Teaching-Learning of Expand-Reduce skills in the context of Software Design

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

by

Patil Deepti Reddy

(Roll No. 134388002)

Supervisors:

Prof. Sridhar Iyer

Prof. Sasikumar M.

Inter-disciplinary Programme in Educational Technology

INDIAN INSTITUTE OF TECHNOLOGY BOMBAY

2019

Dedication Sheet

Dedicated to

my Family members

and

the Almighty

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.

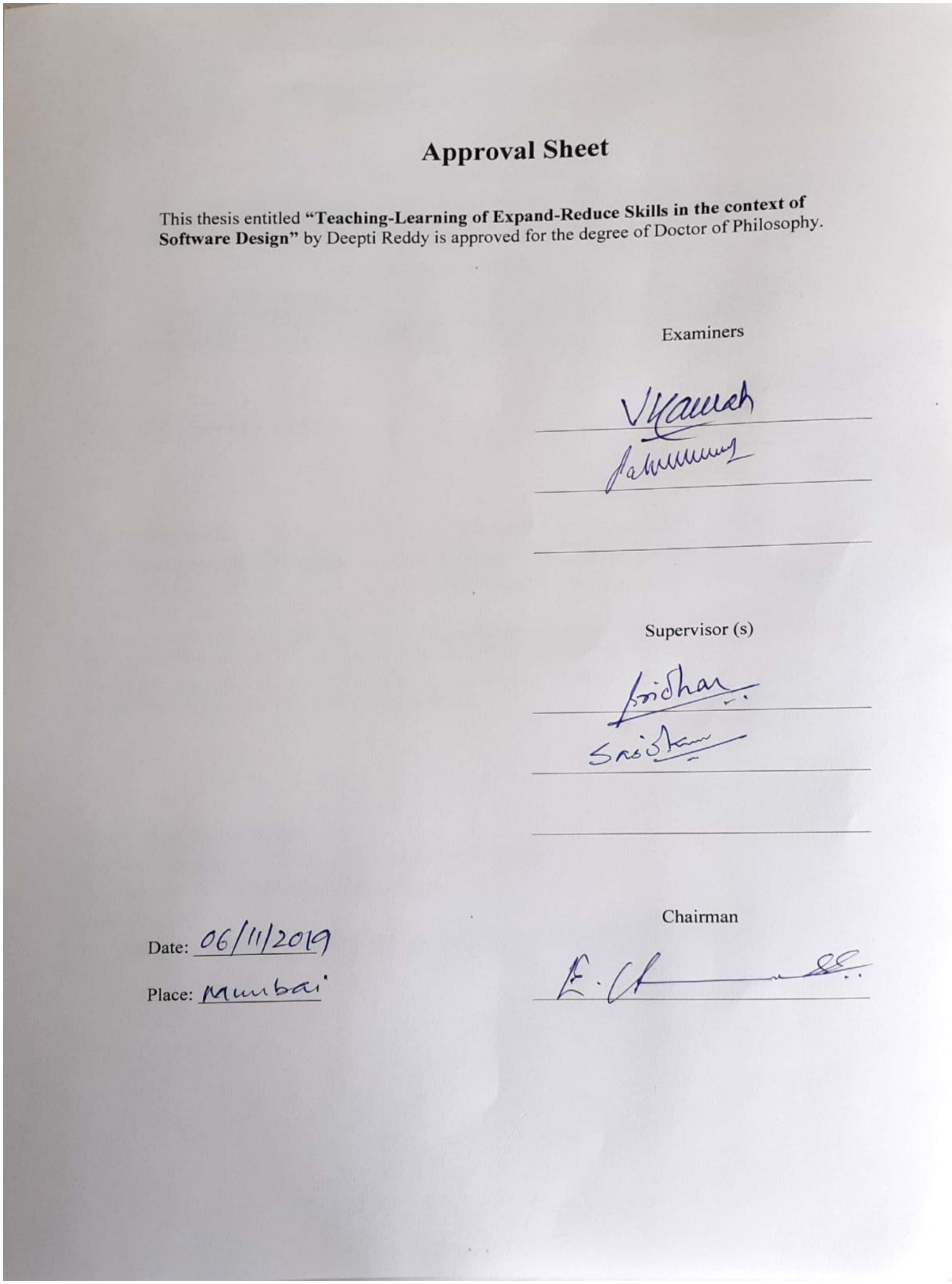
(Signature)

(Name of the student)

(Roll No.)

Date: 06/11/2019

Thesis Approval Sheet



Approval Sheet

This thesis entitled **"Teaching-Learning of Expand-Reduce Skills in the context of Software Design"** by Deepti Reddy is approved for the degree of Doctor of Philosophy.

Examiners

Supervisor (s)

Chairman

Date: 06/11/2019

Place: Mumbai

Abstract

Software design problems are ill-structured, in which the problem space and solution space are not well-defined. In problem space, the requirements are not well defined, and a designer has to formulate incomplete requirements into specific data models, main software functions, and sub-functions. In solution space, there may be multiple solution paths, alternative design options, and the criteria to evaluate and select an optimal solution may not be clearly stated. The issues in software design are that the quality of the design is heavily dependent on the expertise and experience of the designer. Novices tend to think at the programming level and reduce early to the solution design, which affects the quality of the design in many ways.

Research studies have shown that systematically expanding and reducing the problem-solution spaces improves the quality of the solution design. In this thesis, we refer to the ability to expand the problem-solution space and eventually reduce towards a solution, as expand-reduce (ER) skills. In problem space, the ER skill is the ability to explore the problem as a whole and reduce the problem into sub-problems based on the goal to be achieved. In solution space, the ER skill is the ability to generate alternative solutions and reduce to one solution by evaluating and selecting based on selection criteria.

The existing research studies have established the importance of ER skills in solving ill-structured problems, but not much research is done in the direction of teaching-learning of ER skills to novices. To address this research gap, the broad research objective of this thesis is: "teaching-learning of ER skills to novices using technology-enhanced learning environment." Our research is scoped to teaching-learning of ER skills to undergraduate computer engineering students in the context of solving software design problems using appropriate data structures and algorithms.

We have used design-based research (DBR) methodology to design the technology-enhanced learning environment, named Fathom, for teaching-learning of ER skills. We have completed three DBR cycles to design-evaluate-redesign our intervention based on the feedback from learners and practitioners. Four research studies were carried out, which included two exploratory studies and two pretest-posttest experimental studies (N_{total}=200). Both quantitative

and qualitative data were collected and analyzed. Quantitative data was used to measure the learning of ER skills by comparing the learner's performance in pretest, intervention, and posttest. Qualitative data in the form of log data, screen capture, and focus group interviews were used to analyze the behaviors exhibited by learners while interacting with the learning environment and student perceptions about learning ER skills.

The results showed that the Fathom was effective in learning ER skills for novices. The major contributions of this thesis are: providing insights about the cognitive biases of novice towards applying ER skills, identification of effective cognitive and metacognitive scaffolds in technology-enhanced learning environment, identification of ER cognitive tools and, the development of learning environment for learning ER skills, in the context of software design.

Keywords: Expand-reduce skills, technology-enhanced learning environments, software design, ER cognitive tools, Cognitive and metacognitive scaffolds, novices.

Table of Contents

List of Figures

List of tables

Abbreviation Notation and Nomenclature

ABET	Accreditation Board for Engineering and Technology
UML	Unified Modeling Language
DFDs	Data Flow Diagrams
ER	Expand-Reduce
TELE	Technology-Enhanced Learning Environment
ZPD	Zone of Proximal Development
DBR	Design-Based Research
HTML	Hyper Text Markup Language
PHP	Hypertext Preprocessor
SODA	Strategic Options Development and Anlaysis
MACRAME	Multiple Actor RepresentAtion ModElling
AHP	Analytic Hierarchy Process
DS	Data Structure
DSA	Data Structures and Algorithms
TELoTS	Technology-Enhanced Learning of Thinking Skills
PBLE	Problem-Based Learning Environment

Chapter 1

Introduction

1.1 Background and motivation

ABET is a non-profit organization that accredits engineering programs based on the quality standards of the profession for which that program prepares students. According to ABET guidelines, one of the intended outcomes of engineering education is *an ability to design systems, components, or processes meeting specified needs for broadly-defined engineering problems appropriate to the discipline* (ABET 2014). During engineering education, engineering students should acquire the ability to solve design problems.

The design problems are ill-structured and complex because they possess conflicting goals, multiple forms of problem representation, multiple solutions methods, non-engineering constraints, and unanticipated problems (Jonassen, 2006). In computer engineering discipline, software design is one of the important competencies required for a software engineer to sustain in the industry (Tang, 2008; 2010; Carman, 2007).

Software design problems are ill-structured, in which the problem space and solution space are not well-defined. Some of the characteristics of a software design problem in terms of the problem-solution spaces are shown in figure 1.1.

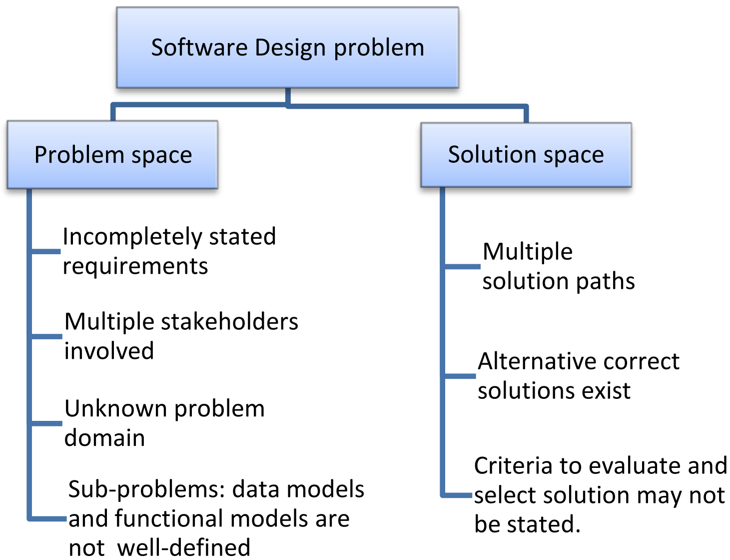


Figure 1.1. Software design problem characteristics.

In problem space, the requirements are broadly defined, and the designer has to elicit specific requirements from various stakeholder's perspectives (Pressman, 2010), the stakeholders involved may not be initially known as the domain in which the software is to be designed may be unfamiliar to the designer (Adelson, 1985; Tang, 2008), and designer has to formulate incomplete requirements into specific data models, main software functions and sub-functions (Guindon, 1990; Pressman, 2010). The solution space may have multiple solution paths and multiple alternative design options, and the criteria to evaluate and select an optimal solution may not be clearly stated (Jonassen, 2006).

Based on various expert studies in software design *(*Adelson, 1985; Guindon, 1990; Tang, 2008; 2010), we will illustrate the experts' approach to solving the software design problem- *"Design a library record keeping system."* The expert initially explores the problem space by simulating various scenarios and using various representations like diagrams, tables, list, etc., as shown in figure 1.2. This process helps them to derive new requirements and constraints which are not stated in the problem. In solution space, multiple solutions exist as the data can be represented in various ways, like flat data model (arrays, linked list), hierarchal data model (B-trees), network model (graphs), entity-relation model and object-relation model. Similarly, the operations can be implemented using various algorithms (linear search, binary search, etc.). The designer has to choose

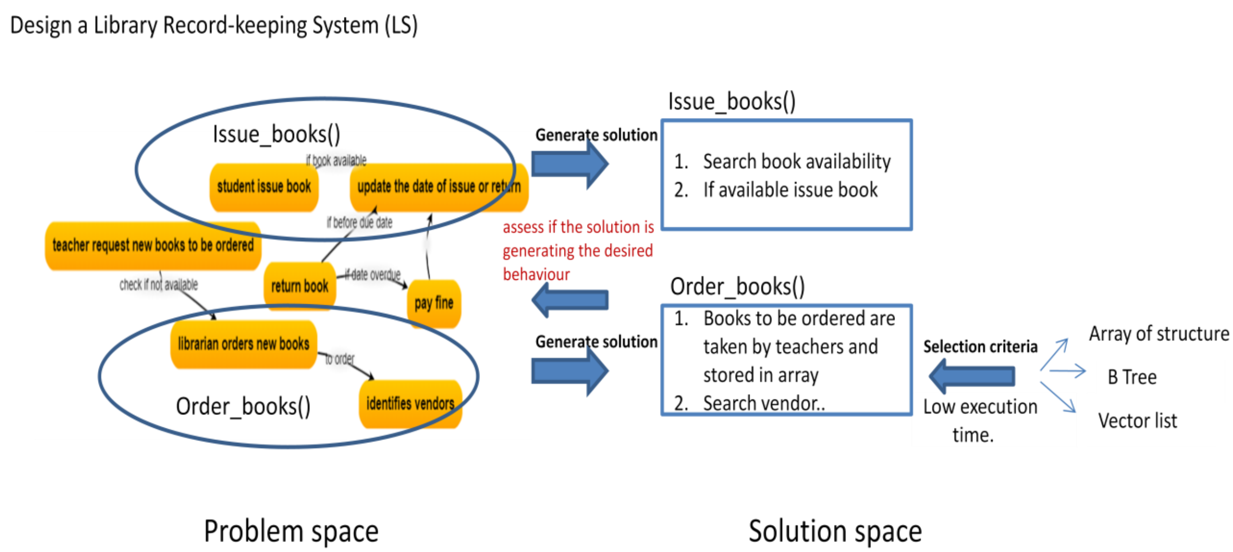


Figure 1.2. Illustration of the experts' problem-solving process

appropriate data structure representation and algorithms by critically evaluating the options based on the criteria (reliability, cost, execution time) and constraints (high reliability, low execution time) applicable to the given problem domain (Tang, 2010; Guindon, 1990). The experts reduce the solution space by doing trade-off analysis of solutions against the selection criteria to select the single solution. They go back and forth between problem-solution spaces, for example, after generating a solution, they test the solution against achieving the stated constraints and in the process go back to problem space to generate new requirements.

The novice approach to solving the problem *"Design a library record keeping system"*, will be that they tend to focus on one of the salient requirement in the library, *search a book*, and tend to write a program to achieve the requirement without thinking of other related requirements such as, *librarian issues a book*, *update book status*, etc.. They lack the ability to understand the problem as a whole from the perspective of various stakeholders involved, which leads to incomplete problem formulations (Adelson, 1985). In solution design, novices tend to select the design option without searching and evaluating from all available design options, which may affect the solution quality (Tang, 2010). They also have difficulty in identifying the selection criteria and constraints, which leads to the premature selection of design options.

One issue in software design is that the quality of the design is heavily dependent on the expertise and experience of the designer (Adelson, 1985; Tang, 2008). The expert designers are adept at applying various cognitive skills in problem-solution space and co-evolve both problem and solution space in an interleaved manner. In problem space, the designers who do initial problem exploration by simulating various problem scenarios can identify more problem statements and constraints. In solution space, the process of identifying alternative solutions and decision making based on selection criteria helps to design effective solutions (Tang, 2008). However, novices have difficulty in solving design problems as they think at programming level and reduce early to the solution design, which affects the quality of the design in many ways like defining the problem too narrowly, solving the wrong problem, failure to see how various sub-problems relate to one another, and anchor on the single solution without explicitly evaluating other alternative (Carmen, 2007; Ellspermann, 2007).

