 2004) and build integrated models of the design (Petre, 2009). This task of connecting between various representations involves a high level of cognitive load (Hungerford et al., 2004). Additionally there exists no single representation/view, which represents the model created for SCD (Niepostyn & Bluemke, 2012). Given this scenario, novices are unable to make connections between the various representations. This results in creating unconnected and incomplete representations.

5.2. Theoretical Foundations

5.2.1 Frameworks to support integrated model building in SCD

Novices are unable to integrate representations to build solution models. The function-behaviour-structure (FBS) design framework (Gero & Kannengieser, 2014) can be considered an appropriate framework to alleviate novices' difficulties of integrating representations. Though we have reasonably argued in chapter 2, section 2.6, in this subsection we revisit the argument for the same.

The FBS framework (Gero & Kannengieser, 2014) models designing in terms of three design elements: function (F), behaviour (B) and structure (S). Functions, describe what the design is for; behaviours, describe what it does; and structures, describe what it is. Along with FBS elements the framework has two sets of

behaviours, expected behaviour (Be) and behaviour derived from structure (Bs). These elements are connected to each other by a set of transformation processes. The set of processes include – (1) formulation which transforms functions to expected behaviours, (2) synthesis which maps expected behaviour to the structure, (3) analysis of structures which leads to generation of behaviours of structures, (4) evaluation of expected behaviour and behaviours extracted from structures, (5) documentation which contains the formal design description. There are three types of reformulation – (6) reformulation of structures, (7) reformulation of expected behaviour and (8) reformulation of functions, which are done to evolve the problem and solution together.

Philippe Krutchen (Krutchen, 2005) mapped one of the software engineering practice processes, rational unified process (RUP), to the FBS framework. The mapping is presented in Table 5.1 below. Each of the transformation processes in the FBS framework corresponds to activities in the design phase. For example, the formulation process in the FBS design framework where the expected behaviours (Be) of the solution are extracted from the functionality (F) is termed as requirement definition in the software engineering process. Thus, through the process of creating FBS elements and establishing relationships between them, appropriate activity in software design can be triggered. At the same time the learner gets to integrate the different representations in UML. Therefore we adopt the FBS design framework as the theoretical basis for the pedagogy of the intervention.

Table 5.1 Mapping FBS design framework to software engineering process

FBS Process	Mapping FBS elements	Software Engineering Process (Rational Unified Process)
Formulation	F->Be	Requirement definition
Synthesis	Be->S	Design analysis and implementation
Analysis	S->Bs	Testing and reviewing activities
Evaluation	Be<->Bs	Assessment
Structural	S->S	Refinement of design

Reformulation		
Behavioral Reformulation	S->Be	Requirements change
Functional Reformulation	S->F	Change in needs
Documentation	S->D	Implementation

5.2.2 External Representation

Design is considered as ill-structured problem solving (Dym et al., 2005) because design problems have ill-defined goals, states and solution steps. Problem solving is a cognitive task that benefits from distributed representation (Zhang, 1997). The representations are split across as the internal representation in the mind of the problem solver and an external representation. External representations that match the task fulfil important functions during problem solving. Literature also suggests that visual diagrams facilitate problem solving more than written notes (Larkin & Simon, 1987).

Problem solving process is dynamic in nature at and every step of the process there is a need to manage partial solutions and which lead to the next problem-solving step. Visualization need not be limited to representation of objects, but can be used at every step of solution. Visualization can be used to understand the problem, model solution, plan and make predictions. In the first step visualization can help bring out knowledge about a problem's structure.

In a study with professional software designers (Cherubini et al., 2007) it is said that software developers *create visualizations to understand, to design and to communicate*. Creating informal visualizations are practices that are adopted by experienced software developers. So creation and manipulation of visualisations are productive actions while creating designs.

In the case of FBS design framework, learners needs to represent function, structure, behaviour and establish the relationship between them. While doing so the learner keeps moving between the problem and solution planes. Visualization would serve as an external representation for the breakdown process as well as effective to and fro movement between the problem and solution planes.

In the proposed pedagogy aimed at novices' for creating integrated SCD the FBS framework can be represented as an external representation. The details of the FBS design framework manifesting as an external representation is presented in the next section.

5.3. FBS graph based pedagogy

The novice difficulties emerging from study 1, such as fixation and lack of integration, motivated us to design a FBS based pedagogic intervention for software conceptual design. The learner needs to represent function, structure, behaviour and establish the relationship between them. The literature on external representations and their effects on conceptual design, as discussed in section 5.2.2, led us to include representation creation/manipulation in the pedagogy. Among the various representations we chose a graph as the graph allows for -

- traversal/creation from top-down to bottom up is possible
- connections can be made between any pair of dyads

These characteristics of a graph provide the freedom to the novices to start from any design element F/B/S and traverse in any direction. Additionally the graph is a representation already known to third year computer engineering undergraduates who are our target population. As the linking between the F/B/S design elements happens the FBS framework design processes such formulation, synthesis, analysis, evaluation, reformulation emerges. The FBS framework manifests as a FBS graph in our intervention. For the design problem 'Create a software conceptual design for a mood based music player', the following Figure 5.1 is an example of a FBS graph.

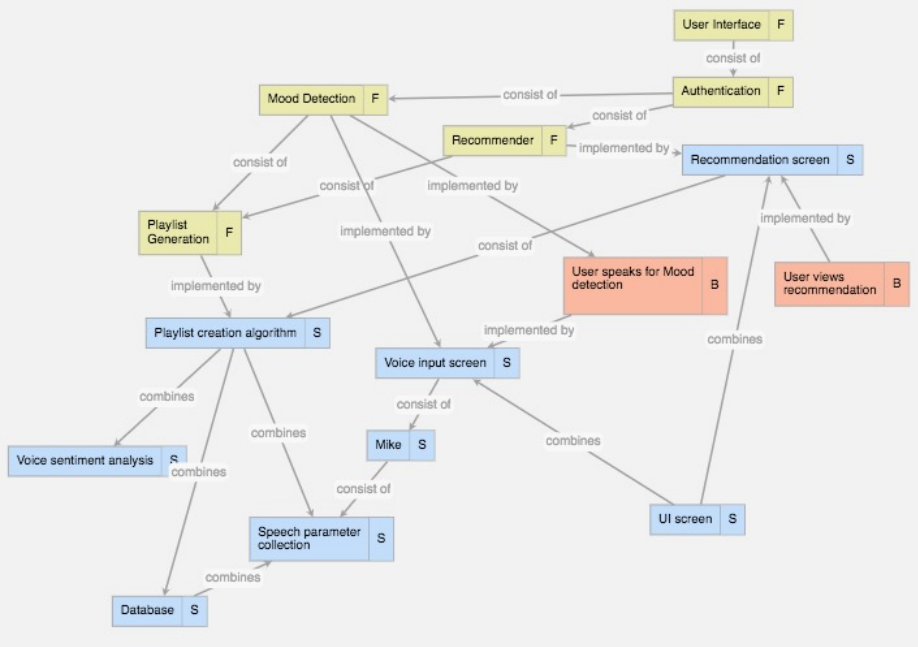


Figure 5.1 Sample FBS graph for the problem - design a mood based music player

From the Figure 5.1, we observe that the nodes are tagged as F/B/S design elements. The creation and traversal of the graph depends on the preference of the reader. The graph additionally does not have any structural limitation (top-down or bottom up). These are the characteristics of the graph that make it suitable for the FBS design framework based external representation.

Based on Chapter 2, 3, 4 the broad conjecture guiding our preliminary designs is - ***"By constructing a FBS graph students will be able to create integrated models for solution design".*** In the FBS graph based pedagogy, the FBS graph is a directed graph. The nodes are the F/B/S design elements. Specific connectors listed below connect them-

- *consists of* – this connector connects the similar nodes such as F->F, B->B, S->S
- *combines* – this connector connects several similar nodes to a node e.g. in the Figure 5.1 we see that several structure nodes combine to form 'speech parameter collection'.
- *implemented by* – this connector connects F->S, F->B, B->S

In this chapter we describe the preliminary learning interventions, and their associated qualitative studies (Study 2 & 3). The aims of the studies are to:

- identify supports to enable novices to create integrated SCD

- identify learner scaffolds required to complete tasks and interactions in FBS graph based intervention

The analysis of the above along with the evaluation of the artifacts produced by the learner will

- inform the task design
- inform the design of the learning environment
- the tools and scaffolds required in the learning environment

5.4. FBS based learning intervention – I

"By constructing a FBS graph students will be able to create integrated solution models as SCD".

Based on this conjecture we have created preliminary designs and conducted two qualitative studies with the intervention (study 2 & 3). The following learning objectives have been set for the FBS based pedagogy for integrated (Table 4.3) SCD:

- Learning Objective 1 (a) - build a syntactic conceptual model of FBS
- Learning Objective 1(b) – build a semantic conceptual model of FBS by interpreting the meaning of the design elements individually (F/B/S), their relationship together and relate it to the UML diagrams
- Learning Objective 2 (a) - refine their conceptual model of FBS by linking the FBS design elements and comparing FBS graphs based on criteria
- Learning Objective 2 (b) – utilize the scaffolds, prompts and criteria evaluation to build strategies for creating FBS graph
- Learning Objective 3 – apply the FBS conceptual model, strategies and criteria to create a FBS graph

The learning objectives are designed in a way that they progressively take the learners through the FBS graph as one of the methods to create integrated SCD for a given design problem. Learners initially build a conceptual model of FBS graph (learning objective 1 (a) & (b)). Then they are provided opportunities to build and evaluate existing FBS graph (learning objective 2 (a) & (b)). Then they apply the strategies learnt to create a FBS graph for another design problem (learning objective 3). By gradually taking them through the progressive planes of doing, evaluation and synthesis we support the novices towards building the integrated models of the conceptual design via the FBS graph.

5.4.1 Task design and learner activities in FBS graph based learning intervention I

We implemented the FBS graph pedagogy with a web page and IHMC CMAP tool. The webpage in this intervention is static and provides information such as definitions of the nodes (F/B/S) in the graph. It also provides the definition of different connectors, as described in the previous section. The webpage consisted of the tabs: phase 1 (learning objective 1), information, phase 2 (learning objective 2) and phase 3 (learning objective 3). The Figure 5.2 below shows the snapshot of the learning intervention I.

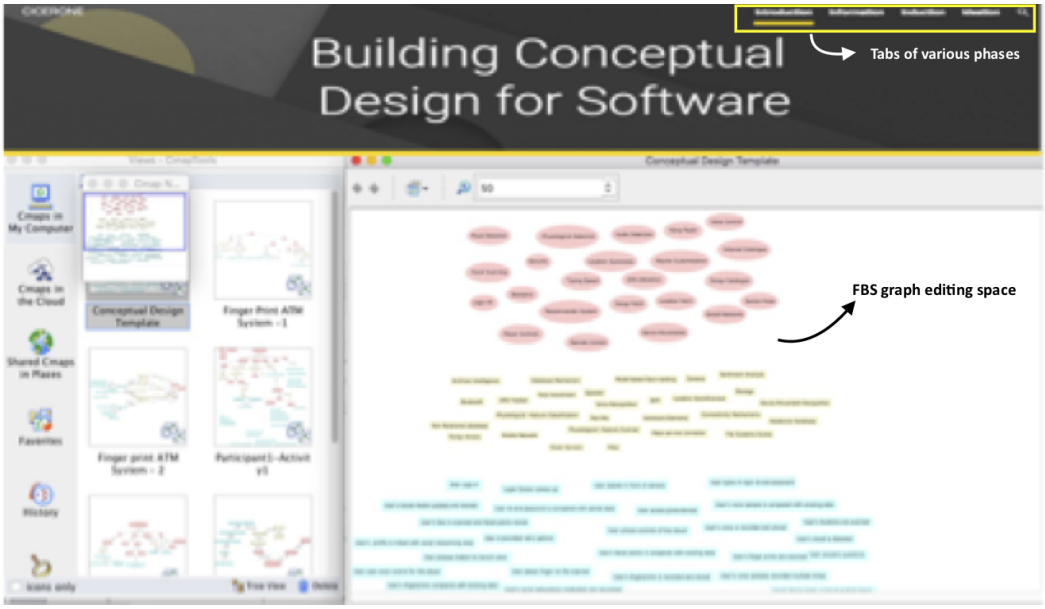


Figure 5.2 FBS graph based intervention I screenshot

The phase 1 tab initiates the learner to the context, learning tasks and objectives. The content is presented as a video. The information tab gives the definitions, examples of FBS. It also provides a set of sample FBS graphs for a design problem (Design a fingerprint ATM). We provided two sample FBS graphs for the same problem (fingerprint ATM). First sample FBS graph was in a top down approach and the second graph started from behaviour nodes. There are two design tasks in the intervention – (i) connect all the FBS elements provided for a design problem (design a mood based music player), (ii) set their own design problem (pet project) and create FBS elements and connect them. The two tasks (task no 2 & 3) as seen in the Table 5.2 below were presented to the learners as videos embedded in the webpage. The participants performed these design tasks on the IHMC CMAP tool.

Table 5.2 FBS graph based intervention I - task design and learner activities

Phase No	Task Design	In FBS intervention I	
		Problem	Task
1	Create FBS Conceptual Model in Problem 1 Context	Design a Fingerprint ATM system	No specific task. The building of FBS conceptual model is an implicit task which the learner would build by themselves by looking at the sample FBS graphs
2	Connect FBS Design Elements in Problem 2 context	Narrative Problem - Suraj has joined a product development firm. The firm encourages its employees to take up pet projects. Since Suraj listens to music almost every day, he wants to design a mood based automatic player. He has come up with various design elements, which are put up on the template. He has a presentation in an hour to show his design. Can you help him to connect the conceptual design elements to form a comprehensive and logical	FBS nodes in the IHMC Cmap tool. The FBS nodes need to be connected in the IHMC CMAP tool.

		conceptual design?	
3	Create and Connect FBS Design Elements in Problem 3 context	Narrative Problem- Suppose you are in the position of Suraj, employed in such a firm. State your pet project that you will choose. Decide on the conceptual design elements and how you would connect them to show your conceptual design.	FBS nodes are created and connected. The FBS graph is created and connected in the IHMC CMAP tool.

5.5 Study 2 – Qualitative Evaluation of FBS graph based intervention I

The broad RQ in this study was, "How to support novices' while learning SCD in FBS based intervention?" The research questions guiding this study was -

2.a After interacting with the FBS based learning intervention, what are the kinds of FBS graphs that learners create?

2.b What difficulties do learners' experience while using FBS based learning intervention for SCD?

5.5.1 Method

The participants ***(n=2)*** for this laboratory study were selected via purposive sampling. The participants of the workshop were post-graduate students with design experience. Due to the complex nature of SCD, prerequisites for participants of the workshop were determined. The participants were already exposed to undergraduate engineering courses such as 'Structured Object Oriented Analysis and Design' (semester 5) and 'Software Engineering' (semester 6) in their third year of engineering. These two courses cover topics of software design approaches, software-modeling tools, characteristics of software solution etc. As the course contents included such concepts, it was appropriate to consider that they had prerequisite knowledge for the SCD activity. The participation in the workshop was voluntary.

5.5.1.1 Study procedure and design problems – Study 2 was conducted as a post-test only study individually with two participants. The participants completed the activities in the learning intervention as mentioned in the Table 5.2. After doing so, the last activity was given as a post-test. Both the participants took 2.5 hours to

complete the activities. We conducted a reflective semi-structured interview by asking questions such as – *'How did you do this task? What are the difficulties that you faced in this task?'* These questions made the participant reflect on the difficulties that they faced during the task.

5.5.1.2 Data sources and analysis – In study 2, for RQ 2.a. we utilized the artifacts that the participants created. We also transcribed and analysed the participant's interview responses to answer RQ 2.b.

To analyse the participant generated artifacts (FBS graph) we then adapted the quality framework of conceptual model (Lindland et al., 1994) to the FBS graph. Lindland et al. (1994) created the framework to define quality as it relates to conceptual model. In our context the FBS graph is the integrated model of the solution software.

The framework of Lindland et al. (1994) attempts to evaluate the quality of a conceptual model of design along the three dimensions of – syntax, semantics and pragmatics. In the framework syntax refers to the model syntactic constructs and the relationship between the constructs. Semantics refers to the meaningful relationship between the constructs. Pragmatics refers to the ease in participation and utility of the model. In our intervention, we consider the FBS graph as the conceptual model. With the quality dimensions of the framework (Lindland et al., 1994) in place, we adapted them to the FBS graph quality parameters. We have operationalized the syntactic dimension to measure the syntactic qualities of the FBS graph. To be able to do this, we have included the criteria of complexity, levels and connectivity. Complexity measures the number of nodes in the graph, connectivity measures the graph connectedness and levels measure the depth of the FBS graph. All the parameters in syntax measure the comprehensiveness of the FBS graph. The semantic quality measures the degree of FBS graph validity. We have operationalized this dimension into validity, consistency and level adjacency. Validity measures the uniqueness and correctness of each FBS connection in the graph. Consistency measures the correctness of all the FBS branches together. Level adjacency measures that the depth is uniform across all branches of the FBS graph. The last category, which is the pragmatic category, measures the usefulness of the conceptual model. We have operationalized this as the extent to which the FBS graph can be mapped to formal UML representations such as use case, class diagram and sequence diagram.

In Table 5.3, we present the FBS graph operationalization of the framework by Lindland et al. (1994). We expand each dimension into the constructs as described in the previous paragraph. The performance levels in each of the constructs are also explained.

Table 5.3 Rubric for FBS graph evaluation based on Lindland et al. (1994)

Criteria	Target Performance	Needs Improvement	Inadequate	Missing
Syntax				
Complexity	More than or equal to 12 nodes each for F, B, S.	Only 4 nodes each for F, B, S	Only 2 nodes each for F, B, S	Only a node each for F, B & S
Levels	Two levels in the function, structure and behaviour sub graph are present in the FBS graph	Only two levels in function and behaviour sub graph are present in the FBS graph	Two levels in either function or behaviour sub graph are present in the FBS graph	There are no levels in all the three - function, behaviour and structure sub graph
Connectivity	All the nodes in the FBS graph are connected	Some of the nodes are connected but there exists nodes in the FBS graph that are not	FBS elements are grouped together to form disconnected forests	There are listing of FBS elements in the graph space
Semantic				
Validity	All FBS branches are unique , relevant to the problem and satisfy the	There are unique relevant FBS branches. However the problem	There are some repetitive FBS branches, which do not satisfy the problem	All the FBS branches are repetitive, irrelevant and do not satisfy the problem

	problem requirements	requirements are not satisfied.	requirement. There are some irrelevant FBS branches also.	requirement.
Consistency	A combination of FBS elements, sub-graphs and branches are not contradictory to one another.	A combination of FBS elements and sub graphs are not contradictory but some FBS branches are contradictory.	The structure nodes are inconsistent	All FBS elements are contradictory to one another.
Level Adjacency	All the adjacent pair of nodes in the graphs are at the same level	Only the nodes in F-F & B-B are at the same level	Only the nodes in F-F are at the same level	At any level of F, B, S there exists no adjacent nodes at the same level
Pragmatism				
Formal Realization	All the design elements of the FBS graph along with their relations are mappable to the appropriate formal representation (UML diagrams)	Only some of the design elements of the FBS graph along with their relations are mappable to the appropriate formal representation (UML diagrams).	Only the FBS design elements but not their relationship mappable to the appropriate formal representation	None of the FBS graph elements and their relations are mappable to the appropriate formal representations

To analyse the interview responses for RQ 2.b, we employed thematic analysis (Braun & Clarke, 2017) to inductively come up with themes of difficulties in the FBS based learning intervention I.

5.5.2 Study 2 - Results

In this section we begin by describing each participant's process in the FBS learning intervention I as case studies. We then present the results of the thematic analysis of the participants phase and task wise in the FBS based learning intervention I. We close this section by summarizing the answers for the research question based on the results.

5.5.2.1 Participant 1

Participant 1 first initially looked at the phase I tab then quickly browsed through the information page. Participant 1 moved to the phase II task in 5 mins. While doing the task (design a mood based music player) participant 1 utilized the details in the information tab. More particularly, participant 1 viewed both the FBS sample graph 1 and 2. However the strategy of FBS graph 1 for the induction task was chosen. The utilization of the top down approach of FBS graph 1 as quoted by participant 1 in the reflection interview. To create the FBS graph in phase II participant 1 took about 1 hour. Participant 1 was fixated with mood detection and populated the graph with all the functions and structures of all mood detection as shown in the Figure 5.3. In the syntactic criteria the FBS graph reaches the levels of target performance. However it is also likely that the participant may have been confused to utilize all the available nodes for the FBS graph. It is important to point out that the participant had used the behaviour nodes only for three of the mood detection methods (voice, facial recognition and biometric). The branch of function node 'Biometric' looks futile considering the other branches. The FBS graph did not reach the target performance in the semantic criteria. The FBS graph lacked both validity and consistency. There was no feedback given to the participant about the performance in the task. The participant then proceeded to the reflection task.

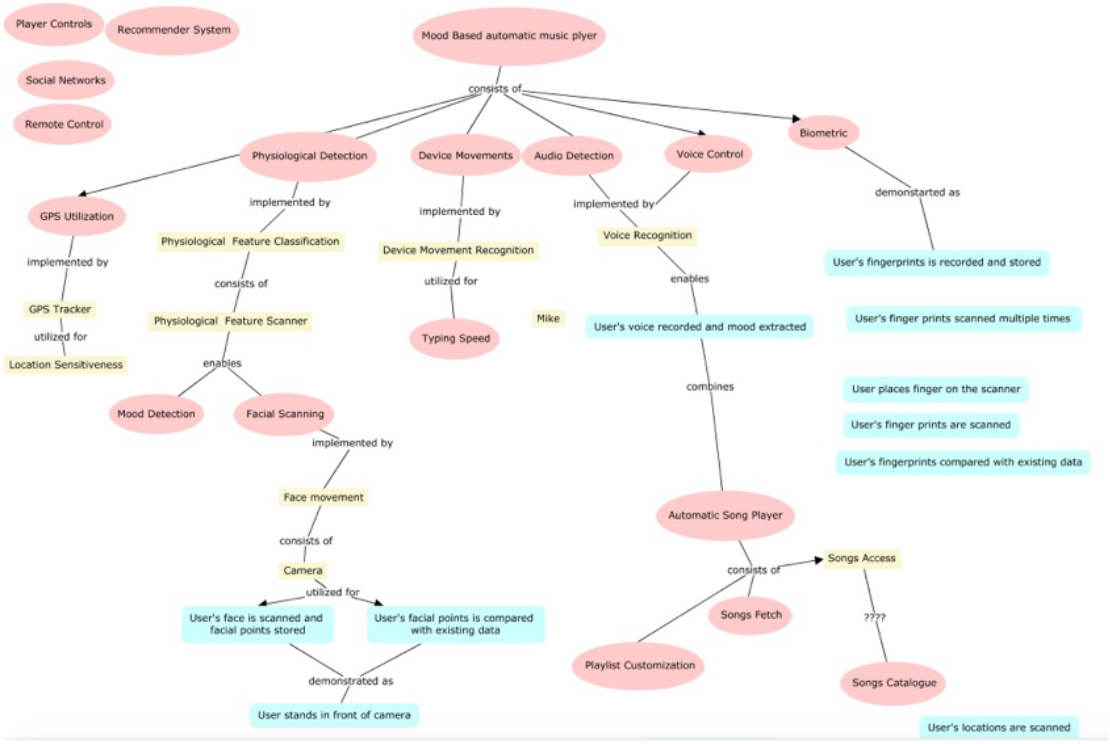


Figure 5.3 Participant1 FBS graph for phase II task in FBS graph based intervention I

In the phase III task participant posed the following design problem

"Generation of an Adaptive Scratch programming System for students based on their selection of type of tutorials, activities done during the tutorials, grades, attendance and type of Practice sessions chosen. Based on this a particular type of Scratch programming session will be selected for every student."

While solving the above design problem participant 1 utilized one branch from the sample FBS graph (fingerprint ATM). This could indicate reusability. However the usefulness of fingerprint scanning in a programming environment is questionable. The identification of functions and generation of user behaviours is evident in the FBS graph but it is lean in terms of structures (2 structure nodes). In the syntactic category (complexity), the FBS graph does not reach the target performance. In the semantic category the FBS graph does not reach the target performance for the validity criteria. This FBS graph needs improvement in the level of comprehensiveness (behaviour, structure nodes) and does not fulfil the problem requirements in the conceptual design. Participant1's SCD was at the target performance of syntax, semantics, and pragmatic categories (Figure 5.4).

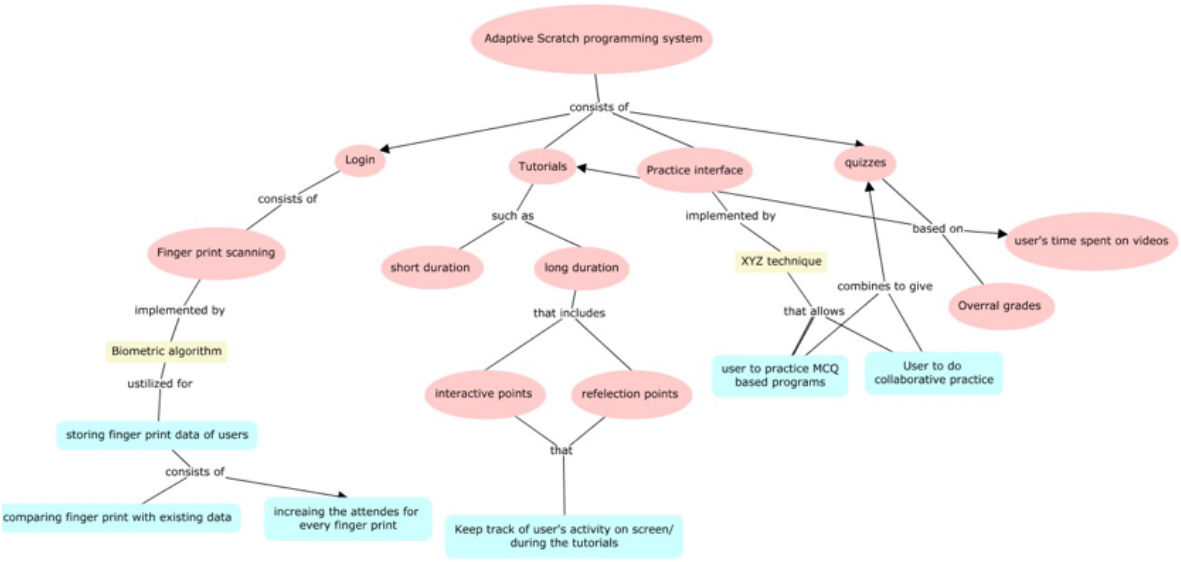


Figure 5.4 Participant1 FBS graph for phase III task in FBS graph based intervention I

Participant 1 began with functions and the design process of problem formulation (F->Be). In phase II as well as phase III, participant 1 did not create the FBS graph at the expected performance levels. In phase II, participant 1 was fixated to a part of the problem 'mood detection' and utilized nodes only pertaining to that problem. In phase III, participant 1 created unrelated branches in the FBS graph and the FBS graph did not reflect the complete solution.

5.5.2.2 Participant 2

The participant 2 went through each of the 4 tabs individually and systematically. The participant paused in between the video and took notes. Participant 2 also went through the introduction tab, paused the video several times to take notes. Then went through the information tab and spent about 15 mins in this tab. In the information tab the participant 2 went through the definitions, connectors and FBS sample graphs (1 & 2). During this time the participant had analysed both the sample FBS graphs. After analysing the sample FBS graphs, the participant had made skeleton graphs based on the sample. This participant then went on to the phase II task. In the phase II task, participant 2 started with the behaviour nodes. Participant 2 then moved to search for mood recognition in a search engine. Participant 2 focused on behaviour nodes in the conceptual design template. Participant 2 then searched for a relational database using the search engine. After searching participant 2 asked whether he can add any more nodes than the ones existing. Participant 2 then built an FBS graph (Figure 5.5) starting with behaviour nodes first, connected them, and then moved to the structure

nodes. The FBS graph is lean in terms of nodes of F/B/S. So it does not reach the target level for the criteria in syntax (complexity and levels). The FBS graph is not detailed enough, so the semantic criteria of validity remain unfulfilled.

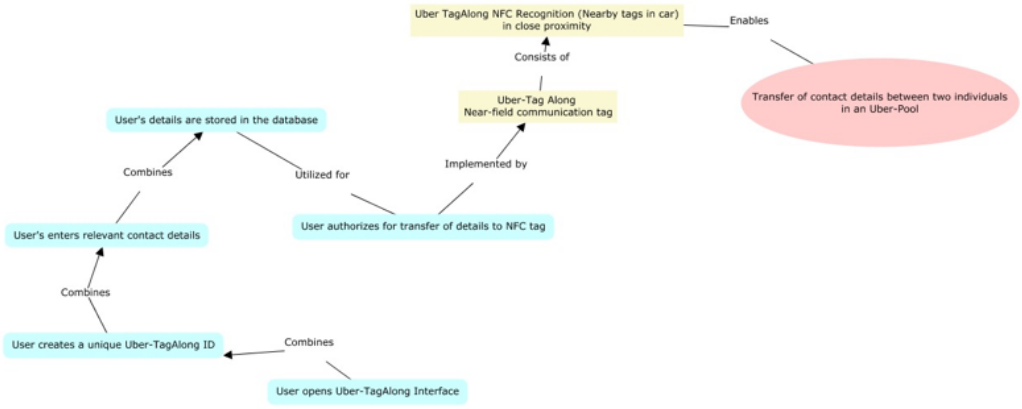


Figure 5.5 Participant 2 FBS graph for task 2(induction task) in FBS graph based intervention I

After going through the phase III task video participant 2 started writing the problem and its corresponding solution as below:

"The Problem:

Aayushi uses Uber-Pool daily for commuting to her office. Often, she engages in vivid and animated conversations with a fellow rider/passenger but is quite hesitant to ask for their name/phone number/business card or their profession in general. How can the social conversation hindrance in the communication or sharing of details between two co-riding individuals in an Uber-Pool be reduced?

Solution:

An NFC (Near-Field Communication) tag can be provided by Uber to their daily Uber-Pool riders. The tag can be placed/attached to the back of the rider's phone. The tag will include relevant details/information such as name, email-id, phone number, profession, social media details. When brought in close proximity to the NFC tag of another rider in the Uber-Pool, there will be an exchange of details between the two individuals. Why? By use of NFC tags, a lot of time will be saved in physical noting down of phone numbers, email IDs. Not all riders (students) will have their business cards to be shared readily. Also, this might be useful for those who are shy and not willing to directly ask for someone's phone number or business card. They can just say 'Can I tag your details for taking this conversation further?'

To come up with the solution, participant 2 referred to the internet for UBER NFC, NFC business cards. After getting the information, the participant drew a rough sketch of the FBS graph. While making the rough sketch of the FBS graph participant 2 referred to the sample graphs and the connector types of FBS graph 1 more frequently. The participant then used the CMAP tool to create behaviour nodes first. The participant populated the CMAP interface with some behaviour nodes, then connected the behaviour nodes. The participant then draws a structure node and a function node and connects them. The FBS graph is lean in terms of nodes of F/B/S. So it does not reach the target level for the criteria in syntax (complexity and levels). The FBS graph is not detailed enough, so the semantic criteria of validity remain unfulfilled. As seen in Figure 5.6, participant 2‘s FBS graph is neither comprehensive nor cohesive. In all the categories of conceptual model quality (Lindland et al., 1994), the FBS graph does not meet the target performance.

Participant 2 in task phase II as well as phase III started with the behaviour nodes and then moved to the structure nodes. In both the phases participant 2 spent a lot of time taking notes about the FBS conceptual model, however was unable to translate it to a FBS graph. In both phases the FBS graph did not reflect the model of the software solution to the respective problems.

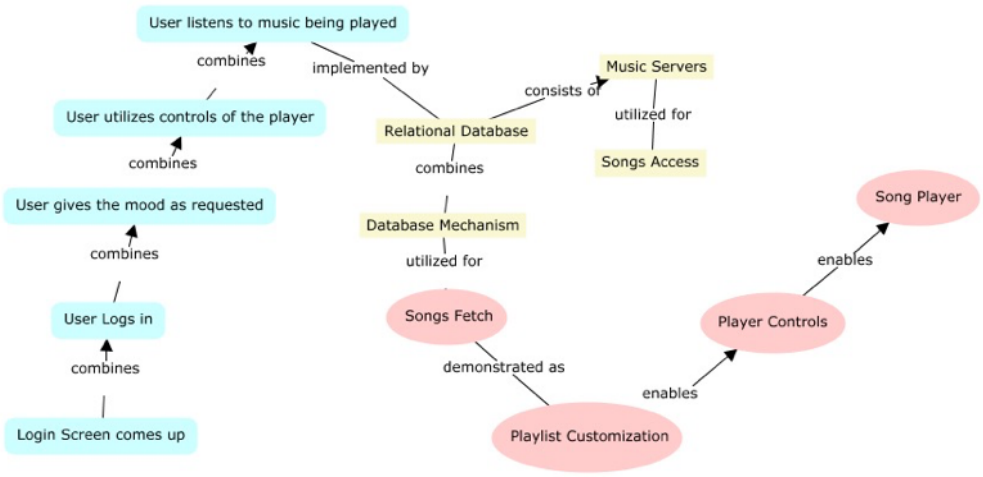


Figure 5.6 Participant 2 FBS graph for task 3 (ideation) in FBS graph based intervention I

In the next section we detail the results of the analysis of participants’ interview responses to bring out the difficulties that they face in FBS based intervention I (RQ 2.b)

5.5.2.3 Thematic Analysis of participants' responses

In this section we present the participants' reflections on difficulties in the FBS based learning intervention I phase and task wise -

Building the FBS Conceptual Model (Phase I, Task 1, Learning Objectives 1.a. & 1.b)

- To build the FBS conceptual design the participants were provided with the definition of the terms, connectors and some sample FBS graphs for a design problem. It is interesting to note that both the participants utilized different sample FBS graphs for the same design problem. Participant1 utilized the top-down FBS graph, whereas participant 2 used the sample graph 2. Participant 2 went ahead to say that for the conceptual design task it was easier to start from behaviours, move to structure and then functions. Participant 2 articulated that the sample graphs only provide the idea of use case to him. He compared the use cases in the sample graphs and then evaluated if they were applicable to him and dismissed them. Participant 1 stated that the FBS graph helped to plan.
- Participant 1 stated that the FBS graph connectors provided were deterring to connect the FBS elements in the conceptual design template. This indicates that maybe the – i) FBS conceptual model that this participant built might not be the intended one, or ii) language in the connectors are not very intuitive to the learners which facilitates connection.
- Both the participants agreed that there was a lot of information, which needed to be sieved through, to build the FBS conceptual model. This can also be seen in participant 2's process description. The participant 2 almost spent ~ 45 minutes in the Information tab alone.

Connecting the FBS design elements (Phase II Task 2, Learning objectives 2.a & 2.b)

In this phase the tasks required the participants to refer to the sample FBS graphs and definitions a number of times.

- Participant 1 pointed to the lack of any directions for strategy to connect the nodes, but later found one 'FBS grouping'. Participant 2 also followed their own strategy of starting with the behaviours.
- Participant 1 demonstrated 'conceptual fixation' where the FBS graph only consisted of different ways to implement mood detection.
- Participant 2 demonstrated planning, monitoring the tasks but mentioned that was unable to translate it to the FBS graph. So mentioned the requirement for a 'self-evaluation checklist' for the FBS graph.

- Both the participants mentioned that they would require scaffolds to create a FBS graph in this task. They also mentioned that if they were to be allowed to collaborate the FBS graph would be richer in terms of ideas.

Creating FBS design elements (Phase III, Task 3, Learning objectives 3.a & 3.b)

- Participant1 mentioned that this task gave some pointers to generate FBS elements.
- Participant 2 stated that the reflection task made the processes explicit.
- However both participants mentioned the need for scaffolds and prompts to create the FBS graph as a solution for the design problem.
- The participants also mention the need for a self-evaluating framework to evaluate the FBS graph.

5.5.2.4 Answering the research questions

2.a After interacting with the FBS based learning intervention I participants are unable to create FBS graphs at the target level performance of the rubric

In study 2, participants (n=2) went through the FBS graph based learning intervention I. Participants need to create FBS graphs in phase II and phase III. We evaluated the participants' FBS graphs based on an adapted rubric (Table 5.3) of conceptual model quality (Lindland et al., 1994). Both the participants did not create FBS graph at the target level of the categories syntax, semantics and pragmatics.

In phase II task, participants were unable to reach target performance in semantic category. In this task the FBS nodes were already provided to the participants and they had to pick and connect them. In phase III the FBS graphs were not in the target performance of both syntactic and semantic categories.

2.b While using the FBS based learning intervention I participants have difficulties in building the FBS conceptual model and the FBS graph.

The sample FBS graph, in phase I has utilities, such as planning, and strategy building however it does not fulfil the objectives of helping the participants' build the syntactic and semantic model of the FBS graph. In phase II participants have mentioned the need for scaffolds or collaborators to link the various FBS nodes to create FBS graphs. Similarly in phase III as well the participants have mentioned the need for scaffolds. Participants in the study also mentioned the need for a self-evaluation checklist for the FBS graph.

5.5.2.5 Summary of Study 2 results

- The participants when left alone to build the FBS conceptual model from the definitions and sample FBS graphs picked the graph type that appealed most to them. However the participants point out a lack of scaffolds to build the conceptual model of the FBS graph.
- Participants mentioned the need for a collaborator to discuss ideas in phase II so that they could connect as many nodes and create a richer FBS graph.
- Participants pointed out the requirement of an evaluation framework to be made available, so that they can self-evaluate the FBS graph in phase II and III.

5.6 FBS graph based intervention II

The results from study 2 prompted us to redesign the task and features of the FBS based learning intervention. In this section we describe the changes we made for this specific intervention of FBS based learning intervention II:

Phase I - The participants in the previous study pointed out that building the FBS conceptual model was overwhelming as there was a lot of information they had to process. There was no assessment of the conceptual model they built which could scaffold the process. To be able to do so we prepared a worksheet, which has a set of question prompts. These question prompts are designed systematically to :

- Interpret syntactically and semantically meaning of the design elements individually F/S/B: This has a set of tasks starting from listing the FBS nodes from graph, evaluating statements with respect to FBS nodes from the graph and describing their own understanding of FBS individually
- Interpret syntactically the relationship of F-B-S: This has a set of tasks starting from identifying connectors and stating a FBS link from the graph
- Interpret semantically the relationship of FBS: Abstracting the relationship of FBS

Phase II – The participants in the previous study indicated that an evaluation framework to self-evaluate would have helped them to construct a richer FBS graph. So for this task the participants were provided with the rubric (Table 5.2) based on the quality of conceptual model (Lindland et al., 1994). The participants were told to apply the learning from the previous task (reflection/sample FBS graph). This was provided to the participants to perform self-evaluation of the FBS graphs created. The participants previously also had mentioned that they would prefer collaborators to discuss ideas and connect the various FBS nodes. So we made this phase as a

collaborative phase, where all the three participants could discuss and create one FBS graph.

Phase III – In the previous study the participant 2 had mentioned that working on a familiar problem (pet project) was easier as they could apply the unknown concepts of FBS and FBS graph creation to a known problem. So for this task we retained the task of setting their own problem and constructing the FBS graph.

The Table 5.4 below captures the tasks and the features in the FBS based learning intervention II.

Table 5.4 FBS graph based intervention II - tasks and learner activities

Phase No	Task Design	In FBS based learning intervention II	
		Problem	Task
1	Create FBS Conceptual Model in Problem 1 Context	Design a Fingerprint ATM system	Worksheet that has a set of question prompts. These question prompts are designed systematically to * Interpret syntactically and semantically meaning of the design elements individually F/S/B * Interpret syntactically the relationship of F-S-B * Interpret semantically the relationship of FBS
2	Connect FBS Design Elements in Problem 2 context	Design a automatic mood based music player	Use the template FBS nodes to connect them and create one FBS graph collaboratively. Rubric based on conceptual design quality (Lindland et al., 1994) adapted for FBS graph provided to

			the participants to perform self-evaluation of the FBS graphs created.
3	Create and Connect FBS Design Elements in Problem 3 context.	Retained the task of setting their own problem and constructing the FBS graph.	FBS nodes are created and connected. The FBS graph is created and connected in the IHMC CMAP tool.

5.7 Study 3 – Qualitative Evaluation of FBS based learning intervention II

The broad RQ in this study was similar to study 2, "How to support novices' while learning SCD in FBS based intervention?" The research questions guiding this study was -

2.a After interacting with the FBS based learning intervention, what are the kinds of FBS graphs that learners create?

2.b What difficulties do learners' experience while using FBS based learning intervention for SCD?

5.7.1 Method

Participants - The participants ***(n=3)*** for this laboratory study were selected via purposive sampling. The participants of the workshop were final year computer engineering students. The participants were recently exposed to undergraduate engineering courses such as 'Structured Object Oriented Analysis and Design' (semester 5) and 'Software Engineering' (semester 6) in their third year of engineering. These two courses cover topics of software design approaches, software-modeling tools, characteristics of software solution etc. As the course contents included such concepts, it was appropriate to consider that they had prerequisite knowledge for the SCD activity. The participation in the workshop was voluntary.

Study procedure and Design problems – Study 3 was conducted as a post-test only study individually with three participants. The participants completed the activities in the learning intervention as mentioned in the Table 5.4. After doing so, the last activity was considered as post-test. The participants took 3 hours to complete the activities. We conducted a reflective semi-structured interview by asking questions such as – *'How did you do this task? What are the difficulties that you faced in this task?'* These questions made the participant reflect on the difficulties that they faced during the task.

Data sources and analysis – In study 3, similar to study 2, for RQ 2.a. we utilized the artifacts that the participants created. We used the adapted rubric of FBS graph based on quality of conceptual model (Lindland et al., 1994). Refer to section 5.5.1.2 where we have adequately discussed the framework of conceptual model and our operationalization of the framework to FBS graph. The FBS graph is a conceptual model of the software solution that the participants have created. The Table 5.3 is the rubric of the FBS graph that is used to evaluate the final task in the intervention.

We also transcribed and analysed the participant's interview responses to answer RQ 2.b. We employed thematic analysis (Braun & Clarke, 2017) to inductively come up with themes of difficulties in the FBS based learning intervention II. In the next section we describe the study 3 participants' task-wise approach.

5.7.2 Study 3 – Results

In this section we begin by describing participant's process in each of the phases of the FBS learning intervention II. We then present the results of the thematic analysis of the participants phase and task wise in the FBS based learning intervention II. We close this section by summarizing the answers for the research question based on the results.

5.7.2.1 Phase wise analysis of participant's process

Phase 1 – Using the worksheet the participants built the conceptual model of FBS as they were able to answer all the questions based on the FBS graph.

Phase 2 – In this task, the participants started the activity of FBS nodes connection by collaborating with each other. After 45 minutes into the task, the participants were fixated with the behaviours of authorization and profile creation. As seen in the Figure 5.7, the FBS graph is rich in terms of behaviours. But all the behaviours pertain only to the functions of login and biometric. All the participants were debating and

discussing the behaviours and flow with regard to profile creation. This fixation led to the task time expanding to 2.5 hours. At the end of 2.5 hours, we called for a task time end. By the end of this the participants had only grouped function and behaviour nodes. During the collaboration activity all the participants did not pay attention to the rubric, which was provided, to guide connecting the FBS nodes and the final FBS graph artifact expected from them. This FBS graph does not reach the target levels in syntactic (complexity and levels) as well as semantic (validity and consistency).

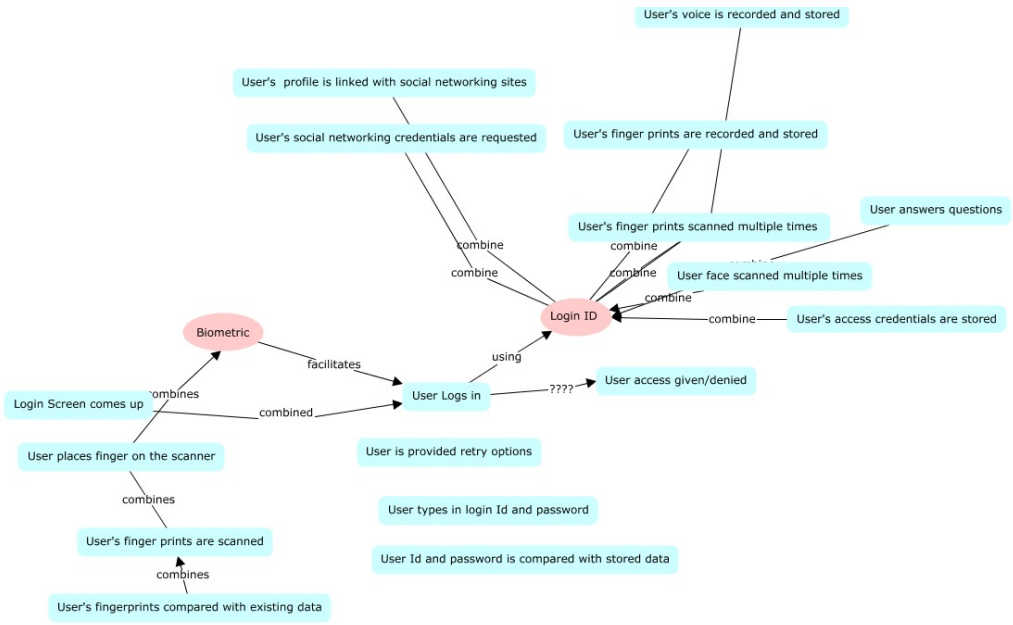


Figure 5.7 FBS graph for phase 2 in FBS graph based intervention II

Phase 3 – All the participants chose the design problem of "A Classroom Information and Assignment Submission system". This was because the 3 participants were building together the same software for their final year project. All the participants preferred to perform this activity individually on a paper rather than the IHMC Cmap tool. It took the participants ~ 4 hours to get to this task. This could have resulted in the low motivation and engagement in this task. However one participant created a FBS graph on pen and paper, at the target level of the criteria syntactic, semantics and pragmatic (Figure 5.8 below). The Figure 5.8 indicates that participants created many nodes for F/B/S and connected them. So the FBS graph reached the target levels in complexity, connectivity and levels. In the semantic category the FBS graph had enough details about the working of the solution. The design solution reaches the target performance in all criterions in the semantic category (validity, consistency and

level adjacency). The FBS graph is also convertible to the UML diagrams of use case, class diagram and sequence diagram. In the post interview, the participant revealed that the FBS graph helped to plan and tag the ideas of the solution software design.

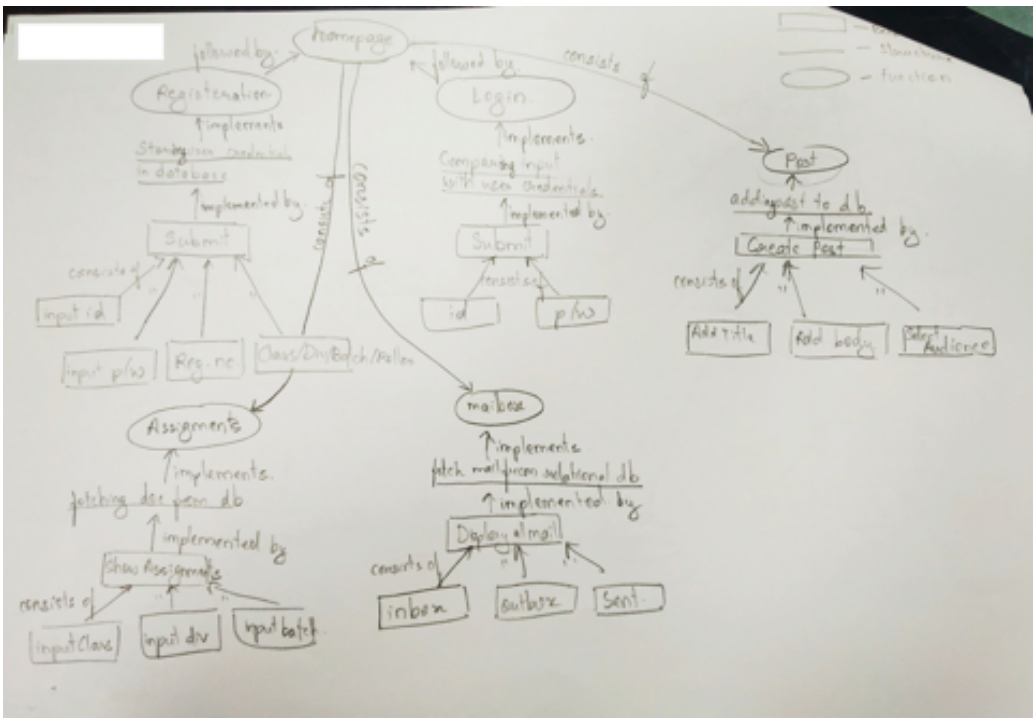


Figure 5.8 Study 3 - participant creating a target FBS graph based on rubric (Lindland et al., 1994)

5.7.2.2 Answering the research questions

Based on study 3, we present the answers to the question below:

2.a After interacting with the FBS based learning intervention, what are the kinds of FBS graphs that learners create?

In study 3, participants (n=3), went through the FBS graph based learning intervention II. In phase II we see that participants could only connect function and behaviour nodes. This indicates that even though all the FBS nodes are provided participants tend to begin from problem formulation (F->Be). Additionally we see that even though the phase II task was collaborative participants did not create a FBS graph at the target performance of syntax, semantic and pragmatic categories. In phase III one out of the three participants created a FBS graph at the target level of all the categories.

2.b What difficulties do learners' experience while using FBS based learning intervention for SCD?

In phase II, where the participants had to create a FBS graph from a given set of nodes fixation was a deterrent. The participants were fixated towards the functionalities of login and biometric authentication. These functionalities are only a part of the functionalities in the solution design.

In phase II and III the participants did not utilize the rubric provided for this task to evaluate the FBS graph during the task. The rubric would have provided the necessary pointers to create a rich FBS graph. From participants' interview responses we come to know that the rubric usage needs to be nudged. Participants also mentioned that they need scaffolds to create a new FBS graph specifically in phase III.

5.8 Summary of results of study 2 and study 3

Study 2 is based on the FBS based learning intervention I. In this study participants failed to create an FBS graph in the target level of the syntactic and semantic categories of the rubric (Table 5.3). Based on the results from this study we made changes to the phases in task design and provided additional tools. This led to the FBS based learning intervention II. Study 3 is based on this learning intervention. In study 3 there is a slight increase in performance of the participants in creating FBS graphs. However, participants still need scaffolds and prompt to create FBS graphs at target level of the rubric (Table 5.3). So we pay attention to participant responses to the problems that they face in each of the phases of the FBS based learning interventions.

We captured participants' response to semi-structured interview questions on the difficulties in the FBS graph based intervention II and I. These responses were transcribed and thematic analysis was done based on the tasks in the interventions. In each of the studies the difficulties in the tasks are captured in the Table 5.5 below.

Table 5.5 Difficulties that novices faced in FBS graph based learning intervention I & II

Task #	Difficulties that novices face in FBS based learning intervention	
	Study 2	Study 3
Task 1	Building FBS based conceptual model based on sample FBS graphs	-
Task 2	• Strategies to create and	• Strategies to create and

Task 3	connect the FBS elements in the FBS graph • Self-evaluation tool/instrument to evaluate the created FBS graph	connect the FBS elements in the FBS graph • Nudge to utilize the rubric to self-evaluate the created FBS graph

In this research cycle**,** we created FBS based preliminary learning interventions and conducted small qualitative laboratory studies (study 2 & 3). The aim of study 2 and 3 was to support novices' while learning SCD in FBS based intervention. The results from this study provide the input to the features and task design for the next version of the FBS based intervention.

However before we summarize this chapter we need to be cognizant about the limitations of study 2 and 3, which are discussed in the next section.

5.7.1 Limitations of study 2 and 3

The criticism that the sample in both the studies are small to make decisions about the pedagogy could arise. In study 2, the sample consists of post-graduate students with design experience. In study 3, the sample consisted of final year computer engineering students. The sample in both these studies consists of people with sufficient software design experience. So the difficulties that they would have had would be found across the undergraduate engineering population as well. So the findings could be generalized to the extent of undergraduate computer engineering students with limited software design exposure. However, there could have been different design decisions across study 2 and study 3 findings. The design decisions that we have taken are based on the FBS graph based pedagogy and the principles we derived from the theoretical foundations as discussed in section 5.2.

5.7.2 Chapter Summary

The literature on expert's conceptual design and specifically software design processes inform us that creating, evaluating and refining the integrated view of the solution leads to good design. Additionally, we chose the FBS design framework (Gero & Kannengieser, 2014) to support creation of conceptual design. In the learning intervention we manifest the framework as an external representation namely the FBS graph. So, FBS graph based pedagogy is the basis of our learning intervention. In FBS

graph based intervention I and II, concept map and worksheet are the pedagogical tools. From our studies (study 2 & 3) we found the difficulties that novices have with the FBS graph based interventions. The results from the studies helped us to come up with the features of a FBS graph based technology enhanced learning environment (TELE). In the next DBR cycle we utilize these findings and integrate it into a learning environment. We then evaluate the learning outcome, in this case the SCD that learners create after interacting with the learning environment.

The Table 5.6 below maps the learning objectives (section 5.4), the tasks and the corresponding features in the TELE. In the Table 5.6, we also conjecture how will the learner learn and the learning design principle that could arise. We discuss these conjectures and principles in the next chapter in detail.

Table 5.6 Mapping the learning outcome, principles and features in the learning environment

Learning outcome	Task	Feature in TELE	How will the feature facilitate learning?	Learning Design Principle
Interpret syntactically and semantically the meaning of FBS graph	Task 1	• Sample FBS graph as an interactive simulation • Worksheet questions	FBS graph will be systematically interpreted to answer the worksheet questions	• Authentic Problems • Worked Example • Question Prompts to systematically interpret the sample FBS graph
Evaluate and modify the	Task 1	Evaluation of FBS graph	Evaluation and	Self-evaluation would promote

FBS conceptual model		based questions in the worksheet	refinement of self's FBS conceptual model	self-regulation (Cho, 2004)
Apply the FBS conceptual model to connect given FBS	Task 2 & 3	• Rubric provided to evaluate and guide the FBS graph. • Performanc e indicators for the desired characteristi cs of SCD	The evaluation with the rubric helps the learner to analyze the FBS graph for desired characteristics of SCD	Scaffold to guide the connection of given FBS
Create and Construct the FBS graph	Task 2 & 3	Prompts and scaffolds to create FBS graph via a pedagogical agent	Cognitive tools to trigger the cognitive processes of mental simulation, abstraction and associatio n	Cognitive Tools for Structure, Behaviour and Function
		Reflection and planning tasks and questions	Opportunities to abstract the process of conceptual design to be made applicable in other similar design problems	Metacognitive Prompts for planning and monitoring

Chapter 6

DBR 2 Problem Analysis and Design of 'think & link'

6.1 Summarizing reflections from DBR iteration 1

In the problem analysis phase of DBR 1 (chapter 4) from study 1, we found that novices have difficulty in utilizing formal representations such UML diagrams. Some novices were unsuccessful in completing the SCD task. Their design solutions addressed only a specific sub-problem, as they were fixated to a particular structure, function or behaviour. The successful novices in study 1 employed all the design strategies in the FBS framework and some expert-like processes such as retrieval and anchoring to known structures, mental simulation of behaviours (end user & system).

The difficulties in novices motivated us to design a FBS based pedagogical intervention for software conceptual design (chapter 5). The FBS framework manifests as a FBS graph in our intervention. We created two preliminary learning interventions based on FBS graph pedagogy. Study 2 and 3 were qualitative case studies using the preliminary interventions. The results from the studies informed the pedagogical design and features of the learning environment in this iteration. In this iteration we design and develop a learning environment based on the findings from earlier cycle. We then evaluate the learner's outcome after they have interacted with the learning environment.

The learning objectives for the FBS based pedagogy for creating integrated SCD are identified in chapter 5 (section 5.4). To achieve learning objective 1, from study 2 we found that learners are unable to implicitly build syntactic and semantic models of FBS graph. So to scaffold and help novices to be able to do this in study 3 we introduced a worksheet, which consists of questions about the FBS graph at various levels -list, evaluate and abstract. These questions guide the learner through the FBS graph thereby building a syntactic and semantic model of FBS (learning objective 1). For learning objective 2 and 3, we realized that the tasks, activities and scaffolds in FBS based preliminary learning interventions II and I were not enough. So in this iteration, we focus on the design and development to address these learning

objectives. The achievement of all the learning objectives manifests as integrated SCD. In this iteration we have focussed on the learners' SCD creation post the intervention. To design the learning environment we reviewed the literature of problem solving and design, which are elaborated in the next section.

6.2 Literature review

To build the scaffolds and features in the learning environment that will support teaching and learning of integrated SCD we reviewed the literature in worked examples, improvable models and metacognition. In the next paragraph we present our reasoning.

From our study 2 and 3 we identified that FBS model building needs to be supported. FBS model building, as a graph, is not very explicit for learners. Since learners encounter the FBS conceptual model for the first time during the intervention, we provided a sample FBS graph as a worked example. However from study 2 we see that a static worked example does not help the learner abstract the conceptual model of FBS. So with the worked example, we need to build interactions around the FBS graph and make it an improvable model. Using the sample FBS graph in the learning environment, as an improvable model, the learners are expected to progressively build on it. We also want the learners to abstract the process of building FBS graphs, so we provide opportunities for evaluation, reflection and planning. The opportunities for meta-cognition provided to learners will help them to abstract the FBS based SCD creation process. So we refer to the theory of worked examples, improvable models and metacognition. This would help us in identifying relevant features and activities in the learning environment.

6.2.1 Worked Examples

The instructional studies (Atkinson et al., 2000) into the use of worked examples in the teaching of problem solving, describe a typical learning process. Given the worked example, learners begin to learn to solve by analogy. They refer to the examples and relate them to the problem to be solved. After relating, the learners abstract the rules and knowledge to build their own strategies. The learners employ and test their strategies multiple times and make adjustments to them.

Worked examples are instructional devices that provide an expert's problem solution for a learner to study (Atkinson et al., 2000). As instructional devices, they typically include a problem statement and a procedure for solving the problem;

together, these are meant to show how other similar problems might be solved. In a sense, they provide an expert's problem- solving model for the learner to study and emulate. Examples typically present solutions in a step-by-step fashion. In many cases, worked examples include auxiliary representations of a given problem, such as diagrams. Since viewing of worked examples is a passive process, questions arise about the transferability of problem-solving skills acquired from merely viewing worked examples (Chi & Glaser, 1985).

Worked examples have been utilized in the teaching and learning of programming (Mark Guzdial, 2012). In his popular blog in CS education (computinged), Mark Guzdial famously said, *"Practice is better for learning facts, worked examples are better for learning skills"*. Additionally, Morrison (2020) adds that, "*Worked examples give learners concrete examples of the procedure being used to solve a problem, showing the explicit steps in the problem-solving process*". To support learners to construct FBS graphs, in our learning environment we employ worked example support. We provide FBS graph to learners for building conceptual models of FBS.

In the FBS graph based pedagogy, we have provided scaffolds to facilitate this process using worked examples. The worksheet has questions that provide the learners opportunities for referring to the FBS graph nodes and connectors then abstracting their syntactic and semantic relationships. Along with the worksheet questions, the FBS graph is provided as a manipulable simulation. In the next section we present the literature on specific cases of worked examples in engineering design known as improvable models.

6.2.2 Improvable Models

Improvable models are designed to be used as primary generators (Darke, 1979; Lawson, 2006) to help students as they start solving an engineering design challenge. These models provide the initial example idea that students can use as a seed for new ideas. One example of an improvable model is the suboptimal system (SS) seed model. It represents suboptimal design due to inefficient design decisions that encompass all parameters of the complete system being represented by the designed model. Students can critique this example model and use it as a seed for iteratively designing an optimal model. The SS seed model visually represents all the design parameters and constraints that define the system. This visual representation will

likely provide an essential scaffold for the students by allowing them to inspect these elements and productively use them for making trade-off decisions, scaffolding their engagement with the disciplinary practices while they are solving the engineering design challenge (Quintana et al., 2004).

According to Wood et al. (2001), improvable models can be used iteratively for understanding the current solution, which is essential for modifying, redesigning, and generating an optimized solution, by students while solving an engineering design challenge. Similar to worked example models, improvable models present sample solutions that will likely guide students' attention by visually representing all the design parameters, both the input and outcome parameters of an engineering system, required for solving the challenge.

The FBS graph is initially provided as a worked example to the learners as a scaffold for them to build the FBS conceptual model. In the next step, the FBS graph is provided as an improvable model. *An improvable model of a FBS graph* would be a scaffold for the learners as they can inspect the visualization to add their own ideas as FBS elements (Dasgupta, 2019). The FBS graph acts as a visualization that can be manipulable by the learners. For the FBS graph based pedagogy, in phase I the manipulations would need to support the building of FBS conceptual models in the learners. The interactions with the FBS graph need to be designed in such a way that the learners are able to inspect the FBS graph elements. In phase II learners build on the existing FBS graph by adding their solution ideas as FBS graph elements. By editing the FBS graph learners build strategies for creating integrated solution design.

6.2.3 Role of metacognition

Investigators have highlighted that metacognition plays an important role in problem solving, social cognition, and various types of self-control and self-instruction (Flavell, 1979). Research has shown that students must be scaffolded in order to articulate and reflect on their inquiry (Quintana et al., 2004) and problem solving (Kim & Hanna, 2011). Elaboration question prompts have been successfully used in ill-structured problem solving to get students to elaborate and explain their thinking (Ge & Land, 2004).

In order to create integrated SCD learners must, at each phase of the FBS graph building process, evaluate their FBS graph for their utility to give the desired conceptual design and plan the next modelling tasks (Jonassen, 1997). This is also

consistent with the multilevel and multifaceted model of metacognition (Efklides, 2008). According to Efklides (2008) metacognition model, for effectively achieving learning objectives of a cognitive task at doing level, the learner must explicitly do metacognition tasks like evaluation and reflection after every few cognition tasks. Based on the literature of metacognition in problem solving and design we have summarized the metacognition features that need to be incorporated for teaching and learning of software conceptual design:

- There exist progressive levels/planes of cognition – doing, evaluation, and synthesis. Learning of FBS based software conceptual design will happen when a learner is explicitly taken through all three levels of cognition.
- To be able to abstract the process of FBS software conceptual design learners need to be provided opportunities for metacognition like planning and control.

From the theory of worked examples, improvable models and metacognition we have identified scaffolds and tasks to be incorporated in the learning environment. In the next section we discuss the tasks and learner activities designed.

6.3 Task design and learner activities in 'think & link' prototype

Based on the study1, study 2; study 3 findings and the review of literature in this chapter we redesigned the FBS based learning intervention. We named the learning intervention as 'think & link'. 'think & link' has evolved from the FBS based learning interventions II and I described in chapter 5.

The goal of 'think & link' is for the learners to be able to create integrated SCD. Integrated term refers to the appropriate utilization of respective UML diagrams and their links with each other. One of the routes that we have undertaken is that learners build FBS based conceptual model of the solution via a FBS graph. By building a FBS graph learners will be able to create integrated SCD. 'think & link' has three phases. The Figure 6.1 below captures the three phases in 'think & link'. The learner task sequence is designed based on the learning objectives elaborated in chapter 5 (section 5.4).

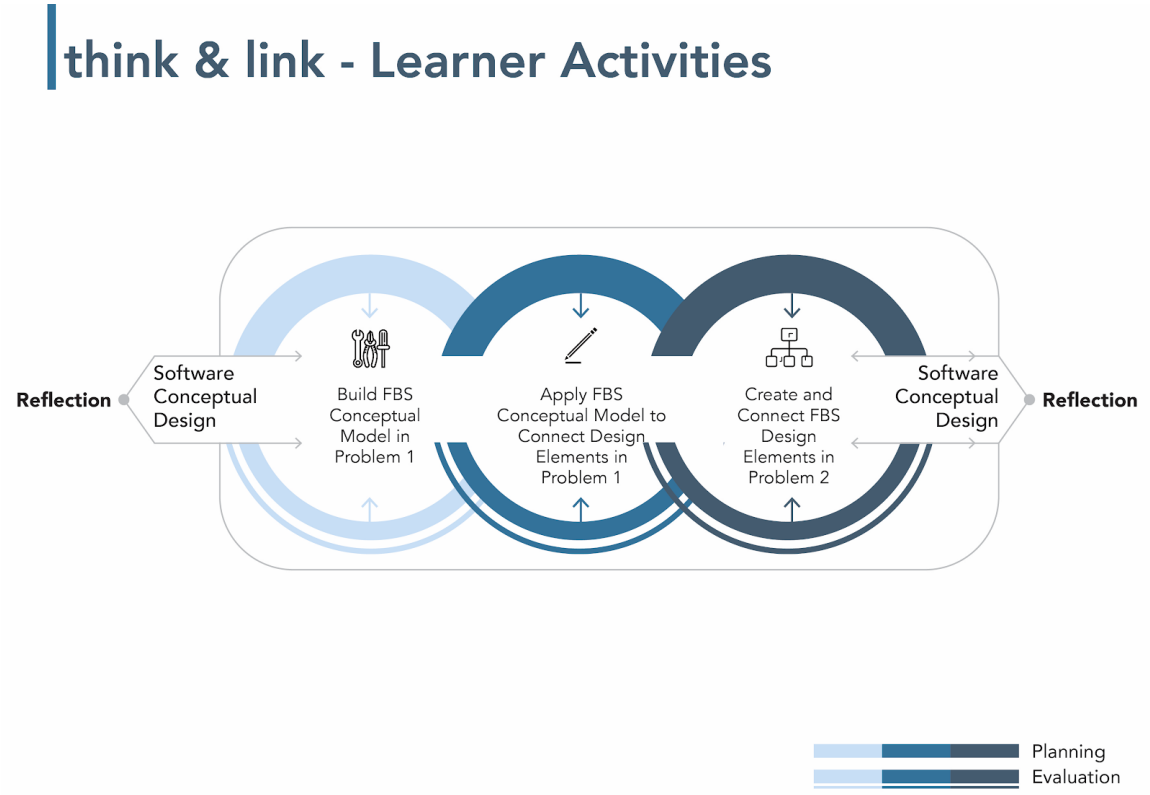


Figure 6.1 Task design in 'think & link'

In phase 1 the problem context (mood based music player), a non-editable FBS graph and a series of questions (activity) is provided to the learners to construct the FBS conceptual model (learning objective 1). In phase 2, the learners are inducted to edit the FBS graph and evaluate it in the same problem context (learning objective 2). In phase 3, the learner creates a FBS graph in the new problem context set by them (learning objective 3).