To address the issues faced by novice designers in solving software design problem, we extrapolated literature review at a broader level to identify the skills and techniques proposed in solving ill-structured problems across domains. Literature in ill-structured problem solving suggests that in problem space, expansionist-reductionist skills are useful to structure ill-defined problem into a well-defined problem. The expansionist approach involves understanding the system as a whole by identifying the parts and interrelationship between the parts, and reductionist approach involves decomposing the problem into subproblems which may be more solvable (Ackoff, 1979; Volkema, 1983). The solution space can be expanded by generating alternative solutions using various techniques like brainstorming, analogous thinking, attribute listing, etc. (Liu, 2004). Eventually, the solution space can be reduced by evaluating alternative solutions against the selection criteria techniques like trade-off analysis, analytic hierarchy process, and the decision matrix (Pugh, 1991).

In this thesis, we refer to the ability to explore the problem space or solution space and eventually reduce towards a solution, as expand-reduce(ER) skills. In problem space, the ER skill is defined as the ability to explore the problem as a whole and reduce the problem into sub-problems based on the goal to be achieved. In solution space, the ER skill is the ability to generate alternative solutions and reduce to one solution by evaluating and selecting based on selection criteria (Figure 1.3.).

The research studies in software design suggest the need to apply ER skills to improve the quality of the design, especially when the designer is not having prior experience in designing or if the problem domain is unknown. Much of current research is focused on eliciting the cognitive processes used by designers while designing (Adelson, 1984; Carmen, 2007; Tang, 2008; 2010), and less research done in the direction of teaching-learning of these skills to novices.

The engineering education curriculum is more focussed on teaching software design using formal design artifacts like UML diagrams and DFDs, than the cognitive skills or thinking that are used informally by the expert designers before creating formal design artifacts (Tang 2010). The focus is not given on teaching ER skills explicitly, and it is assumed that students learn these

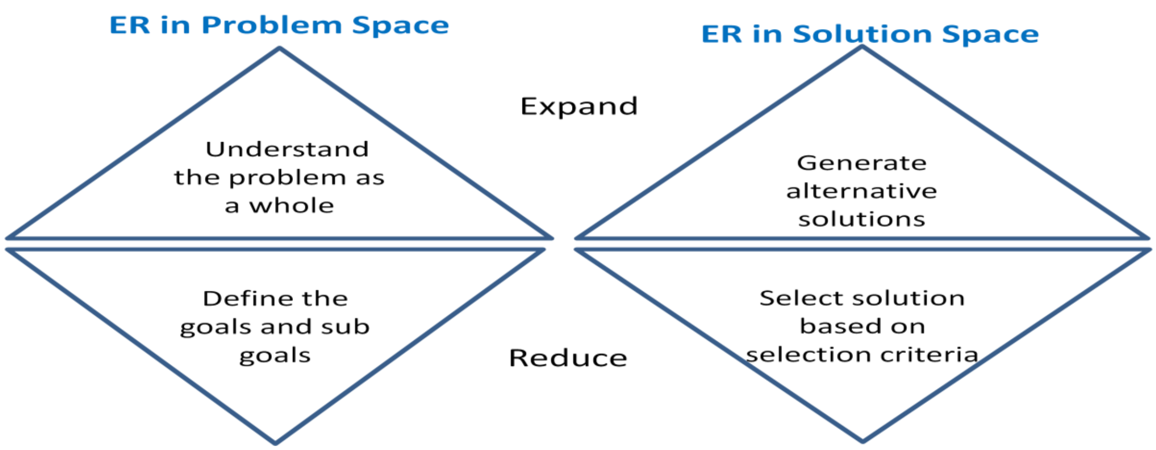


Figure 1.3. Expand-reduce skill in problem space and solution space

skills on their own while solving well-defined problems (Jonassen, 2006). In this thesis, we aim to address this gap by designing and developing a learning environment for the teaching-learning of ER skills to novices.

1.2 Research objective

In this sub-section, the overview of the research in ER skills in the context of solving a software design problem and the need for training novices in ER skills is summarized, followed by the formulation of our research goal and objectives-

1. The literature on problem-solving (Ackoff, 1979; Volkema, 1983; Liu, 2004; Ellspermann, 2007) has well established the need for ER skills in solving ill-structured problems to generate quality and quantity problem statements and solutions. The think-aloud studies done on experts to elicit the implicit cognitive skills reveal that they systematically expand-reduce in both problem and solution space, while novices lack these skills (Adelson, 1984; Tang, 2008). In figure 1.4, the topmost triangle represents this research area.
2. Research studies (Jonnasen, 2006; Dorst, 2001; Dym, 2005) suggests that students at undergraduate engineering level lack the skills to solve design problems effectively as they are not trained in the cognitive and metacognitive skills related to solving these problems.
3. The computer science and engineering (CSE) education is more focussed on using innovative teaching-learning techniques for teaching concepts (Jagadish, 2012; Janitor, 2010; Ramon, 2004; Reddy, 2015; Velázquez, 2012; Warendorf, 1997) and less focused on teaching complex problem solving skills to undergraduate engineering students (Cooperrider, 2008; Dym, 2005).

The current research work in ER skills and the gap in teaching-learning of the ER skills to a novice is summarized in figure 1.4. The literature review in the three areas: ER skills in solving design problem; the need of training these skills to novices; and current education scenario; suggests the need of teaching-learning of ER skills to novices and there are very few research studies done on teaching-learning of ER skills to novices.

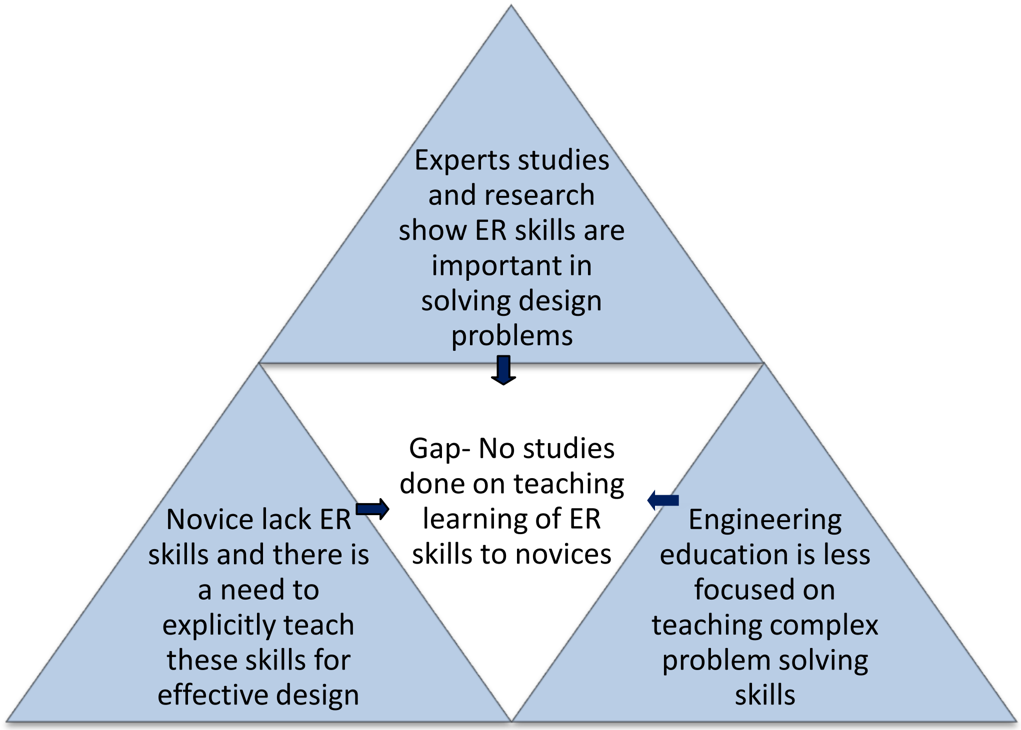


Figure 1.4. Research gap in teaching-learning of ER skills to the novice.

Based on the research gap identified, our broad and specific research goals are defined:

Broad research goal-

Design technology-enhanced learning environment (TELE) for effective teaching-learning of ER skills to novice learners.

Specific research goal-

Design technology-enhanced learning environment (TELE) for teaching-learning of ER skills to undergraduate computer engineering students in the context of software design problem-solving in data structures and algorithm course.

1.3 Research Methodology

To achieve the research goal, we may have to conduct research studies with novices to understand the difficulties and level of scaffolding needed in the learning of ER skills and design the intervention by adding, removing or modifying the design features based on the feedback from learners. The development of the TELE and its evaluation is a continuous and cyclic process.

Design based research (DBR) is a framework that helps the researcher to analyze the real world problems, integrate design principles with technological advances into solution development, test and refine learning environments as well as define new design principles (Reeves, 2006; Amiel, 2008). We used design-based research (DBR) methodology, as it is aligned with our requirements and iterate through design-evaluate-redesign phases.

The DBR cycle consists of four phases: problem analysis, solution development, evaluation, and reflection, as shown in figure 1.5. In the problem analysis phase, the problem is analyzed by the researchers and practitioners in collaboration to formulate the research questions. In the solution

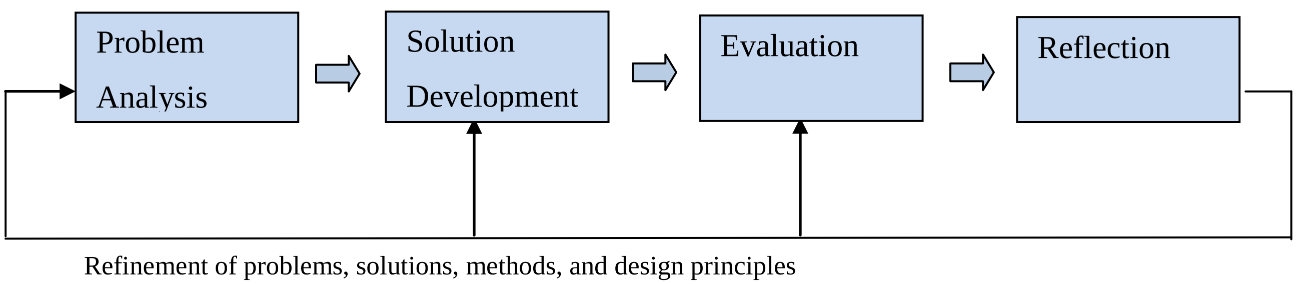


Figure 1.5. Stages in each cycle of DBR

development phase, the solution is developed based on existing design principles, and technological innovations suggested in the literature. In the evaluation phase, the solution is tested with practitioners in the real world, followed by a reflection phase to identify the limitations and accordingly redesign the intervention. The outcome of DBR is to produce design principles which can be used by other researchers.

In this thesis, we have completed three DBR cycles as summarized in figure 1.6. In first DBR cycle, we addressed two sub-goals:(1) assess the level of a novice in applying ER skills in the context of solving a software design problem, and (2) identify the effective scaffolding techniques to facilitate novice in doing of ER skills. These sub-goals were formulated as the current research has stated that novices lack the ability to solve ill-structured design problems (Jonnasen, 2006; Dorst, 2001; Dym, 2005), however there are very few studies done to identify the level of scaffolds to be provided in the learning environment for teaching-learning of ER skills to novices.

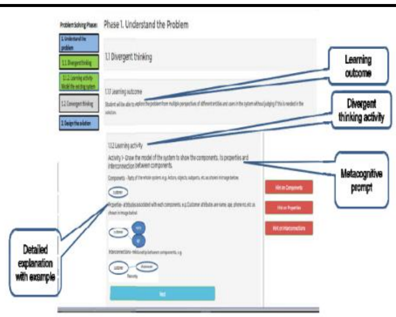
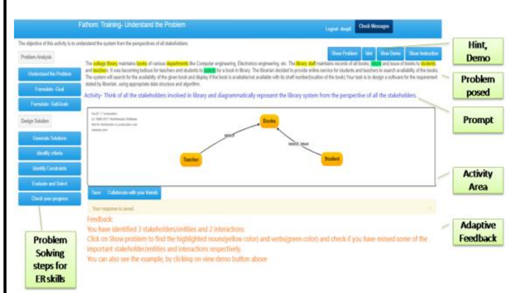
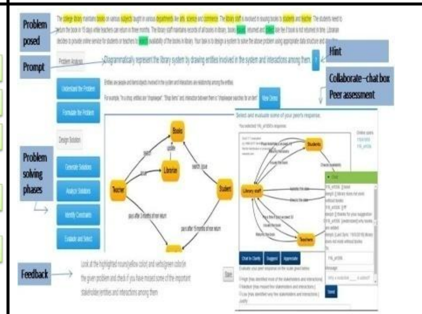
	DBR Cycle 1	DBR Cycle 2	DBR Cycle 3
Problem analysis	1. To assess the level of ER skills in novice. 2. Literature survey to identify the ER cognitive tools and learning techniques, **Study 1(N=40), exploratory study**	To redesign the TELE with an improved level of scaffolding and introduce reflection activities.	To revise the metacognitive activities for effective monitoring and controlling of ER skills.
Solution	Fathom-Ver1	Fathom-Ver2	Fathom-Ver3
			
Evaluation	**Study 2-** 49 engineering students, exploratory study **Results-** Students were having difficulty in doing ER	**Study 3-** 49 students, experimental pre-posttest. **Results-** Students could do ER skills but faced difficulty in evaluation of ER skills**.**	**Study 4-** 50 students, experimental pre-posttest. **Results-** Students could do ER skills and monitor ER skills.
Reflection	Prompts are not effective if learners do not have prior experience in ER cognitive skill. Improve the design by adding more scaffolds.	Novice learners are not able to self-evaluate to monitor and control ER skills. Improve metacognitive support.	The pedagogical support in terms of cognitive and metacognitive support and cognitive tools are effective in doing and learning of ER skills.

Figure 1.6. Design-based research cycles applied in this thesis

To achieve the first sub-goal we did study1 to answer Research Question (RQ):

RQ1- What is the level of a novice in applying ER skills in solving software design problem?

Based on the results of study1 which informed us about the level of the novice in applying ER skills, the first version of technology-enhanced learning environment (TELE), Fathom-Ver1, was designed and developed. The main design features of Fathom-Ver1 was to provide structured guidance towards applying ER skills in solving software design problem. Fathom-Ver1 was evaluated using study 2, and the research question asked was:

RQ2- What are the difficulties faced by novices in performing ER skills using Fathom-Ver1?

Based on the difficulties faced by the learners in performing ER skills while solving a software design problem, the design features of Fathom-Ver1 were improved in the next DBR cycle. In the second cycle of DBR, the focus was on learning and transfer of ER skills. The Fathom-Ver2 was redesigned and evaluated using study 3. The research question asked was:

RQ3- How effective are the design features of Fathom-Ver2 in learning ER skills?