'think & link' consists of tasks at progressive planes of cognition – doing, evaluation and synthesis. These planes of cognition directly correspond to the phase 1, 2, and 3 in 'think & link'. The tasks are sequenced such that the learners are explicitly taken through all planes of cognition. As depicted in the Figure 6.1, the learning tasks are interspersed with planning, reflection and evaluation prompts and tasks. This is done to trigger metacognitive processes of monitoring, control and planning. The Table 6.1 captures the different phases, the features that the phase offers and the activities that the learner needs to complete.

Table 6.1 'think & link' – task design and learner activities

Task	Feature in TELE	Learner activities
Task 1	• Sample FBS graph as an interactive simulation • Worksheet questions	Learners' have to refer the FBS graph and complete the worksheet questions
Task 2	• Recap of the FBS relationship abstraction	Learners' refer to the worksheet activity and form the sentences that abstract FBS design elements and their relationship. For example, learners need to create sentences such as – *"Functions implements behaviour which is utilized by structure"*
	• Sample FBS graph as an interactive simulation • Prompts and scaffolds to create FBS graph via a pedagogical agent • Information videos on FBS and FBS graph	Learners' edit the given FBS graph to add F/B/S elements and connect them
	• Rubric provided to evaluate and guide the FBS graph. • Performance indicators for the desired characteristics of SCD • Reflection and reasoning questions to abstract	Learners' evaluate the FBS graph based on the rubric. Learners' reflect on the evaluation with the rubric to analyse the FBS graph for desired characteristics of SCD.
Task 3	• Text box to provide the problem that the learners	Learners' edit the problem statement and create FBS graph by adding F/B/S

	intend to create SCD for • Prompts and scaffolds to create FBS graph via a pedagogical agent • Information videos on FBS and FBS graph	elements and connecting them
	• Rubric provided to evaluate and guide the FBS graph. • Performance indicators for the desired characteristics of SCD • Reflection and reasoning questions to abstract	Evaluate the FBS graph based on the rubric. Reflect on the evaluation with the rubric to analyze the FBS graph for desired characteristics of SCD.
Task 1, 2 & 3	• Planning questions before every task	Opportunities to abstract the process of conceptual design to be made applicable in other similar design problems

We created a prototype, conducted heuristic evaluation for usability, redesigned the interface and then took it to students for evaluation. The screen shots of the initial prototype and features are presented in the Figure 6.2, 6.3, and 6.4 below.

In phase 1, the sample FBS graph for a design problem (mood based music player) is provided. As shown in the Figure 6.2, the sample FBS graph, the pedagogical agent along with the prompts to complete the task is shown to the learners. The task in phase 1 is to complete the worksheet questions, which is provided after clicking on the next button.

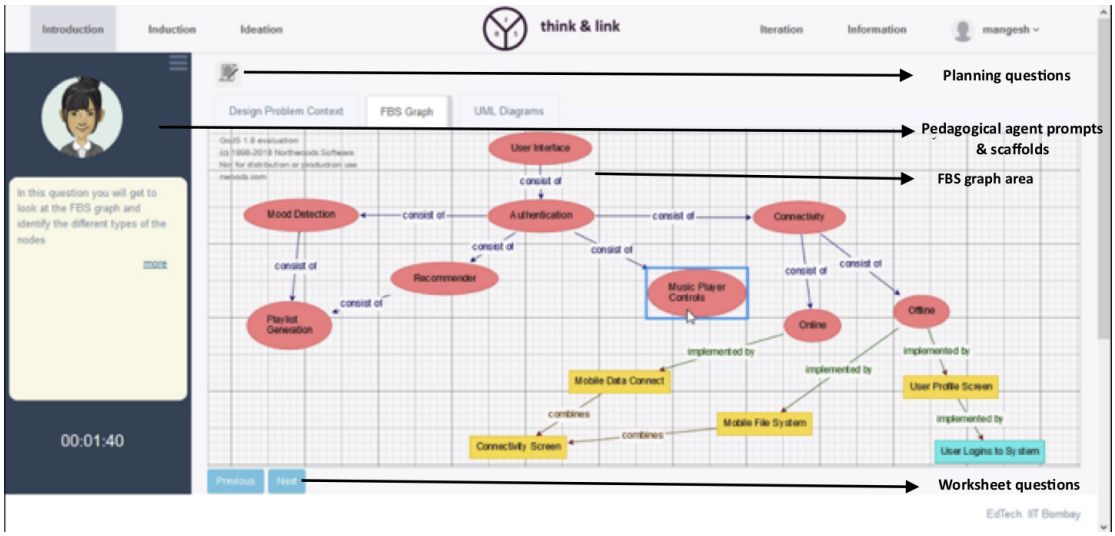


Figure 6.2 Task 1 - FBS graph as a worked example in 'think & link' prototype

In phase 2 the learners are first provided with space to recap their FBS conceptual model, which is the outcome of phase 1. As shown in the Figure 6.3, the phase 2 starts with the activity of reflection of the FBS conceptual model. A word cloud consisting of various FBS elements and the connectors are provided. Learners need to create sentences using the words from word cloud that reflect the FBS conceptual model.

Figure 6.3 Task 2 - Recap of FBS conceptual model in 'think & link' prototype

After completing the reflection activity, in phase 2 learners would need to add the FBS nodes, connectors and evaluate the FBS graph. As seen in Figure 6.4, learners are provided with the rubric as a wheel. The performance indicators for a specific target are provided just below the wheel. The target levels are provided just

below the FBS graph space. The learner needs to click on the criteria wheel option, select the target level, look at the performance indicators and choose the right level. Once the learner completes this, they would have to provide a reason for their choice and plan for the changes that they would make in the FBS graph. The learner can also edit the FBS graph and move on to the next evaluation criteria.

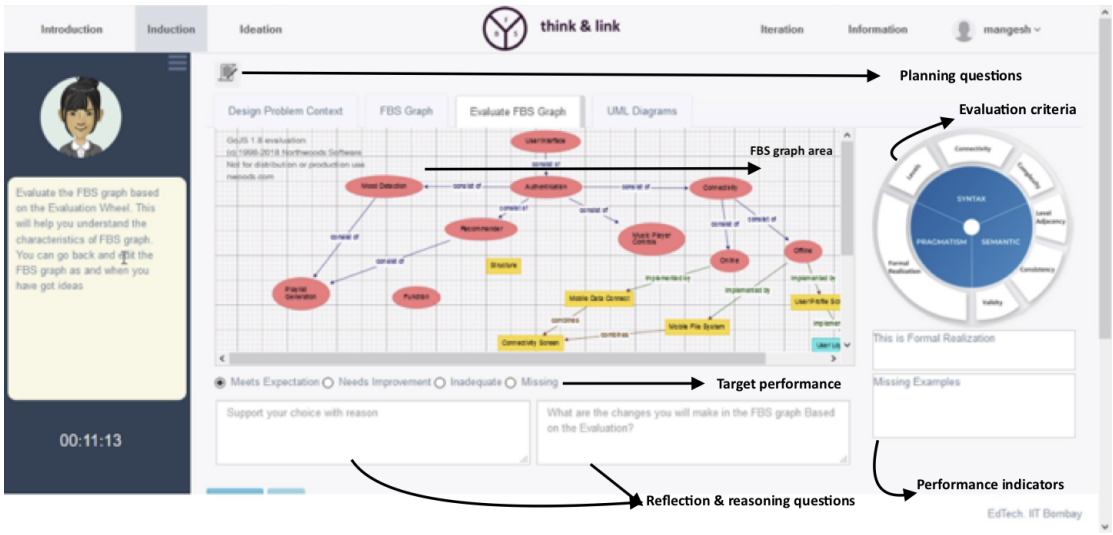


Figure 6.4 Task 2- Evaluation of FBS graph based on the rubric in 'think & link' prototype

Phase 3 had similar activities as in phase 2, however the design problem needed to be set by the learner. The learners' set their own design problem and start creating a new FBS graph from a blank graph area. The learners need to create a new FBS graph and evaluate it (see Figure 6.4). Across the phases, the learners are provided with the prompts from pedagogical agent based on the context of the task and their performance. So the agent is adaptive. The learners are also provided with planning and reflection questions. To complete the task in each phase learners are provided with information in the form of videos. All the videos are available in the information page of 'think & link'.

6.4 Heuristic evaluation and user experience redesign of 'think & link' prototype

We designed and created the learning environment, 'think & link' incorporating the task structure, learner activities and features as discussed in the previous section. We performed a heuristic evaluation for usability and redesigned the learning environment.

Learner's engagement with the learning environment needs to be designed for *"non-automatic, effortful thus metacognitively guided process"* (Salomon et al., 1991). Design of an online learning environment can influence the student learning process and experience (Gearing, 2012). Poor design and usability are found to be detrimental to learner motivation, which then leads to high rates of attrition (Minocha & Sharp, 2004; Ssemugabi & De Villiers, 2010). However, designing tasks to facilitate learning is distinctly different from designing for basic user tasks. Traditional user-centred design is focused on helping people complete tasks they already know whereas learner-centred design focuses on helping understand novel knowledge (Quintana et al., 2008). Given that the content is highly intertwined in the context the application of best practices in learner-centred design becomes challenging. It is important that the learning environment is evaluated for usability.

There exist various usability evaluation methods. These methods are helpful to understand the problems in the design, which creates difficulties for learners and impacts their learning performance. We used one such popularly used usability evaluation method- heuristic evaluation. Heuristic evaluation involves experts to evaluate a user interface based on predetermined usability criteria (Neilsen, 1992). After heuristic evaluation, specific usability problems were identified and the redesign of the user experience in 'think & link' resulted in new user interfaces. The procedure for evaluation and redesign is as depicted in Figure 6.5. It starts with understanding the users of the learning environment. In the case of 'think & link', undergraduate computer engineering students are the users. After profiling the users, the next step was to set usability goals for the learning environment. There are available standard tools created by HCI designers such as Usability Goal setting Tool (Joshi, 2009). We utilized the tool to set goals and evaluate the designed interface against them. A usability expert did the evaluation. After the evaluation the interfaces that violated the goals were redesigned based on design heuristics. We collaborated with a usability expert with educational and industry experience in usability. The details of each step are presented below in the next subsections.

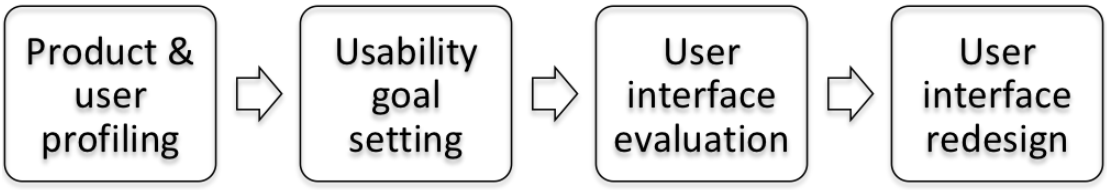


Figure 6.5 Process of heuristic evaluation and redesign

6.4.1 Product and User profiling

The process begins with the designers creating the product profile and one or more user profiles. These form the input for the usability goal setting. The product profile contains questions related to the product domain, platform, cost, and targeted users. By answering these questions, the goal of the learning environment and its intended usage was made explicit. This step helped define 'think & link' as a product in the learning domain, available free of cost for users having internet access with a desktop. The user profile involved questions related to end-users' age, technology savviness, and the expected frequency of use. This step provided reconfirmation that third and fourth-year computer science engineering undergraduates were the end users of 'think & link'. Additionally, by answering these questions the designers were able to identify the minimum criteria for qualifying as a user and any situation that disqualifies them. 'think & link' would require the learners to have prior knowledge of UML diagrams. The deterrents for the usage of 'think & link' are lack of internet and a desktop/laptop.

6.4.2 Usability goal setting

The next step in this process was to set and prioritize the goals of the product. By goal, we refer to the criteria against which the product will be evaluated. E.g., "think & link provides scaffolds and affordances to create an integrated FBS graph". We utilised an available tool for usability namely, Usability goal setting tool (UGT). UGT helps specify high-level usability goals and break them down into concrete, measurable goal parameters (Joshi, 2009). UGT provides a list of 30 goals consisting of 6 categories – learnability, speed of use, ease of use, ease of communication, error-free use and subjective satisfaction. We adapted the list to our context and added a goal each in the category ease of communication and error-free use. We added 2 more goals in the new category of teachability. So in total, we had a set of 34 goals in seven categories.

Once the goal list was ready the usability expert and I assigned weights (0-5) to the goals. The weights indicated the priority of the goal for the product. The higher goal weight indicated the criticality of the goal to the product. We individually assigned the weights to the goals based on the profiling (product & user) and the teaching-learning objectives. After individual weights assignment, we had several rounds of discussion and arrived at a list of goals and their goal assignments for 'think & link'. The list of goals and their criticality assigned is presented in Figure 6.6 below.

6.4.3 User interface evaluation

For the redesign of the actual environment, we considered all goals with weight above 3. These goals were a unique selling point (5) and critical (4 & 3) and are recommended for product evaluation. Each of the screens was evaluated against the goals and the goals that misaligned were noted down.

Goal #		0	1	2	3	4	5
	Learnability						
1	Findability options/data/information should be visible/easy to find				x		
2	User should take less time to learn (e.g. in < 10 mins, in < 2 hrs of practice, in < 2nd attempt)			x			
3	Users should be able to learn on their own				x		
4	Product should be internally consistent				x		
5	Products should be consistent with other products, older methods/past habits of users				x		
6	Product should be consistent with earlier version	x					
7	User should remember / retain critical, but infrequent tasks				x		
	Speed of Use						
8	User must be able to do the primary task / the most frequent tasks quickly, easily, at all times					x	
9	User should be able to navigate quickly and easily						x
10	Product should not load user's memory / product should not put cognitive load on a user						x
11	Flexibility: user should control the sequence of tasks				x		
12	User should be able to complete frequent / critical tasks in specific time / no of steps / in less efforts					x	
13	Product should be personalized for the user automatically			x			
14	Product should be localised for specific market segments		x				
15	User should be able to customize the product for himself		x				
	Ease of Use						
16	Interface should clearly communicate the conceptual model					x	
17	Intuitiveness:User should be able to predict the next step/task					x	
18	No entry barrier: user must be able to complete critical tasks			x			
19	Product should require no unnecessary tasks				x		
20	Product should automate routine tasks / minimise user task load			x			
21	Product should always be on, always accessible		x				
	Ease of Communication						
22	Information Architecture: Information should be well aggregated, well categorised, well presented				x		
23	Communication should be clear / user should easily understand text, visuals						x
24	Modes of communication needs to be redundant through multiple channels			x			
	Error-free Use						
25	Product should give good feedback / display its current status						x
26	Product should not include errors				x		
27	Product should tolerate users' errors / forgiving interface / should prevent errors						x
28	Product should help user recover from errors / help users troubleshoot problems					x	
29	Product should be to handle constraints on impossible situations to avoide error					x	
	Subjective Satisfaction						
30	User should feel in control of the product / behavioural appeal			x			
31	User should feel emotionally engaged with product / brand / product should be fun / reflective appeal					x	
32	User should find the product aesthetically appealing / product should have visceral appeal				x		
	Teachability						
33	User should be able to learn control options available in youtube videos						x
34	Product should have consistency in mapping of controls and corresponding actions						x

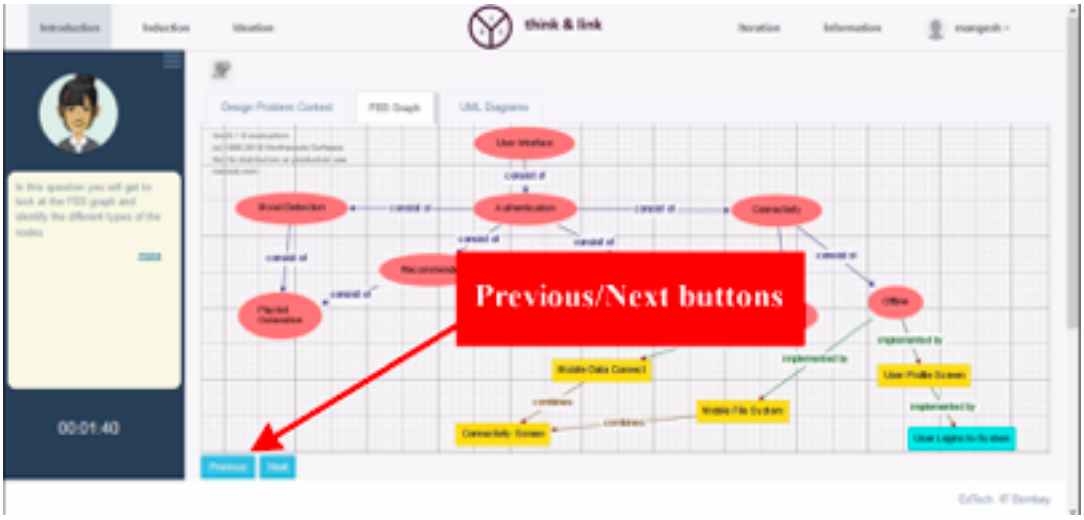
Figure 6.6 Usability goal setting

6.4.4 User interface redesign

Based on the defined list of goals, in each of the screens, the design changes were recommended. For example, in Figure 6.7, we can compare the old and redesigned version of a screen. Here the goal 'user should be able to navigate quickly and easily', was found to be misaligned. One of the major challenges was to design a navigation menu, whose conceptual model could be quickly grasped by the learner. For instance, the next/previous button in the old design was recommended to go along with the

main tab menus in the top such that the sub-menus could form a branched structure. Considering the goal category of 'ease of communication', the buttons were also given contextual labels like intro/task instead of labels like next/previous. Similarly, all the design changes were then collated to create the redesigned interface. In the next section, we discuss the changes for the misaligned goals.

Old screen



Redesigned screen

Figure 6.7 Introduction screen, before and after heuristic evaluation and redesign

6.4.5 Redesigned user interface

The usability expert examined each screen of the application for violation of the usability goals mentioned in the previous section. For the redesign of the interface, each element on the screen was manipulated for its size, shape, and colour, to build an

optimized visual hierarchy. Some of the changes done in the interfaces for the category of goals are discussed in Table 6.2. The prototype of the redesigned interfaces was created on the tool Adobe XD. The newer interfaces are available in the link - https://thinknlink.tech/

Table 6.2 User experience redesign in select screens

Category of Goals	Specific Changes
Learnability	• Creation of conceptual model with a visual hierarchy built for screens and custom colour palette for consistency across the product model • Creation of phase and task information structure • The affordance and emphasis provided in the selected elements of the screen indicate the actions that the learner could perform (e.g. button labels)
Speed & Ease of use	• Removal of unnecessary information in the pages which distracts the core function to be performed on that screen • Task pages do not have a scroll down. All the task information is provided on the fixed screen. • The consistency and colour mapping of the buttons guide the learners to take the appropriate action in the FBS graph screen
Ease of Communication & Error-free usage	• Task Information provided by the pedagogical agent to the learner is given in small sentences and also grab the attention of learners • Task progress indicators in phase II & III graph evaluation wheel convey the progress and task to be completed • The layout of the multiple groups of elements on the evaluation screen is positioned to direct a

	specific flow of action

6.5 Features in 'think & link'

'think & link' is a web-based, self-paced, FBS framework based learning environment for teaching and learning of integrated SCD. 'think & link' consists of scaffolds for learners to create, modify and evaluate a FBS graph for design problems. The task sequences in 'think & link' are based on the learning objectives (see section 5.4 in chapter 5). There are three phases in 'think & link' (see Figure & Table 6.1).

'think & link' has the following features:

- FBS graph manipulator and editor – This feature is present throughout the three phases. However in the first phase alone the editor options are not provided to the learners. The graph manipulator displays the FBS graph for the problem with color-coded nodes (see Figure 6.8). The clickable options on the right panel help the learner to display similar nodes, links and adjacent nodes. In the edit mode the right panel extends clickable options to add function, structure and behaviour nodes. Dragging the cursor from the source node and placing it on the destination node creates the link. The link can be annotated with tags - 'implemented by, consists of, and combines' by right clicking on the link. Using the activity, manipulator and clickable options in the introduction phase learners build FBS conceptual models. This model helps the learners conceptually link F/B/S together. By editing FBS graphs learners build strategies to create and establish links between F/B/S for a given design problem.

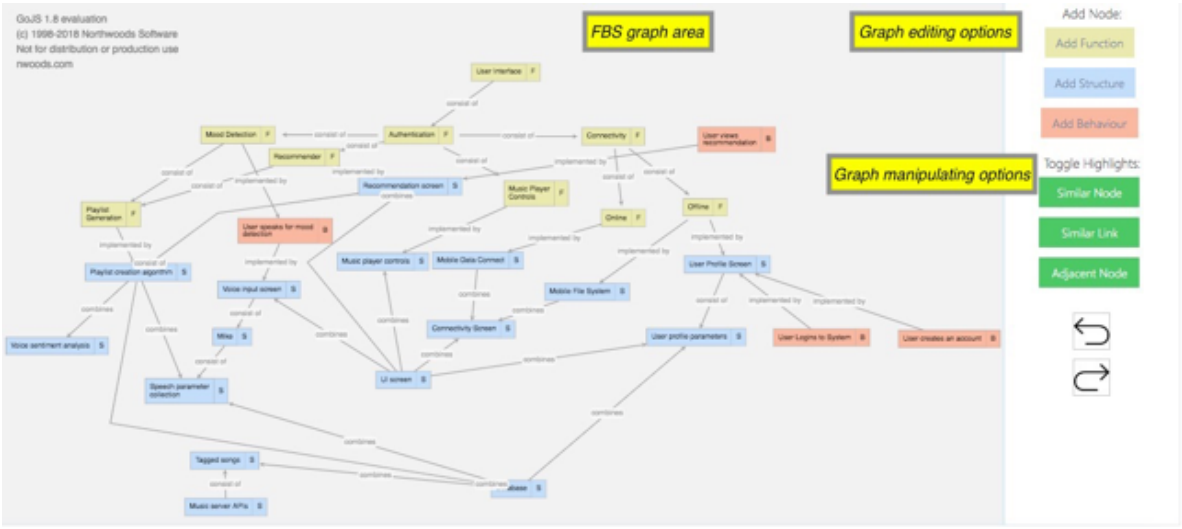


Figure 6.8 FBS graph editor options

- FBS graph evaluator – This feature is present in the phase 2 and 3. It aids in the self-evaluation of FBS graphs based on criterion of syntactic, semantic and pragmatic categories. The criteria of the conceptual model were adapted from Lindland et al. (Lindland et al., 1994). The categories include properties like connectivity, complexity, consistency, validity, consistency, levels and formal realization. All these parameters are adapted and presented in the context of the FBS graph. The evaluation categories are presented as a clickable wheel (see Figure 6.9). Clicking on an evaluation criteria the performance levels (meets expectation, needs improvement, inadequate, missing) are presented as radio buttons. The explanation of the criteria and the respective selected performance level is presented to the learner. The learner has to select the performance level after evaluating the FBS graph. The learner needs to support the performance level choice with reason and state the corresponding changes in the FBS graph that the learner would make. The learners' self evaluate the categories of SCD in the context of the FBS graph. Learners are also required to provide reasoning for the evaluation and reflect on the changes in the FBS graph.

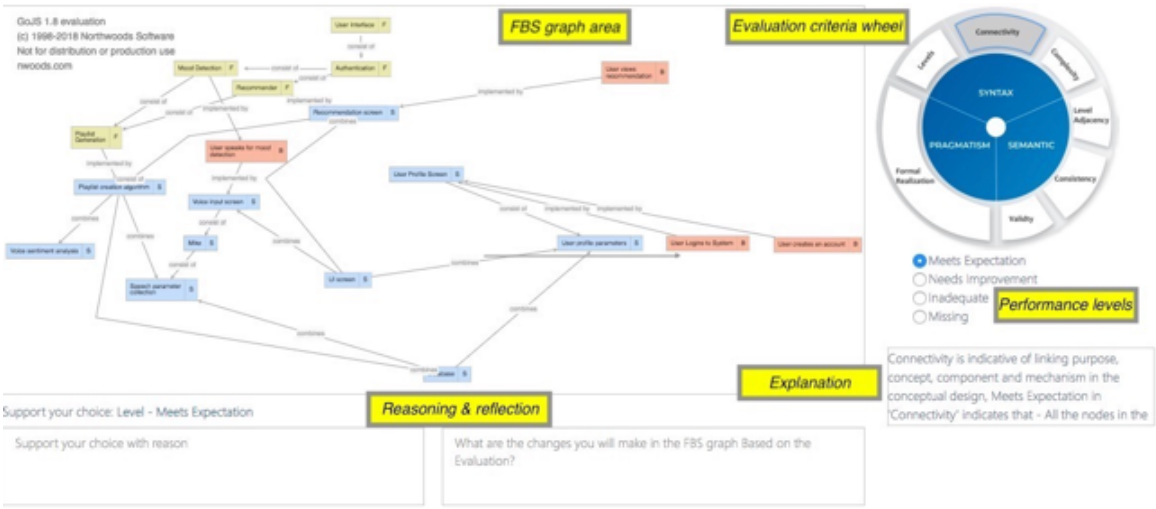


Figure 6.9 FBS graph evaluator options

- Resources for Information - At each and every step of tasks in ‘think & link’ there are videos, which contain task specific knowledge required to complete the task (see Figure. 6.10). Additionally there is also a ‘Information’ page in the learning environment that includes a collection of videos about the context of the learning environment like SCD, FBS framework etc.

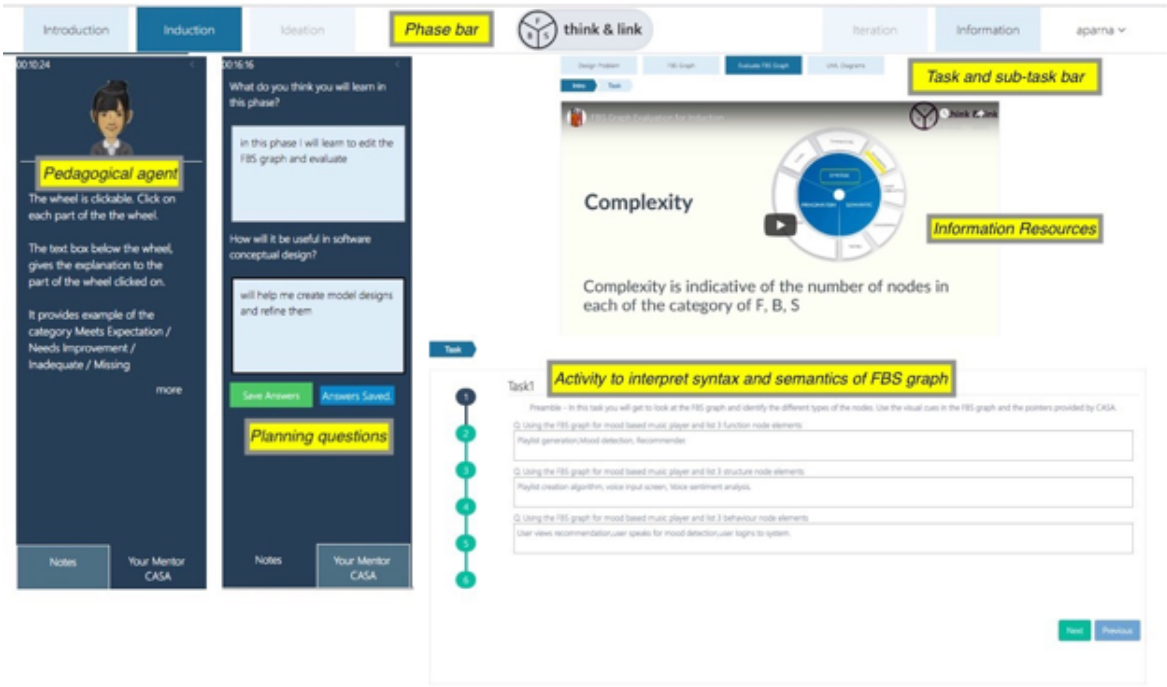


Figure 6.10 Features of 'think & link'

- Planning questions – It is important for the learner to reflect, evaluate and monitor their process during design. By doing this learners will be able to

imbibe the process of SCD along with the strategies. As mentioned earlier the learners are taken through the three planes of cognition – doing, evaluation and synthesis. In the planning activity the learners are required to reflect on the task ahead and plan (see Figure. 6.10). Example questions that they encounter are – *What will you do in this phase? How will this task be useful for creating software conceptual designs?*

- On the left side of 'think & link' (see Figure. 6.10) a vertical column is dedicated to the pedagogical agent CASA (conceptual design assistant). CASA is present all through the phases. CASA provides procedural prompts related to the task that the learner is currently performing. The prompts provided by CASA are adaptive and are dependent on the learner's actions and progress in the task. The learner's task progress is monitored and CASA provides appropriate scaffolds to complete the task at the desired level. CASA also provides cognitive prompts, which aid the learner in creating the FBS design elements and linking them. The Table 6.3 below provides a sample of prompts from each phase based on learner's actions.

Table 6.3 Sample of CASA prompts in each phase of 'think & link'

Phase	Learner's action	CASA prompt	Nature of prompt
Phase 1	First time the 'FBS graph' tab is enabled	i. Here you will get to look at the FBS graph to answer questions in the Worksheet. ii. View the 'FBS Graph - Video' to get to know what is a FBS graph. iii. Click on 'more' to get additional prompts on FBS-graph	Information prompt
	next button (Task 3 in activity) & Task button	Use the previously identified FBS (Function, Behaviour, Structure) design elements, definitions and your understanding of FBS to answer the question.	Procedural prompt to complete tasks

Phase 2	FBS graph editing task & > 3 Function nodes created	i. You might want to list the structures that will implement the functions ii. Imagine how possibly the end user will interact with the system with such capabilities	i. Cognition prompt to associate structures ii. Cognition prompt to simulate behaviours
Phase 3	FBS graph 'Task' tab If (> 5 mins time in the FBS graph time in the FBS graph & no action) then trigger	i. Recollect experiences with similar systems which you have encountered some mood basic music apps or music speaker ii. Try recollecting some algorithms or physical components that were presented in those systems iii. How are those systems/applications/algorithms/components different from the present problem? iv. Who is the end user of this application? and How will the end user interact?	i. Cognition prompts to create Function nodes ii. Cognition prompts for structure iii. Cognition prompts to adapt/reuse known structures iv. Cognition prompts for simulating behaviours

6.6 Summary

To alleviate the novice difficulty in SCD we designed and developed a function-behaviour-structure (FBS) design framework (Gero & Kannengeiser, 2014) based learning environment - 'think & link'. The goal of 'think & link' is to scaffold and help novices create integrated solution designs for software design problems. 'think & link' is implemented using Javascript, MySQL, and PHP. 'think & link' is available in this link https://thinknlink.tech/. To access 'think & link' student interface, create a login id or this student credential can be utilized: user id – Prathiksha, password – seokjin. To access the teacher interface in 'think & link' this credential can be used: user id – etiitb, password – thinknlink2019.

In ‘think & link’ the FBS design framework manifests as a manipulable graph known as the FBS graph. ‘think & link’ consists of activities grouped as three phases and provides scaffolds/prompts to complete them. Based on literature about experts design processes, theoretical frameworks and studies 1, 2 and 3 we propose the following conjectures about how ‘think & link’ supports novices creation of integrated SCD –

1. Conjecture 1 - Our first conjecture is related to the outcome of SCD, which satisfies the criteria for software design described in Section 4.1.3 in Table 4.3.

If an individual student builds syntactic & semantic interpretation of FBS design elements; creates, connects & evaluates FBS design elements using strategies and associate FBS graph to UML diagrams, they will be able to create integrated software conceptual design.

2. Conjecture 2 – Our second conjecture is related to the learning process of SCD,

If an individual student evaluates FBS graph, associates FBS graph to UML diagrams and writes planning, evaluation and reflection statements, they will be able to understand and abstract the process and strategies for software conceptual design.

Based on these conjectures we evaluated the learning environment ‘think & link’. The evaluation of ‘think & link’ is presented in the next chapter.

Chapter 7

DBR 2 Evaluation of 'think & link'

'think & link' is a self-learning web-based learning environment designed for teaching-learning of integrated SCD for novices. In the previous chapter, we described the design of 'think & link' emerging from problem analysis (chapter 4) and evaluation of preliminary learning environment designs (chapter 5). In chapter 5, the RQ 2 (2.a & 2.b) was about the difficulties that novices face in the FBS graph pedagogy. We conducted multiple iterations of the design of FBS graph based pedagogy and evaluated them with study 2 and 3. The findings informed the design and development of the learning environment 'think & link'. In this chapter we elaborate the study to evaluate 'think & link'. The evaluation serves the purpose of examining the conjectures that we have made in chapter 6.

There are two studies (Study 4 & 5) that we describe in this chapter. Studies are quite similar procedure wise but have different participants as they were conducted at different institutes. We conducted study 5, as we wanted to reconfirm our findings with participants from varied institutes.

7.1 Study Method

7.1.1 Research Questions

The broad research goal of the two studies is to identify the changes in novices' SCD understanding and process. The specific research questions are:

3.a. After interacting with 'think & link' what are the categories of SCD that learners' create?

3.b. After interacting with 'think & link' what are the changes in learners' understanding of SCD?

3.c. After interacting with 'think & link' what changes in the process of creating SCD do the learners' perceive?

3.d. How do the learners' use the features in 'think & link'?

The RQ 3.a. is towards testing the conjecture 1, that after learners interact with 'think & link' they would create integrated SCD. The RQs 3.b and 3.c is towards testing

conjecture 2 about the understanding and processes of SCD. The RQ 3.d. is to examine the interactions that learners perform while using 'think & link'.

7.1.2 Study Participants

7.1.2.1 Study 4

The study was conducted as a hands-on one-day workshop in an urban private engineering institute. The workshop was conducted in a computer laboratory of the institute, which had individual student desktops with access to the internet. The participants for the workshop were selected via purposive sampling. Due to the complex nature of SCD, prerequisites for participants of the workshop were determined. Only participants in their final year of computer engineering (CS) and information technology (IT) were considered for the workshop. Students in the fourth year of CS and IT undergraduate engineering programs complete courses such as 'Structured Object Oriented Analysis and Design' (semester 5) and 'Software Engineering' (semester 6) in their third year of engineering. These two courses cover topics of software design approaches, software-modeling tools, characteristics of software solution etc. As the course contents included such concepts, it was appropriate to consider that they had prerequisite knowledge for the SCD activity. The participation in the workshop was voluntary. An online workshop registration form to students via an instructor in the institute was floated. 30 final year students registered for the workshop. However on the day of the workshop only 20 students turned up (CS=15, IT=5: male=16, female=4). Students were provided with a consent form, which contained details about the study and data collection. Students were provided with the option of discontinuing the study at any time. The students that signed on the consent form were part of the study. Students were provided with a certificate after completion of the workshop. The objective was to obtain a typical representation of learners from the age group (19-22) with appropriate domain exposure. However the participants are representative of Indian urban engineering students.

7.1.2.2 Study 5

For study 5 we replicated study 4 with participants from another nearby institute. We conducted the study as a hands-on workshop in another urban private engineering near the institute. The engineering institute is located in the same city as our institute.

The sampling technique employed was convenience sampling. 22 students in the second year (male= 16, female=6) of their undergraduate degree in CS and IT attended the workshop. Students had just finished their semester exams, and voluntarily signed up for the workshop to learn about software design. Similar to study 4, students were provided with a consent form. Only students who signed on the consent form were part of the study. Students were provided with a certificate after they completed the workshop. Software modeling concepts and tools are prerequisites for 'think & link' and as second year undergraduates they have not yet encountered these courses/concepts. So we introduced these concepts as a separate module for these participants. These participants are also representative of Indian urban engineering students.

7.1.3 Study Design and Procedure

7.1.3.1 Design Problems

Both the studies, 4 and 5, were a single group pre-post test design. In study 4 and 5, we utilized the same set of design problems for pre-test and post-test. The design problems for the pre-post test are provided in the Table 7.1. The design problems were problems based on the familiarity of software systems usage among the students. For example the systems such ATM, music player are familiar to participants as they encounter such systems in their day-to-day lives. These two problems were selected, as the participants would be familiar in terms of usage, at least partially, to the software systems. In these problems the functional specifications are open-ended, and the familiar part of the problem (ATM, music player) gives indication for the functional decomposition. By open-ended we mean that no requirements were provided to the students. Participants had to assume the requirements and constraints from the problem and solve the problem. The indications for functional decomposition make the design problem tractable for novices.

Table 7. 1 Design problems for pre and post-test

Test	Design Problem
Pre-test	Design a mood based automatic music player
Post-test	Design a fingerprint ATM system

As we have different design problems that the students worked on, it is important to establish similarity among the problems so that we can evaluate the SCD and compare the design strategy among the participants. We provided the problems to an expert, with several years of software design expertise, who reviewed the problems. The expert vetted them as problems that are equally matched in terms of complexity, time taken to solve, and amount of code that needs to be written. When compared with the design problems category in literature, these software design problems are in between the innovative and creative design problem category (Brown & Chandrasekharan, 2014).

7.1.3.2 Study 4

Before the study a registration form was floated. In the registration form participants answered an open-ended questionnaire aimed to capture their prior conception of SCD. Participants solved a pre-test at the start. They individually created a SCD on pen and paper for the design problem - 'Design a mood based automatic music player'. They were free to use the internet for this task. After completing this task, participants utilized the individual desktop to access 'think & link'. 'think & link' has three phases which the participants completed in ~4.5 hours. After completing all activities in 'think & link', participants for an hour solved another SCD for – 'Design a fingerprint ATM system'. The pre and post SCD problems can be considered equivalent, as they are similar in terms of complexity and time taken to solve. After completing the post-test, participants were asked to respond to questions about understanding of term software conceptual design, usefulness and usability of 'think & link'. Multiple semi-structured focus group interviews with all participants were conducted for about 30 mins each. The focus group interviews were conducted in small groups of 2-4 participants. The participants spent around seven hours in the workshop.

7.1.3.3 Study 5

The study procedure of 4 and 5 is captured in Figure 7.1. Study 5 participants did not have exposure to UML modeling, so there was a preliminary background building session for them, one day before the study. A day before the study the workshop participants went through a module on software modeling using the Unified Modeling Language. The instructor for the session was a professor from the institute who

teaches the course software engineering. The instructor taught them about four important representations - use case diagram, class diagram, activity diagram, and sequence diagram. They used the online tool draw.io to create the representations. The instructor also provided them with some practice problems to model. After the session we asked the participants to respond to a survey designed to capture their conceptions of software conceptual design. This was followed by a pre-test.

In the pre-test, participants were provided the problem -'Create a software conceptual design for mood based music player'. After the pre-test participants were provided with access to 'think & link'. After completing the activities in 'think & link' the participants were provided a post-test; 'Create a software conceptual design for fingerprint based ATM'. After the post-test, we collated participants' perceptions about software conceptual design and 'think & link'.

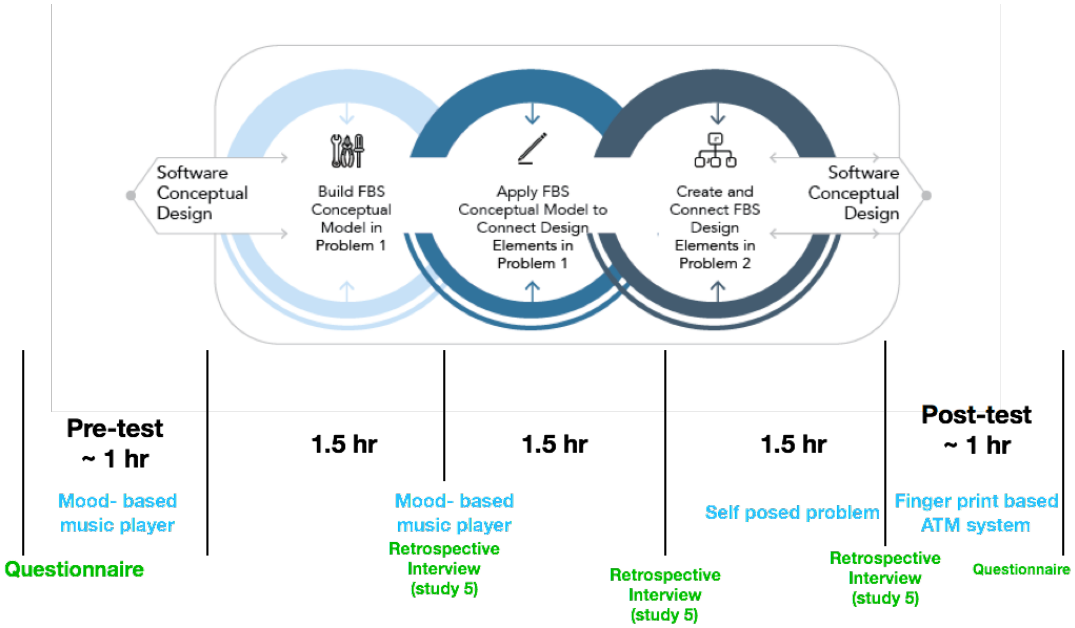


Figure 7.1 Study 4 & 5 procedure

7.1.4 Data Sources

7.1.4.1 Study 4

The online registration form for the workshop contained prior conception open-ended questions to capture participants' understanding of the term software conceptual design. The question that participants were asked was, "What is your understanding of 'software conceptual design'?" The sketches and notes for the pre and post paper based activities were collated as artifacts of the design activity. While the participants interacted with 'think & link' their actions in the learning environment were logged.

At the end participants' conceptions about software conceptual design, usability and usefulness of 'think & link' were also captured as responses to online questionnaires. The workshop closed with multiple focus groups semi-structured interviews where the participants were asked to respond to questions about their design processes and features utilized. Table 7.2 below maps the measure, data source and the analysis technique.

7.1.4.2 Study 5

All the data sources in study 4 were also included in study 5. In study 5, after the UML modelling session participants' filled in the open-ended questionnaire to capture understanding of SCD. They also solved the pre-test design problem of 'mood based music player'. The sketches and notes of the pre-test were collated. Participants then interacted with the learning environment. While the participants interacted with 'think & link' their actions in the learning environment were logged. Random selection of 6 participants was done. After completion of every phase these participants were interviewed. The semi-structured interview contained questions about the task they completed, how they performed the task, what features in 'think & link' were utilized, what was their strategy to complete the task. They were also asked questions about understanding of SCD and if it has changed from prior understanding. This was done to understand the learners' process of creating SCD closer to the completion of activities in 'think & link'.

Once the participants completed activities in 'think & link' they were provided with the post-test problem. After completion of post-test the participants' conceptions about software conceptual design, usability and usefulness of 'think & link' was also captured as responses to online questionnaires. Apart from the 6 participants other participants participated in multiple focus groups semi-structured interviews where the participants were asked to respond to questions about their design processes and features utilized. Table 7.2 below captures the data sources, mapping to research questions and the study.

Table 7.2 Mapping data source, RQ and study 4 and 5

Data Source	RQ 3.a (categories of SCD that learners' create)	RQ 3.b (changes in learners' understanding of SCD)	RQ 3.c (changes in the process of creating SCD do the learners' perceive)	RQ 3.d (learners' use the features in 'think & link')	Study
Pre-post artifact	✓	-	-	-	4 & 5
Pre-post answers to question about understanding of SCD (questionnaire)	-	✓	-	-	4 & 5
Focus group interview transcripts about design processes and features utilized	-	-	✓	-	4 & 5
Retrospective interviews about feature usage and change in understanding of SCD	-	-	✓	-	5
'think & link' system logs	-	-	-	✓	4 & 5

7.2 Data Analysis

Data collected in study 4 and 5 is elaborated in the previous section. In study 4 and 5, we followed a similar analysis process for each of the research questions. So in this section, we describe the data analysis for each research question.

7.2.1 RQ 3.a After interacting with 'think & link' what are the categories of SCD that learners' create?

To answer RQ 3.a we used the participants' pre-post design artifacts (Table 7.2). We utilized the categories of software conceptual design by Eckerdal et al. (Eckerdal et al., 2006) to analyse the pre-post design artifacts. The categories are presented in the Table 7.3 below. These categories are results of a phenomenographic analysis of the phenomenon 'produce a design'. Eckerdal et al., (Eckerdal et al., 2006) set out to gather the understanding of this phenomenon by giving the final year project students a design problem and collecting the design artifacts. The category consists of content and representation indicators in the artefact. The category 4 represents the use of multiple integrated representations and well-developed solutions. The category 4 is the target category after the intervention. These categories of student designs were utilized to classify the pre and post SCD of participants.

Table 7.3 Criteria to evaluation the design artifacts of SCD

Category #	Category	Content (indicators)	Representation (indicators)
0	Restatement	• Restate requirements from task description • No design content other than stated in the description	List or Bulleted items Informal design
1	Skumtomte	• Add a small amount to restating task • Unimportant	Informal design

		implementation details • No overall system view and any work on modules	
2	First step	Some significant work beyond restatement	• Formal notation representing components • Design of one of the system's components like GUI or Database
3	Partial design	• Understandable description of parts and overview • Description of parts may be incomplete or superficial • Communication between parts may not be completely described	• Formal notation representing working of the solution • Illustration of relationship between the parts
4	Complete Design	• Well developed solution • Understandable overview • Solution parts description includes explicit communication between them • Formal representations as well as text	Multiple formal notations such as Use case, Class diagram, component diagram and all the representations are linked

Once the participants' artifacts are classified, there emerged a need to explicate the participants' transition across the categories in pre and post-test. We used the tool interactive stratified attribute tracking, iSAT (Majumdar & Iyer, 2016) on the pre-post artefact categories result. iSAT is a web-based tool (Majumdar & Iyer, 2016) that helps users visualize and interact with the data. It has been used for analysing data from different research contexts (Majumdar & Iyer, 2016) and to take instructional decisions. Here we use the tool to analyse pre-post artifacts category transitions. iSAT helps to visualize the proportion of the participants who created artifacts in a specific category. iSAT further assists to explicate the transitions between the categories in both the pre and post-tests. It traces what proportion of any category in pre-test has transitioned to the post-test categories. An example of the visualisation, iSAT, from study 4 is presented in the Figure 7.2 below. The columns represent the test, pre and post. In each column the cells represent the category and the corresponding proportion of participants in the particular category. The connecting bands between the two columns correspond to the transition across the categories from pre to post test.

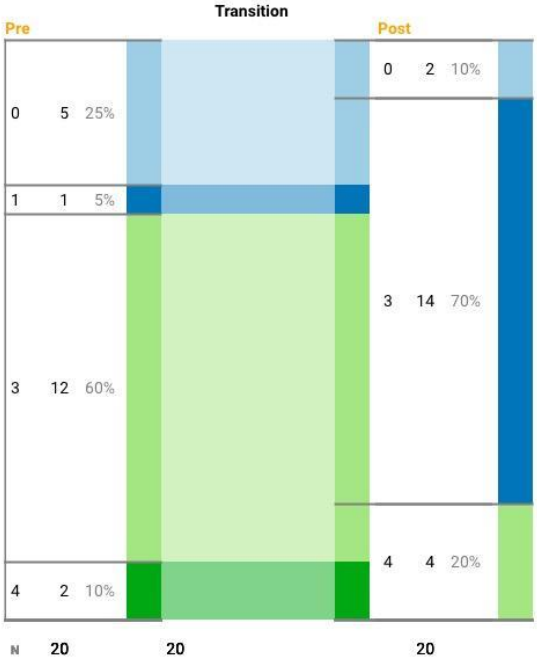


Figure 7.2 Example of an iSAT diagram

7.2.2 RQ 3.b After interacting with 'think & link' what are the changes in learners' understanding of SCD?

To analyse the open-ended responses to the questionnaire, we used the guidelines provided by Clarke and Braun (Braun & Clarke, 2017). The open-ended responses to both pre-post questions were already in textual format since they were collected online. Each participant's pre-post responses were collated. We went through the responses and initially coded them. The unit of analysis was a sentence. The codes represented the relevant qualities of SCD. After initial coding, codes were discussed and refined by us together. This iterative process continued until both of us agreed to the codes. In the next step, the codes were then clustered together to form themes. The responses falling in a specific theme and code were compared to arrive at the answers to RQ. Table 7.4 below presents the responses, their corresponding codes and the clustered themes.

Table 7.4 Snapshot of participant responses and their corresponding codes/themes

Participant responses to –'What is your understanding of software conceptual design?'	Corresponding code capturing the quality of SCD	Clustered themes
Design model of the solution	model	outcome
Drawing conceptual and schematic diagrams	drawings	
Early phase in which we identify constraints, requirements, goals and users	extracting problem characteristics	activities
It is a systematic approach to create solution details instead of just throwing things on paper	systematic approach	
Use online and offline drawing applications	tools	tools
Detailed plan of ideas which can be explained to others	plan	outcome

Conceptual design is going deep to find solution and form connections to various solution parts and modules	integration of solution parts	activities

7.2.3 RQ 3.c After interacting with 'think & link' what changes in the process of creating SCD do the learners' perceive?

In the semi-structured focus group interview conducted after the post-test in both the studies (study 4 & 5) we asked the participants the questions - "What according to you is a process of creating software conceptual design? How did you understand this? Has the process changed from previous?" The transcripts of the responses were collated and we performed inductive thematic analysis (Braun & Clarke, 2017). We initially coded for the activities that they mention. The unit of analysis was a sentence. After initial coding, codes were discussed and refined by both of us together. This iterative process continued until both of us agreed to the codes.

7.2.4 RQ 3.d How do the learners' use the features in 'think & link'?

To answer this RQ we used 'think & link' system logs.

System logs - 'think & link' stores the events happening in the system in the database. There are two kinds of events that are logged. The first sets of events are the user-generated events. The clicks on menu/feature buttons are user generated. Internal system events such as worksheet saved, phase completion are also logged. The event is logged as a row in the database. There are many columns in the row. However, we will focus on the relevant columns for a logging row, which are log_id, user_id, phase, subphase, subsubphase, event, event_time

- log_id : this is a unique number generated by the system for every log in the system. This uniquely identifies each row.
- user_id : Every user in the system gets a unique number associated. The logs are associated with the user based on this number. A log entry is associated with a user based on this number.
- phase : 'think & link' has three phases. The log entry is associated with which phase the log entry was generated in. The phase names are logged under this column. For example, 'Introduction' is a phase

- subphase : Each phase has 4 subphases in 'think & link'. The subphases indicate the task that the user is performing in the phase. For example, 'FBS graph' is a subphase.
- subsubphase : In a subphase there are two tabs. One has information in the form of the video and the other the task itself. For example, in 'FBS graph', 'intro' is a subsubphase.
- event : Each event description is captured in this column. For example if the participant had clicked on the video then the description 'FBS graph introduction video is playing' is captured here.
- event_time : This column logs the date and time. The time is tracked down to the second.

All of these values are present for the whole session and the participants in the study 4 and 5. We utilized the trajectory miner package (TraMineR) in R. Utilizing the TraMineR package we create time stamped event sequences (Ritschard et al., 2014). Using this sequence we create ordering of sequences in each phase. We then extracted the order of the successive elements in sequences that are shared by at least 5% of the logs in each phase. Additionally we also looked for the most frequent sub sequences in a window. We compared the frequent sub sequences across the post-test categories of participants. The script for both these analyses is available in the Appendix.

7.3 Results

To answer RQ3a, we evaluated the pre-post solutions of the participants (study 4 & 5) using the criteria by Eckerdal et al. (Eckerdal et al., 2004). To answer RQ3b, we turn to the participants' response pre-post to the open-ended question - 'What is your understanding of software conceptual design?' The responses falling in a specific theme and code were compared to arrive at the answers to RQ3b. Answers from focus group interviews (study 4 & 5) and from the reflective interviews were used to answer RQ3c. The system logs are the data sources to understand how the participants use the features (RQ 3d) in 'think & link'.

7.3.1 RQ 3.a After interacting with 'think & link' what are the categories of SCD that learners' create?

Study 4

In pre-intervention design solutions twelve out of the twenty participants' artifacts are in the category 3 (see Figure 7.3), which corresponds to the usage of diagrams representing the flow of the solution. Most of the participants created flowcharts describing the flow of the functionality mood detection. Participants also have written plain text exploring the problem. The text is mostly a restatement of the problem in many different ways. This corresponds to the five (Figure 7.3) participants in the category 0. Two participants created multiple artifacts like activity diagram, sequence diagram and class diagram that falls in category 4. Some of the participants have spent time in analysing only sub-problem mood detection. However they have not utilized any formal representation, which contributes to category 0. We do not have any participants in the category 2, which corresponds to static formal representation of components. Post-Intervention SCD categories - fourteen of the participants' artifacts are in the category 3. Participants have created flowcharts, which fall in category 3 as in the pre intervention case. There are no artifacts in the category 1 & 2, which indicates that participants are unable to utilize the formal representations for function and structure separately. There is a drop in category 0 and a moderate jump in category 4 (Figure 7.3).

RQ 3.a After interacting with the TELE what are the categories of scd that learners' create?

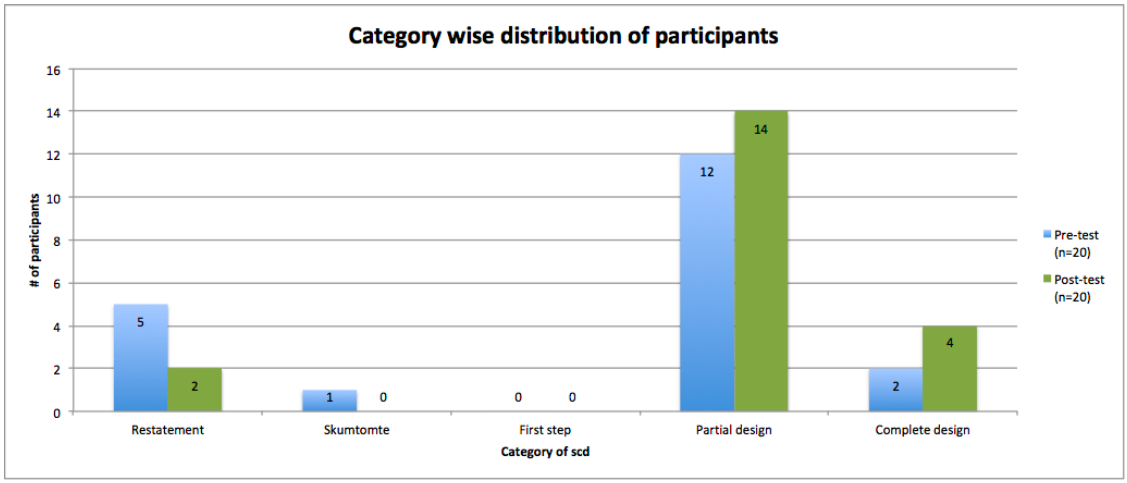


Figure 7.3 Study 4 - Comparison of pre-post artifact categories generated

To analyse the transitions across the pre-post categories, we used the tool iSAT, mentioned earlier (section 7.2.1). The observations from analysing the pre –post transitions (Figure 7.4) are:

- none of the participant slid down to lower category in post test

- majority of the participants in both pre and post fell in the category 3 as they utilized the representation of flowchart
- participants' in the post intervention moved to the category 3 from category 0 (Figure 7.4). The majority of the dynamic representations utilized by participants were flow charts, so that could explain the shift. A participant moved to category 3 from category 1. This movement could indicate that participants are able to create dynamic representations than the static formal representations.
- participants' in the post intervention moved to the category 4 from category 3 (Figure 7.4). During the intervention the understanding that software conceptual design comprises multiple artifacts depicting the functional, behavioural and structural view could be the reason for the shift.

There was no drastic shift in the participants' artifacts, i.e. all the participants in post intervention did not create SCD in the category 4.

RQ 3.a After interacting with the TELE what are the categories of scd that learners' create?

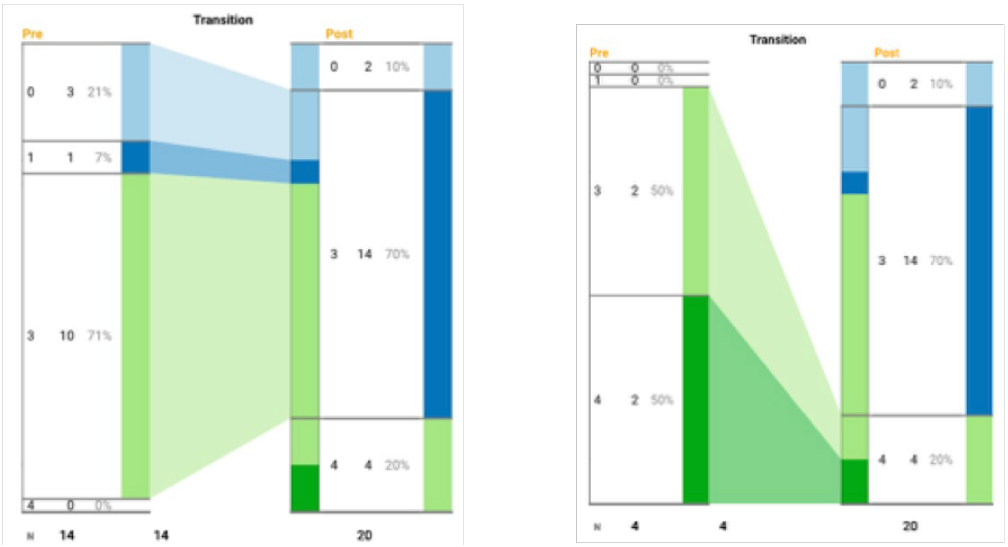


Figure 7.4 Study 4 - Pre-post category transitions

Study 5

In pre-intervention design solutions seven of the 18 participants' artifacts are in the category 3 (partial design, see Figure 7.5), which corresponds to the usage of diagrams representing the flow of the solution. In this category participants created multiple representations using use case, class diagram and activity diagram. However the links between these representations are missing, due to which they fall in the

category of 3. In this study 7 out of the 18 students created only class diagrams and used text to express the working. This resulted in them falling into category 2 (first step, see Figure 7.5). 3 participants added unimportant details to the solution e.g. 'music is an important feature of our life as it soothes us'. Such details do not provide information about the problem or the solution. So they fell in the category of 1 (Skumtomte, see Figure 7.4). A participant in category 0 (restatement, Figure 7.4) merely restated the problem as text.

Post intervention we see that a participant created complete designs (category 4). The participant's solution in this category had different representations like sequence diagram, class and use case. These representations were linked utilizing the F/B/S design elements and establishing the relationship between them. 14 participants post intervention created solutions (category 3) by utilizing multiple representations using use case, class diagram and activity diagram. However the links between these representations are missing. A participant in category 2 (first step) only came up with a use case representation. 2 participants in category 1 (Skumtomte) only added insufficient details to the problem and did not utilize any formal representation for the solution.

RQ 3.a After interacting with the TELE what are the categories of scd that learners' create?

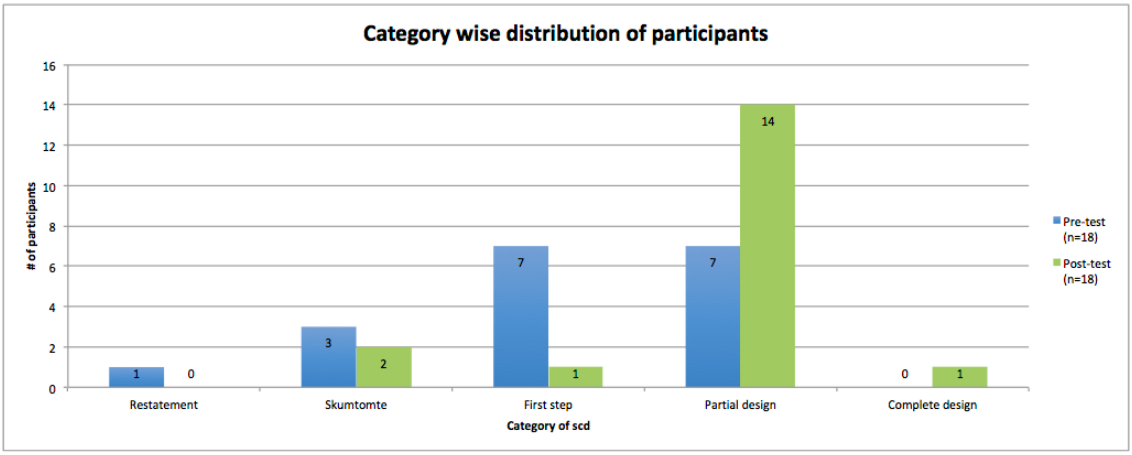


Figure 7.5 Study 5 - Comparison of pre-post artifact categories generated

To analyse the transitions across the pre-post categories, we used the tool iSAT, similar to study 4. The observations from analysing the pre –post transitions (Figure 7.6) are:

- majority of the participants in both pre and post fell in the category 3 as they utilized multiple representations to depict the solution design

- 3 participants from category 1 in pre-test moved to category 3 in post-test. The participants in pre-test added unimportant details to the solution but after the intervention moved to creating solution designs using multiple representations.
- 7 participants from category 2 in pre-test moved to category 3 in post-test. They started out by creating only static representations like class or component representations. But after the intervention participants shifted to utilizing multiple representations
- 1 participant in category 3 in pre-test moved to category 2 post-test. In the pre-test this participant created a use case diagram, class diagram and activity diagram for the problem mood-based music player. The diagrams were however not integrated so the solution fell into the category 3 (partial design). In the post-test the participant however only created a use case diagram for the fingerprint ATM problem. So the post-test solution fell into category 2. In both the studies this was one participant whose post-test solution slid in a lower category than the pre-test.

Similar to study 4, in study 5 also there was no drastic shift in the participants' artifacts, i.e. all the participants in post intervention did create SCD in category 4.

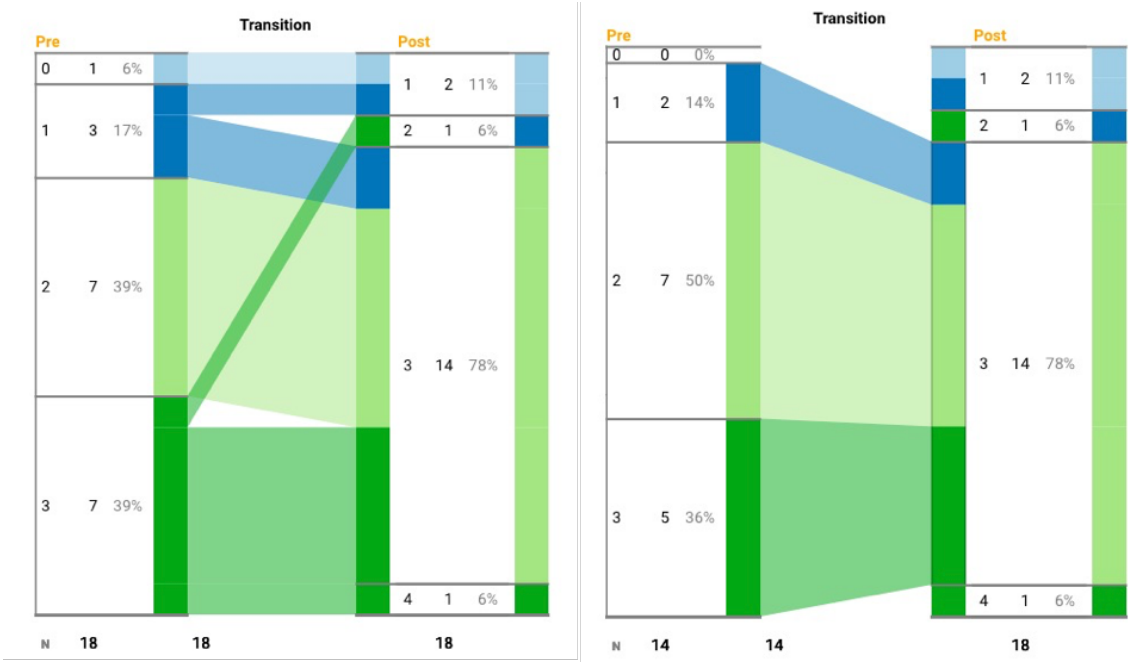


Figure 7.6 Study 5 - Pre-post artifact category transitions

Summarizing Study 4 & 5 results

Answering RQ3a - Participants create formal representations that depict the dynamic working aspects of the solution

From both the studies we observe that participants moved away from informal designs. The decrease in category 0 (restatement) and category 1 (Skumtomte) indicates that. Participants who created informal design and simple representations after intervention moved to behaviour based representations. Participants who created only behaviour-based representations went on to create multiple integrated formal representations. The target category is 4, which indicates usage of multiple integrated representations. Not all of the students in both studies reached this category. The majority of the participants reached the category 3, which indicates usage of behaviour based representations such as activity diagrams, sequence diagrams. Such representations represent the dynamic aspect of the design solution.

The performance of category 3 participants in study 5 is better than performance of participants in the same category in study 4. For example, in study 4 the participants in category 3 used dynamic representations like flowchart, however the participants in study 5 in the same category used static as well as dynamic representations like class diagram and sequence diagram. Study 4 participants are final (4^{th}) year engineering students. They would have learnt the courses on UML modelling in their 3^{rd} year. However, study 5 participants are 2^{nd} year engineering students and have had a session on UML modelling closer to the study. This could be one of the reasons for utilization of the formal representations in the pre and post-test. However similar to study 4, study 5 participants in the post-test did not achieve the target level of the

7.3.2 RQ 3.b After interacting with 'think & link' what are the changes in learners' understanding of SCD?

Study 4

The open-ended responses and the themes emerging from the coding of all pre-intervention responses resulted in broad themes of outcome, activities and tools. The themes emerging from the coding of all post-intervention responses resulted in themes of outcomes and activities. Responses under the post-intervention themes indicate a gradual refinement of understanding and shift of perspective for the participants.

Below we discuss the shift in understanding with example responses, their corresponding code and theme.