We saw in study 3 that learners had difficulty in evaluating their skills and take appropriate action to improve. Thus, in the third cycle of DBR, the solution was redesigned to scaffold metacognitive skills of evaluating the ER skills. The Fathom-Ver3 was redesigned and evaluated. The research question asked for study 4 was RQ4:

RQ4- How effective are the design features of Fathom-Ver3 in learning ER skills at cognitive and metacognitive level?

The results showed that students were effectively doing ER skills during the intervention and were able to transfer these skills. They were able to monitor and take appropriate control action plan to improve the ER skills during the intervention.

1.4 Solution Overview

Our solution approach is grounded on the constructivist approach stated by Vygotsky's scaffolding theories on the zone of proximal development (ZPD) (Roblyer, 2012). The ZPD is the difference between novice and expert in performing certain skills, and the novice can learn the skills only when scaffolding is provided by a teacher or peer with a higher skill set.

The technology-enhanced learning environment (TELE) named Fathom is our solution for the teaching-learning of ER skills. Fathom's pedagogical features are designed to support novices in learning ER skills in the context of software design problem solving using appropriate data structures and algorithms. The design rationale is to scaffold novice towards the learning of ER skills using cognitive and metacognitive processes and tools used by experts.

The pedagogy of Fathom is designed based on the integration of technological affordances, and principles suggested in learning sciences literature on effective cognitive and metacognitive instructions for novices (Bannert, 2013; Xun, 2014). The overall pedagogical features of Fathom are:

1. The learners are systematically guided to do ER skills by providing support at three levels (figure 1.7): Understanding the skill: Systematic guidance with prompts, explanation of new terms, and worked example to illustrate how to use ER skills (green color boxes in figure 1.7).
2. Practice ER skills: Learners performs the ER skill in the context of solving a software design problem (blue color box in figure 1.7)
3. Engage in metacognitive activity: System generated formative feedback, collaborate with peers to evaluate, monitor, and reflect on the skill (Pink color boxes in figure 1.7).

The annotated screenshot in figure 1.8, shows the design features implemented in Fathom.

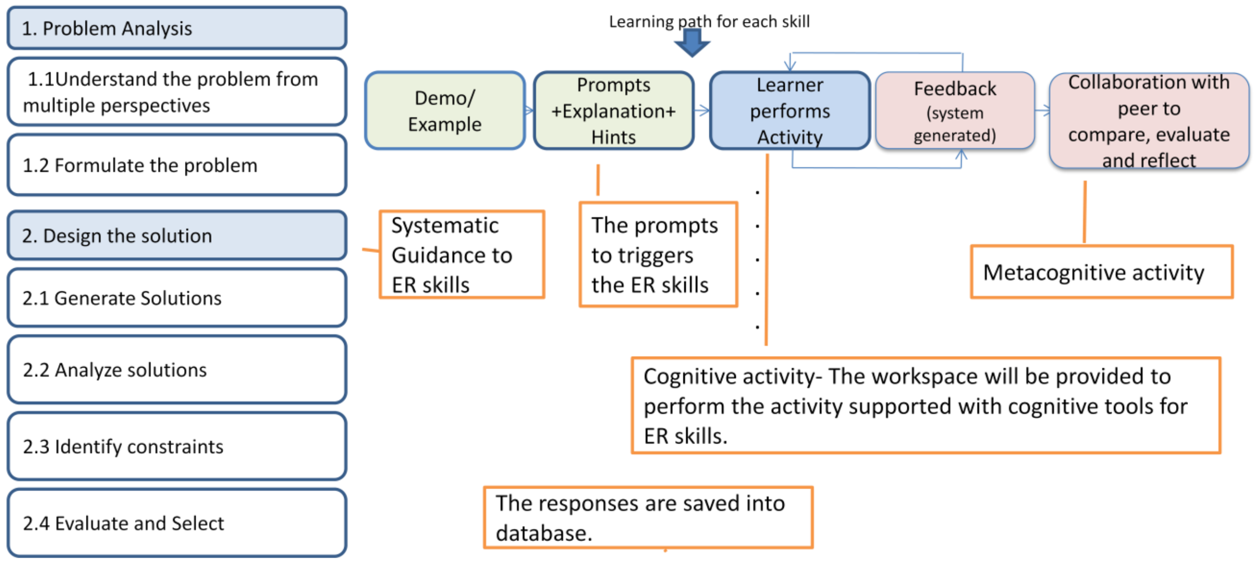


Figure 1.7. Solution design of Fathom

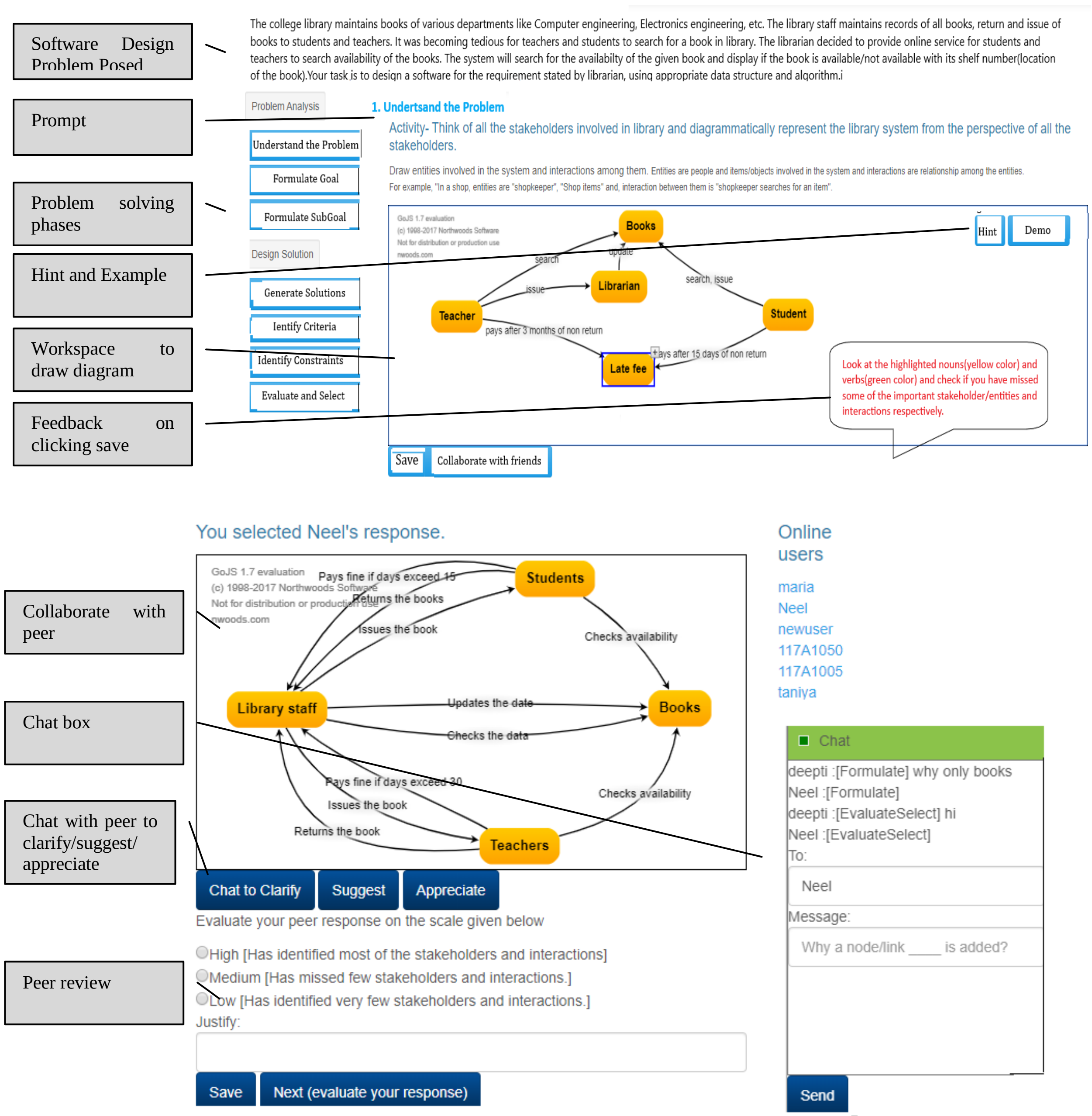


Figure 1.8. Screenshot of Fathom

We have identified the cognitive processes and tools related to ER skills from literature based on expert studies on software design (Tang, 2008;2010; Carman, 2007; Adelson, 1985), engineering design(Howard, 2008; Madhuri, 2015; Saaty 2008; Pugh, 1991), technical problem solving(James, 1995) and creative problem solving (Liu, 2004; Basadur, 2000). Fathom is equipped with cognitive tools to enable learners to systematically expand-reduce problem-solution spaces, as shown in figure 1.9.

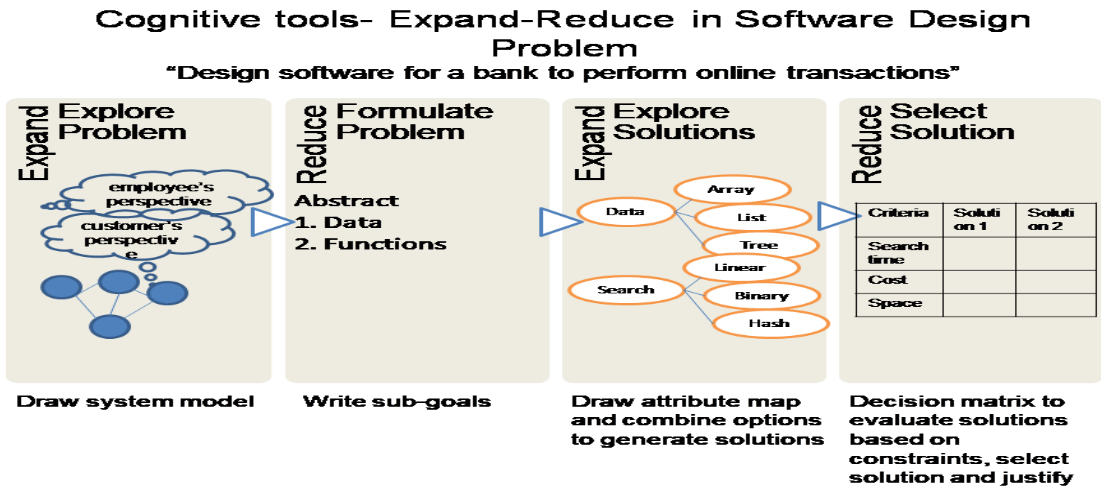


Figure 1.9. Cognitive tools for learning ER skills in software design

1.5 Scope of the thesis

We scope the research done in this thesis in terms of design of learning environment for ER skills, learner characteristics, problem and domain, context, and technology.

Learner group: The learners considered are undergraduate engineering students from second year onwards, with specialization in the field of computer engineering or information technology.

Problem and Domain: The problem selected for practicing ER skills are software design problems in data structures and algorithms course. A real-world context is posed in the problem and the software has to be designed using appropriate data structures and algorithms, for example, "Design software for college library to search availability of books, using appropriate data structure and algorithm."

Context: Fathom is designed for self-learning for novices, and the role of the instructor is not active. The instructor can facilitate students in using Fathom in the lab, either in between or at the end of the data structures course.

Technology: Fathom is developed and deployed using client-server architecture with front-end technologies: HTML5, JavaScript, JQuery, Bootstrap framework, and backend technologies: PHP and MySQL.

1.6 Contributions

This thesis makes contributions in terms:

1. Providing insights about the cognitive biases of novice towards applying ER skills in the context of solving a software design problem. Engineering educators and educational technology researchers can use these implications.
2. Identification of effective cognitive and metacognitive scaffolds in a technology-enhanced learning environment for doing and learning of ER skills by a novice. These findings are useful for engineering educators, instructional designers, educational technology, and learning science researchers.
3. Identification of ER cognitive tools for doing and learning of ER skills in the context of solving a software design problem. These are useful for engineering educators, instructional designers, and educational technology researchers.
4. Educational technology researchers can use the research studies with replicable research methods, data collection methods (quantitative and qualitative), assessment instruments (rubric to assess ER skills), and findings.
5. Development of a system, Fathom, that can be used by
    1. Engineering students to practice and learn ER skills,
    2. Engineering educators for teaching-learning of ER skills to their students.
    3. Educational technology researchers to conduct studies

1.7 Structure of the thesis

The thesis is organized as follows:

In chapter 2, we set the context by discussing the literature on software design problem solving and issues faced by designers, followed by discussion and definition of ER skills and the importance of these skills in problem-solving. Further, the research studies related to ER skills in software design are discussed to establish the research gap and our research goals.

In chapter 3, the overview of design-based research (DBR) methodology and its relevance to our research goals is discussed. The overview of three DBR cycle, corresponding research questions and studies are discussed.

In chapter 4, the first DBR cycle 1: problem analysis phase, is discussed. In problem analysis, the study 1, to understand the level of novices in applying ER skills is reported followed by the literature survey to identify ER cognitive tools and techniques proposed by researchers and experts in various areas of problem-solving. Based on the literature survey, the learning and design conjectures are identified for the design of the solution.

In chapter 5, the design of the initial version of Fathom-Ver1 is discussed, followed by the evaluation study to investigate the effectiveness of Fathom-Ver1 in performing learning activities designed to learn ER skills.

In chapter 6, the DBR cycle 2 phases are discussed. In problem analysis, the principles for effective scaffolding techniques for novices are identified. In solution design, the revised version of Fathom-Ver2 is discussed, followed by the evaluation study to investigate the effectiveness of Fathom-Ver2 in learning ER skills and the reflection phase in which the implications of current study and feedback for next cycle is established.

In chapter 7, the DBR cycle 3 is discussed. In problem analysis, the principles for effective metacognitive scaffolding techniques for novices are identified. In solution design, the revised version of Fathom-Ver3 is discussed, followed by the evaluation study to investigate the effectiveness of Fathom-Ver3 in learning ER skills and reflection phase.

In chapter 8, we summarize an overview of research, answers to RQs, claims, limitations, and generalization of our research to other domains. Finally, in chapter 9, we conclude with contributions of the thesis, future work, and final reflection.

Chapter 2

Literature Survey

This chapter provides an overview of ill-structured problem-solving literature that led us towards the characterization of expand-reduce(ER) skills followed by the difference in expert and novice approach in applying the ER skills, and the current state of engineering education. The overall literature review informs us towards formulating the research gap, research goals, and solution design.

The overall flow of literature review mapped to various areas is shown in figure 2.1.