A. Refinement of understanding about SCD

The participants' developed the understanding about SCD being a *'combination of all UML diagrams'* (code: drawings, theme: outcome) rather than just thinking about *'conceptual & schematic drawings'* (code: drawings, theme: outcome). There is a refinement in the outcome of SCD as an integrated solution from participants' responses such as- (i) *doing a conceptual design going deep into what actually the problem is and form connections to various solution parts actually*, (ii) *what will be the back end, what will be the front end, how will front end access back end*. The other observed refinement in participants' understanding about SCD is that *'.... need to understand intricacies for implementing minor details'* (code: details of solutions, theme: activities) whereas in pre response we see the response as *'creating design modules'* (code: module creation, theme: activities). Earlier participants' understanding about activities in SCD was 'documentation' whereas now they refine their understanding as SCD involves design and creation ('*in software engineering we didn't design anything, it was just documentation'*).

B. Perspective shift

The participants' developed alternative views about SCD, which will be useful during solving design problems. The first shift in perspective is about SCD being a systematic approach – '*it is a systematic way instead of just throwing things on paper'* (systematic approach- activities). Pre-intervention participants' design for 'customer requirements in modules'. After intervention participants' acquire the perspective of designing for understanding of other '*designers as well as programmers or developers'*. The participants' also develop the perspective about the cognitive process involved in the activity – *'we need to first imagine how will end user use it, then create use cases, identify components'*. The participants' view SCD as a stage '*before coding'* where they are required to *'mention all steps so that it is as close to the real software',* whereas, in pre-intervention response they view it as a '*phase extracting problem chara*cteristics'.

Study 5

The open-ended responses and the themes emerging from the coding of all pre-intervention and post-intervention responses resulted in broad themes of outcome and activities. Responses under the post-intervention themes indicate a gradual shift

towards refinement of solution characteristics from problem characteristics by the participants.

A. Refinement of solution characteristics

Some of the participants before the intervention define SCD as *blueprint, road map, sketches, and representations* for communicating ideas. Before the intervention participants also held the perception of SCD consists of activities primarily pertaining to design problems. In the pre-intervention questionnaire, participants describe SCD as *problem analysis*, *goal identification*, *and problem solving*. One of the participants describes SCD as '*problem analysis by understanding people's needs'*. In the post-intervention questionnaire many participants move towards describing solution characteristics. They define the outcome of SCD to be *easy to understand*, *easy to implement*, *modifiable.* Some of the participants in the post-intervention questionnaire also define SCD in terms of FBS design elements. Participants mention that in SCD *'identify function, structures and behaviours'*. They utilize the FBS graph evaluation parameters as SCD output characteristics - *'it should be consistent, connectivity, to have all functionality, define the mechanism of (working) system'*. These parameters are based on the quality of conceptual models proposed by Lindland et al. (Lindland et al., 1994). In the post-intervention questionnaire participants have utilized their takeaways from the FBS conceptual model as well as FBS graph creation and evaluation tasks. Participants to describe the activities in SCD as well as SCD solution characteristics utilize these takeaways.

We also zoomed into the pre-post responses of the participant in study 5 whose post-test artifact slid into a lower category (section 7.3.1). There was no major change in the pre-post understanding in SCD.

Summarizing Study 4 & 5 results

Answering RQ3b: Participants exhibit shift in perspective, refinement in understanding of SCD and solution characteristics

The pre-post open-ended responses indicate that learners' have undergone a shift in their understanding of SCD. The open-ended responses collated before the commencement of the workshop capture the conception participants had before the exposure to 'think & link'. Although the coding themes in pre-post almost remain the same, the responses themselves were contrasting in nature. This indicates a conceptual integration (Vosniadu, 2019), in which the practices and understanding from the activities in 'think & link' are combined with participants' earlier

conceptions. We observed the shift in understanding in study 4 participants, who were final year engineering students. Similar shift in understanding was observed in study 5 participants as well. So if participants have been exposed to UML modelling and then learn integrated SCD building in 'think & link', they undergo a shift in understanding of SCD. This shift is synonymous with the disciplinary practices in software design.

7.3.3 RQ 3.c After interacting with 'think & link' what changes in the process of creating SCD do the learners' perceive?

Study 4

The data sources to answer this RQ were utterances based on the focus group interviews. Participants reflected on their change in process of creating SCD after the intervention.

- Participants start with the notion of designing the solution to meet customer requirements. However, post the intervention the participants refine their understanding as designing for communicating solutions to developers as well. This is evident in the shift 'designing for ease of implementation'.
- Participants start with the notion of SCD pertaining only to the solution. Post intervention participants improve their understanding about SCD pertaining to details about the problem as well as solution. The theme 'expanding problem and solution' captures this improvement.
- Participants initially describe the solution as a set of modules. However, post intervention they speak about connecting the modules. So participants move from disparate to interlinked solution parts, the theme in which such an improvement is captured
- Participants reflect on the whole aim of the SCD, which was previously used for documentation. However, as they work on different problems, they realize that in this phase, they design and create solutions that fulfil requirements.

We see (Figure 7.7) that such shifts and refinement are not trivial, but they bring about the subtle changes that participants perceive about the process of creating SCD.

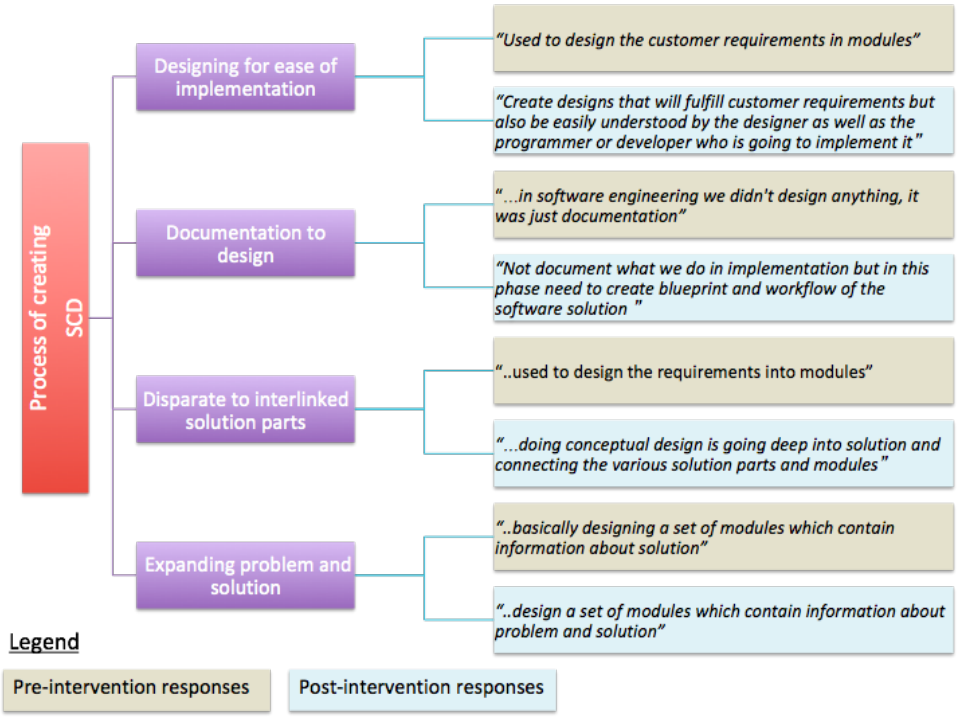


Figure 7.7 Study 4 - Participants' perception of processes in SCD

Study 5

Similar to study 4, the themes were extracted from the focus group interviews. Participants reflected (Figure 7.8) on their change in process of creating SCD after the intervention.

- Similar to study 4 results, participants reflected that SCD needs to be helpful for developers to implement the solution. Design needs to communicate the intended working of the solution as well as satisfy the problem requirements.
- Participants reflected that along with the explanation of the working of the solution the SCD should also be able to explain the requirements of the problem to the developers.
- In study 5 participants reflected on the process of balancing the features in the solution. A participant mentioned before the intervention, in the design solution, as designers they would offer features to satisfy the client. After the intervention, participants realize that having a lot of features would increase the complexity of the implementation.
- Additionally, participants reflected on the change in strategies. Post intervention participants describe FBS strategies for design, whereas pre-intervention the UML diagrams were utilized where they get stuck while creating the solution.

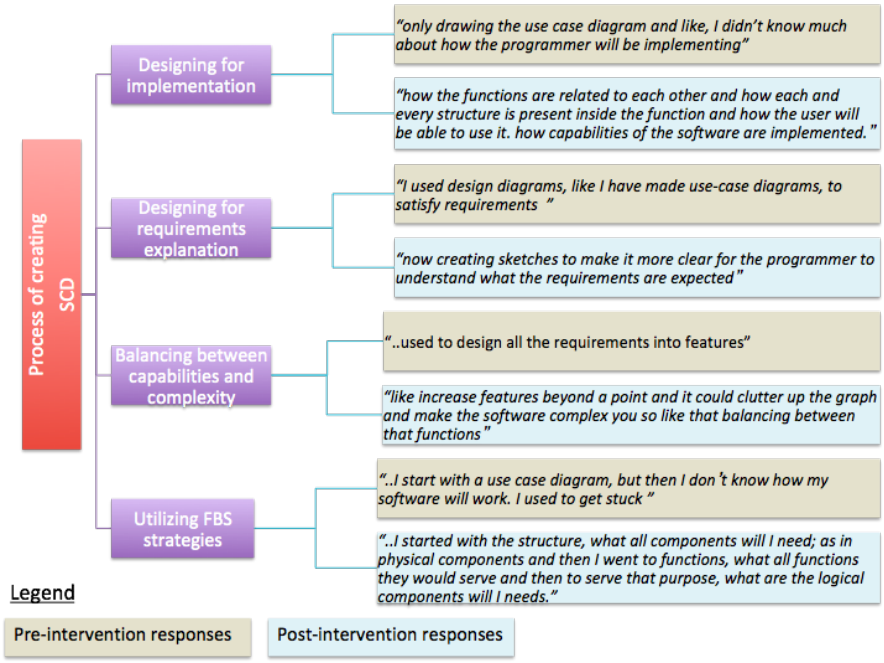


Figure 7.8 Study 5 - Participants' perception of processes in SCD

Summarizing Study 4 & 5 results

Answering RQ3c: Participants exhibit shift in design strategies, refinement in stakeholders, goals of process of SCD

The gradual change in the participants' understanding about the processes in SCD is indicative of conceptual change (Nussbaum, 1989). Study 4 participants responded broadly about the process of SCD to actually designing where both the problem and solution are expanded. Participants in study 4 also shifted their understanding that the SCD process should extract solution details at the same time maintain coherence in the solution.

Study 5 participants revealed shift in design strategies during SCD. The participants realized that during the design process they would need to balance between the features and implementation complexity. They reflected about how FBS design elements will be utilized by them during the SCD. All these changes indicate that participants perceive a shift in the process of SCD. We also zoomed into the lone participant whose post-test solution design slid into a lower category (section 7.3.1). In the pre-intervention strategy the participant mentions concepts such as 'abstraction ', 'data structures', 'modularity' as strategies to create SCD. Post-intervention the participant mentions the use of FBS graph and UML diagrams.

7.3.4 RQ 3.d How do the learners' use the features in 'think & link'?

To answer this RQ, we combined all the logs from study 4 as well as study 5. Instead of comparing the learner actions study wise, we wanted to examine the learner actions in each phase. We also intend to compare the learner's actions according to their performance in the post-test. The results to this question are presented phase wise, starting with phase 1.

7.3.4.1 Phase 1 – Learner actions sequences

In this phase we expect the learner to individually engage in the activity of building FBS conceptual models in problem context1 (mood based music player). The features available in 'think & link' in this phase are the FBS graph manipulator, activity sheet and the various video resources. We expect that the learners seek appropriate information as and when necessary, utilize the FBS graph manipulator and complete the activity sheet. As we see in the graph below, the X-axis represents the time-stamp in seconds. The Y-axis represents the events that have been logged in the database. The participants start with information, and then they read the problem context. The participants also go through the video about the task in this phase. After which the participants move to the graph section. This movement to the graph section is prompted by CASA. In the graph section, participants go through the introduction video. After this, the participants start solving the activity sheet.

In the Figure 7.9, we see a section highlighted in the graph area. This indicates that all participants have gone back and forth between the FBS graph as well as the task sheet. This indicates that to complete the task sheet, the participants utilized the FBS graph manipulations.

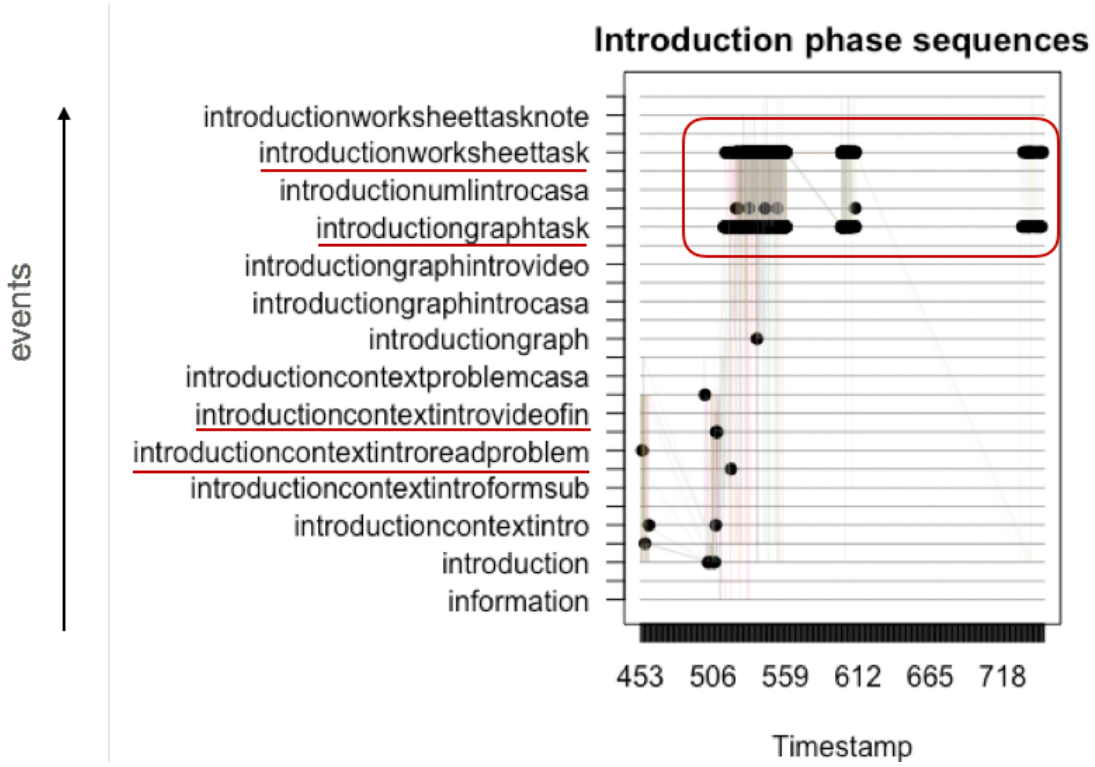


Figure 7.9 Participants' sequence of actions in Phase 1

Apart from the order of the successive events in the phase, we also looked at the most frequent event sequence pattern in this phase. The Table 7.5 represents the most frequent event sequence pattern. The sequences in the Table 7.5 have occurred in all the action sequences (support =1). The first row indicates the pattern that is enforced in 'think & link' phases. Participants need to plan for the phase, before they start activities. The participants' plan for the phase by looking at the task video as indicated by the event 'introductioncontextintrovideo'. The event 'introductioncontextintroformsub' indicates that participants have completed planning for the phase.

The next row indicates the most frequent sequence, which is the graph and the worksheet task. The row 4 indicates the sequence in the reverse order, which indicates that participants move back and forth between the graph task and worksheet task. Row 3 in the Table brings the sequence of how the participants get to the worksheet task- they view the task video, complete filling in the plan for the phase, and move to the graph task.

Table 7.5 Most frequent sub sequences in phase1

Most frequent sub sequences occurring in Phase 1
(introduction, introductioncontext)-(introductioncontextintrovideo)-(introductioncontextintroformsub)
(introductiongraphtask, introductionworksheettask)
(introduction)-(introductioncontextintroformsub)-(introductiongraphtask, introductionworksheettask)
(introductionworksheettask)-(introductiongraphtask)-(introductionworksheettask)

We see that the action sequences are common across all the post-test category participants. We also decided to examine differences between the FBS conceptual models created by participants. Looking at the different answers to the abstraction task in phase 2, we did this. In Table 7.6 we present the examples in each of the post-test category participants (Table 7.3). Even though the participant action sequences are similar, there is a difference in the conceptualization of the FBS model. The post-test category participants indicate a difference in their understanding of the FBS conceptual model. Participants in

- informal design category (category 1 & 2)- abstract only relationship between dyads (row 2 in Table 7.6)
- partial design (category 3) - either don't abstract or start with Functions (row 3 & 4 in the Table 7.6)
- complete design (category 4) - triads & start with Structures (row 5 in the Table 7.6)

We also looked at how the participant whose post-test category of SCD slid into a lower category performed at this task. The response of the participant for this task is – *'Function is implemented by behaviour and implemented by structure.'* The participant has abstracted the relationship of the triads starting with function, which explains the usage of only use case diagrams in the post-test (section 7.3.1).

In the next section we examine the participants actions in the phase 2 of 'think & link'.

Table 7.6 Comparison of participants' semantic interpretation of FBS conceptual model

Post-test category	Abstraction of FBS relationship
Informal design (category 1 & 2)	• *Function Implements Structure, structure is utilized to achieve the Behaviour, Structure demonstrates the Behaviour which is implemented using function* • *Function consists Function, Structure implemented by Behaviour, Function combines Structure, Function represented Structure*
Partial design (category 3)	• *mood detection implemented by user speaks for mood detection implemented by voice input screen consist of mike used by end user*
	• *Function is achieved by Structure utilized by Behaviour* • *Function is implemented by Structure which gets utilized during user Behaviours*
Complete design (category 4)	• *Structure consist of function & implemented by behaviours* • *Functions are implemented by structures, which utilize behaviour. Behaviour combines with structure to implement functions.*

7.3.4.2 Phase 2 – Learner actions sequences

In the phase 2 we expect the learner to individually engage in the activity of editing and evaluating the FBS graph in problem context1. The features in 'think & link' that are available to the participants are the FBS graph editor; FBS graph self-evaluator and the various information resources. The prompts and scaffolds from the pedagogical agent based on the learner actions are also available. We expect the

learners to utilize these resources and complete the tasks of editing and evaluating FBS graph.

As we see in the graph below, the x-axis represents the time-stamp in seconds. The y-axis represents the events that have been logged in the database. The participants start with looking at the task introduction. Completing the planning task as well as recap task. Then the participants move to the evaluation tab immediately. They watch the evaluation task video. As they evaluate they move towards the FBS graph-editing task. Then we see in the Figure below, participants move back and forth between the FBS graph evaluation and editing tasks. The participants end this phase by watching the UML creation video.

In the Figure 7.10, we see two sections that are highlighted. This indicates the frequent back and forth between certain tasks and/or features. The first one is between the context task and the evaluation video. Participants refer to the context task video, which indicates the tasks that need to be completed in this phase. The evaluation intro video provides details of the evaluation task like usage of the evaluation wheel; FBS graph evaluation criteria to name a few. The next section of the graph area, which is highlighted, is the frequent back and forth between the evaluation task and graph task. It is interesting to note that the participants start with the evaluation task and then move to the FBS graph task. The FBS graph tab is familiar to the participants, as they have used it in the previous phase. Additionally, it comes ahead of the evaluation tab; even then the participants start with the evaluation task.

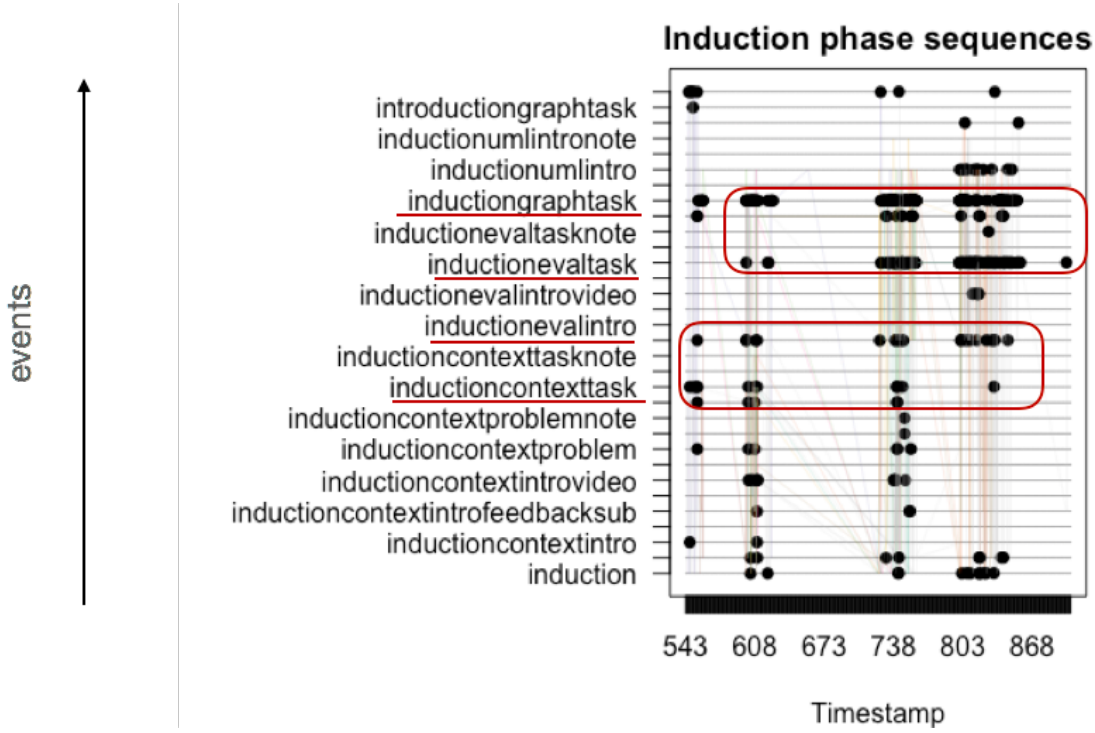


Figure 7.10 Participants' sequence of actions in Phase 2

Apart from the order of the successive events in the phase, we also looked at the most frequent event sequence pattern in this phase. The Table 7.7 represents the most frequent event sequence pattern. The first row, similar to Table 7.5 as in Phase 1, indicates the enforced task in 'think & link'. Participants need to plan for the phase, before they start activities. In row 2 we see that after completing the planning task, participants move towards graph edit task and evaluation task. Row 3 and 4 indicates that to complete the phase, participants use the linear path of graph task, evaluation task, UML video and phase completion. There isn't much back and forth between the graph task and evaluation task. However as seen in the Figure above participants while evaluating tend to refer to the graph task.

Table 7.7 Most frequent sub sequences in phase 2

Most frequent sub sequences occurring in Phase 2
(induction, induction context)-(inductioncontextintrofeedbacksub)
(induction)-(inductioncontextintrofeedbacksub)-(inductiongraphtask)-(inductionevaltask)
(inductiongraphtask)-(inductionevaltask)-(inductioneval)-(inductionphasefin)

(inductiongraphtask)-(inductionevaltask)-(inductionumlintro)

We also looked at the difference (see Table 7.8) between the sequences of actions among the post-test category participants. Among the participants in informal design category (category 1 & 2), we see that those participants did not edit the graph at all. In the post-test category 'partial design' (row 2) participants we see that they edit the graph and then evaluate. There is no back and forth between the graph task and evaluation task. While we look at the post-test participants in category 'complete design; we see that they moved back and forth between - graph task & evaluation task, phase 2 & phase 1.

Table 7.8 Comparison of sub sequences based on post-test performance

Post-test category	Event sub sequences
Informal design (category 1 & 2)	(induction)-(inductioncontexttask)-(inductionevaltask)-(inductionphasefin)
Partial design (category 3)	(inductiongraphintro)-(inductiongraphtask)-(inductionumlintro)-(inductionphasefin) (inductiongraphtask)-(inductioneval)-(inductionevaltask)-(inductionphasefin)
Complete design (category 4)	(inductiongraphtask)-(inductionevaltask)-(inductiongraphtask) (inductiongraphtask)-(introductiongraphtask)-(inductiongraphtask) (introductionworksheettask)-(inductioncontext)-(inductioneval)-(inductiongraphtask)

7.3.4.3 Phase 3 – Learner actions sequences

In the phase3 we expect the learner to engage in the activity of creating and connecting FBS design elements in a new problem context2. Additionally in this phase the learner is expected to be able to understand the underlying links between various UML diagrams from the FBS graph. The learner is provided with tools such as FBS graph editor, FBS evaluator, prompts and scaffolds from pedagogical agent. The learner in this phase is allowed to formulate a design problem of their own and create a FBS based SCD. We expect the learners to utilize these resources and complete the tasks of editing and evaluating FBS graph.

As we see in the graph below (see Figure 7.11), the X-axis represents the time-stamp in seconds. The Y-axis represents the events that have been logged in the database. The participants start with looking at the task introduction. This is evident from the viewing of the task introduction video. After this, participants complete the planning task, which is a compulsory task. After this the participants edit the problem. In this phase participants are expected to frame a design problem for themselves. We see that participants do this. After this the participants are seen looking at the introduction of graph evaluation, graph editing and UML. These introductions are presented as videos. Participants seem to be reviewing the FBS graph editing, evaluation and UML concepts. After this participants are seen to go back to the phase context to understand the goals of this phase. After having done that participants go back and forth between FBS graph creation and evaluation. To complete the phase, participants again move back to review goals of the phase and edit the FBS graph.

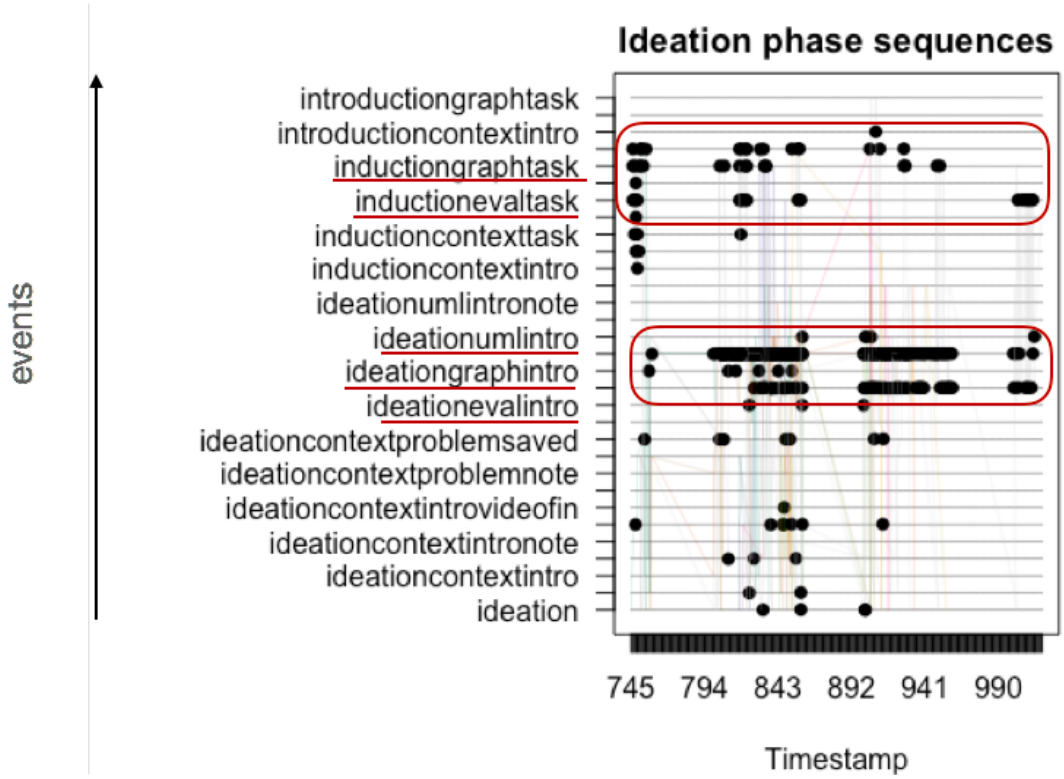


Figure 7.11 Participants' sequence of events in Phase 3

Apart from the order of the successive events in the phase, we also looked at the most frequent event sequence pattern in this phase. The Table 7.9 represents the most frequent event sequence pattern. The first row similar to all phases indicates the enforced planning task. The first row also indicates that after completing the planning task participants move on to the graph task and evaluation task. The second row indicates the linear sequence of graph, evaluation tasks and UML intro. Sequences in row 3 and 4 indicate that the participants read the problem, edit it if required, then proceed to the FBS graph task, moving to evaluation task later.

Table 7.9 Most frequent sub sequences in phase 3

Most frequent sub sequences occurring in Phase 3
(ideation)-(ideationcontextintrofeedbacksub)-(ideationgraphtask)-(ideationevaltask)
(ideationgraphtask)-(ideationevaltask)-(ideationumlintro)
(ideationcontextproblemread)-(ideationcontextproblemsaved)-(ideationgraphtask)-(ideationevaltask)

(ideationcontextproblemsaved)-(ideationgraphtask)-(ideationevaltask)

We also looked at the difference between the sequences of actions among the post-test category participants. The informal design category (category 1 & 2) participants have followed the linear path in this phase. The linear path is that of starting with the graph task, then evaluating the graph and completing the phase. In the partial design category (row2, category 3), we see that participants have referred to the previous phase's evaluation task to complete this phase's evaluation task. When we examined the edits to the FBS graph, we see that the participants only edited the FBS graph to add behaviour nodes. Complete design category participants (row 3) have referred to the previous phase's graph task (inductiongraphtask) to complete this phase's graph task. Last row of the Table shows that after formulating the problem, participants' refer to the previous phase's graph task.

Table 7.10 Comparison of subsequence in phase 3 based on post-test performance

Post-test category	Event sequences
Informal design (category 1 & 2)	(ideation)-(ideationgraphtask)-(ideationevaltask)-(ideationphasefin)
Partial design (category 3)	(inductionevaltask)-(ideationevaltask)
Complete design (category 4)	(inductiongraphtask)-(ideationgraphtask)-(ideationevaltask) (ideation, ideationcontext)-(ideationcontextproblemsaved)-(inductiongraphtask)

7.3.4.4 Answering RQ3d : Participants' usage of features in 'think & link'

- Phase 1 - All participants utilized the features of FBS graph, information resources and pedagogical agent prompts to complete the worksheet task. However there is a difference in the participants' understanding of the FBS framework, based on their post-test performance. This indicates that the FBS conceptual model abstraction differs according to participants' post-test

performance. Some of the participants are unable to interlink FBS elements together.

- Phase 2 - All participants utilized the FBS graph evaluator to complete the task. Some of the participants who utilized the FBS graph editing options did not edit the graph to include all FBS design elements. However, some participants did not utilize the FBS graph editing options at all. The participants who did not utilize the FBS graph edition option did not perform at the desired level in the post-test.
- Phase 3 - Majority of the participants utilized all features in this phase in a linear fashion. However as conjectured not all of them went back and forth between the feature tabs and phases. So even though the option of going back and forth is provided only certain participants utilized it. Such participants were found to perform at the target level in the post-test.
- The participant in study 5, whose post-test category slid in phase 2 did not edit the FBS graph and directly completed the evaluation task. In the phase 3 the participant regenerated the FBS graph for mood basic music player already available in phase 1 and 2. The participant did not utilize the tasks and features as intended.
- In each phase of 'think & link' we see differences in the participants' usage of features and sequence of tasks.

7.4 Discussion

From study 4 and 5 in the post-test we see,

- slight increase in participants creating representations describing behaviour and multiple integrated formal representations
- slight decrease in participants creating informal designs

Additionally while comparing the pre and post test performance we also observe that, participants who

- created informal design and simple representations after intervention moved to behaviour based representations
- created only behaviour based representations went on to create multiple integrated formal representations

After intervention, participants shift to create formal behaviour based representations that depict the dynamic aspects of SCD. The pre-post open-ended responses indicate that learners' have undergone a shift in their understanding of

SCD. The open-ended responses collated before the commencement of the workshop capture the conception participants had before the exposure to 'think & link'. Although the coding themes in pre-post almost remain the same, the responses themselves were contrasting in nature. This indicates a conceptual integration (Vosniadu, 2019), in which the practices and understanding from the activities in 'think & link' are combined with participants' earlier conceptions. Although the pre-post change in categories of design solution observed at the end of the workshop was not significant, we see the shift in participants' outcome of SCD. There was no 'gestalt shift' in the participants' artifacts, i.e. all the participants in post intervention did not create SCD in category 4 (Table 7.3). The gradual change in the participants' understanding about the outcomes and activities in SCD is indicative of refinement in understanding of SCD (Nussbaum, 1989) as follows

- The participants' have refined the understanding of SCD consisting of design activities rather than just documentation. In 'think & link' participants are to build, evaluate and refine FBS models as SCD for design problems. The deliberate practice activity of SCD creation for three problems in the workshop (mood based music player, final year project and finger-print ATM system) brought the change that SCD is design and not documentation.
- Participants have refined their understanding of SCD as creating solution details and linking the solution parts. The FBS graph creation includes the activity of creating individual F/B/S nodes and linking them together. These activities could have led to participants to refine their understanding.
- Participants have also developed refinement of understanding in the outcome of SCD as an integrated view combining all UML diagrams. In 'think & link' as participants edit and create the FBS graph, a visualization of the FBS graph as an integrated model is provided. This visualization informs the learner that an FBS graph is an integration of function (use case diagram), structure (class/component diagram) and behaviour (activity/sequence diagram). Additionally the learning environment provides procedural steps in the form of a video to create UML diagrams like use case, class diagram and sequence diagram from the FBS graph. This conceptual change could alleviate the 'lack of integration' difficulty in novices.
- The participants' view SCD as consisting of cognitive processes of mental simulation for creating use cases and then identifying structures that could

implement the functionalities. In 'think & link', CASA, the pedagogical agent provides cognitive prompts, based on expert processes (Ball et al., 2010), for creating as well linking FBS graph elements. The prompts are provided to the participants based on their actions. This conceptual change could alleviate the novice difficulties of fixation as it provides them with mental tools to generate functionalities as well as identify ways to implement them.