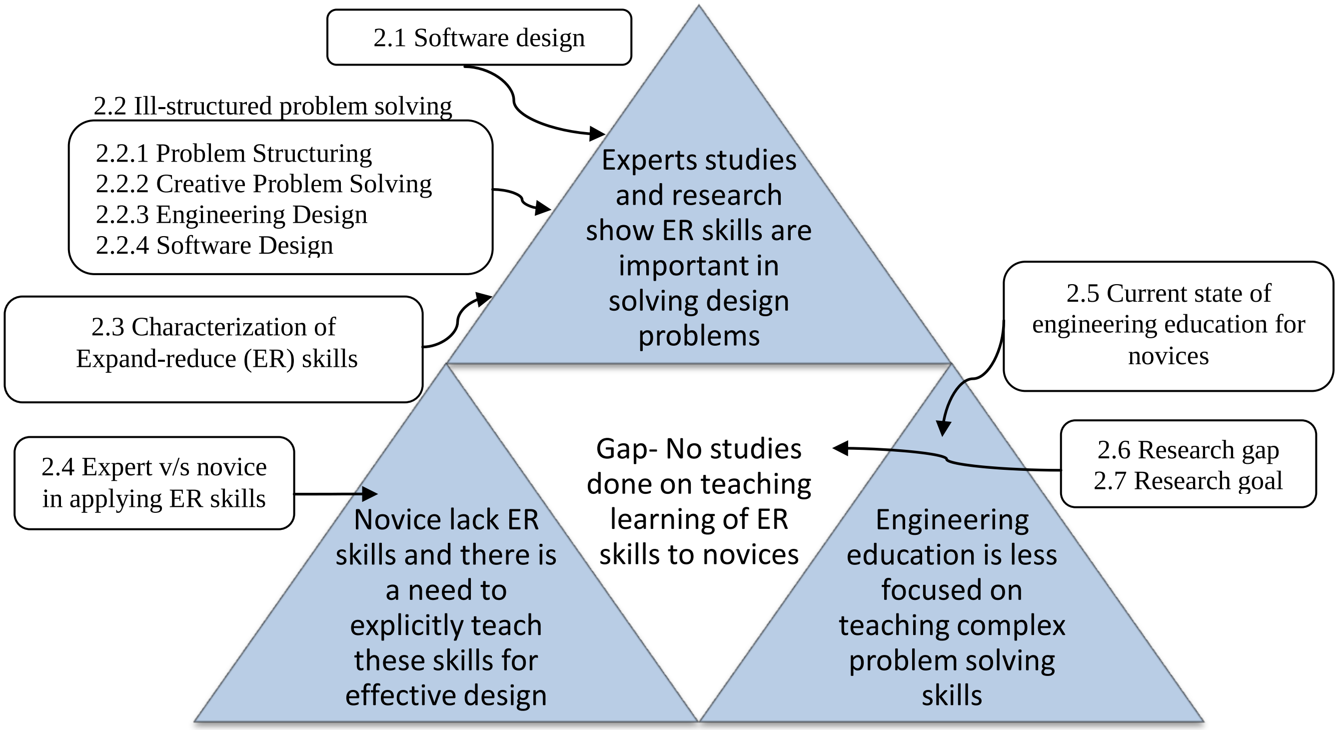


Figure 2.1. Overview of the literature review

2.1 Software design problems

Software design problems are ill-structured in nature as these problems may not have a well-defined problem and solution space (figure 2.2). In problem space, the stakeholders; goals; sub-goals; and constraints are not always well-defined or initially known to the designer. In solution space, such problems have alternative solutions; multiple solution paths; and multiple criteria for evaluating solutions (Jonassen, 2000).

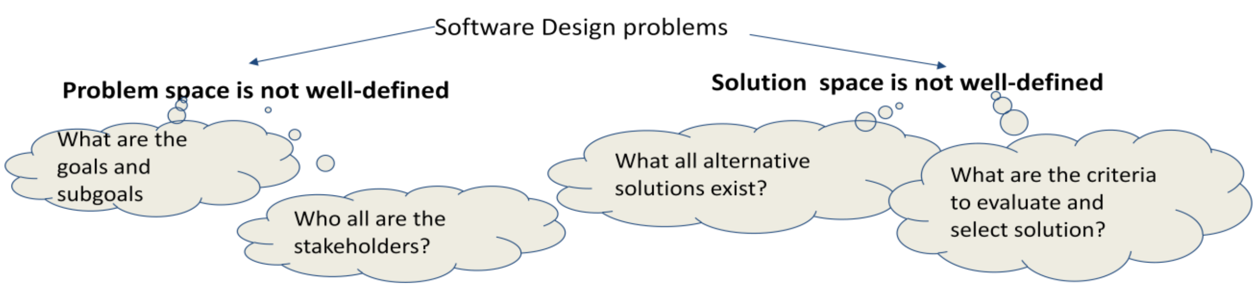


Figure 2.2. Software design problem characteristics

For example, in the software design problem *"Design software for a manufacturing company to automate report generation,"* the problem space is not well-structured as the requirement are broadly defined and designer has to formulate incomplete requirements into specific data models, main software functions and sub-functions. The solution space may have multiple solution paths and multiple alternative design options, and the criteria to evaluate and select an optimal solution may not be clearly stated. The designer has to choose appropriate data structure representation and algorithms by critically evaluating the options based on the criteria and constraints applicable to the given problem domain (Pressman, 2010; Tang, 2010; Guindon, 1990).

Software design is the process followed by designers to design software for a given real-life problem. The design process is iterative and broadly consists of following phases – Establishing a need, Analysis of requirements, Design, and Implementation (Pressman, 2010).

Establishing a need phase is where the requirements of the system are elicited by identifying stakeholders, recognizing requirements from multiple stakeholders' viewpoints, and collaborating to make a final decision on requirements needed.

Analysis of the requirement phase is where the requirements are elaborated into models that depict the overall design. The data model depicts the information domain for the problem, flow-oriented models represent the functional elements of the system (operations and attributes), and scenario-based models of the requirements are from the point of view of various system actors.

The design phase is where the final drawings or model of the system or product are completed. For example, in software engineering, the design model provides details about software architecture, data structures, interfaces, and components that are necessary to implement the system.

Implementation phase is where the designs are implemented by programmers using appropriate programming frameworks and later deployed and tested with the actual users.

In the software development process, the design is the most important phase as stated by Guindon (1990), *"the most expensive errors to correct in a software development project are those made during high-level system design and not during coding."* During the design phase, if the designer is not exploring the problem-solution spaces, then one may end up with incomplete or sub-optimal design (Tang, 2010).

During design, the design decisions made by the designers are implicit and based on the experience and expertise of the designer (Guindon, 1999). The novice problem solvers with no prior experience in design may not think of the problem space fully and end up with incomplete problem formulation (Ackoff, 1979; Dennis, 1999; Adelson 1985) in terms of eliciting requirements from the viewpoint of various stakeholders, identifying the modules and sub-modules, and data items. If the solution space is not explored in terms of all alternative design options and evaluating and selecting based on the selection criteria, then one may end up with wrong or sub-optimal solution (Carman, 2007; Tang, 2008). The issues in solving design problem, which is ill-structured are summarized in figure 2.3.

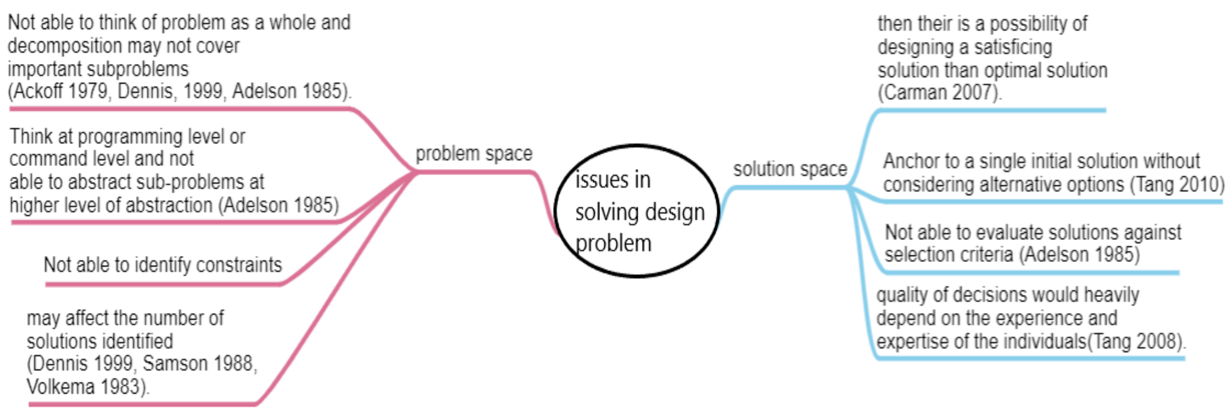


Figure 2.3. Issues in solving design problems if problem-solution spaces are not explored

In the next sub-section, we have reviewed the literature on ill-structured problem solving to explicate the cognitive processes and techniques used in problem-solution spaces to address the issues shown in figure 2.3.

2.2 Cognitive processes and techniques in ill-structured problem solving

The quality of the design depends on how well the problem-solution spaces co-evolved. The problem space has to be expanded enough to understand the system as a whole before you reduce to formulate the sub-problems such that they are directly solvable. In solution space, the alternative design options should be explored, evaluated against the constraints identified in the problem, and select the optimal options (Tang, 2010).

In this sub-section, we have reviewed the ill-structured problem-solving literature to explicate various cognitive processes and techniques proposed to expand-reduce the problem-solution spaces effectively. The various areas of ill-structured problem solving literature reviewed are: problem structuring (Ackoff, 1991; Volkema, 1983;Dennis, 1999;Kunene, 2005; Ellspermann, 2007); creative problem solving (Liu, 2004; Basadur, 2000, Ellspermann, 2007), engineering design (Howard, 2008; Pugh, 1991; Saaty, 2008; Madhuri, 2015) and software design (Adelson, 1985; Guindon, 1990; Tang, 2008; 2010).

2.2.1 Problem structuring

Problem structuring is a preliminary step for solving ill-structured problems. As stated earlier, in ill-structured problems, the stakeholders involved are not directly stated, the problem space is not well-defined, and the goals and specific goals to be achieved are not clearly stated. Research suggests that problem structuring processes are useful approaches to formulate ill-structured problems (Basadur, 2000; Dennis, 1999; Ellspermann, 2007; Kunene, 2005).

It is argued in the literature that the quality of the problem formulation depends on the ability to see the whole problem space and then decompose the problem into subcategories (Ackoff, 1979; Dennis, 1999). This process is categorized as expansionist and reductionist (Volkema,1983) approach in which the problem space is expanded before reducing (decomposing) the problem into subproblems. The expansionist approach involves understanding the system as a whole by identifying the parts and interrelationship between the parts, and reductionist approach involves reducing the problem into sub-problems (Ackoff, 1979).

Risks of problem structuring if the problem space is not expanded are (Dennis, 1999; Samson, 198; Volkema, 1983)-

1. Decomposition of tasks may not completely cover the problem space; it may miss important subcategories
2. It may be more difficult to create good, holistic solutions without considering all subcategories of the problem simultaneously. It leads to the inability to see the whole picture and its interconnections.
3. If the problem definition is narrow, it may affect the number of solutions identified.

Several techniques are used to expand and reduce the problem space. Strategic Options Development and Analysis (SODA) is a method used by consultants to solve complex and messy problems of their clients (Eden, 2001). It relies on the concept of cognitive mapping to explore a problem area. A cognitive map is used to represent a person's thinking about an issue and is a directed graph, consisting of nodes (ideas) and arcs (connections between ideas). Typically, there are over 100 nodes on a map. SODA is implemented through the use of interviews conducted by a facilitator with individuals concerned with the problem area in question. The individual maps can then be combined to produce group maps.

MACRAME (Multiple Actor RepresentAtion ModElling) is a methodology that considers the problem from the viewpoint of various stakeholders involved in the problem situation. Problem is formulated using actorial structure from general level to a specific level. The structure shows how various stakeholders and actors are interrelated to each other (Norese, 1995).

Why-What's Stopping (WWS) is a problem structuring technique to formulate ill-structured problems. It combines expansionism, reductionism, and creative thinking processes. The expansionist approach allowed user to explore the problem by asking question "How might we..", for example, if the problem statement is "How might we retain new customers?", the broadening question would be, "Why would we want to retain new customers?" and the narrowing question would be, "What is stopping us from retaining new customers?" The empirical study was done to investigate the effects of training in the use of WWS technique on the quality and quantity of problem statements. Results showed significant improvement in the quantity of problem statements generated and no significant difference in quality and uniqueness of problem statement in experiment group compared to the control group (Ellspermann, 2007).

The cognitive processes, tools, and techniques proposed in problem structuring literature to explore problem space are usually used by system analysts and experienced professionals to identify and formulate the problem and sub-problems to be solved in an organization.

2.2.2 Creative problem solving

The creative problem solving is another area of research in ill-structured problem solving, which focuses on creative thinking skills. The creative problems are non-routine problems for which the solution or the solution path is unknown. The creative problems are solved using creative problem-solving models, broadly consisting of three stages: Understand the problem; Generate ideas; and Planning for action (Treffinger, 1995).

The divergent and convergent thinking skills are integral to the creative problem-solving process to generate quantity and quality solution (Basadur, 1990; Howard, 2008). Divergent thinking is associated with generating alternative solutions, and convergent thinking skills are -pros & cons analysis of various solutions, making suitable assumptions for a given problem, selection of accurate solution based on constraints identified in a given problem and justifying selected solution (Madhuri, 2015).

Divergent thinking techniques

Various divergent thinking techniques are proposed to generate ideas (Liu, 2004; Oslapas, 1993):

- Analogical thinking: transfer an idea from one context to a new one,
- Brainstorming: encourage to individually generate ideas with a focus on quantity and not quality in less amount of time,
- Mind mapping: variant of brainstorming, where ideas are represented in pictures as well as words
- Attribute listing: identify attributes of a subject and think up ways to modify, reverse, combine, or improve on each.

A research study has shown that brainstorming is effective in a group interaction compared to individuals working alone. In a group, people hear about other people’s ideas, which otherwise they would not have thought of and build on these ideas generating more ideas. It is even more effective when people are working on sub-problems to generate ideas rather than on a problem not divided into sub-problems. This is explained as cognitive phenomena as individuals presented with the problem as a whole, would tend to focus on a small portion of the solution space and generate few ideas. They explore a few related sub-categories in depth rather than think of a broader range of sub-categories of the overall problem (Kunene, 2005).

Brainstorming is widely used to generate ideas in a group(Kunene, 2005), while other creative thinking techniques like attribute listing map or analogous thinking has not been widely used or tested in generating ideas for solving ill-structured problems (Ellspermann, 2007).

2.2.3 Engineering design

Engineering design is the process to design systems, components, or processes meeting specified needs for broadly-defined engineering problems appropriate to the discipline. It is a decision-making process (often iterative), in which the basic sciences, mathematics, and engineering concepts are applied to convert resources optimally to meet a stated objective (ABET, 2014).

The engineering design process broadly consists of four phases: Establishing a need; Analysis of task; Design; Implement (Howard, 2008). It is implicitly assumed in the engineering design process that designers generate alternative design options, evaluate and select. However, it is argued in the literature that by integrating the engineering design and creative problem-solving process by explicitly generating ideas and evaluating them, the quality of design improves. These models are referred to as divergent-convergent models (Howard, 2008; Cooperrinder, 2008; Basadur, 1990).

The divergent thinking techniques were discussed in the previous sub-section: creative problem-solving process, and in this section, we will discuss various convergent thinking techniques proposed in the literature. Convergent thinking techniques are used to evaluate the alternative ideas on various criteria and select appropriate idea. For example, taking decision on buying equipment for a company involves evaluating all alternatives based on various selection criteria-cost, usability, etc. Some of the tools widely used for evaluating alternatives are decision matrix (Pugh, 1981); and Analytic Hierarchy Process (AHP) (Saaty, 1980).

i. Decision matrix (Pugh, 1981) is used to take a design decision by evaluating alternative designs based on multiple criteria and decide which one best meets all the criteria. The decisions that are based on multiple criteria is complex, thus resulting in inconsistent and irrational decisions. The Pugh Matrix provides a simple approach to compare alternatives designs against each criterion.

The alternative designs and criteria are listed in a table, as shown in figure 2.4. The figure shows the use of a decision matrix to evaluate and select from the alternatives.

	Design1	Design2	Design3	Design4
Criteria1	1	3	4	3
Criteria2	2	5	4	2
Criteria3	4	3	1	3
Criteria4	3	2	1	4
Total score	10	13	10	12

Figure 2.4. Example of decision matrix

In figure 2.4, four design alternatives are evaluated against four criteria. The designs are rated on a scale, for example, 1-5 against each criterion and the total rating for each design is calculated. The design with the highest rating is selected. The alternative is to weight the criteria based on its importance over other criteria and multiply the rating with the weight of each criterion, before summing.

The advantage of Pugh's decision matrix is that it is easy to use. While this method will fail if the selection of criteria is incomplete or incorrect as the effectiveness of Pugh's matrix is related to the quality of the selection criteria. If the selection criteria are not correct or incomplete, then the selection decision can go wrong.

ii. Analytic Hierarchy Process (AHP) is another popular technique proposed by Saaty (1980) which is used to choose among multiple criteria and choices, for example, select a car by evaluating alternative choices (Baleno, Honda city, Ford aspire) against the multiple criteria (cost, style, mileage, reliability). The basic principle of AHP is to do a pairwise comparison of criteria and perform matrix calculation (Eigen-vector) to find the relative importance of one criterion over another. The steps followed are:

a. The Saaty's scale (Table 2.1), is used to calculate the importance of one criterion over other. For example, if the criteria "cost" is somewhat more important than "style," then "cost" is rated at 3 and "style" is rated at 1/3. These pairwise comparisons are carried out for all factors to be considered, and the matrix is completed as shown in Table 2.1 (second table):

Table 2.1. Saaty's rating scale with example

Intensity of Importance	**Definition**	**Explanation**
1	Equal importance	Two factors contribute equally to the objective
3	Somewhat more important	Experience and judgment slightly favor one over the other.
5	Much more important	Experience and judgment strongly favor one over
7	Very much more important	Experience and judgment very strongly favor one over the other. Its importance is demonstrated in practice.
9	Absolutely more important.	The evidence favoring one over the other is of the highest possible validity.
2,4,6,8	Intermediate values	When compromise is needed

	Style	Cost	Mileage	Reliability
Style	1	1/3	5	1
Cost	3	1	5	1
Mileage	1/5	1/5	1	1/5
Reliability	1	1	5	1

b. In next step, the Eigen-vector is calculated by standard method as (0.232, 0.402, 0.061, 0.305). These four numbers correspond to the relative values of style, cost, mileage, reliability. The 0.402 means that cost is the most preferred criteria over all others, and the next preference is given to reliability (Saaty, 1980; Coyle, 2004).

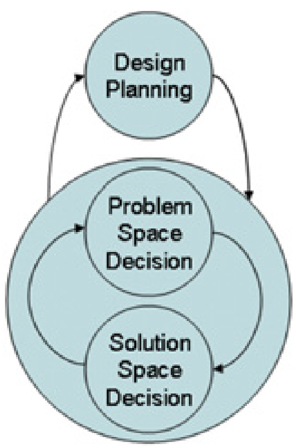
The main advantage of the AHP is its ability to rank choices in the order of their effectiveness in meeting conflicting objectives. The drawback is the ambiguity in the interpretation of the Saaty's rating scale and complexity of the mathematic calculations.

2.2.4 Research studies in software design

In this section, we have explicated the important cognitive processes and behaviors exhibited by designers while solving software design problems (Adelson, 1985; Guidson, 1990; Tang, 2010; 2008). Four expert studies were reviewed, which are focused on eliciting expert behaviors while designing software for familiar and unfamiliar domains (Adelson, 1985), problem-solving heuristics used by experts to design satisficing solution (Guindon, 1990) and design reasoning techniques to improve software design quality (Tang, 2010; 2008).

The designers solved design problems by iterating and co-evolving between problem-solution spaces. Figure 2.5 describes an iterative design decision-making process consisting of two stages: design planning and problem-solution space decision. Stage 1 is when designers plan the design session by scoping the problems and prioritizing the issues. Stage 2 is where designers solve the design problem by iterating and co-evolving between problem-solution spaces. The problem space decision making involves problem analysis and formulation. Solution space decision making is about generating solutions and other potential options, evaluating between different options, and selecting the most appropriate solution (Tang 2010).

a



b

Design Planning: 1. Scoping 2. Identify Key Design 3. Priortise Design Problems **Problem Space Decisions:** 1. Analyse Problem 2. Formulate Problem **Solution Space Decisions:** 1. Generate Solution 2. Evaluate Solution 3. Select Solution	**Reasoning Techniques:** • Conte(X)tualise situation • (S)implify situation • Sc(E)nario generation • (C)onstraints consideration • (T)rade-off analysis • (O)ption generation • (I)nduction • (D)eduction

Figure 2.5. (a) iterative design decision-making model, (b) general design activities and reasoning tactics (Tang, 2010).

The cognitive skills and behaviors exhibited by designers in problem and solution space based on four expert studies are discussed followed by comparison of the studies:

1. Problem space related cognitive skills:

In problem space, the designers start with problem analysis to identify the information relevant to a problem, followed by problem formulation to articulate what problems and sub-problems need to be solved or how to proceed to solve them. To identify the sub-problems to be solved, the techniques used by designers for exploring and expanding problem space are:

i. Formation of mental models: Expert designers form an internal working model of the design-in-progress which is capable of supporting mental simulation. The evidence of these models is the experts generated external models in the form of sketches which depicted some form of actions in the system (Adelson, 1985).

ii. Systematic expansion of the mental model by co-evolving between problem-solution spaces: The nature of the mental model change as the design is progressed. The model started at an abstract level and expanded to the next level of detail. For example, while designing email-system, the designer started with a model depicting the flow of information as commands (READ, WRITE) at an abstract level and progressed to a detail level by representing information as the flow of tokens. This progression from one level of abstraction to the next immediate level is known as systematic expansion of the mental model. This mental model is used to identify sub-problems and its corresponding solutions.

The prior domain knowledge plays a major role in the simulation of the mental model and its assessment. In problem- *design library record keeping system,* where the designers did not have prior experience in solving a similar problem, the mental model formed was not as detailed as the one in the familiar domain. The designers only represented how the system would appear to the user and did not consider other issues of how the system would interact with the operating system. The simulations were done by representing commands (read, reply, etc.), but simulations done were local considering each command in isolation and not as a whole in terms of how different commands interact with each other. The person was not doing global simulation on how commands are affecting other commands (Adelson, 1985).

iii. Requirement understanding by simulating scenarios: Experts tend to generate requirements from the incomplete problem statements by simulating scenarios in the given problem (Guindon, 1990; Tang 2010). For example if the problem is to "design a logic for a lift", then designers simulate the scenario of lift moving up or down across floors using diagrams. In the process of drawing, designers deduce new requirements; for example, the lift should locate the location of the floor, or constraints like "no single point of failure" or "low cost," which may not be part of specifications. Experts come up with subproblem and its associated solution to satisfy the new requirements. These requirements allow designers to reduce the solution space and do a quick trade-off analysis to select a solution based on the constraints (Guindon, 1990).

iv. Initial problem exploration: The study by Tang (2010) showed that for the design task: "design traffic simulation program for students to understand traffic lights, condition and flow", the team that started upfront with the solution without exploring the problem space sufficiently formulated only few problem statements. Whereas another team that spent more time on exploring their plan, designed layout of the road and other design issues, generated a larger number of problem statements.

2. Solution space decisions:

In the solution space, the three general steps followed are: generating solution options, evaluating the available options, and selecting a solution from the available options.

i. Generating solutions:

Designers select design strategies and notations based on their prior knowledge. For example, in the study by Guindon (1990), one of the designers selected entity-action notation and DFDs to structure the problems and used it as a guide towards design decomposition. This behavior is consistence with his software engineering background. Another designer chose state transition diagram as his design notation, as he has experience as an electrical engineer and worked with problems similar to the lift problem.

Different design behavioral patterns were observed depending on the experience of the designers. The experienced designers may not consciously generate multiple solution options explicitly, but consider alternative solution options during various contexts like during problem refinement, constraint specification, and while considering new scenario (Tang, 2010). While, for inexperience designer's, the initial design became their final design. They appeared to adhere to their initial designs, from where they continued to design for additional requirements. After each decision point, the inexperienced participants especially spent little efforts on reassessing the consequences of additional changes to the initial design. However, the study by Tang (2008) showed that when designers were prompted to think of alternatives and justify the selection, the quality of the design solution improved. The need to justify their decision made them to constantly assess the design decision against the requirements and constraints, especially for inexperienced designers.

The studies have also shown that designers generated more design options when a higher number of sub-problems are identified in problem space (Tang, 2010; Kunene, 2005).

ii. Evaluating alternative solutions:

- **Trade-off analysis:** Regarding evaluating alternative solutions, experienced designers rarely considered more than one alternative to the problem. Even if they considered, they rapidly eliminated without in-depth evaluation using a quick trade-off analysis (easy to design). For example, in the lift example, the designer did not do an in-depth analysis of design alternative- distributed versus centralized control process but quickly chose distributed based on the requirement of no single point of failure. Designers tend to elaborate only one main design idea, usually selected early in the design process. The selection is attributed to a similar solution that has worked in the past and the limited time to design (Guindon, 1990).
- **Preferred evaluation criteria:** Designers select preferred evaluation criteria to reduce the set of possible designs to consider and use it as a primary guide in searching for a satisfying solution which may not be an optimal solution. For example, in the lift example, the designer selected reliability as preferred evaluation criteria and used it to select a distributed control system and to reject the centralized control system.

  Preferred evaluation criteria if wisely selected, can effectively reduce the complexity of the design process. However, in need to reduce the search space, inexperienced designers may early commit to wrong choices that are difficult to recover later on. Delayed commitments until more constraints on the design have been uncovered may be more desirable (Guindon, 1990).

iii. Selecting a solution

Some of the skills that are used by designers to evaluate and select a solution effectively are:

- **Explicit reasoning:** Explicit reasoning is the use of explicated arguments to support or reject a design problem and a solution choice. The empirical study by Tang (2008) showed that the test group that was equipped with design reasoning produced a higher quality design than the control group, especially for inexperienced designers.

- **Representing constraints, plans, and note-taking:** Designers keep track of constraints, critical requirement, and assumptions in the problem statement by making a note of it. They referred back to their notes to check if all the critical requirements were met in the final review of the solution design (Adelson, 1985; Guindon, 1990). The designer simulated solution design and compared them with what was demanded by the requirements.

3. **Comparison of the expert studies in software design**

The four expert studies in software design process reviewed are compared along with the parameters: objective, participants, design problem solved and cognitive processes used, shown in table 2.2.

Table 2.2. Comparison of the expert studies in software design

Expert study	**Objective**	**Participants**	**Design problem solved**	**Cognitive processes identified**
The role of the domain experience in software design (Adelson, 1985)	1. To identify the behaviors and skills of a designer when designing in a familiar domain and an unfamiliar domain. 2. Difference between novice and expert behavior.	**Experts-** 8years of experience in designing communication systems. **Novice-** several years of experience in programming and two years in designing.	1. Design an electronic mail system. 2. Design a library record keeping system 3. Design an interrupt handler.	Form the mental model, systematic expansion of model, simulate the model and assess, plan solutions, note-taking.
Knowledge exploited by experts during software system design (Guindon, 1990)	To describe how designers exploit heuristics and personalized evaluation criteria to constraint a design process and select a satisfactory solution.	**Two designers**, 1. Five years of experience in designing real-time systems. 2. Ten-year experience with the communication system	A lift control system- design a logic to move lift between floors with the given set of rules.	Simulate the problem scenarios to create new requirements, constraints, partial solutions, and test-cases. The requirements were the basis for reducing the design alternatives.
What makes software design effective (Tang, 2010)	Study how designers do the planning, co-evolve problem-solution space, and use reasoning techniques in software design and its influence on design	**Two teams** of two experienced software designers each.	Design traffic simulation program for students to understand traffic lights, condition, and flow.	Initial problem space exploration, minimal context switching, problem-solution co-evolution, and explicit reasoning can help effective

	effectiveness.			software design.
Design reasoning improves software design quality (Tang, 2008)	The goal of this work is to investigate if there is any quality improvement to software design when design reasoning is applied.	**Twenty participants** with an average of eight years of experience.	design a UI for the monitoring system	Designers prompted to think of alternatives and justify the selection, led to improved design quality, especially for inexperienced designers.

Summary

Based on the four expert studies discussed above, the cognitive processes used by designers in problem and solution space are summarized:

Problem space-It is observed in studies that the designers expand problem space by creating and simulating mental models at various levels of abstraction. Eventually, from these mental models, the sub-problems are decomposed.

Designers create a mental model of the system by using external representations like drawing a state transition diagram or listing the entities and their respective properties and actions. The drawing chosen is based on prior experience of the designer; for example, if the designer has prior experience with designing control systems, then he may prefer state transition diagram and a software designer may prefer entity-action representation or data flow diagrams. If the problem to be solved is*: design a library management system for a college*, then different representations that can be used by the designers are state transition diagram (figure2.6), or listing entities and corresponding actions (figure 2.7).