From RQ3d we found differences in the participants' feature usage as well as sequence of tasks. In the results section we grouped the participants according to their post-test artifact evaluation. The Table 7.11 captures the differences elaborated in the results section.

Additionally the participant whose post-test SCD slid to a lower category also did not perform the intended activities in phase 2 and phase 3. This could have resulted in the slide in the post-test performance. This result indicates the checks and scaffolds that need to be built in the learning environment to ensure that the participant performs the designed activities.

Table 7.11 Comparison of participants' actions and sequences in 'think & link'

Post- test categories / Phases in 'think & link'	Informal category (1 & 2)	Partial design category (3)	Complete design category (4)
Phase 1 – Abstraction of FBS relationship	abstract only relationship between dyads e.g. *Function Implements Structure*	either don't abstract or start with functions e.g. *Function is achieved by Structure utilized by Behaviour*	abstract the relationship and often start with structures e.g. *Structure consist of function & implemented by behaviours*
Phase 2 – Editing and evaluation of sample FBS graph	do not edit the graph and directly start evaluation	edit graph and then evaluate, however while examining their edits it is only addition of either a function or behaviour	move back & forth between evaluation & graph edit tasks. Their edits involves all design elements F/B/S. They also move across the phases 1 & 2

Phase 3 – Creation and evaluation of FBS graph	follow linear progression of activities, which are editing the problem statement, creating the FBS graph and evaluating it	For evaluation task refer to the evaluation done in the previous phase. Move across the phases.	move back & forth between problem setting, graph edit & evaluation tasks. They also move across the phases 2 & 3

7.5 Summary

This chapter summarizes the research studies and results of study 4 and 5. Study 4 and 5 were conducted to evaluate 'think & link' and test the conjectures that we made in chapter 6. Based on the investigation and study results from 4 and 5, our conclusions of conjecture 1 is as follows –

If individually students' build syntactic & semantic interpretation of FBS design elements; create, connect & evaluate FBS design elements using strategies and associate FBS graphs to UML diagrams they are able to utilize formal behaviour based representations that depict the dynamic aspects of SCD.

We found that when novices are exposed to FBS based intervention, abstract FBS relationships, utilize all the features in the teaching-learning system to create their own strategy, create dynamic and multiple representations as solutions for SCD problems. Participants, who did not utilize the FBS editing graph options, did not exhibit significant change in the pre-post SCD category.

Our conclusions based on investigation of conjecture 2 is as follows –

If individually students' evaluate FBS graph, associate FBS graph to UML diagrams and write planning, evaluation and reflection statements, they are able to understand and abstract the disciplinary processes and strategies of software conceptual design.

In study 4 and 5, we examined participants' understanding as well as reflections on process. Comparison of pre-post responses to understanding of SCD as well as reflection in the SCD processes indicates conceptual change (Nussbaum, 1989). The gradual change in the participants' understanding about the outcomes and activities in SCD is indicative of the conceptual change (Nussbaum, 1989). Learners seem to have a disciplinary practice-based shift in their understanding of SCD.

We noticed that participants utilized the features of 'think & link' differently. When the system logs are grouped based on post-test category performance, a distinct difference in the activity performance is noticed. These performance differences can be the input for the redesign of 'think & link'. We need to build checks and scaffolds in the activities so that participants perform the intended activity at the requisite level.

In the next chapter, we integrate the findings from all our studies to propose a local learning theory of conceptual change.

Chapter 8

Fostering conceptual change in software conceptual design

"One way to conceptualize undergraduate education is as a process of moving students along the path from novice toward expert understanding within a given discipline" (National Research Council, 2012). It is important to begin by identifying what students do, how their practices contrast with the disciplinary practices (i.e., experts' heuristic practices) and how to foster conceptual change in students. Conceptual change covers the description and analysis of learners' progress from prior conceptions to disciplinary practices (Von Aufschnaiter & Rogge, 2015).

8.1 Unpacking 'conceptual change'

Concepts provide a means through which humans make sense of the world (Nersessian, 2010). Concept or conception here refers to knowledge which the learner has to learn or an understanding that learner holds at a particular point in time (Von Aufschnaiter & Rogge, 2015). The term "*concepts in conceptual change research often refers to a broader scope than isolated and static concepts*" (Chi, 2008). The 'knowledge-as-elements' view proposes that naive knowledge consists of unstructured collection of independent elements, which have been developed by intuitive interaction with the world (diSessa, 1993). The 'revision, refinement and reorganization' of the elements bound in the context are termed as 'conceptual change' (diSessa, 2014).

Conceptual change is a meaningful lens for our work as we begin with learners' knowledge about SCD via UML modelling. Then we utilize the FBS graph based pedagogy to 'revise, refine and reorganize' the concepts so that learners' create integrated SCD. The practical steps that are recommended for 'conceptual change' in a given context, includes: 1) opportunities for learners to realize the alternate conceptions, 2) carefully examining and understanding the learner's prior conceptions to plan for teaching-learning, and 3) utilizing their prior conceptions in the teaching-learning. In addition, the contextual sensitivity of conceptions also needs to be taken into account. In this thesis the first research goal is to understand novice learners'

SCD processes. We have utilized this understanding to design 'think & link' that supports novices learning of SCD via the FBS design framework.

To capture 'conceptual change', the experiment form of research, in which different forms of instruction are compared in effect on performance of a test, is inadequate. Tools and methods to capture learners' conceptions include concept inventories (CIs); in-depth interviews, concept maps, and concept sketches; surveys; and observations of students. The nature of processes or "mechanisms" through which the concepts and conceptual change happens need to be captured. In this thesis throughout study 1-5 we have captured learners' conceptions via open-ended questionnaires, interviews and SCD solution artifacts. In study 1 specifically video recording of learners' SCD task was utilized to capture their design processes.

To foster disciplinary conceptual change, learners need to be involved in conceptual engagement (Dole & Sinatra, 1998). Learners need to be engaged in an effortful and mindful process. Learners' should be encouraged to construct their own knowledge and skills through active engagement rather than being passive listeners. In order to reorganize what they know, learners must become self-regulated and effortful, analysing and synthesizing new information (Jonassen & Easter, 2013). In this thesis we have designed a FBS graph based pedagogy. This is operationalized as a web page 'think & link'. In 'think & link' learners model SCD as a FBS graph. The FBS graph is an integrated view of the solution design. The learners are provided with opportunities to map this FBS graph to prior concepts of UML modelling.

8.2 'Conceptual change' in this thesis

Novices and experts exhibit a conceptual difference in understanding and doing SCD. Novices view SCD as solution generation specific to a functional, structural or behavioural view (Lakshmi & Iyer, 2018). The disciplinary practices evident from expert literature involve problem understanding and generating cohesive solutions fulfilling all requirements (Ball et al., 2010). Specific intervention was designed for teaching and learning of this disciplinary practice. Deliberate instruction induced conceptual change is attempted in this paper. Often a pre-test/post-test design using a research-based concept inventory is employed to measure conceptual change (Singer et al., 2012). However, in-depth information about the nature of conceptual change is often missing in such cases. Students conceptual change is investigated by interpreting written answers or statements, utterances, and/or drawings (Von

Aufschnaiter & Rogge, 2015). Descriptions, dialogue, sketches also capture such changes (Singer et al., 2012).

In this thesis we can follow the conceptual change starting from understanding novices' design strategies, cognitive processes and difficulties (chapter 4), then move towards designing intervention for the integrated SCD creation (chapter 5 & 6) and then observe the conceptual change in novices' SCD outcomes and understanding of SCD (chapter 7). The Figure 8.1 captures this journey. It starts from understanding novices' difficulties, then designing learning environments for novices to engage in disciplinary practices and then capturing the conceptual change that novices undergo after interacting with 'think & link'.

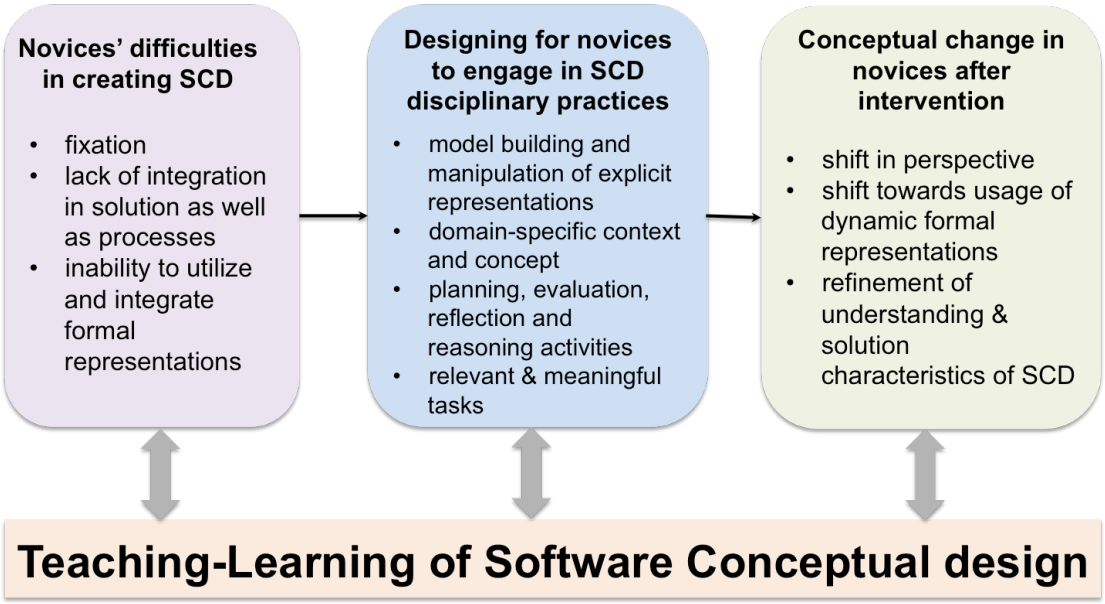


Figure 8.1 Conceptual change in this thesis

8.2.1 Novice design processes and difficulties in software conceptual design

Prior studies on novice difficulties in SCD (Eckerdal et al., 2006) (Chren et al., 2019; Thomas et al., 2014) indicate that novices (i) only rewrite problem statements during the design phase, (ii) are unable to utilize formal representations of UML to model SCD and (iii) are unable to utilize multiple UML diagrams for integrated view of solution.

We conducted an exploratory study (chapter 4) to understand novice design strategy and their problem solving process. We utilized the FBS design framework to code and represented their actions as linkograph. We also coded for cognitive

processes, wherein we used the conceptual design cognitive processes as the inductive framework (Hay et al, 2017). The results from the study (chapter 4) show that -

- novices' have the cognitive abilities like experts to mentally simulate scenarios, retrieve prior software systems' implementation, analogical reasoning, problem structuring, problem analysis
- novices' are however unable to integrate various solution parts to present a cohesive solution
- some novices' fixate and repeat certain functions, structures or behaviours. They also fixate on solving a part of the problem in many ways
- novices' utilize either problem or solution based strategies and are unable to integrate them

8.2.2 Designing for novices to engage in SCD disciplinary practices

Conceptual change arises from interaction between experience and current conceptions during problem solving. Conceptual change results most consistently from extended problem solving or some higher-order cognitive activity (Nersessian, 1999). So the pedagogy to foster conceptual change needs to involve problem(s) solving. In this thesis we have contextualised SCD in the various design problems as discussed chapter 1, section 1.4.1.

To foster conceptual Duit et al. (2008) have recommended design principles for a learning environment. In the below bulleted items we examine the design principles and their operationalization in 'think & link'.

1. The learning environment needs to integrate domain-specific concepts that learners need to know. To foster the disciplinary practice of integrated solutions we provided a learning opportunity for learners to integrate UML representation via the FBS framework. Initially, we provide them the introduction to the FBS conceptual model as a representation. This representation scaffolds the novices to integrate the different representations of use case, class and sequence diagram.
2. Learners need to be actively involved in a process of meaning and knowledge construction rather than passively receiving information. In 'think & link' learners are introduced to the FBS conceptual model via the FBS graph. This model helps them to think about solutions using these design elements. The FBS graph as a representation gives the opportunity to model their ideas as

F/B/S and establish the relationships between them. This construction of the FBS graph also triggers the appropriate activity in software design.

3. Learners need to be supported in their active learning and need to be guided towards the acquisition of self-regulated processes. In 'think & link' the planning, evaluation, reflection and reasoning activities are provided to learners. Before every phase in 'think & link' the learners are provided with an activity where they need to plan what they learn and how it would be meaningful to SCD. At the end of phase, learners can if they need, edit the plan and their takeaways from the phase. Every phase has the learners to complete a task, during their task, learners are provided with opportunity for evaluation and reflection. This helps them self-regulate their learning and at the same time abstract the process of learning.
4. Learners need to be provided with the information regarding the relevance and meaningfulness of the tasks. In 'think & link', learners are provided with explanatory videos about the relevance of SCD in software design. They are motivated to create FBS graph as it would help them create integrated solutions that satisfy the requirements. In pre-test learners are provided with a problem context where they create their own solutions. In phase 1 & 2, the same problem context as the pre-test is repeated so that the learners have the opportunity to implement their solution ideas in the FBS graph. In phase 3, learners are given the opportunity to define their own problem and create their solutions as an FBS graph. In study 3 & 4, final year engineering students were encouraged to frame their final year project as the problem context in phase 3. So the FBS graph created in phase 3 is useful for their project ideas.

8.2.3 Conceptual change after using 'think & link'

Outcome of SCD - After using 'think & link' learners' start utilizing the representations that are dynamic in nature. The solution that they create represents the working of the solution in several conditions. The dynamic representations start from simple ones like flowchart, to complex ones like activity diagram and sequence diagram. In study 4 (chapter 7), the participants were all final year computer engineering and information technology undergraduates. In the pre-test as well as post-test most of the study 4 (chapter 7) participants utilized flowchart. This could be attributed to their practice of using flowcharts to represent the working of the software

solution. In study 5 (chapter 7), which had participants in second year, we saw that in pre-test participants utilized static representations like class or component diagrams. In the post-test they shifted towards activity or sequence diagrams. Learners have started creating SCD using formal dynamic representations that capture the internal working of the solution.

Learner understanding of SCD – Conceptual change that learners undergo in the categories of SCD understanding are:

- Refinement of understanding of SCD - In study 4 and 5 (chapter 7) participants recognized that the outcome of SCD as integrated solution- "*combination of all UML diagrams", "connecting the solution parts", "how will front end access back end".* Participants also recognized that SCD is a phase of "*creating/doing design*" rather than just "*documentation*".
- Perspective shift - Participants prior to 'think & link' aim at fulfilling all the customer requirements. But post they acquire the perspective of designing for understanding of other "*designers as well as programmers or developers".* Participants get the perspective of SCD being a "*systematic approach".* Additionally, they develop the perspective about SCD as a stage "*before coding"* where they are required to *"mention all steps so that it is as close to the real software",* whereas, in pre-intervention response they view it as a "*phase extracting problem chara*cteristics".
- Solution characteristics - Participants now are able to characterize the solution as - understandable, implementable, and simple. The participants also apply the evaluation characteristics of the FBS graph to the solution outcomes like - *consistent, connectivity, to have all functionality, define the mechanism of (working) system'*. These parameters are based on the quality of conceptual models proposed by Lindland et al. (Lindland et al., 1994).

Perceptions about SCD process - Participants in study 4 (chapter 7) also shifted their understanding that the SCD process should extract solution details at the same time maintain coherence in the solution. Study 5 (chapter 7) participants revealed shifts in design strategies during SCD. The participants realized that during the design process they would need to balance between the features and implementation complexity. They reflected about how FBS design elements will be utilized by them during the SCD. All these changes indicate that participants perceive a shift in the process of SCD.

'think & link' fosters conceptual change in the SCD outcome, learner SCD understanding and perceptions about SCD processes. These changes are captured in the Figure 8.1

8.3 Summary

Conceptual change research begins with addressing learners' understanding of a given topic and the change in understanding as a result of instruction (von Aufschnaiter & Rogge, 2015). The design, development and implementation of FBS graph based intervention via 'think and link' fosters conceptual change. The gradual change in the participants' artifacts, understanding about the outcomes and activities in SCD is indicative of the conceptual change (Nussbaum, 1989). Learners seem to have a disciplinary practice-based shift in their understanding and perception of processes in SCD. However, the process of conceptual change appears to be gradual and complex.

In this chapter we have summarized our research goals via the conceptual change theory lens. It is important to utilize such theory lenses where the learners are at the centre. Conceptual change theories help computing education researchers to move beyond identifying individual errors and alternate conceptions. It helps advance the domain-specific theory by understanding the process of knowledge acquisition, application and evolution of alternate conceptions (Qian & Lehman, 2017).

Chapter 9

Discussion

9.1 Overview of Research Goals

In this thesis we started with the broad goals of -

- developing understanding of novice processes in software conceptual design
- using the understanding to design a technology enhanced learning environment to support novices learning of SCD.

These goals were further expanded to specific RQs as shown in the Figure 9.1. The Figure 9.1 is reproduced here again to recap the DBR cycles. As seen from Figure 9.1 we started by collecting initial requirements from study 1 which informed us about the novices design strategies and cognitive processes while creating SCD. With these results from study, and principles from theory, we designed initial FBS graph based intervention. Study 2 and 3 helped us identify difficulties of learners while using FBS graph based interventions. Design-based research cycle 1 starts from study 1 and ends at the results of study 2 and study 3.

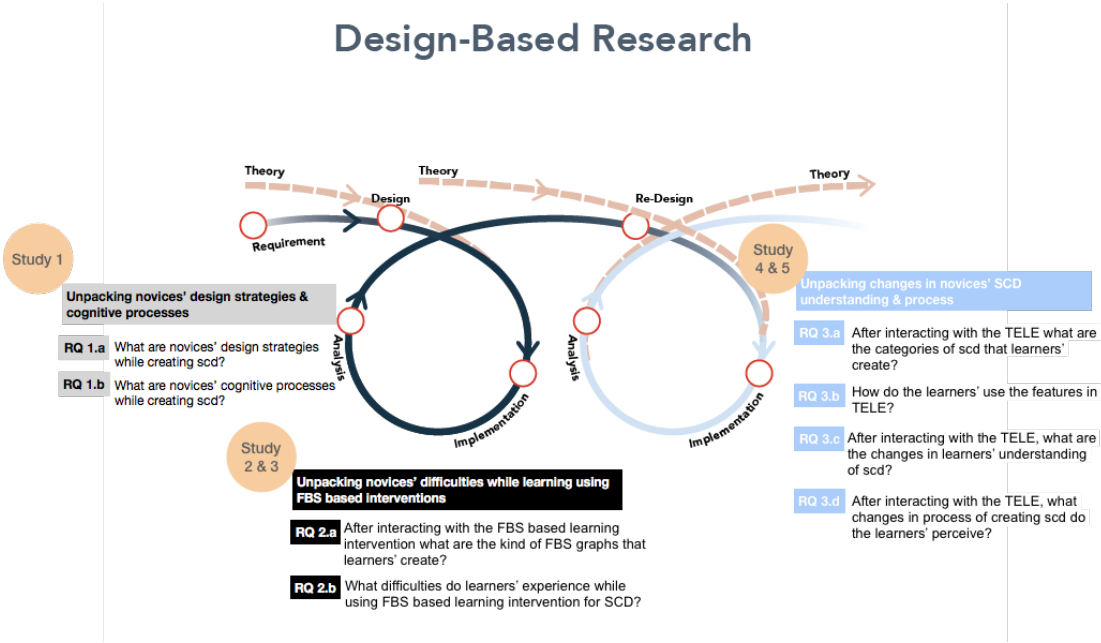


Figure 9.1 Design Based Research cycles in this thesis

The results from cycle 1 helped us design the web-based prototype of 'think & link'. We conducted heuristic evaluation for usability for the prototype. This led to the

newer version of ‘think & link’. ‘think & link’ was then taken to undergraduate computer and information technology students. Study 4 the participants were final year computer engineering and information technology students. Study 5 the participants in second year of the same branches. The results from study 4 and study 5 helped us identify the changes in novices’ understanding and process of SCD after they have used ‘think & link’.

The ‘gestalt shift’ of creating integrated solution designs, as SCD in participants after using ‘think link’ was not observed (study 4 & 5). However, we see that the students have undergone changes in their understanding as well as SCD processes as perceived by them. In the next section we summarize our findings and present the process of conceptual change in learners, with extensions to teaching and learning of SCD.

9.2 Summary of findings from DBR iterations

The two iterations of a design-based research project include characterization of novices design strategies and cognitive processes, designing for novices to build integrated SCD and conceptual change after students interacted with the learning environment. We present the summary of our findings in Table 9.1. Each of these findings is presented in sections below.

Table 9.1 Summary of findings

Goal	Findings	Study #
Unpacking novices design strategies and cognitive processes during creation of SCD	Unsuccessful novices • unable to utilize formal representations • fixated to F/B/S design elements • employ either problem or solution based design strategies • unable to trigger cognitive processes for creating SCD Successful novices • start from solution generation strategies • anchor existing/previous solutions and adapt them to the given problem context	Study 1 - Exploratory Qualitative Study

	• employed strategies such as evaluation, test case generation, and simulating failure scenarios • utilized cognitive processes such as retrieval, simulation, abstraction and association	
Unpacking novices' difficulties while learning using FBS based interventions (Suitability of FBS design framework argued in Chapter 5, section 5.2)	• Novices require scaffolds and prompts to build FBS conceptual model • Novices require scaffolds to create FBS design elements and the FBS graph	Study 2 & 3 - Lab study
Designing FBS framework based pedagogy for novices to create integrated SCD	• FBS graph presented as improvable model • Tasks with the FBS graph to build FBS conceptual model • Scaffolds for FBS graph editing and creation • Evaluation task of FBS graph with opportunities for reasoning and reflection	Study 2 & 3 - Lab study
Identifying changes in novices' SCD understanding & process after interacting with FBS based TELE	Participants • shift to create dynamic and formal representations of SCD • observed refinement in understanding of SCD • shift in perspective about SCD processes • refinement in characteristics of solution in SCD • develop strategies for process of SCD	Study 4 & 5 - Field study

Identifying features' in TELE that learners' utilize leading to changes in learners' understanding and processes of SCD	Participants • utilize the tools & feature in phase 1 however there is a difference in their abstraction of FBS conceptual model • utilize the features of 'think & link' differently. Participants', who did not edit the FBS graph suitably, did not exhibit significant change in the pre-post SCD category.	Study 4 & 5 - Field study

9.3 Mapping the findings to tasks, features and scaffolds in 'think & link'

The gradual change in the learners' understanding about the outcomes and activities in SCD is indicative of conceptual change (Nussbaum, 1989). In this section we explain the aspects of the FBS graph pedagogy and the features of 'think & link' impacting the conceptual change.

'Post the intervention participants create dynamic and formal representations of SCD'

In the learning environment participants are provided with examples of end-user behaviour and system's behaviour. The improvable model of the FBS graph provides the learner with such examples. The information videos about the FBS graph as well provides such resources. In the FBS graph, the learners need to create behaviour nodes and make connections with the structure and function nodes. This could have triggered the process that in conceptual design phase they need to think about how the structures will interact and be utilized. Participants are also provided with procedural videos to create formal representations such as use case, class and sequence diagram from the FBS graph. These aspects of 'think & link' trigger the learners towards creating representations that depict the working of the system.

'Post- intervention participants' have refined the understanding of SCD'

In 'think & link' participants are to build, evaluate and refine FBS models as SCD for design problems. The deliberate practice activity of SCD creation for three problems in the workshop (mood based music player, final year project and finger-print ATM system) could have brought the change that SCD is design and not documentation.

In ‘think & link’, the FBS graph creation includes the activity of creating individual F/B/S nodes and linking them together. These activities could have led to participants to refine their understanding of SCD as creating solution details and linking the solution parts.

In ‘think & link’ as participants edit and create the FBS graph, a visualization of the FBS graph as an integrated model is provided. This visualization informs the learner that an FBS graph is an integration of function (use case diagram), structure (class/component diagram) and behaviour (activity/sequence diagram). Additionally the learning environment provides procedural steps in the form of a video to create UML diagrams like use case, class diagram and sequence diagram from the FBS graph. Performing such tasks and utilizing the scaffolds in ‘think & link’ lead the participants to have developed refinement of understanding in the outcome of SCD as an integrated view combining all UML diagrams.

‘Post-intervention participants develop strategies for process of SCD’

As the learner create the FBS graph, the pedagogical agent provides the learner with scaffolds and prompts to create nodes and links in the FBS graph. Additionally learners are provided with planning, reflection and evaluation prompts to abstract their process of learning SCD. These features have led to participants developing and abstracting the strategies for processes of SCD.

In ‘think & link’, CASA, the pedagogical agent provides cognitive prompts, based on expert processes (Ball et al., 2010) , for creating as well linking FBS graph elements. The prompts are provided to the participants based on their actions. Utilization of such prompts to create the FBS graph has led the participants to abstract the processes of creating SCD and provides them with mental tools to generate functionalities as well as identify ways to implement them.

9.4 Addressing the research goals

The research goals of this thesis are to develop an understanding of novice processes in SCD and use it to design a technology enhanced learning environment to support novices learning of SCD. The Table 9.2 summarises the claims of the thesis.

In this thesis we have identified that

- novices have the difficulties of fixation and lack of integration while creating software conceptual design

- successful novices exhibit structure based design strategies and expert like cognitive processes

We have created a learning environment to address the learning of SCD. The results from our studies indicate that the following features and scaffolds in learning environment fosters the creation of SCD

- affordances for sketching, evaluation, procedural information to integrate UML representations
- adaptive prompts for to trigger cognitive processes and strategies

Base on study 4 and 5 we claim that

- novices assimilate SCD disciplinary practices in outcome, understanding as well as processes after explicit training in FBS based intervention

Table 9.2 Claims and evidence

Claims	Evidence
• Novices fixate when they utilize only either F, or B, or S based design strategies • Successful novices exhibit structure based design strategies and expert like cognitive processes	Study 1
Following features and scaffolds in learning environment fosters the process of creation of SCD • sketching feature to create and connect FBS design elements, evaluation feature to evaluate connected FBS elements, assimilation of UML & FBS design elements, planning & reflection opportunities to abstract SCD process • adaptive prompts for to trigger design strategies and cognitive processes of mental simulation, abstraction , association • tasks in the different planes of cognition – doing, evaluation and synthesis	Study 1, 2 & 3
Novices assimilate SCD disciplinary practices in understanding as well as processes after explicit training in FBS based	Study 4 & 5

intervention	

9.5 Generalizability

The goal of this thesis was to understand novice design processes and support them in the context of software conceptual design. So the findings from novice study, design of the pedagogy as well its evaluation are important aspects. So we examine thesis work for generalizability around these aspects.

9.5.1 Novice design strategies and cognitive processes

We identified that novices were successful in the task, when they employed structure-based strategies. The successful novices also employed cognitive processes such as mental simulation, abstraction and association. In case of unsuccessful novices, they employed mostly function-based strategies. Additionally the cognitive processes were limited to information seeking. We now examine whether these findings are likely to exist in other problem contexts and complexity.

The claims about the novice design strategies and cognitive processes are tied to the software design problem contexts. However, they can be generalized to the context of general software design. As stated in section 1.5 about the scope of the thesis, the design problems have been chosen based on familiarity of software systems usage among the students. In these problems the functional specifications are open-ended, and a part of the problem (ATM, payment systems, recommender system, music player) gives indication for the functional decomposition. If we carry out novice studies with similar other design problems, we are likely to find similar results as study 1. Hence the findings of novice difficulties can be extrapolated to other similar design problems.

However the findings of the study cannot be extended to design problems, which lead to a major invention or completely new products. The findings about novice processes from this thesis cannot be generalized to problems in the category of creative problems.

The findings of novice difficulties such as fixation and lack of integration likely to be found in other design disciplines. These findings support the already existing literature available in novice design. From this thesis, the characterization of

fixation among novices as thinking in terms of only functions or structures or behaviours is applicable across the other design disciplines.

9.5.2 FBS graph based pedagogy

The FBS graph can be utilized as external representation in integrating multiple views for tasks in software design such as comprehension, evaluation and decision-making. It can also be used for understanding complex systems in computer science such as networks, operating systems. FBS graph can be utilized as external representation of a program. The pedagogy designed for teaching and learning of software conceptual design can be adapted to use for other design tasks in programming such as code comprehension.

In the practice of engineering, fluency with creation of representations, usage of multiple representations and translation across representations are some of the essential skills (Johri & Lohani, 2011; Aurigemma et al., 2013). The FBS design framework is based on the deep understanding of what (structure), the how (behaviour) and the why (function). The design of the FBS graph based pedagogy is based on FBS design framework and external representation. The FBS graph allows the flexibility to start from any of the design elements and does not impose any arrangement. In the FBS graph pedagogy the FBS graph is the external representation that integrates the isolated UML representations of use case, class diagram and sequence diagram. Adapting the FBS graph based pedagogy to other task contexts is possible.

9.5.3 'think & link'

'think & link' has been instantiated as a self-paced, web-based learning environment. In this thesis, it has been used in the context of a workshop. However, it can be used through the semester with the software engineering course or corresponding laboratory course as an accompanying tool. It can also be used in the final year project lab, where the final year engineering graduates can create SCD of their final year project.

'think & link' has an instructor-authoring interface, so the problems and FBS graph can be changed. So for other design problems, the learning environment can still be utilized. The scaffolds and prompts are based on the FBS graph and not specific problem based.

In ‘ think & link’ the scaffolds and prompts are based on the FBS graph. The FBS graph is an integrated view of the other UML diagrams. So for comprehension of the UML diagrams the FBS graph could be utilized. In other design problem contexts such as programming, could utilize the FBS graph to create and comprehend programs that are devoid of programming language syntax (purpose-first programming).

In contexts other than design, if the problem context can be represented by the causal structure of the FBS framework, then ‘think & link’ could be used. For example, in case of troubleshooting networks, the first step is to understand the problem context by identifying the FBS elements. This can be represented as a causal structure of FBS graphs. The learners can then construct a FBS graph for another network troubleshooting scenario.

9.6 Limitations

In this section we elaborate on the limitations of this thesis.

9.6.1 Learner characteristics

The participants of the study were from urban engineering colleges, with medium of instruction as English. In think & link most of the instruction, information and task-related details are in English. Learners’, whose medium of instruction is not English, might have difficulty in working with a learning environment. In such cases there would be different learning outcomes. Apart from language we have not considered any of the other learner characteristics such as experience, motivation, and persistence.

Our sampling was a convenience sampling of the participants in all the studies. All participants volunteered for the study. We did not consider gender ratio and also our analysis did not differentiate participants' process, outcome and perceptions based on gender. So the effect of gender on SCD and the learning outcome of intervention need to be investigated in future.

9.6.2 Design Problem Characteristics

The FBS graph based pedagogy and the scaffolds are applicable to the type of design problems as discussed in section 1.5.1. The problem contexts chosen in this thesis are based on usage familiarity. The four problems used in the five studies were selected, as the students would be familiar in terms of usage, at least partially, to the software

systems. The indications for functional decomposition make the design problem tractable for novices. Familiar problems have been used in this thesis. The proposed pedagogy may not apply to creative problems.

9.6.3 Singular perspective – cognitive

In this thesis we have employed several methods appropriate to the research question. Our results depend on the theoretical lens through which we view SCD. We believe that design is created as the designer interacts with self, creates representations outside, evaluates, makes changes and so on. So it was important to understand the interactions with self as well as the designer's environment. Any other theoretical lens would lead to other results. In this thesis we have addressed the individual's cognitive aspects with respect to SCD and support novices teaching and learning of SCD.

9.7 Implications

9.7.1 Theory of novices' SCD practices

In this thesis, we analysed the design strategies and cognitive processes using the FBS design framework and linkography. We found that novices were unable to create software design as they fixate on F/B/S design elements and unable to create integrated SCD. The design strategies of novices were either F/B/S based. We found that novices were successful when they used structure based design strategies and triggered cognitive processes like expert software designers. In the context of software design our work offers insight about novices' software design processes.

Creating integrated solutions and mapping the different UML models are disciplinary practices in software design. The novice study offers insight on novice's current conceptions about SCD. To support and foster integrated model building, we created FBS graph based pedagogy. Our work offers insights on how this pedagogy fostered the disciplinary practice among novices. The results from novice study, FBS graph pedagogy and novices' conceptual change could be a part of the discipline based education theory of software design.

9.7.2 Teaching and learning of SCD

The design features of 'think & link' could be utilized by teachers/researchers who want to develop conceptual change in novices' understanding of SCD. The results from study 4 and 5 have implications for the teaching and learning of software conceptual design. The major implications are discussed below:

- Explicitly think and link design elements – UML modeling has various representations pertaining to each view of the solution. The learners need to be made aware that the representations are linked. Linking these representations will help the learner build cohesive design solutions. FBS graph can be used as external representation for teaching and learning of SCD. The teaching and learning needs to include activities pertaining to creating such integrated representations.
- Scaffolds for cognitive processes – Teaching and learning of conceptual design in software engineering predominantly revolves around tool usage and understanding the syntax of the representation (UML diagrams). In think & link the prompts could be broadly classified as information, procedural and cognitive. Instructors could provide such prompts during the process of SCD taking into account the learner's actions and the context. Learners need to be provided with explicit scaffolds for the cognitive processes like mental simulation, abstraction and association (Ball et al., 2010).
- Deliberate practice of SCD – Learners need to understand the reason for conceptual design before actually implementing the software using programming languages. They would need to be provided with design problems that are tractable at the same time necessitating the conceptual design activity. During the process of practice, learners need to be gradually taken through the planes of cognition – doing, evaluation and synthesis. Opportunities for reflection and abstraction of the process of SCD need to be provided to the learners.

In this chapter we have shown how the research goals have been addressed by integrating all the findings. We also provided the implications and limitations of our findings. In the next chapter we discuss the future work.

Chapter 10

Conclusion

Characterization of novices' design processes and designing a learning environment to support novices learning of SCD are the research goals of this thesis. These research goals led to the two iterations of a DBR project. We conducted five studies in the two iterations of DBR. Study 1, 2 and 3 were part of DBR cycle 1. In this cycle we used the FBS design framework lens to - understand novices design processes in SCD and build preliminary interventions for novices to learn SCD. The findings from this cycle were utilized to create FBS graph based pedagogy in DBR cycle 2. This was implemented as a web-based learning environment named 'think & link'. Study 4 and 5 were conducted to understand and evaluate the effects of 'think & link'. In this chapter we provide the conclusions of this thesis and the future work that we envision.

10.1 Contributions of thesis

The Table 10.1 summarises the contributions of this thesis and provides the implications of the contributions as well.

Towards theory of software design education - This thesis provides detailed characterization of novice design strategies and cognitive processes in software conceptual design as discussed in section 9.7.1. These results have implications for computing and software education researchers to utilize these results and methods to add to the theory of novice software designers. Learning scientists and design educators could examine the similarity of these results in their respective contexts and settings.

Towards pedagogy of software conceptual design - This thesis describes a FBS graph based pedagogical design of a learning environment for supporting novices' software conceptual design creation. Instructional designers and developers can adapt this design to develop technology-enhanced learning environments for teaching and learning of software conceptual design.

This thesis identifies a set of scaffolds necessary for teaching and learning of software conceptual design. Instructional designers and engineering educators can use these scaffolds for teaching and learning of other software design problems.

Towards teaching practices of software conceptual design - 'think & link' is an instantiation of the FBS based pedagogy. It helps learners to create integrated multiple representations by thinking in terms of FBS for a given design problem context. 'think & link' can be used throughout the semester with the software engineering course or corresponding laboratory course as an accompanying tool. 'think & link' has a teacher-authoring tool for extending the FBS graph contexts to different design problems.

Table 10.1 Thesis contributions and implications

Contributions	Implications for
Characteristics of novices' design strategies and cognitive processes while creating software conceptual design	Researchers in computing education research, learning science and design education
Identified a set of features and scaffolds for novices teaching-learning of FBS based software conceptual design	Instructional designers and software engineering educators
Pedagogical design of a FBS based learning environment for teaching-learning of software conceptual design	Instructional designers and software engineering educators
Identified the usage of features in the learning environment by engineering undergraduates	Instructional designers, Researchers in building TELE

think & link is an instantiation of the FBS based pedagogy. It helps learners to create integrated multiple representations by thinking in terms of FBS for a given design problem context. We have provided a teacher-authoring tool for different FBS graph contexts.	Software engineering students and software engineering educators

10.2 Future Work

This thesis work has brought out several new research questions and so our work can be productively taken forward in multiple directions. We begin with the results of the last study.

10.2.1 Mining for learner actions and FBS graph in 'think & link'

All learner actions in 'think & link' are logged and stored in the database. With the FBS graph we could explore how learners engage with FBS design elements. We could at the same time compare the action sequences as learners engaged with the FBS design elements in the FBS graph. As our samples were smaller, there were limits to the kind of analyses possible. However, with large data collection and over semester-long studies, we could create learner models about FBS design elements usage from the data. The analysis could also then categorize productive and unproductive actions with respect to FBS graph. We could make predictions of learners' actions and based on which category they fall into the actions could then be scaffolded and guided towards productive disciplinary practices via the FBS graph. This project could be a master's level project for a master's student with computer science education and learning analytics interests.

10.2.2 Adaptive visual dialogue agent for 'think & link'

The pedagogical agent CASA provided the prompts and scaffolds for the creation and evaluation of the FBS graph. As learners navigate and assimilate the FBS graph, a visual dialogue agent that can prompt the learners, get the learners to reflect, respond and provide feedback. The dialogue between the agent and learner needs to be visual as well. The agent currently in 'think & link' is based on learner actions but the

prompts placed on the left side of the learning environment do not bring the learner's attention. This could be the reason that the learners continue with their own actions and do not follow the recommended actions while editing or creating the FBS graph. This agent could model "persuasive interactions between the learner and the system".

10.2.3 Large study for understanding of novice design strategies and cognitive processes in SCD

In this work, we have brought out the novices' design processes in software conceptual design with a small set of participants. In this thesis we have established a process of analysis of software conceptual design. A larger study similar to the scaffolding project (Eckerdal et al., 2006) can be undertaken. It would involve evaluation of artifacts, novice processes and reflection. The replication of study 1 in this thesis on a large scale would add to the theory about novices' difficulties in software design. This theory would be interesting for researchers in computer education and software design educators. This project can be undertaken as a multi-institutional study by a post-doc researcher.

10.2.4 Unpacking the conceptual change through large scale implementations of 'think & link' in classrooms

In this thesis we conducted two studies (study 4 & 5) as workshops for a day or utmost two days. It would be interesting to take 'think & link' as a teaching-learning tool in a semester-long course of UML modeling or software engineering. From study 5 we observed that second year undergraduate engineering students who were recently exposed to UML modeling had performed better than study 4 participants who were final year engineering students. So as and when computer and information technology students learn UML modeling, 'think & link' could be used as a teaching-learning tool. The evaluation that happens over the semester-long study would provide deeper understanding about the gradual conceptual change. This project could be a master's level project for a student with computer science education interests.

10.2.5 Implementing assessment in 'think & link'

In each phase of think & link we observe differences in the participants based on their post-test category. In phase 1 we observe differences in participants' understanding of FBS framework. Some participants abstract the FBS framework starting with

structures and some other abstract with respect to the functions. However be the nature or sequence of abstraction, we would need to ensure that learners understand FBS as a logical unit. Based on learners' response to answers in the recall task at the end of phase1, we would need to assess the responses. Based on the assessment appropriate feedback needs to be provided to the participants.

In phase 2 we observe that participants in the lower categories of post-test did not edit the FBS graph. We need to assess the FBS graph for edits, so that we ensure that participants complete the task of editing the FBS graph in phase 2. Additionally we would need to provide assessment of the FBS graph in phase 2 as well as phase 3. This evaluation would need to be compared with the participants' self-evaluation. The comparison between the automated assessment as well as participants' self-evaluation could lead to participants creating FBS graph at the target level of the rubric. Additionally the assessment of the FBS graph by the system would ensure that participants are corrected in case of inaccuracies in the FBS graph.

To make sure that learners' takeaway the intended goals of the environment, assessment needs to be incorporated into the three phases. Based on the assessment of these, appropriate prompts and scaffolds need to be added to the existing version of 'think & link'. 'think & link' can then be evaluated with research questions similar to study 4 and 5.

10.2.6 Designing for reflection in SCD among learners and instructors

Learning analytics dashboard (LAD) is the integration of the learning analytics approach with the concept of a dashboard, which visualizes the information regarding learning data, learning patterns, and behaviours (Teasley, 2017). With the help of a learning analytics dashboard the learners can monitor and reflect on their learning processes (Klerkxx et al., 2017). They can then compare their progress to their learning goals and make necessary adjustments. In 'think & link' we log the learner actions and post-hoc we have analysed the learner's progress. If we could analyse their current actions and map it to the learning goals it would serve as visualization for meta-cognitive reflection and evaluation. Learners plan their goals before every phase begins, they can compare their plans to the actions and make necessary adjustments to their actions or goals.

A similar dashboard can be provided to teachers about the progress of the students in 'think & link'. The learners' understanding about FBS and the evaluation

of the FBS graphs could be provided to the teacher. The teacher based on the data available can choose to intervene and provide guidance to the learners. Such a dashboard for a teacher would be helpful to also access the active engagement of the learners in the learning environment when it is being used in laboratory classes.

Previous research about providing such features to learners as well as instructors has shown positive effects on final scores (Kim et al., 2016). However, we conjecture that the features such as personalized dashboard along with feedback and explanation on how to improve and interpret the results could lead to better learning and awareness during the process of the learning. Testing this conjecture could be an interesting area of research using 'think & link'.

10.2.7 Role of affect in SCD

We did not systematically incorporate or investigate the role of motivation and interest in this thesis as mentioned in section 9.6.1. For studies 1-5, we purposively sampled motivated and interested participants and so the lack of motivation and interest was not a factor in the software conceptual design process. It is important to investigate the effect of these and other potential affective factors on software conceptual design. One way to investigate these affective factors is to intermittently track novices' self-reported motivation, interest, self-efficacy, to name a few, as they are completing activities and correlate these affective factors with their productive and unproductive action patterns. This will throw light on how affective factors impact performance, and how performance in turn influences these affective factors. These findings may also suggest what features are required in 'think & link' in order to trigger and maintain the relevant affective factors such as interest, motivation and self-efficacy among others.

10.2.8 Role of collaboration in SCD

In this thesis we have focused only on individual learners. However, we know that collaboration is known to be a disciplinary practice in software design. In the workplace, software engineers need to collaborate with other designers, developers and team members to complete the process of design, development, implementation and deployment of software. Additionally collaboration while working on ill-structured problem solving is known to trigger learning processes, strategies that are not available while they work alone (Schwartz, 1995). Collaboration has been found

to be very useful in design especially in the early conceptual phases of software development (Brooks, 1986).

However productive collaboration needs design and it does not happen automatically. So it would be an interesting area to explore how collaboration can be incorporated in the FBS graph based pedagogy. There are different phases in 'think & link'. We would like to understand at which phase in the 'think & link', would the collaboration be more beneficial for software conceptual design. The broad research questions of interest would be -

1. What is the effect of collaboration in software conceptual design outcome?
2. How should learners collaborate while learning software conceptual design?
3. What would be the nature of collaboration while learning software conceptual design that would aid in disciplinary practices?
4. At what phases of 'think & link' should the learners collaborate while learning software conceptual design?

10.2.9 Taking turns in design – Role of switching perspectives while design (end user & system)

In this thesis we focussed on working with problems that participants are familiar with. While solving such problems, we noticed that participants' take perspectives of self-using similar systems. The participants also take perspective of the system working and come up with the internal working of the system. This aspect of perspective taking could be more examined in the context of creating software conceptual design solutions.

Perspective taking is a cognitive process in which individuals adopt others' viewpoints in an attempt to understand their preferences, values, and needs (Parker & Axtell, 2001). Theories suggest that by engaging in perspective taking designers obtain a clearer, more integrative understanding of what types of ideas will be useful to the stakeholders (Grant & Berry, 2011). This aspect of the cognitive process needs deeper understanding in the context of software design. It is important to understand how novices engage in perspective taking. At the same time, how do expert software designers engage with perspective taking? How do expert software designers switch between perspectives? This aspect of perspective taking in the context of software conceptual design would add to the theory on the disciplinary practices of software design.

10.3 Final reflection

This thesis has been an attempt to unpack novice design processes and support them in the context of software conceptual design. The aim of initiating the undergraduate engineering students in the disciplinary practices employed by practicing software engineers is fulfilled through the learning environment 'think & link'. We utilized the theoretical lens of FBS design framework to understand novices' software conceptual design processes. At the same time we utilized it to design pedagogy to support novice design processes.

In the process of this thesis, I have undergone change in the perspective of learners/ novices. I have begun to understand that novices have the capabilities to think and do as experts, however it is the difficulties that need to be unearthed and alleviated. I believe that the teaching and learning is a rich and complex phenomena intertwined within a context, with the learner at the centre. The context needs to provide opportunities for learners to perform actions and reflect on them. In this phenomenon we have the learners, the material they interact with and the context they interact in. The learner is a social being and comes with identity and agency. There emerges a strong need to acknowledge the existence of learner 's identity, agency, intuition and complex reasoning, the last two of which can be recruited for formal learning. The material that the learner interacts with needs to be malleable which allows participation, supports expressiveness and transformation. The context provides meaningful activity for the learners to interact with the material and make sense of the interaction. The contexts need to be designed as 'enrichment frames' consisting of meaningful activities, which are open ended, connected to intuition, malleable, localized and contextualized. The learners would go through a pedagogical experience where they bring in their relevant knowledge, engage in enrichment frames, reflect and make connections beyond the discipline.

Going back to my belief of teaching and learning to be contextual, most of the learning theories have been developed based on the research in USA/European countries. I would be interested to study the relevance of the learning theories in the context of Asian countries and maybe create theories of learning in the sub-continent contexts. This thesis is the beginning in that direction.

Appendix

A. Consent form

Consent to Participate in Educational Research

[Title: Understanding software conceptual design]

You have been asked to participate in a research study conducted by T. G. Lakshmi from the Inter-Disciplinary Program in Educational Technology at the Indian Institute of Technology Bombay (IITB). The purpose of the study is to gather requirements for developing TEL systems to teach software conceptual design. You were selected as a possible participant in this study because of your educational background as a 3rd year engineering student.

· **PARTICIPATION AND WITHDRAWAL**

Your participation in this study is completely voluntary and you are free to choose whether to be in it or not. If you choose to be in this study, you may subsequently withdraw from it at any time without penalty or consequences of any kind. The investigator may withdraw you from this research if circumstances arise that warrant doing so.

You will not be compensated for the participation. You should read the information below, and ask questions about anything you do not understand, before deciding whether or not to participate.

· **PURPOSE OF THE STUDY**

The study is designed to understand software conceptual design processes and gather requirements for developing a TEL system. The TEL system intends to develop software conceptual design in computer science engineering undergraduates.

· **PROCEDURES**

If you volunteer to participate in this study, we would ask you to do the following things:

1. Come up with a conceptual design for a software system
2. Participate in interview

Participating in this research study is voluntary. You have the right not to answer any question, and to stop your participation in the study at any time. We expect that the study will take about 3 hours. Your interactions will be audio and video recorded. A screen recording of your interactions with the computer will be done.

· **POTENTIAL BENEFITS**

- Apply concepts learned to a real-world problem
- An immersive experience of a requirements gathering and software design and activity

· **CONFIDENTIALITY**

Any information that is obtained in connection with this study and that can be identified with you will remain confidential and will be disclosed only with your permission or as required by law. We will not use your name in publications; however, we may need to use your academic performance details if you give us permission.

· **IDENTIFICATION OF INVESTIGATORS**

If you have any questions or concerns about the research, please feel free to contact T.G Lakshmi (lakshmiganesht@gmail.com) or Prof. Sridhar Iyer, CDEEP IITB (sri@iitb.ac.in) with any questions or concerns.

SIGNATURE OF PARTICIPANT

I understand the procedures described above. My questions have been answered to my satisfaction, and I agree to participate in this study. I have been sent a copy of this form.

_____
Name of Participant

_____ _____
E-mail address Contact No.

_____ _____
Signature Date

B. Sample interview questions for study 4 & 5

The following sets of questions were used for the focus group interview after the participants have completed the post-test. The interview began with broad questions, and then to more specific questions.

Questions for Focus group interview

- Broadly can you talk about what you did in 'think & link'?
- Lets go through each phase
    - What was the phase?
    - What did you do in there?
    - How did you solve this task/question?
    - What were the strategies used to solve the tasks/question?
    - What was the role of the given info/tool/feature in 'think & link' in solving the task? What would have happened if this info/tool/feature was not present? What would you have done?
        - How did you use the info/tools/feature to solve the task?

        - How was the info/tool/feature of ‘think & link’ useful/not useful in solving the task?
        - What more did you need to complete the task?
        - How would you recommend changing/adding/removing to the features of think & link to do the task?
- What was the approach you took to solve the 'finger print atm system' (post-test) design problem?
    - How did you solve problems like these previously?
    - Was there a difference in solving the problems before and after using ‘think & link’?
    - Compare the UML diagram you drew for your final year project and then the FBS graph that you drew
    - Any changes you will make?
    - Were there any differences in your approach of conceptual design before and after the usage of the system? If yes, then can you say what they were?
    - By going through the FBS graph did it change the way you think about software conceptual design?
    - Will you use FBS graph for software conceptual design?
- What according to you is a process of creating software conceptual design? How did you understand this? Has it changed from previous? If yes, why has it changed and how did it change?
- What features in ‘think & link’ helped you understand this?
- What is the role of this conversation on your learning today?
- How would you redesign ‘think & link’ for a student such as yourself to better learn software conceptual design?

C Scripts for analysis of log data

We present the R script used for analysis of the log data. In RQ 3.d. (chapter 7) we have mentioned the results of the analysis. Here we present the R script using the TraMineR library.

```
#using the library#
library(TraMineR)
#setting the workspace#
setwd("~/Documents/Lakshmi/Seminar/Learning Analytics/SAKEC/“)
#reading the source file#
mvad <- read.csv(file = "tse-sequence-intro.csv", header = TRUE)
#creating a time stamped event sequence#
mvad.seqe <- seqecreate(id=mvad$user_id,timestamp = mvad$event_time, event =
mvad$event)
#extracting subsequences found in 50% cases with 4 as number of events in a
window#
mvad.subseqee <-seqefsub(mvad.seqe,pmin.support=0.5, max.k = 4)
#writing subsequences into a file#
df <- mvad.subseqee$data
df$subseq <- as.character(mvad.subseqee$subseq)
write.csv(df,’subsequences-intro.csv')
#setting screen size#
par(mar=c(4,15,2,1))
#ordering successive sequences#
seqpcplot(mvad.seqe,
  filter = list(type = "function",
                value = "cumfreq",
                level = 0.8),
  order.align = "last",
  ltype = "non-embeddable",
  cex = 1.5, lwd = .9,
  lcourse = "downwards"
```

References

Abelson, H., & Greenspun, P. (2001). Teaching software engineering-lessons from MIT. In *Proceedings 10th International World Wide Web Conference*.

Abernethy, K., Kelly, J., Sobel, A., Kiper, J. D., & Powell, J. (2000, March). Technology transfer issues for formal methods of software specification. In *Thirteenth Conference on Software Engineering Education and Training* (pp. 23-31). IEEE.

Ahmed, S., Wallace, K. M., & Blessing, L. T. (2003). Understanding the differences between how novice and experienced designers approach design tasks. *Research in engineering design*, *14*(1), 1-11.

Ahmed, S., Wallace, K. M., & Blessing, L. T. (2003). Understanding the differences between how novice and experienced designers approach design tasks. *Research in engineering design*, *14*(1), 1-11.

Akayama, S., Demuth, B., Lethbridge, T. C., Scholz, M., Stevens, P., & Stikkolorum, D. R. (2013, September). Tool Use in Software Modelling Education. In *EduSymp@ MoDELS*.

Altadmri, A., & Brown, N. C. (2015, February). 37 million compilations: Investigating novice programming mistakes in large-scale student data. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (pp. 522-527).

Armarego, J. (2009). Constructive Alignment in SE education: aligning to what?. In *Software Engineering: effective teaching and learning approaches and practices* (pp. 15-37). IGI Global.

Atkinson, R. K., Derry, S. J., Renkl, A., & Wortham, D. (2000). Learning from examples: Instructional principles from the worked examples research. *Review of educational research*, *70*(2), 181-214.

Atman, C. J., & Bursic, K. M. (1998). Documenting a process: The use of verbal protocol analysis to study engineering student design. *Journal of Engineering Education*, *87*(2), 121-132.

Atman, C. J., Adams, R. S., Cardella, M. E., Turns, J., Mosborg, S., & Saleem, J. (2007). Engineering design processes: A comparison of students and expert practitioners. *Journal of engineering education*, *96*(4), 359-379.

Atman, C. J., Cardella, M. E., Turns, J., & Adams, R. (2005). Comparing freshman and senior engineering design processes: an in-depth follow-up study. Design studies, 26(4), 325-357.

Atman, C. J., Chimka, J. R., Bursic, K. M., & Nachtmann, H. L. (1999). A comparison of freshman and senior engineering design processes. *Design studies*, *20*(2), 131-152.

Aurigemma, J., Chandrasekharan, S., Nersessian, N. J., & Newstetter, W. (2013). Turning experiments into objects: The cognitive processes involved in the design of a lab on a chip device. *Journal of Engineering Education*, *102*(1), 117-140.

Bakker, A., & Van Eerde, D. (2015). An introduction to design-based research with an example from statistics education. In *Approaches to qualitative research in mathematics education* (pp. 429-466). Springer, Dordrecht.

Ball, L. J., Onarheim, B., & Christensen, B. T. (2010). Design requirements, epistemic uncertainty and solution development strategies in software design. *Design Studies*, *31*(6), 567-589.

Ball, L. J., St. BT Evans, J., Dennis, I., & Ormerod, T. C. (1997). Problem-solving strategies and expertise in engineering design. *Thinking & Reasoning*, *3*(4), 247-270.

Barab, S. (2014). Design-based research: A methodological toolkit for engineering change. In *The Cambridge Handbook of the Learning Sciences, Second Edition* (pp. 151-170). Cambridge University Press.

Barab, S. A., & Squire, K. (2004). Design-based research: Putting a stake in the ground. The Journal of the Learning Sciences, 13(1), 1-14. doi:10.1207/s15327809jls1301_1

Beckman, K., Coulter, N., Khajenoori, S., & Mead, N. R. (1997). Collaborations: closing the industry-academia gap. *IEEE software*, *14*(6), 49-57.

Bergmann, R. (2003). *Experience management: foundations, development methodology, and internet-based applications* (Vol. 2432). Springer.

Bhatta, S., Goel, A., & Prabhakar, S. (1994). Innovation in analogical design: A model-based approach. In *Artificial Intelligence in Design'94* (pp. 57-74). Springer, Dordrecht.

Bogdan, R. C., & Biklen, S. K. (2007). Research for education: An introduction to theories and methods.

Bracewell, R. H., & Sharpe, J. E. E. (1996). Functional descriptions used in computer support for qualitative scheme generation-'Schemebuilder'. *Ai Edam*, *10*(4), 333-345.

Brand-Gruwel, S., Wopereis, I., & Vermetten, Y. (2005). Information problem solving by experts and novices: Analysis of a complex cognitive skill. *Computers in Human Behavior*, *21*(3), 487-508.

Braun, V., & Clarke, V. (2017) Thematic analysis, The Journal of Positive Psychology, 12:3, 297-298, DOI: 10.1080/17439760.2016.1262613

Brooks, E (1986). *No Silver Bullet*. In Proceedings of the IFIP Tenth World Computing Congress

Brown, D. C., & Chandrasekaran, B. (2014). *Design problem solving: knowledge structures and control strategies*. Morgan Kaufmann.

Buckley, B. C. (2000). Interactive multimedia and model-based learning in biology. *International journal of science education*, *22*(9), 895-935.

Buckley, B. C., Gobert, J. D., Kindfield, A. C., Horwitz, P., Tinker, R. F., Gerlits, B., ... & Willett, J. (2004). Model-based teaching and learning with BioLogica™: What do they learn? How do they learn? How do we know?. *Journal of Science Education and Technology*, *13*(1), 23-41.

Bürgin, R., & Ritschard, G. (2014). A decorated parallel coordinate plot for categorical longitudinal data. *The American Statistician*, *68*(2), 98-103.

Chakrabarti, A., & Bligh, T. P. (2001). A scheme for functional reasoning in conceptual design. *Design Studies*, *22*(6), 493-517.

Chakrabarti, A., Siddharth, L., Dinakar, M., Panda, M., Palegar, N., & Keshwani, S. (2017, January). Idea Inspire 3.0—A tool for analogical design. In *International Conference on Research into Design* (pp. 475-485). Springer, Singapore.

Cherubini, M., Venolia, G., DeLine, R., & Ko, A. J. (2007, April). Let's go to the whiteboard: how and why software developers use drawings. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 557-566).

Chi, M. T. (2008). Three types of conceptual change: Belief revision, mental model transformation, and categorical shift. *International handbook of research on conceptual change*, *61*, 82.

Chi, M. T., & Glaser, R. (1985). *Problem-solving ability* (p. 27). Learning Research and Development Center, University of Pittsburgh.

Cho, M. H. (2004). The Effects of Design Strategies for Promoting Students' Self-Regulated Learning Skills on Students' Self-Regulation and Achievements in Online Learning Environments. *Association for Educational Communications and Technology*.

Chren, S., Buhnova, B., Macak, M., Daubner, L., & Rossi, B. (2019, May). Mistakes in UML diagrams: analysis of student projects in a software engineering course. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)* (pp. 100-109). IEEE.

Chrysikou, E. G., & Weisberg, R. W. (2005). Following the wrong footsteps: fixation effects of pictorial examples in a design problem-solving task. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *31*(5), 1134.

Clancy, M. J., & Linn, M. C. (1999). Patterns and pedagogy, in 'The proceedings of the thirtieth SIGCSE technical symposium on Computer science education'.

Cobb, P., Confrey, J., DiSessa, A., Lehrer, R., & Schauble, L. (2003). Design experiments in educational research. *Educational researcher*, *32*(1), 9-13.

Collofello, J. S. (2000). University/industry collaboration in developing a simulation-based software project management training course. *IEEE Transactions on Education*, *43*(4), 389-393.*Conference on Computer Science Education* (Koli Calling 2005), TUCS General Publication

Cross, N. (2004). Expertise in design: an overview. *Design studies*, *25*(5), 427-441.

Cross, N., Christiaans, H., & Dorst, K. (1994). Design expertise amongst student designers. *Journal of Art & Design Education*, *13*(1), 39-56.

Cross, N., Christiaans, H., & Dorst, K. (Eds.). (1996). *Analysing design activity*. Wiley.

Cunningham, K. (2020, August). Purpose-first Programming: A Programming Learning Approach for Learners who Care Most About What Code Achieves. In *Proceedings of the 2020 ACM Conference on International Computing Education Research* (pp. 348-349).

Darke, J. (1979). The primary generator and the design process. *Design studies*, *1*(1), 36-44.

Dasgupta, C. (2019). Improvable models as scaffolds for promoting productive disciplinary engagement in an engineering design activity. *Journal of Engineering Education*, *108*(3), 394-417.

Dasgupta, C. (2019). Improvable models as scaffolds for promoting productive disciplinary engagement in an engineering design activity. *Journal of Engineering Education*, *108*(3), 394-417.

Davis, A., Overmyer, S., Jordan, K., Caruso, J., Dandashi, F., Dinh, A., ... & Ta, A. (1993, May). Identifying and measuring quality in a software requirements specification. In *[1993] Proceedings First International Software Metrics Symposium* (pp. 141-152). IEEE.

Dinar, M., Shah, J. J., Cagan, J., Leifer, L., Linsey, J., Smith, S. M., & Hernandez, N. V. (2015). Empirical studies of designer thinking: past, present, and future. *Journal of Mechanical Design*, *137*(2).

DiSessa, A. A. (1993). Toward an epistemology of physics. *Cognition and instruction*, *10*(2-3), 105-225.

DiSessa, A. A. (2014). *A history of conceptual change research: Threads and fault lines*.

Dole, J. A., & Sinatra, G. M. (1998). Reconceptalizing change in the cognitive construction of knowledge. *Educational psychologist*, *33*(2-3), 109-128.

Drappa, A., & Ludewig, J. (2000, June). Simulation in software engineering training. In *Proceedings of the 22nd international conference on Software engineering* (pp. 199-208). Dublin, Ireland. Amsterdam: Elsevier, pp. 1069-1076.

Duit, R., Treagust, D., & Widodo, A. (2008). Teaching science for conceptual change: Theory and practice. In *International handbook of research on conceptual change* (pp. 629-646). Routledge.

Dym, C. L., Agogino, A. M., Eris, O., Frey, D. D., & Leifer, L. J. (2005). Engineering design thinking, teaching, and learning. *Journal of engineering education*, *94*(1), 103-120.

Eckerdal, A., McCartney, R., Mostro¨m, J.E., Ratcliffe M., & Zander, C. (2006b). Comparing student software designs using semantic categorization.

Eckerdal, A., McCartney, R., Moström, J. E., Ratcliffe, M., & Zander, C. (2006). Can graduating students design software systems?. *ACM SIGCSE Bulletin*, *38*(1), 403-407.

Eckerdal, A., McCartney, R., Moström, J. E., Ratcliffe, M., & Zander, C. (2006a). Categorizing student software designs: Methods, results, and implications. *Computer Science Education*, *16*(3), 197-209.

Efklides, A. (2009). The role of metacognitive experiences in the learning process. *Psicothema*, *21*(1), 76-82.

Ellis, H. J. (Ed.). (2008). *Software Engineering: Effective Teaching and Learning Approaches and Practices: Effective Teaching and Learning Approaches and Practices*. IGI Global.

Erden, M. S., Komoto, H., van Beek, T. J., D'Amelio, V., Echavarria, E., & Tomiyama, T. (2008). A review of function modeling: Approaches and applications. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AI EDAM*, *22*(2), 147.

Fiorineschi, L., Rotini, F., & Rissone, P. (2016). A new conceptual design approach for overcoming the flaws of functional decomposition and morphology. *Journal of Engineering Design*, *27*(7), 438-468.
Flavell, J. H. (1979). Metacognition and cognitive monitoring: A new area of cognitive–developmental inquiry. *American psychologist*, *34*(10), 906.

Fry, H., Ketteridge, S., & Marshall, S. (2008). Understanding student learning. In *A handbook for teaching and learning in higher education* (pp. 26-44). Routledge.

Galle, P. (2009). The ontology of Gero's FBS model of designing. *Design Studies*, *30*(4), 321-339.

Gasparini, A. (2015, February). Perspective and use of empathy in design thinking. In *ACHI, the eight international conference on advances in computer-human interactions* (pp. 49-54).

Gero, J. S., & Kannengiesser, U. (2014). The function-behaviour-structure ontology of design. In *An anthology of theories and models of design* (pp. 263-283). Springer, London.

Gero, J., Kan, J. and Pourmohamadi, M. 2011. Analysing design protocols: Development of methods and tools. . *In ICORD 11: Proceedings of the 3rd International Conference on Research into Design Engineering*. (2011), 3-10.

Gibbs, G. R. (2007). Thematic coding and categorizing. *Analyzing qualitative data*, *703*, 38-56.

Giering, J. A. (2012). Use of evaluation to design quality online learning: understanding the shared experience.

Gilhooly, R. H. K. (1997). Introduction domains, paradigms, and methods in the study of expertise. *Thinking & Reasoning*, *3*(4), 241-246.

Gnatz, M., Kof, L., Prilmeier, F., & Seifert, T. (2003, March). A practical approach of teaching software engineering. In *Proceedings 16th Conference on Software Engineering Education and Training, 2003.(CSEE&T 2003).* (pp. 120-128). IEEE.

Goel, A. K., de Silva Garza, A. G., Grué, N., Murdock, J. W., Recker, M. M., & Govindaraj, T. (1996, June). Towards design learning environments—I: Exploring how devices work. In *International conference on intelligent tutoring systems* (pp. 493-501). Springer, Berlin, Heidelberg.

Goel, A., Rugaber, S., & Vattam, S. (2009). Structure, behavior & function of complex systems: The SBF modeling language. *International Journal of AI in Engineering Design, Analysis and Manufacturing*, *23*(1), 23-35.

Goldschmidt, G. (2004). Design representation: Private process, public image. In *Design representation* (pp. 203-217). Springer, London.

Goldschmidt, G. (2014). *Linkography: unfolding the design process*. Mit Press.
Goold, A., & Horan, P. (2002, February). Foundation software engineering practices for capstone projects and beyond. In *Proceedings 15th Conference on Software Engineering Education and Training (CSEE&T 2002)* (pp. 140-146). IEEE.

Grant, A. M., & Berry, J. W. (2011). The necessity of others is the mother of invention: Intrinsic and prosocial motivations, perspective taking, and creativity. *Academy of management journal*, *54*(1), 73-96.

Halling, M., Zuser, W., Kohle, M., & Biffl, S. (2002, February). Teaching the unified process to undergraduate students. In *Proceedings 15th Conference on Software Engineering Education and Training (CSEE&T 2002)* (pp. 148-159). IEEE.

Harman, G. (1986). Change in view Cambridge MA.