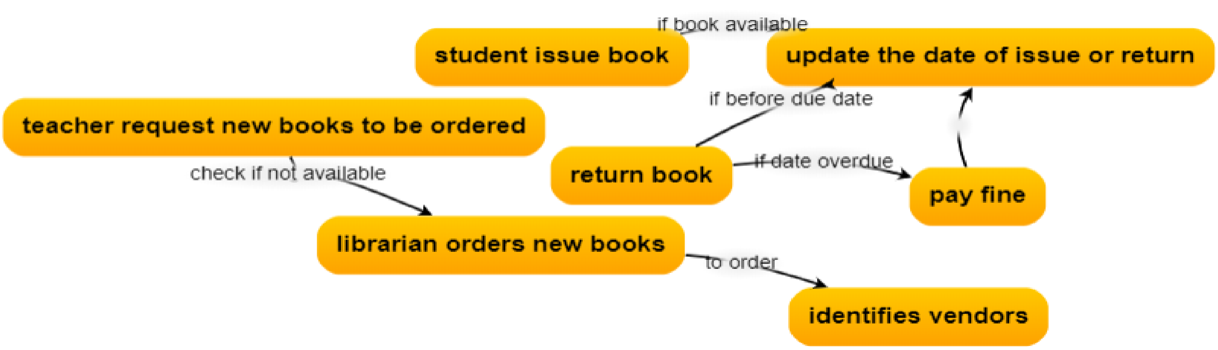


Figure 2.6. State transition diagram of the library management system

Entity	Action
Librarian	Issue books, return books, maintain cards, order books, calculate fine.
Teacher/student	Issue/return books, re-issue book, request new books
Book	Update date of issue or return

Figure 2.7. Entity-action list of the library management system

Solution space- After initial problem exploration and stating the requirements, designers generate solutions for each requirement by following three general steps: generating solution options, evaluating the pros and cons of the available options when there is more than one solution option, and selecting a solution from the available options. The solution design is often interleaved with problem space, as designers may identify new requirements or constraints while designing a solution and go back and forth in problem-solution spaces.

The experienced designers are good at identifying the selection criteria; they may not do an exhaustive search in solution space and quickly reduce to few alternatives based on the selection criteria; quickly do trade-off analysis of the available options and take design decisions. However, the inexperienced designer benefits more by explicitly generating all possible alternatives; explicitly stating the selection criteria; evaluating alternative options against the selection criteria; and taking design decision to select by explicitly justifying the selection. This process helps the designer to backtrack to other alternative solution if the current solution is later found to be unviable.

The cognitive skills followed by the designers is shown diagrammatically in figure 2.8.

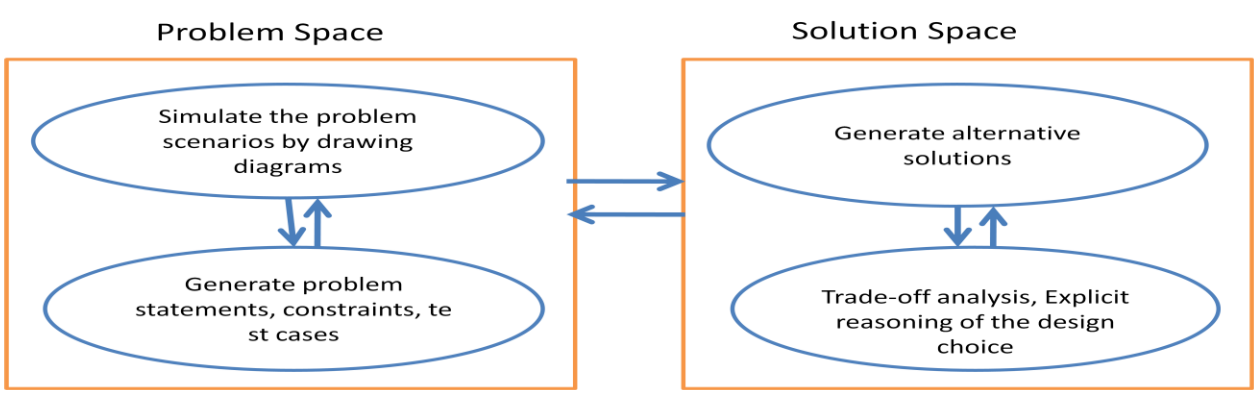


Figure 2.8. Cognitive skills in problem and solution space.

2.3 Expand-reduce skills in solving design problems

The ill-structured problem-solving literature states the process of understanding the system as a whole from the perspective of multiple stakeholders viewpoint as expansionist thinking, and the process of decomposing the problem into sub-problems that are directly solvable as reductionist thinking (Ackoff, 1991; Volkema, 1983; Ellspermann, 2007). In creative problem solving, the emphasis is given on expanding the solution space by generating alternative ideas/solutions using various divergent thinking techniques (Liu, 2004; Basadur, 2000), and in engineering design process, the emphasis is on using convergent thinking techniques to evaluate and select alternative solutions based on multiple criteria (Howard, 2008; Pugh, 1991; Saaty, 2008; Madhuri, 2015). The software design literature is reviewed to identify the cognitive processes used by experts in the design process. The cognitive tools and processes related to the expand-reduce problem and solution space are summarized in figure 2.9.

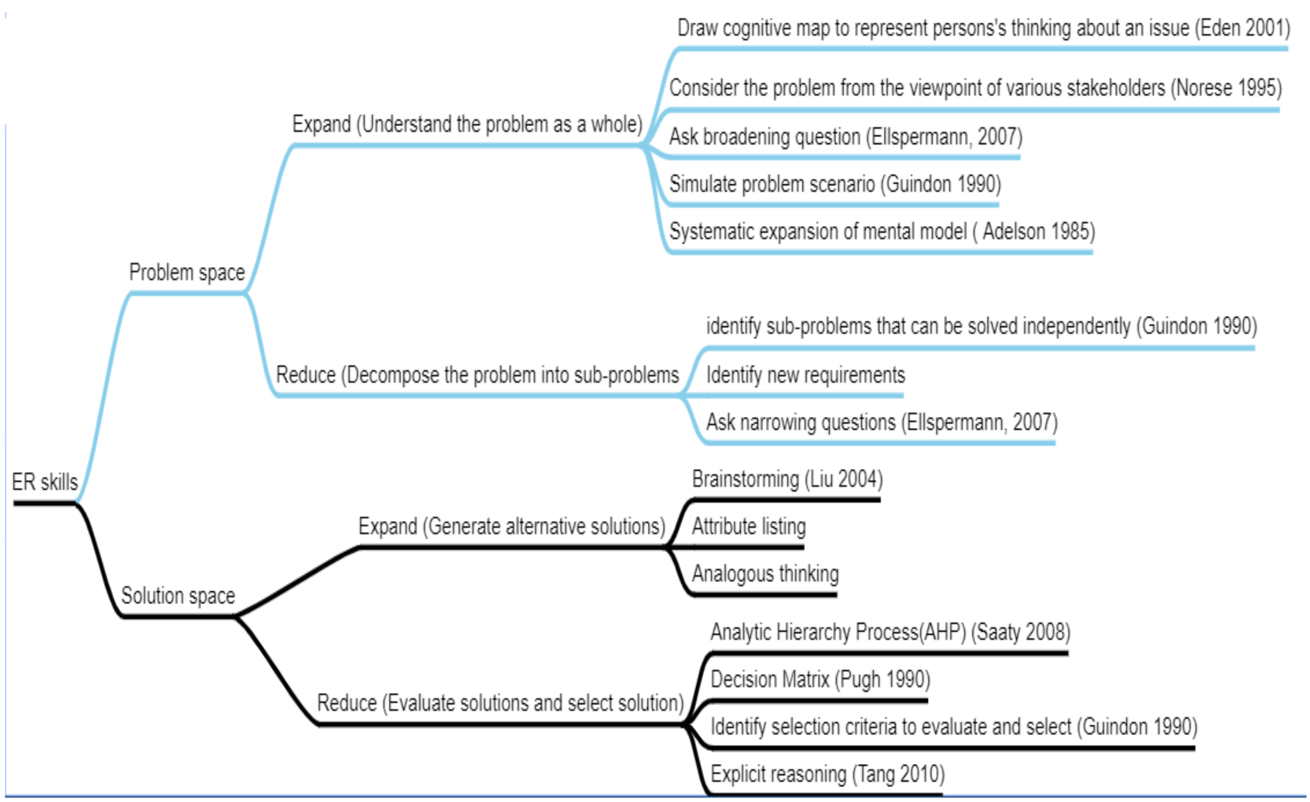


Figure 2.9. Summary of the ER skills and tools used in problem and solution space

In this thesis, we characterize expand-reduce skills based on expansionist-reductionist thinking in problem space (Ackoff, 1991; Volkema, 1983; Ellspermann, 2007) and divergent-convergent thinking in solution space (Liu, 2004; Basadur, 2000, Howard, 2008; Pugh, 1991; Saaty, 2008; Madhuri, 2015). We define expand-reduce(ER) skills as the ability to explore the problem space or solution space and eventually reduce towards a solution. In problem space, the ER skill is the ability to explore the problem as a whole and reduce the problem into sub-problems based on the goal to be achieved. In solution space, the ER skill is the ability to generate alternative solutions and reduce to one solution by evaluating and selecting based on selection criteria (Figure 2.10).

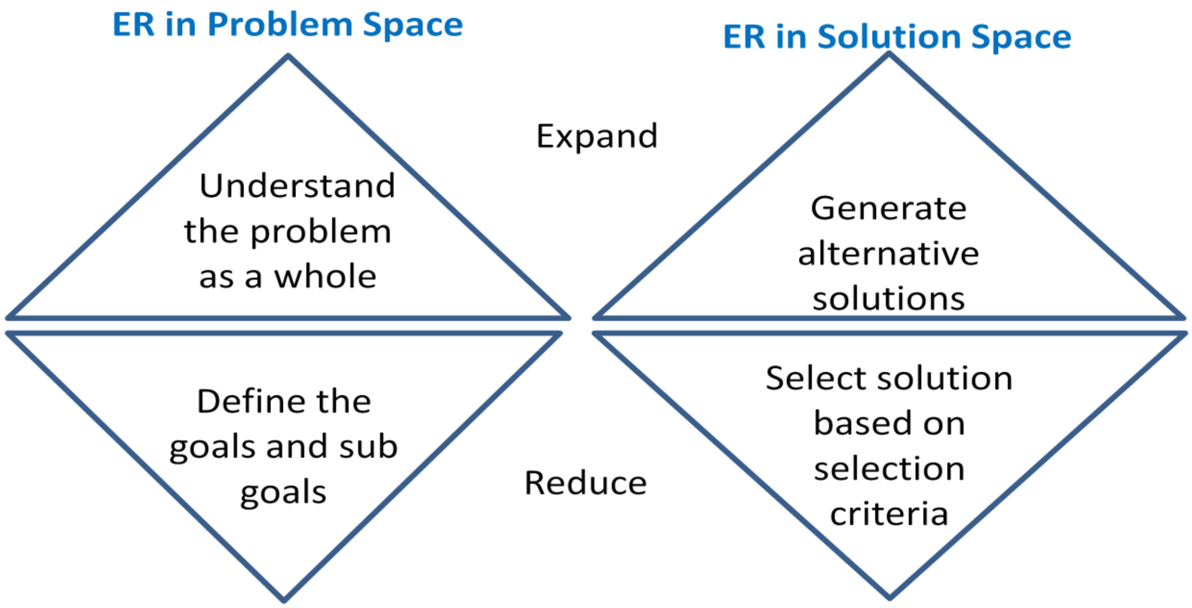


Figure 2.10. Expand-reduce skills in problem-solution space.

2.4 Comparison of expert and novice cognitive skills

The comparison of novice versus expert behavior in solving design problems identified in the four expert studies are summarized below (Adelson, 1985; Guindon, 1999; Tang, 2008; 2010):

i. Formation of mental models: Novice considers commands at a programming level and not able to form or simulate the mental model. Novice considers some or single sub-problem or command at a time and is not able to consider all sub-problems and interactions of the system.
ii. Novices have only low-level representations at the programming knowledge level, whereas experts have representations at both abstract and concrete levels

iii. Experts do not tend to generate a wide range of alternatives, they intuitively rule out unviable design alternatives, whereas inexperience designers benefit from explicitly considered design options (Tang, 2008).

iv. Experts use design heuristics to guide their search for a design solution; for example, may not commit to the design decision until sufficient information on the requirement and constraints is available. However, novices may differ in using the heuristics efficiently; for example, the novice may commit early on decisions (Guindon, 1999).

v. The inexperienced designers tend to adhere to their initial design, which eventually became final design. As the design progresses, they tend to ignore the mandatory requirements and do not assess the design against the requirements. However, experienced designers are good at identifying the selection criteria and selecting the right option. This expert behavior also depends on their prior experience with similar problems.

The research studies in software design suggest the need to apply ER related cognitive processes to improve the quality of the design, especially when the designer is not having prior experience in designing or if the problem domain is unknown. The research studies were more focused on eliciting the cognitive processes used by designers while designing, but to our best knowledge, there is no research done in the direction of teaching-learning of these skills to novices. In the next sub-section, the issues in engineering education at the undergraduate level are discussed to understand the state of formal education in teaching-learning of complex problem-solving skills to novice engineers.

2.5 Engineering education

"Engineering is a profoundly creative process. A most elegant description is that engineering is about design under constraint. The engineer designs devices, components, subsystems, and systems and, to create a successful design, in the sense that it leads directly or indirectly to an improvement in our quality of life, must work within the constraints provided by technical, economic, business, political, social, and ethical issues."(From the book: The engineers of 2020)

One of the outcomes of engineering education is the ability to identify, formulate and solve engineering problems; design a product under a given set of constraints, which directly or indirectly solves some real-life problems and improve quality of our lives (ABET, 2014). The design problems are ill-structured and the skills needed to solve design problems need to be taught explicitly to students during engineering education. However, engineering education is more focussed on teaching content and solving well-structured problems.

The well-structured problems are directly solvable by applying known concepts and principles, and they have obvious correct or incorrect answers. The ill-structured problems are not directly solvable; the problem is vaguely defined, possess multiple solutions and solution paths. It is implicitly assumed that by solving well-structured problems, students can transfer skills to solve ill-structured workplace problems. However, research studies have shown that solving well-structured problems does not readily transfer to ill-structured problems (Jonnasen,2006; Sharp, 1991).

The book - "The engineers of 2020", addresses the issues in engineering education, in which it is observed by an industry that the gap between engineers in practice and engineers in academics has developed and grown. This is the reason that today's engineers are not adequately prepared to enter and sustain themselves in today's workforce. The curriculum should be redesigned to reduce this gap by focussing on the development of skills to solve complex problems, use technology to enhance learning, and make life-long student learner.

In this thesis, we address to reduce the gap between engineers in practice and engineers in academics by identifying the cognitive skills that designers apply while solving software design problems and design a learning environment for teaching-learning of complex problem-solving skills to novices.

2.6 Research gap in teaching-learning of ER skills to novices

The gap in teaching-learning of ER skills to novices is summarized based on:

i. Research studies in problem-solving that have established the importance of ER skills in solving design problems efficiently (Ackoff, 1979; Volkema, 1983; Liu, 2004; Ellspermann, 2007, Adelson, 1984; Guindon, 1990; Tang, 2008).

ii. Research studies have shown that novice learners cannot apply ER skills effectively in solving ill-structured problems, but limited research is done to address the problem of teaching-learning of ER skills to novice learners (Jonnasen, 2006; Bennart, 2013).

iii. Engineering education is not focussed on the teaching-learning of thinking skills to solve ill-structured problems to undergraduate students (Cooperrider, 2008; Dym, 2005).

The research gap identified is the need for teaching-learning of ER skills to novices. Based on the research gap identified, our broad and specific research goals are defined:

Broad research goal-

Design technology-enhanced learning environment (TELE) for effective teaching-learning of ER skills to novice learners.

Specific research goal-

Design technology-enhanced learning environment (TELE) for teaching-learning of ER skills to undergraduate computer engineering students in the context of software design problem-solving in data structures and algorithm course.

2.7 Research goal

To address the research gap in teaching-learning of ER skills to novices, we conjecture that designing a learning environment to trigger ER skills in the context of solving software design problems will foster the learning of ER skills to novices. Our broad research question is:

Broad research question- How to teach ER skills to novice learners in the context of solving software design problem?

The specific research questions (RQ) are:

i. How to assess the level of the novices (undergraduate engineering student) in applying the ER skills?

ii. What is the level of scaffolding that is needed in teaching-learning of ER skills?

iii. How can technology affordances be used in learning ER skills?

To train novice in ER skills, we have scoped our problem to software design problem solved using appropriate data structures and algorithm. The learners trained are second-year computer engineering students who have completed data structures and algorithms course.

The software design problems used for training novices in our research are:

Problems Posed	
Shop-inventory problem	*The local automobile retail shop sells parts for different car models- oil, filters, brakes, batteries, etc. The shop owner notices that if any part is getting out of stock, then there is a risk of losing customers. So he wants to have a software developer create an inventory control program that tracks the quantity of all the parts and creates a report of the parts that needs to be ordered, so there is minimal risk of items getting out of stock. Each night the program creates a report of the parts whose quantity is below a certain value so that they can be ordered to bring the stock levels back up to the right number.* *Come up with multiple possible solutions by using appropriate data structures and operations for solving the above problem. Justify which solution is most efficient for the above stated problem.*
Library problem	*The college library maintains books of various departments like Computer engineering, Electronics engineering, etc. The library staff maintains records of all books, return and issue of books to students and teachers. It was becoming tedious for teachers and students to search for a book in the library. The librarian decided to provide online service for students and teachers to search the availability of the books. The system will search for the availability of the given book and display if the book is available/not available with its shelf number (location of the book). Your task is to design software for the requirement stated by the librarian, using appropriate data structure and algorithm.*
Bank problem	*A new branch of bank XYZ has opened in a city. The bank provides the following services to customers: maintaining customer accounts, withdraw amount, deposit amount, and check the balance in their account. The bank manager wants a software system to be designed for its customers to provide a service of checking their account balance, while other services will be offline. The system should take customers' account_no as input and display the balance in the account.*

Your task is to design software for the above requirement using an appropriate data structure and algorithm.

In these problems the real-life scenario is posed, for example, bank, library or a supermarket and the requirements to be achieved are stated at a very broad level, for example, “Design software system to search for book availability in the library.” The problem is not well-structured as the data items and operations (sub-problems) are not clearly defined in the problem. The solution space consists of searching and selecting appropriate data structure and algorithms for each data item and operations based on the selection criteria. The solution space is not well-defined as multiple solutions exist, and the selection criteria, sub-problems, and constraints are not directly stated.

Chapter 3

Research Methodology

Based on the literature survey, we have stated our research objective as “to design technology-enhanced learning environment for teaching-learning of ER skills to novices in the context of software design problem-solving.” To achieve our research objective which is interdisciplinary, it is important to ask the right research questions based on the principles suggested in various disciplines like educational technology, learning sciences, problem-solving, software design, etc. To answer the research questions, the learning environment has to be designed, and research studies have to be conducted to evaluate our intervention and contribute back our findings in terms of an innovative solution, new principles or reconfirming the existing principles.

In this section, we will discuss the research methodology used to achieve the stated research objective and overview of our research.

3.1 Research Objective

To achieve the stated research objective, namely, teaching-learning of ER skills to novices in the context of software design problem using Technology-Enhanced Learning Environment (TELE), we need to know the level of scaffolds to be designed in the learning environment for the novices to perform and learn ER skills. The sub-goals that we will be addressing to design learning environment are-

1. Assess the level of the novice in applying ER skills in the context of software design problem-solving.
2. Identify the relevant cognitive tools and processes in software design that can be used to teach ER skills to novices.
3. Identify the level of cognitive and metacognitive scaffolding that is needed in teaching-learning of ER skills to novices.
4. Identify the technology affordances that can be used in teaching-learning of ER skills.

To achieve our sub-goals, we may have to conduct studies with novice to understand the difficulties and level of scaffolding needed in the learning of ER skills, and design and redesign the intervention by adding, removing or modifying the design features based on the feedback from the evaluation studies. The development of the TELE and its evaluation will be a continuous and cyclic process.

Design based research (DBR) is a framework that helps the researcher to analyze the real world problem, integrate design principles with technological advances into solution development, test and refine learning environments as well as define new design principles (Reeves, 2006; Amiel, 2008). We will be using this DBR methodology for our research, as it suits our research requirement of working with practitioners and iterating through design-evaluate-redesign phases.

3.2 Design-Based Research (DBR)

The design-based research (DBR) is a framework used in educational research (Reeves, 2006; Amiel, 2008). DBR aims to build strong connections between educational technology, educational research, and real-world problems. The emphasis is on an iterative research process that involves design-evaluate an innovative product or intervention, and systematically refine the innovation while also producing design principles.

In traditional empirical predictive research, the interventions are designed, developed and tested with learners as a single one-shot study, and researchers rarely engage directly with practitioners in the design process and, if this is done, participation is limited. While DBR gives the advantage of evaluating the design with learners throughout the design and development cycles until the final stage of research. DBR framework allows the researcher to understand how integrating technologies into the classroom leads to substantial changes in social organization, student-teacher relationships, and a myriad of other factors that cannot be investigated successfully by predictive research. The research studies are done in schools or colleges in which researchers investigate in real-world settings while attempting to control for critical variables identified through theory and previous research (Amiel, 2008).

The DBR cycle consists of four phases: Problem Analysis, Solution Development, Evaluation, and Reflection, as shown in figure 3.1.

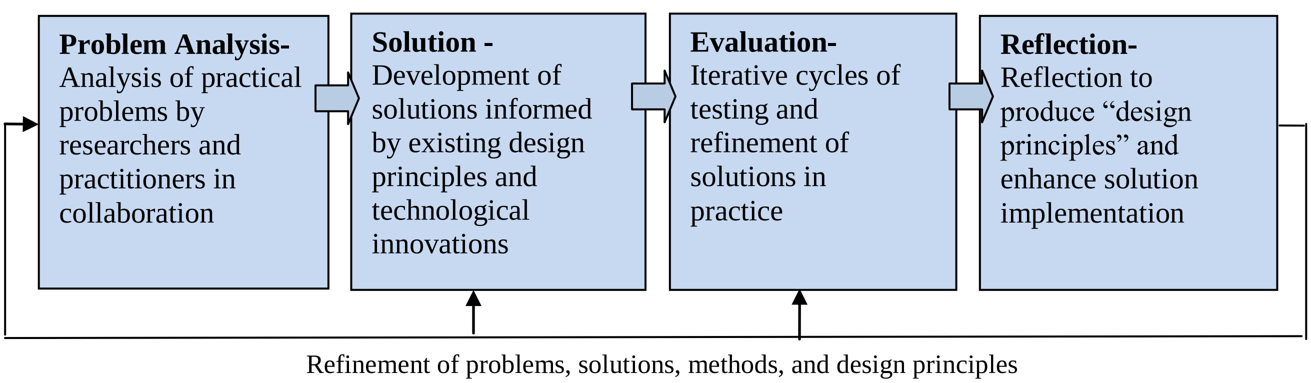


Figure 3.1. Stages in each cycle of DBR (Reeves, 2006)

In the problem analysis phase, the problem is analyzed by researchers and practitioners in collaboration, and the research questions are established that are worth to be investigated further.

In the solution development phase, the design for the learning environment is proposed to address the research problem. This design could be a new set of strategies, or it could be based on research gathered from previously tested design principles.

In the evaluation phase, the solution is tested with practitioners in real-world setting, for example, in a classroom setting with students.

In the reflection phase, the results are analyzed to produce "design principles" and based on the feedback from the evaluation study, redesign the intervention.

The development of design principles will undergo a series of testing and refinement cycles. Data is collected systematically to re-define the problems, possible solutions, and the principles that might best address them. As data is re-examined and reflected upon, new designs are created and implemented, producing a continuous cycle of design-reflection-design. The outcomes of design-based research are a set of design principles or guidelines derived empirically and richly described, which can be implemented by others interested in studying similar settings and concerns. While the ultimate objective is the development of theory, this might only occur after long-term engagement and multiple design investigations.

3.3 Overview of DBR cycles in our research

For the current research problem, we required three DBR cycles, summarized in table 3.1.

3.3.1 DBR Cycle 1

To achieve our research goal of teaching-learning of ER skills to novices, it is important to understand the difficulties faced by novices in doing and learning of the ER skills in the context of solving software design problems and the level of scaffolds to be designed in the learning environment. As a first step towards achieving our research goal, the problem analysis in the first DBR cycle is divided into two sub-goals:

1. To assess the level of a novice in solving software design problem using ER skills,
2. To identify the instructional strategies for teaching-learning of ER skills.

A preliminary study, we will call it study1, was done for our first sub-goal. The research question for study 1 is:

RQ1- What is the level of a novice in applying ER skills in solving software design problem?

Participants and research method: Since the aim of the study is to understand student's difficulties in performing ER skills, we used exploratory research method. Exploratory research is used as it is conducted for a problem that is not studied clearly and aims to acquire more insights to formulate a more precise problem or develop a hypothesis (Shields, 2013). The study was done with 40 second year undergraduate engineering students using the worksheet in which the students were instructed to solve a software design problem in the data structures course:

Problem - *The local automobile retail shop sells parts for different car models- oil, filters, brakes, batteries, and. The shop owner notices that if any part is getting out of stock, then there is a risk of losing customers. So he wants to have a software developer create an inventory control program that tracks the quantity of all the parts and creates a report of the parts that needs to be ordered, so there is minimal risk of items getting out of stock. Each night the program creates a report of the parts whose quantity is below a certain value so that they can be ordered to bring the stock levels back up to the right number.*

Come up with multiple possible solutions by using appropriate data structures and operations for solving the above problem. Justify which solution is most efficient for the above-stated problem.

Table 3.1. Overview of DBR cycles completed in this thesis

	DBR Cycle 1	**DBR Cycle 2**	**DBR Cycle 3**
Problem analysis	• To assess the ability of the novice in applying ER skills and the level of scaffolding required to teach ER skills. **(Study 1)** • To identify the instructional strategies for teaching-learning of ER skills based on a literature survey.	• To redesign the TELE with an improved level of scaffolding and introduce reflection activities. • To assess the learning of ER skills.	• To redesign the TELE to scaffold the learners to regulate the evaluation of the ER skills. • To assess the learning of ER skills.
Solution	**Fathom-Ver1**: The TELE was designed to enable learners to practice ER skills in the context of solving a software design problem.	**Fathom-Ver2:** The activities were redesigned with more scaffolding. The self-evaluation techniques were added to monitor and take	**Fathom-Ver 3:** The metacognitive scaffolds were improved in terms of system generated formative feedback; peer review; self-evaluation;

		control action plan to improve ER skills.	and active collaboration.
Scaffolds	1. Prompts with the definition of new terms, examples, and animations. 2. Cognitive tools- drawing diagrams (on paper), pros and cons analysis, decision table, textbox – writing goal, sub-goal, solutions, justification.	1. Worked example and videos to illustrate the use of the cognitive tool to perform ER skill using a similar design problem. 2. Drawing tools build in TELE. 3. System generated feedback (few activities) 4. Self-evaluation 5. Analyze the new problem.	1. System generated feedback, 2. Peer review to evaluate peers response and self evaluate their response. 3. Active collaboration chat facility to clarify, suggest, and appreciate peer's responses.
Evaluation	**RQ-** How effective are the design features of TELE in doing ER skills? **Study2** with 49 second year computer engineering students. **Exploratory study** **Results-** The activity supported with simulation, prompt with the definition of new terms and example were effective. Prompts are not effective as students had no prior experience of solving a design problem.	**RQ-** How effective are the design features of TELE in learning ER skills? **Study3** with 49 second year computer engineering students. **Experiment-** Pretest-Intervention- Posttest **Results-** Significant improvement from pre-test to intervention score, Significant improvement from pre test to post test. Self-evaluation activity- only 24% of students improved their responses, 76% did not change the response.	**RQ-** How effective are the design features of TELE in learning ER skills? **Study4**with 50 second year computer engineering students. **Experiment-** Pretest-Intervention- Posttest **Results-** Significant improvement from pre-test to intervention score, Significant improvement from pre test to post test.
Reflection	The level of the scaffolding of the activities has to be improved. Our goal in the next cycle is to engage novice in a reflection, activity to evaluate the gap between the current and the desired level and take appropriate action to improve the skill [James, Flavell, Narciss].	The self-evaluation activity was not effective as the students were not able to evaluate their response and take appropriate action with just one intervention. Scaffolding is needed to regulate the evaluation skills in novices.	The pedagogical support in Fathom-Ver3at both cognitive and metacognitive level is effective in performing and learning of ER skills for novices.

Data Collected and analyzed: To measure the level of students in performing ER skills, we evaluated the student's artifacts using the rubric designed to assess ER skills.

Results of study1: The results of study1 showed that students were not able to expand-reduce effectively in problem-solution spaces, which led to weak solution design and reasoning.

Based on the results of study 1, it was evident that students could not apply ER skills on their own. Our next step was to design a learning environment for the teaching-learning of ER skills to novices. We based the design of the learning environment on the instructional design principles by Willis (2009) which states that to design a learning environment, the instructional designer has to make the right choices about guiding theories of learning (e.g., behaviorism, cognitive science, constructivism, etc.); general strategies for teaching and learning (e.g., direct instruction, student-centered instructions, etc); and pedagogies (e.g., problem-based learning, anchored instruction, tutorial, etc.). Since we are developing a learning environment for teaching-learning of thinking skills, the design process is based on Technology Enhanced Learning of Thinking Skills (TELoTS) framework (Murthy et al., 2016), that recommends steps to be followed by the researchers and designers to design an effective smart learning environment for developing thinking skills.

To get insights on the verified instructional strategies suggested in ill-structured problem-solving literature, we did further literature review. Based on the principles of effective instructional strategies for ill-structured problem-solving skills suggested in the literature, we designed and developed the initial solution "Fathom-Ver1", which was equipped with ER cognitive tools. Study 2 was done to identify the challenges that students face while performing ER skills using Fathom-Ver1. The RQ asked in study 2 is:

RQ2- What are the difficulties faced by novices in performing ER skills using Fathom-Ver1?

Participants and research method: We used the exploratory study to understand the student behaviors in terms of how well students are using the design features of Fathom-Ver1 in performing the ER skills, and the difficulties faced by them. Study 2 was done with 49 undergraduate second-year computer engineering students. Students interacted with Fathom-Ver1 for almost 2 hrs, followed by student perception survey and student interviews.