Hatcher, G., Ion, W., Maclachlan, R., Marlow, M., Simpson, B., Wilson, N., & Wodehouse, A. (2018). Using linkography to compare creative methods for group ideation. *Design Studies*, *58*, 127-152.

Hay, L., Duffy, A. H., McTeague, C., Pidgeon, L. M., Vuletic, T., & Grealy, M. (2017). Towards a shared ontology: A generic classification of cognitive processes in conceptual design. *Design Science*, *3*.

Hay, L., Duffy, A. H., McTeague, C., Pidgeon, L. M., Vuletic, T., & Grealy, M. (2017). A systematic review of protocol studies on conceptual design cognition: Design as search and exploration. *Design Science*, *3*.

Hmelo-Silver, C. E., & Pfeffer, M. G. (2004). Comparing expert and novice understanding of a complex system from the perspective of structures, behaviors, and functions. *Cognitive science*, *28*(1), 127-138.

Hmelo, C. E., Holton, D. L., & Kolodner, J. L. (2000). Designing to learn about complex systems. *The journal of the learning sciences*, *9*(3), 247-298.

Hokanson, B. (2001, August). Digital image creation and analysis as a means to examine learning and cognition. In *International Conference on Cognitive Technology* (pp. 226-232). Springer, Berlin, Heidelberg.

Hughes, J., & Parkes, S. (2003). Trends in the use of verbal protocol analysis in software engineering research. *Behaviour & Information Technology*, *22*(2), 127-140.

Hungerford, B. C., Hevner, A. R., & Collins, R. W. (2004). Reviewing software diagrams: A cognitive study. *IEEE Transactions on Software Engineering*, *30*(2), 82-96.

Hutchins, E., & Klausen, T. (1996). Distributed cognition in an airline cockpit. *Cognition and communication at work*, 15-34.

Hyde, K. F. (2000). Recognising deductive processes in qualitative research. *Qualitative market research: An international journal*.

IEEE (1998). *IEEE Recommended Practice for Software Requirements Specifications*.

IEEE Std. 830. The Institute of Electrical and Electronics Engineers, Inc., USA.

IEEE Standards Coordinating Committee. (1990). IEEE Standard Glossary of Software Engineering Terminology (IEEE Std 610.12-1990). Los Alamitos. *CA: IEEE Computer Society*, *169*.

ISO (2001). Software Product Evaluation-Quality Characteristics and Guidelines for their Use, ISO/IEC Standard 9126, Switzerland.

Iwasaki, Y., Fikes, R., Vescovi, M., & Chandrasekaran, B. (1993, January). How things are intended to work: Capturing functional knowledge in device design. In *IJCAI* (pp. 1516-1522).

Jackson, D. (2013). Conceptual design of software: A research agenda.

Jansson, D. G., & Smith, S. M. (1991). Design fixation. *Design studies*, *12*(1), 3-11.

Jiang, H., & Yen, C. C. (2013). Design Thinking in Conceptual Design Processes: A Comparison Between Industrial and Engineering Design Students. *Advances in Industrial Design Engineering*, 29.

Jin, Y., & Benami, O. (2010). Creative patterns and stimulation in conceptual design. *Ai Edam*, *24*(2), 191-209.

Johri, A., & Lohani, V. K. (2011). Framework for improving engineering representational literacy by using pen-based computing. *International Journal of Engineering Education*, *27*(5), 958.

Jonassen, D. H. (2000). Toward a design theory of problem solving. *Educational technology research and development*, *48*(4), 63-85.

Jonassen, D. H., & Easter, M. A. (2013). Model building for conceptual change. In International handbook of research on conceptual change (pp. 580-600).

Joshi, A. (2009). Usability goals setting tool. In *4th Workshop on Software and Usability Engineering Cross-Pollination: Usability Evaluation of Advanced Interfaces, Uppsala*.

Kan, J. W., & Gero, J. S. (2017). *Quantitative methods for studying design protocols*. Dordrecht: Springer.

Kannengiesser, U., & Gero, J. S. (2019). Design thinking, fast and slow: A framework for Kahneman's dual-system theory in design. Design Science, 5.

Karimi, P., Grace, K., Davis, N., & Maher, M. L. (2018, July). Creative sketching apprentice: Supporting conceptual shifts in sketch ideation. In *International Conference on-Design Computing and Cognition* (pp. 721-738). Springer, Cham.

Kavakli, M., & Gero, J. S. (2002). The structure of concurrent cognitive actions: a case study on novice and expert designers. *Design studies*, *23*(1), 25-40.

Kim, J., Jo, I. H., & Park, Y. (2016). Effects of learning analytics dashboard: analyzing the relations among dashboard utilization, satisfaction, and learning achievement. *Asia Pacific Education Review*, *17*(1), 13-24.

Kirsh, D. (2010). Thinking with external representations. *AI and Society*, 25, 441–454.

Klerkx, J., Verbert, K., & Duval, E.(2017). *Learning analytics dashboards*. In C. Lang, G. Siemens, A. Wise, & D. Gašević (Eds.), The handbook of learning analytics (pp. 143–150). Society for Learning Analytics Research (SoLAR).

Kopcha, T. J., Schmidt, M. M., & McKenney, S. (2015). Editorial 31 (5): Special issue on educational design research (EDR) in post-secondary learning environments. *Australasian journal of educational technology*, *31*(5).

Kornecki, A. J. (2000, March). Real-time computing in software engineering education. In *Software Engineering Education and Training, Conference on* (pp. 197-197). IEEE Computer Society.

Kornecki, A. J., Khajenoori, S., Gluch, D., & Kameli, N. (2003, March). On a partnership between software industry and academia. In *Proceedings 16th Conference on Software Engineering Education and Training, 2003.(CSEE&T 2003)*. (pp. 60-69). IEEE.

Kruchten, P. (2005). Casting software design in the function-behavior-structure framework. *IEEE software*, *22*(2), 52-58.

Lakshmi, T. G., & Iyer, S. (2018). Exploring novice approach to conceptual design of software. International Society of the Learning Sciences, Inc.[ISLS]..

Larkin, J. H., & Simon, H. A. (1987). Why a diagram is (sometimes) worth ten thousand words. *Cognitive science*, *11*(1), 65-100.

Lawson, B. (2006). *How designers think: The design process demystified*. Routledge.

Lawson, B. (2006). *How designers think: The design process demystified*. Oxford, UK: Elsevier.

Levy, S. T., & Wilensky, U. (2008). Inventing a "mid level" to make ends meet: Reasoning between the levels of complexity. *Cognition and Instruction*, *26*(1), 1-47.

Li, P. L. (2016). What makes a great software engineer (Doctoral dissertation).

Lindland, O. I., Sindre, G., & Solvberg, A. (1994). Understanding quality in conceptual modeling. *IEEE software*, *11*(2), 42-49.

Lister, R., Adams, E. S., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, M., ... & Simon, B. (2004). A multi-national study of reading and tracing skills in novice programmers. *ACM SIGCSE Bulletin*, *36*(4), 119-150.

Lister, R., Simon, B., Thompson, E., Whalley, J. L., & Prasad, C. (2006). Not seeing the forest for the trees: novice programmers and the SOLO taxonomy. *ACM SIGCSE Bulletin*, *38*(3), 118-122.

Ma, L. (2007). *Investigating and improving novice programmersâ€™ mental models of programming concepts* (Doctoral dissertation, University of Strathclyde, Glasgow, UK).

Majumdar, R., & Iyer, S. (2016). iSAT: a visual learning analytics tool for instructors. *Research and practice in technology enhanced learning*, *11*(1), 16.

Mark Guzdial (April, 2012). https://computinged.wordpress.com/2012/04/04/practice-is-better-for-learning-facts-worked-examples-are-better-for-learning-skills/

Mark Guzdial (April, 2018) https://computinged.wordpress.com/2018/04/06/how-do-students-learn-the-notional-machine-developing-a-mental-model-of-program-behavior/

Marshall, S., & Pennington, G. (2009). Teaching excellence as a vehicle for career progression. *A Handbook for Teaching and Learning in Higher Education*, 485.

Martin, L., & Schwartz, D. L. (2009). Prospective adaptation in the use of external representations. *Cognition and Instruction*, *27*(4), 370-400.

Masur, A., & Salustri, F. A. (2007). The Idea Concept Design Process. In *DS 42: Proceedings of ICED 2007, the 16th International Conference on Engineering Design, Paris, France, 28.-31.07. 2007* (pp. 315-316).

Mathias, J. R. (1995). A study of the problem-solving strategies used by expert and novice designers.

Mc Neill, T., Gero, J. S., & Warren, J. (1998). Understanding conceptual electronic design using protocol analysis. *Research in Engineering Design*, *10*(3), 129-140.

McKenney, S., & Reeves, T. C. (2012). Conducting educational design research. London: Routledge

Medvidovic, N., Rosenblum, D. S., Redmiles, D. F., & Robbins, J. E. (2002). Modeling software architectures in the Unified Modeling Language. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, *11*(1), 2-57.

Minocha, S., & Sharp, H. (2004). Learner-centred design and evaluation of web-based e-learning environments. *In The 7th HCI educators workshop; Effective Teaching and Learning in HCI*. Preston, United Kingdom.

Monsalve, E. S., Pereira, A. X., & Werneck, V. M. B. (2014). Teaching Software Engineering through a Collaborative Game. In *Overcoming Challenges in Software Engineering Education: Delivering Non-Technical Knowledge and Skills* (pp. 310-331). IGI Global.

Moody, D. L., & Shanks, G. G. (1994, December). What makes a good data model? Evaluating the quality of entity relationship models. In *International Conference on Conceptual Modeling* (pp. 94-111). Springer, Berlin, Heidelberg.

Moody, D. L., Shanks, G. G., & Darke, P. (1998, November). Improving the quality of entity relationship models—experience in research and practice. In *International Conference on Conceptual Modeling* (pp. 255-276). Springer, Berlin, Heidelberg.

Moore, T. J., Miller, R. L., Lesh, R. A., Stohlmann, M. S., & Kim, Y. R. (2013). Modeling in engineering: The role of representational fluency in students' conceptual understanding. *Journal of Engineering Education*, *102*(1), 141-178.

Moritz, S. H., Wei, F., Parvez, S. M., & Blank, G. D. (2005). From objects-first to design-first with multimedia and intelligent tutoring. *ACM SIGCSE Bulletin*, *37*(3), 99-103.

Morrison, B. B., Margulieux, L. E., & Decker, A. (2020). The curious case of loops. *Computer Science Education, 30(2)*, 127-154.

National Research Council. (2012). *Discipline-based education research: Understanding and improving learning in undergraduate science and engineering*. National Academies Press.

Navarro, E. O., & Van Der Hoek, A. (2005, April). Scaling up: How thirty-two students collaborated and succeeded in developing a prototype software design environment. In *18th Conference on Software Engineering Education & Training (CSEET'05)* (pp. 155-162). IEEE.

Nersessian, N. J. (1999). Model-based reasoning in conceptual change. In *Model-based reasoning in scientific discovery* (pp. 5-22). Springer, Boston, MA.

Nersessian, N. J. (2010). Mental modeling in conceptual change. *International Journal on Humanistic Ideology*, *3*(01), 11-48.

Nielsen, J. (1992, June). Finding usability problems through heuristic evaluation. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 373-380).

Niepostyn, S. J. (2015). The sufficient criteria for consistent modelling of the use case realization diagrams with a new functional-structure-behaviour UML diagram. *Przegląd Elektrotechniczny Sigma NOT*, *2*, 31-35.

Niepostyn, S. J., & Bluemke, I. (2012, June). The Function-Behaviour-Structure Diagram for Modelling Workflow of Information Systems. In *International Conference on Advanced Information Systems Engineering* (pp. 425-439). Springer, Berlin, Heidelberg. No. 41, Turku, Finland (pp. 57 – 64).

Nuldén, U., & Scheepers, H. (2000). Understanding and learning about project failure and escalation: Simulation in action. In *Second Nordic Workshop on Computer Supported Collaborative Learning and Knowledge Empowerment* (p. 63).

Nussbaum, J. (1989). Classroom conceptual change: philosophical perspectives. *International Journal of Science Education*, *11*(5), 530-540.

Oh, E. (2002, May). Teaching software engineering through simulation. In *Proceedings of the 2002 International Conference on Software Engineering Doctoral Symposium*.

Olivé, A. (2000). An introduction to conceptual modeling of information systems. *Advanced database technology and design. Artech House*, 25-57.

Pahl, G., & Beitz, W. (2013). *Engineering design: a systematic approach*. Springer Science & Business Media.

Pan, J., Liddicoat, A., Harris, J. G., & Shepherd, L. (2010). Assessing Curriculum Improvement Through Senior Projects.

Pan, R., Kuo, S. P., & Strobel, J. (2010). Novice students' difficulties and remedies with the conceptualization phase of design. In *American Society for Engineering Education*. American Society for Engineering Education.

Pan, R., Shih-Ping, K. and Johannes, S. 2010. Novice students' difficulties and remedies with the conceptualization phase of design. *American Society for Engineering Education*. (2010).

Parker, S. K., & Axtell, C. M. (2001). Seeing another viewpoint: Antecedents and outcomes of employee perspective taking. *Academy of Management Journal*, *44*(6), 1085-1100.

Peneul, W., Fishman, B., Chen, B., & Sabelli, N. (2011). Organizing research and development at the intersection of learning, implementing, and design. *Educational Researcher*, *40*(7), 331-337.

Petre, M. (2009, August). Insights from expert software design practice. In *Proceedings of the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering* (pp. 233-242).

Petre, M., van der Hoek, A., & Baker, A. Editorial in Design Studies. *Volume*, *31*, 533-544.

Petre, M. (2013, May). UML in practice. *In 2013 35th international conference on software engineering (icse)* (pp. 722-731). IEEE.

Pfahl, D., Koval, N., & Ruhe, G. (2001, April). An experiment for evaluating the effectiveness of using a system dynamics simulation model in software project management education. In *Proceedings Seventh International Software Metrics Symposium* (pp. 97-109). IEEE.

Posner, G. J., Strike, K. A., Hewson, P. W., & Gertzog, W. A. (1982). Accommodation of a scientific conception: Toward a theory of conceptual change. *Science education*, *66*(2), 211-227.

Pourmohamadi, M., & Gero, J. S. (2011). LINKOgrapher: An analysis tool to study design protocols based on FBS coding scheme. In *DS 68-2: Proceedings of the 18th International Conference on Engineering Design (ICED 11), Impacting Society through Engineering Design, Vol. 2: Design Theory and Research Methodology, Lyngby/Copenhagen, Denmark, 15.-19.08. 2011*.

Pressman, R. S. (2005). *Software engineering: a practitioner's approach*. Palgrave macmillan.

Qian, Y., & Lehman, J. (2017). Students' misconceptions and other difficulties in introductory programming: A literature review. *ACM Transactions on Computing Education (TOCE)*, *18*(1), 1-24.

Quintana, C., Krajcik, J., Soloway, E., & Norris, C. (2002). A framework for understanding the development of educational software. In *The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications* (pp. 823-834).

Quintana, C., Reiser, B. J., Davis, E. A., Krajcik, J., Fretz, E., Duncan, R. G., ... & Soloway, E. (2004). A scaffolding design framework for software to support science inquiry. *The journal of the learning sciences*, *13*(3), 337-386.
Quintana, C., Reiser, B. J., Davis, E. A., Krajcik, J., Fretz, E., Duncan, R. G., ... & Soloway, E. (2004). A scaffolding design framework for software to support science inquiry. *The journal of the learning sciences*, *13*(3), 337-386.

Richey, R. C., & Klein, J. D. (2014). Design and development research. In *Handbook of research on educational communications and technology* (pp. 141-150). Springer, New York, NY.

Ritschard, G., Bürgin, R., & Studer, M. (2013). Exploratory mining of life event histories. *Contemporary issues in exploratory data mining in the behavioral sciences*, 221-253.

Ritschard, G., Bürgin, R., and Studer, M. (2014), "Exploratory Mining of Life Event Histories", In McArdle, J.J. & Ritschard, G. (eds) *Contemporary Issues in Exploratory Data Mining in the Behavioral Sciences*. Series: Quantitative Methodology, pp. 221-253. New York: Routledge.

Ritschard, G., Bürgin, R., and Studer, M. (2014), "Exploratory Mining of Life Event Histories", In McArdle, J.J. & Ritschard, G. (eds) *Contemporary Issues in Exploratory Data Mining in the Behavioral Sciences*. Series: Quantitative Methodology, pp. 221-253. New York: Routledge.

Rittel, H. W., & Webber, M. M. (1973). Dilemmas in a general theory of planning. *Policy sciences*, *4*(2), 155-169.

Rosenman, M. A., Gero, J. S., & Oxman, R. E. (1991). What's in a case: the use of case bases, knowledge bases, and databases in design.

Salomon, G., Perkins, D. N., & Globerson, T. (1991). Partners in cognition: Extending human intelligence with intelligent technologies. *Educational researcher*, *20*(3), 2-9.

Schilling Jr, W. W., & Sebern, M. J. (2013). Teaching software engineering: An active learning approach. *The ASEE Computers in Education (CoED) Journal*, *4*(1), 13.

Schön, D. (1938). The reflective practitioner. *New York*, *1083*.

Schunk, D. H. (1998). Teaching Elementary Students to Self-Regulate Practice. *Self-regulated learning: From teaching to self-reflective practice*, 137.

Schwartz, D. L. (1995). The emergence of abstract representations in dyad problem solving. *The journal of the learning sciences*, *4*(3), 321-354.

Seitamaa-Hakkarainen, P., & Hakkarainen, K. (2001). Composition and construction in experts' and novices' weaving design. *Design Studies*, *22*(1), 47-66.

Shavelson, R. J., Phillips, D. C., Towne, L., & Feuer, M. J. (2003). On the science of education design studies. *Educational researcher*, *32*(1), 25-28.

Shin, J., Rusakov, A., & Meyer, B. (2014). Teaching Software Engineering through Robotics. *arXiv preprint arXiv:1406.4458*.

Sien, V. Y. (2011). An investigation of difficulties experienced by students developing unified modelling language (UML) class and sequence diagrams. *Computer Science Education*, *21*(4), 317-342.

Singer, S. R., Nielsen, N. R., & Schweingruber, H. A. (2012). Discipline-based education research. *Washington, DC: The National Academies*.

Soloway, E. (1986). Learning to program= learning to construct mechanisms and explanations. *Communications of the ACM*, *29*(9), 850-858.

Sonnentag, S. (1998). Expertise in professional software design: A process study. *Journal of applied psychology*, *83*(5), 703.

Sonnentag, S., Niessen, C., & Volmer, J. (2006). Expertise in software design.

Ssemugabi, S., & De Villiers, M. R. (2010). Effectiveness of heuristic evaluation in usability evaluation of e-learning applications in higher education. *South African computer journal*, *2010*(45), 26-39.

Stevens, S. M. (1989). Intelligent interactive video simulation of a code inspection. *Communications of the ACM*, *32*(7), 832-843.

Suwa, M., Gero, J., & Purcell, T. (2000). Unexpected discoveries and S-invention of design requirements: important vehicles for a design process. *Design studies*, *21*(6), 539-567.

Tang, A., Aleti, A., Burge, J., & van Vliet, H. (2010). What makes software design effective?. *Design Studies*, *31*(6), 614-640.

Tang, H. H. (2002). *Exploring the roles of sketches and knowledge in the design process*. Department of Architectural and Design Science, Faculty of Architecture, University of Sydney.

Teasley, S. D. (2017). Student facing dashboards: One size fits all?. *Technology, Knowledge and Learning*, *22*(3), 377-384.

Teel, S., Schweitzer, D., & Fulton, S. (2012). Teaching undergraduate software engineering using open source development tools. *Issues in Informing Science and Information Technology*, *9*, 063-073.

Thomas, L., Eckerdal, A., McCartney, R., Moström, J. E., Sanders, K., & Zander, C. (2014, July). Graduating students' designs: through a phenomenographic lens. In *Proceedings of the tenth annual conference on International computing education research* (pp. 91-98).

Tremblay, G. (2001). Software Design. *SWEBOK*, 35.

Trends in Concept Design, PTC study, July 2011 – http://www.ptc.com/go/concept-design

Umeda, Y., Ishii, M., Yoshioka, M., Shimomura, Y., & Tomiyama, T. (1996). Supporting conceptual design based on the function-behavior-state modeler. *Ai Edam*, *10*(4), 275-288.

Van den Akker, J., Gravemeijer, K., McKenney, S., & Nieveen, N. (Eds.). (2006). *Educational design research*. Routledge.

Viswanathan, V. K., & Linsey, J. S. (2013). Design fixation and its mitigation: a study on the role of expertise. *Journal of Mechanical Design*, *135*(5).

Volet, S. E., & Järvelä, S. E. (2001). *Motivation in learning contexts: Theoretical advances and methodological implications*. Pergamon Press.

Von Aufschnaiter, C., & Rogge, C. (2015). Conceptual change in learning. *Encyclopedia of science education*, 209-218.

Vosniadou, S. (2007). Conceptual change and education. *Human development*, *50*(1), 47-54.

Vosniadou, S. (2019, April). The Development of Students' Understanding of Science. In *Frontiers in Education* (Vol. 4, p. 32). Frontiers.

Vosniadou, S. (2019, April). The Development of Students' Understanding of Science. In *Frontiers in Education* (Vol. 4, p. 32). Frontiers.

Vosniadou, S., Chi, M. T., Ohlsson, S., Cosejo, D. D., Brown, D. E., & Nerserssian, N. J. (2011). New Approaches to the Problem of Conceptual Change in the Learning of Science and Math. In *Proceedings of the Annual Meeting of the Cognitive Science Society* (Vol. 33, No. 33).

Wang, F., & Hannafin, M. J. (2005). Design-based research and technology-enhanced learning environments. *Educational technology research and development*, *53*(4), 5-23.

White, R., & Gunstone, R. F. (2008). The conceptual change approach and the teaching of science. In International handbook of research on conceptual change (pp. 619-628). Routledge.

Wilke, W. (1999). *Knowledge management for intelligent sales support in electronic commerce*. IOS Press.

Winne, P. H., & Hadwin, A. F. (2008). The weave of motivation and self-regulated learning In: Schunk DH, Zimmerman BJ, editors. Motivation and self-regulated learning: Theory, research, and application.

Wohlin, C., & Regnell, B. (1999, March). Achieving industrial relevance in software engineering education. In *Proceedings 12th Conference on Software Engineering Education and Training (Cat. No. PR00131)* (pp. 16-25). IEEE.

Wood, K. L., Jensen, D., Bezdek, J., & Otto, K. N. (2001). Reverse engineering and redesign: courses to incrementally and systematically teach design. *Journal of Engineering Education*, *90*(3), 363-374.

Wood, K. L., Jensen, D., Bezdek, J., & Otto, K. N. (2001). Reverse engineering and redesign: courses to incrementally and systematically teach design. *Journal of Engineering Education*, *90*(3), 363-374.

Wynn, D. C., & Clarkson, P. J. (2018). Process models in design and development. *Research in Engineering Design*, *29*(2), 161-202.

Xun, G. E., & Land, S. M. (2004). A conceptual framework for scaffolding III-structured problem-solving processes using question prompts and peer interactions. *Educational technology research and development*, *52*(2), 5-22.

Yeh, M. K. C. (2018). Examining novice programmers' software design strategies through verbal protocol analysis. *International Journal of Engineering Education*, *34*(2), 458-470.

Yilmaz, S., Daly, S. R., Seifert, C. M., & Gonzalez, R. (2015). How do designers generate new ideas? Design heuristics across two disciplines. *Design Science*, *1*.

Zhang, J. (1997). The nature of external representations in problem solving. *Cognitive science*, *21*(2), 179-217.

Zou, J., & Du, Q. (2013). A Functional Reasoning Cube Model for Conceptual Design of Mechatronic Systems. *Strojniski Vestnik/Journal of Mechanical Engineering*, *59*(5).

List of Publications

In Journals

1. **Lakshmi, T.G.** & Iyer, S (2020). Applying the Function-Behaviour-Structure (FBS) design lens to explore novices' approach in software conceptual design. Computer Science Education. Under review
2. **Lakshmi, T.G.** & Iyer, S (2020). Fostering conceptual change in software design. IEEE Transactions on Education. Under review.

In Peer-reviewed conference proceedings

1. Deepti Reddy, Kavya Alse, **Lakshmi T.G.**, Prajish Prasad, and Sridhar Iyer. (2021). Learning Environments for Fostering Disciplinary Practices in CS Undergraduates. In Proceedings of the 52nd ACM Technical Symposium on Computer Science Education (SIGCSE '21). Association for Computing Machinery, New York, NY, USA, 1287. DOI: https://doi.org/10.1145/3408877.3439677

2. **Lakshmi, T.G.**, & Iyer, S. (2020, Jul). Teaching-learning of software conceptual design via function-behaviour-structure framework.13th Workshop on cooperative and human aspects of software engineering:(CHASE 2020). In 2020 42nd International Conference on Software Engineering (ICSE). IEEE.

3. **Lakshmi, T. G.** & Herold, P. C. (2019, December). Heuristic Evaluation and User Experience Redesign of 'Think & Link' Learning Environment–A Case Study. In 2019 IEEE Tenth International Conference on Technology for Education (T4E) (pp. 166-169). IEEE.

4. **Lakshmi, T. G.** (2018, August). Developing Students' Conceptual Design Skills for Software Engineering. In Proceedings of the 2018 ACM Conference on International Computing Education Research (pp. 278-279).

5. **Lakshmi, T. G.** & Iyer, S. (2018). Exploring Novice Approach to Conceptual Design of Software. In Kay, J. and Luckin, R. (Eds.) Rethinking Learning in the Digital Age: Making the Learning Sciences Count, 13th International Conference of the Learning Sciences (ICLS) 2018, Volume 3. London, UK: International Society of the Learning Sciences

6. **Lakshmi, T. G.**, Prasad, P., & Iyer, S. (2017, July). A System for Developing Operationalization Skills through Problem Decomposition. *In Advanced Learning Technologies (ICALT), 2017 IEEE 17th International Conference on* (pp. 427-429). IEEE.
7. A. Raina, **T. G. Lakshmi** and S. Murthy. (2017). "CoMBaT: Wearable Technology Based Training System for Novice Badminton Players," *2017 IEEE 17th International Conference on Advanced Learning Technologies (ICALT)*, Timisoara, 2017, pp. 153-157, doi: 10.1109/ICALT.2017.96.

8. **Lakshmi, T. G.**, Narayana, S., Penugonda, H., Vaidya, D., Poonia, V., Ganguly, S., & Murthy, S. (2017). Pivoteeing: a flipped approach in a postgraduate solid state devices course. *In Proceedings of the 25th International Conference on Computers in Education*.

9. **Lakshmi, T. G.**, Narayana, S., Prasad, P., Murthy, S., & Chandrasekharan, S. (2016). Geometry-via-Gestures: Design of a gesture based application to teach 3D Geometry. *In Proceedings of the 24th international conference on computers in education* (pp. 180-189). Mumbai, India: Asia-Pacific Society for Computers in Education.

10. S. Narayana, P. Prasad, **T. G. Lakshmi** and S. Murthy. (2016)."Geometry via Gestures: Learning 3D Geometry Using Gestures," *2016 IEEE Eighth International Conference on Technology for Education (T4E)*, Mumbai, 2016, pp. 26-33, doi: 10.1109/T4E.2016.014.

11. K. Alse, **L. Ganesh**, P. Prasad, M. Chang and S. Iyer. (2016). "Assessing Students' Conceptual Knowledge of Computer Networks in Open Wonderland," *2016 IEEE 16th International Conference on Advanced*

Learning Technologies (ICALT), Austin, TX, 2016, pp. 513-517, doi: 10.1109/ICALT.2016.22.

12. **Ganesh, L.** (2014, December). Board Game as a Tool to Teach Software Engineering Concept--Technical Debt. *In Technology for Education (T4E), 2014 IEEE Sixth International Conference on* (pp. 44-47). IEEE.

13. **Ganesh, L.** (2013, December). The effect of comic strips as a supplementary material to teach computer networks. *In Technology for Education (T4E), 2013 IEEE Fifth International Conference on* (pp. 184-191). IEEE.

Acknowledgements

In this section I thank all people in the Ph.D. journey as well as people who lead me up to the journey. I begin this section by acknowledging my privileges, which I have been bestowed with. These privileges start with being born into a family that believes education as an important aspect of life. My mother (Uma Ganesh) always pushed me to find learning opportunities in all endeavours. My father (T V Ganesh) was the one who taught me work ethics by setting an example. He rose into the higher echelons of a public sector bank, by sheer hard work and grit. The values that my parents instilled in me, I would always be grateful to them.

This thesis is the result of the guidance of my supervisor - Prof. Sridhar Iyer. When I look back, it all started with his questions. He let me meander through all the possible pathways, but he always knew when and where to align the research focus. I thank him, for giving me the freedom in my research work and instilling confidence in me. I can only hope to live up to his expectations always from here onwards.

My chief-mentor during Ph.D., who has turned into a mentor for life, Prof. Sahana Murthy was always generous with her time for me. I have shared my research work as well as my self-doubts and always found a patient listener and advisor in her. She took me under her wings and instilled self-belief in all my strengths. She along with my other RPC member, Prof. Sasi Kumar, have provided the critical comments at the right moments. This has strengthened the thesis and helped me grow as a researcher. I thank both of them.

My teachers during the Ph.D. journey have made me the researcher that I am today. Prof. Sanjay - his deep discussions and take on cognition/learning have provided me with perspectives to think. Prof. Maiga –his childlike enthusiasm in building and creating learning environments has inspired the software developer in me. Prof. Chandan – his ability to remain comfortable amidst ambiguity has motivated me to find calmness during uncertainty. Prof Ram – his ability to be gentle while critical has made me realize the importance of being kind while giving feedback. I thank all the teachers for imparting the knowledge and making me realize these qualities.

These six years of research journey that I undertook, my peers in this learning process were Prajish and Soumya, my batch mates. Prajish was extremely supportive

and stood by me at various points of my thesis journey. As we plunged into our individual thesis, his constant checks on me, nudged me to keep working towards the deadlines. Soumya has been my friendly companion. Her love for fun and laughter has made me tide over some of the rocky roads in this journey. I look forward to engaging in many research projects in the near future with both of them. To my batch-mates (batchies), I am thankful for their intellectual engagement and their constant support throughout.

This thesis would not have completed with the design and development support that I received from my student interns (Keval and Swapnil), my colleagues Rahul, Herold and developers Mangesh and Varun. They shared my vision and helped me realize the design of 'think & link'. I thank all of them. The logistics support provided by Pallavi, Vidya, Seema and Prakash (Sr & Jr) is immeasurable. They are the silent supporters of the EdTech department and research scholars (RS), specifically.

The seniors in the department have been a constant source of inspiration. These amazing women inspired me and made me realize that this journey seems possible, especially during difficult times. I thank Madhuri mam – for sharing her journey and encouraging me to pursue EdTech Ph.D.; Gargi mam – for constant encouragement and reassuring words; Mrinal mam – for showing that pursuit of excellence does not mean giving up other aspects of your life; Rekha mam – for finding humour in the process and sharing it with everyone in the lab; Deepti mam - for managing the most stressful times with calm and composure. I thank all of them.

I also thank my seniors, Rwito and Aditi, for their timely and critical reviews of my work. Their comments have shaped many of my papers and ideas. I thank Anurag - my immediate senior, who supported me with all the critical information and guided me through the process. I thank Balraj for his dank humour, Veenita for her trusted advice and JK for all the advice on everything under the sun. Ashu and Rumana, thank you for the kind words whenever I needed them. To all the other RS in the lab, I would like to thank you for your companionship and *joie de vivre*.

My best friend Sush is my rock of Gibraltar. I thank her for being there always for me. For calling on my lapses when I needed them and more importantly not calling on them when I don't. Sush, I can't thank you but I need to. To my friends Jigna, Parth, Animesh, Sushmita, Nisha, Karthik, Harini, Venky, thank you for making me feel part of the gang.

My pranams to Kamalamma, my grandmother, who couldn't see me at the finishing line of this Ph.D. journey. She was the matriarch of my family. I am sure she would be smiling from above. I fondly miss the presence of my best friend, Haripriya. My life would not have been the same without these people. I thank them for their presence in my life.

To my family, starting with my in-laws (Raji amma, Ramani appa) whom I dedicate this thesis. I can't thank them enough for their selfless support. My sister (Kamala) and brother-in-law (Raj) always kept me in their prayers and sent me the positive energy when I needed it the most. I thank my sister-in-law (Sangeeta) and brother (Sai) for believing in me. All my cousins, nieces and nephews (kamalamma korangnaes) for their love and laughter, thank you.

To my children- Ashwath and Aparna. Ashwath I thank you for being the listener, critique and friend. Your euphemisms crack me up, but they are full of worldly wisdom. Ashwath: thank you for being you. Aparna- she has been my constant companion as I began my research journey. In my first year of Ph.D., I was carrying Aparna. As I see her grow as an individual, I also saw my thesis grow. Aparna: thank you for being with me. In both my kids I had wonderful cheerleaders who would celebrate every milestone however small they might be.

Last but not the least, Santosh – my partner in life. Thank you Santosh, for making me realize I have wings and also for being the wind beneath my wings. I look forward to the rest of the journey with you as always by my side.