Data Collected and analyzed:

- To measure the performance level of ER skills by students during the intervention, we evaluated student's responses while interacting with Fathom-Ver1.
- Student's perception survey was used to calculate the percentage of students who agreed or disagreed that the activities were useful in learning ER skills.
- Focus group interview data with four students was transcribed and analyzed to find the students perceptions on usefulness of the activities in learning ER skills and difficulties faced while performing the activities.

Results:

The student perception survey data showed that students perceived that the activities were useful in learning ER skills. However, based on the ER skills performance score, the student's behavior observed during interaction with Fathom-Ver1 and student interview data, we found that they faced difficulty in understanding and performing some of the activities.

3.3.2 DBR Cycle 2

Fathom-ver1 was designed to understand how effective are the learning activities and ER cognitive tools in performing ER skills in the context of solving a software design problem. Based on the difficulties faced by students during study 2, we further decided to improve the scaffolding structure. In the second cycle of DBR, the research goal was on improving the scaffolds to allow learners to perform and learn ER skills effectively. The solution was redesigned based on the design principles of providing cognitive and metacognitive support to facilitate learning and transfer of ER skills. Study 3 was done to evaluate the effectiveness of the redesigned solution Fathom-Ver2. The research question investigated was:

RQ3- How effective the design features of Fathom ver2 are in the learning of ER skills?

The specific RQs are written to investigate each design feature in detail:

- RQ3.1 How effective is Fathom-Ver2 in helping students to perform and learn ER skills?
- RQ3.2 How effective is structured versus a semi-structured sequence of learning activities in performing ER skills?

- RQ3.3 How effective is the feedback mechanism (system generated, self-evaluated) in monitoring and controlling the ER skills?
- RQ3.4 How effective are reflection activities in enabling the student to analyze the ER skills in a new problem?

Participants and research method- Study 3 was done with 52 undergraduate second-year computer engineering students. The research method was experimental pretest-posttest. The problem-solving literature argues between both structured versus open learning environment for learning of problem-solving skills (Xun 2013; Tselios, 2002). To investigate the effectiveness of both in our learning context, we controlled the variable: structured versus a semi-structured sequence of learning activities. The pretest-postest was administered to measure the learning gain of ER skills before and after the intervention.

Data Collected and analyzed-

- The student's performance of ER skills was measured by evaluating the student's artifacts generated during pretest, posttest, and intervention
- Students perception survey on the usefulness of Fathom-Ver2 in learning ER skills.
- Log data to analyze the student's behavior while interacting with Fathom-Ver2.

Results-

RQ 3.1. Significant improvement was seen from pretest to posttest scores in quality of the problem formulation, solution quality, and justification.

RQ 3.2. No significant difference between the experiment and control group.

RQ 3.3. Based on log data analysis, we found that most of the students were revising their responses based on system generated formative feedback in the activity-"understand the problem." However, in other activities, the feedback was generated based on self-evaluation, which is not effective as only 24 % of students were revising their responses.

RQ 3.4. The reflection activity based on analyzing a new problem was not effective as only 10% of students were able to evaluate correctly.

The results showed that students were effectively doing ER skills during the intervention and in the posttest problem, but were not good at self-evaluating and reflecting on the ER skills and were not able to take appropriate action to improve the skill to the desired level.

3.3.3 DBR Cycle 3

The feedback from DBR cycle 2 showed that novices are weak in self-evaluation of skills, and they seek expert feedback to evaluate and regulate their performance. In the third cycle of DBR, the solution was redesigned to scaffold metacognitive skills of evaluating the ER skills and taking appropriate action to improve and reflect by comparing with peer's responses. Study 4 was done to evaluate the overall improved pedagogical features of Fathom-Ver3. The research question asked was:

RQ4- How effective are the design features of Fathom-Ver3 in learning of ER skills at cognitive and metacognitive level?

The three sub-research questions (RQ4.1 to RQ 4.3) were investigated to evaluate the effectiveness of each design feature:

- RQ4.1- Does the cognitive tools of Fathom-Ver3 effective in learning of ER skills?
- RQ4.2- Are the pedagogical features in Fathom-Ver3 effective in learning ER skills?
- RQ4.3-How effective is the metacognitive activities: formative feedback and collaboration activity in enabling students to monitor and evaluate their ER skills?

Participants and research method- Study 4 was done with 50 undergraduate second-year computer engineering students. The research method was experimental pretest-posttest, and the control variable is cognitive tools. Since the Fathom learning activities were designed around ER cognitive tools, we used experimental-control group study to validate the effectiveness of ER cognitive tools in learning ER skills. The pretest-postest was administered to measure the learning gain of ER skills before and after the intervention.

Data Collected and analyzed-

- The student's performance of ER skills was measured by evaluating the student's artifacts generated during pretest, posttest, and intervention

- Students perception survey on the usefulness of Fathom-Ver3 in learning ER skills.
- Log data to analyze the student's behavior while interacting with Fathom-Ver3.
- Focus group interview to find the students perceptions on usefulness of the activities in learning ER skills.

Results

RQ4.1. The significant gain in ER skill scores of experiment group over the control group in the posttest. This shows that the ER cognitive tools were effective in learning ER skills.

RQ4.2. Log data analysis showed that the high performers were effectively using the prompts, hints, worked examples, collaboration, and feedback.

RQ4.3. The analysis of log data showed that the system generated feedback and peer review activity was enabling 80 % (most) of the students to identify the gaps and redo the activity. Further analysis showed that students who scored high were actively seeking the help provided in the environment, and based on feedback and peers collaboration evaluated and improved their ER skills.

In the following chapters, the DBR cycle1,2 and 3 are discussed in detail.

Chapter 4

DBR Cycle 1- Problem Analysis

To achieve our research goal: "teaching-learning of ER skills to novices," it is essential to understand the level of scaffolds to be designed in the learning environment for novices to learn ER skills. As a first step towards achieving our research goal, the problem analysis in the first DBR cycle is divided into two sub-goals:

1. To assess the level of the novice in solving software design problem using ER skills.
2. To review various teaching-learning techniques for ill-structured problem-solving thinking skills.

In this section, we will first discuss the preliminary study performed to understand the level of the novice in applying ER skills. Next, literature is reviewed to identify various teaching-learning techniques for ill-structured problem-solving skills. Both the study and literature review will inform us about the learning conjectures and design conjectures to be addressed in the design and development of our TELE for teaching-learning of ER skills in the context of software design problem-solving.

4.1 Study 1- Understand the level of novices in applying ER skills.

The goal of this preliminary study was to address our first sub-goal: assess the level of novices in applying ER skills while solving software design problems.

The research question we will be investigating in this study is:

RQ1- What is the level of novices in applying ER skills while solving a software design problem?

1. **Study design and participants**

We did a field study with 40 undergraduate computer engineering students. The participants were selected based on purposive sampling techniques, as the pre-requisite is to have students who have completed the data structures and algorithm course. The research study design is a single group exploratory study with two phases, as shown in figure 4.1.:

Phase 1. Students solve a software design problem with no prompts,
Phase 2. Students solve the same software design problem with question prompts to trigger ER skills.

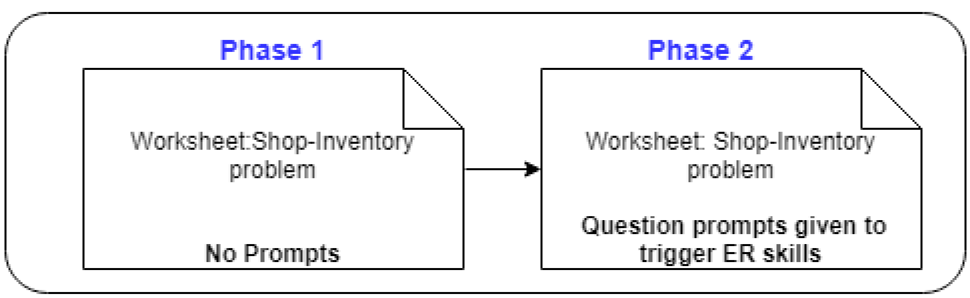


Figure 4.1. Research study1 design

The exploratory study design was used for two purposes:

i. To know how well students can solve software design problem without any support.
ii. To understand how question prompts are effective in triggering ER skills while solving a software design problem. Question prompts were used to enable learners' to think and execute cognitive processes to systematically expand-reduce in problem-solution spaces.

2. Procedure

The procedure for the study is as follows:

Phase1: The participants were given a shop-inventory problem to be solved on a paper:

The local automobile retail shop sells parts for different car models- oil, filters, brakes, and batteries. The shop owner notices that if any part is getting out of stock, then there is a risk of losing customers. So he wants to have a software developer to create an inventory control program that tracks the quantity of all the parts and creates a report of the parts that needs to be ordered, so there is minimal risk of items getting out of stock. Each night the program creates a report of the parts whose quantity is below a certain value so that they can be ordered to bring the stock levels back up to the right number.

Come up with multiple possible solutions by using appropriate data structures and operations for solving the above problem. Justify which solution is most efficient.

The students were asked to write multiple solutions using appropriate data structure and algorithms and select the optimal solution and justify. The participants were given 30 minutes to solve the problem.

Phase 2- Immediately after completing the first worksheet, the students were given the second worksheet to solve the same shop inventory problem with question prompts, as shown in figure 4.2. The participants were given 30 minutes but took almost 45 minutes to solve the problem.

Worksheet Activity of solving Open Problem	
Open Problem: *The local automobile retail shop sells parts for different car models...*	
1. Understand and analyze the problem:	
1.1	**List all the entities and actors [Nouns]:** entities are the existing real things or objects,

	for example, person, organization, table, etc. To list entities look for nouns in the above problem
1.2	**List the data generated from the perspective of each entity listed in 1.1:** data is the information like customer entity will have data: customer-id, name, age.
1.3	**List all the operations that are performed on the above data from the perspective of each entity listed in 1.1:** operations are the actions performed on the data set like customer places an order or cancels an order.
1.4	**What is the requirement in the above problem:** the requirement is the task to be achieved in above-stated problem.
1.5	**Identify the data (listed in 1.2), needed to solve the above requirement:**
1.6	**Identify the operations out of the list in 1.3, needed to solve the problem:**
2. Problem-solving:	
2.1	**List the desirable Data Structures that can be used to solve the above problem:**
2.2	**For each DS, give solution to solve requirement given in 1.4 using data and operations identified in 1.5 and 1.6 respectively:** **Solution 1:** Data Structure and data: Operations and algorithm: **Solution 2:** **Solution N:**
2.3	**Identify the efficient solution based on constraints (space, time, complexity) and requirement in the problem. Justify.**

Figure 4.2. Worksheet with ER related question prompts

The prompts were designed to systematically guide student's problem-solving process towards ER skills in both problem and solution space. In problem space, the prompts 1.1 to 1.3 in figure 4.2 were directing the student to think of the problem space by listing all entities and its corresponding data items and operations involved in the problem. The prompts 1.4 to 1.6 were used to reduce the problem space by explicitly writing the requirements and identifying the data and operations to achieve the stated requirements. In solution space, the ER prompts 2.1 and 2.2 are designed to direct students to expand solution space by listing all alternative data structures and algorithms, generate multiple solutions, and then reduce (prompt 2.3) by evaluating against the constraints and requirements applicable to the given problem, select and justify.

3. Data sources and analysis

As the purpose of the study was to understand how students can solve a software design problem and assess the level of ER skills, we collected and evaluated the student's worksheets solved in both the phases using the rubric shown in table 4.1.

Table 4.1. Rubric to assess ER skills

Worksheet steps	ER skills	Levels of skill demonstrated during intervention and pre-posttest against questions asked.		
		3: High	2: Medium	1: Low
1: Understanding the problem				
Expand Step 1.1, 1.2, 1.3	Understand the problem as a whole	Listed all possible entities and operations	Listed some possible entities and operations	Listed few entities and operations
Reduce Steps 1.4, 1.5	Identify constraints	Most of the constraints and assumptions were identified correctly	Some of the constraints were identified correctly	The constraints were identified but partially correct.
	Selection of solution component	Most of the components were selected correctly	Some of the components were selected correctly	The components were partially identified correctly
2: Problem-solving				
Expand Steps 2.1, 2.2	Generated alternative solutions	Student was able to generate >= 3 ideas/solutions	Student was able to generate 2 ideas/solutions	Student was able to generate 1 ideas/solutions
	Solution quality	The ideas are clearly and correctly explained at a more detailed level by showing the data representation, operations, input, and output data of each operation, and steps to perform the operation.	The ideas are clearly and correctly explained at a more abstract level by showing the data representation, operations, input, and output data of each operation, without the steps to perform the operation.	The ideas are not clearly explained.
Reduce Steps 2.3	Pros and cons analysis	The limitations and advantages of most of the attributes of the concepts analyzed effectively	The limitations and advantages of some of the concepts analyzed	The limitations and advantages of few concepts analyzed

	Justified selected solution	Justified the selected solution based on most of the constraints and assumptions and explained using evidence.	Justified the selected solution based on some of the constraints and assumptions.	Selected the solution by without considering the constraints or assumptions

The rubric is created to assess expand-reduce skills in both problem and solution spaces. The assessment criteria are based on the ER sub-skills derived from the ill-structured problem-solving literature discussed in chapter 2. The assessment criteria used in problem space is derived from various problem-solving literature (Adelson, 1985; Ackoff, 1979; Dennis, 1999; Ellspermann, 2007) which suggests techniques to expand the problem space by understanding the problem as a whole from the perspective of various stakeholders involved and then reduce the problem into subproblems. Similarly, in solution space the assessment criteria are derived from creative problem-solving (Basadur, 1990; Liu, 2004; Oslapas, 1993) and engineering design (Howard, 2008; Madhuri, 2015; Cooperrinder, 2008;) literature which proposes various divergent and convergent thinking techniques to expand by generating alternative ideas/solutions and reduce by evaluating and selecting single solution based on selection criteria and constraints. The assessment of each sub-skill is done at three levels: high (score: 3), medium (score: 2), and low (score: 1).

In phase1 worksheet, the students were asked to generate multiple solutions using appropriate data structures and algorithms and justify which solution is most efficient. The phase1 worksheet was assessed in terms of the number of solutions generated by a student, the solution quality, and ability to justify using the rubric shown in table 4.1.

In phase2 worksheet, the prompts were given to direct student's thinking to systematically expand-reduce in both problem-solution spaces, as shown in figure 4.2. The student's responses for each step in worksheet were evaluated using the rubric shown in table 4.1.

In problem space if the learner is able to expand by identifying all the entities and operations from the perspectives of all stakeholders involved in the system, then he/she is given three (high) marks, if a few are missed then two marks are given and if only a few are identified then one mark is given. The reduce skill is the ability to identify the data and operations needed for solving the given problem and assessed based on correctness, completeness, and the level of abstraction achieved. In solution space, the ability to expand is assessed by the ability to generate all possible alternative solutions. The quality of the solution is assessed on the correctness (selecting appropriate data structure and algorithm), and completeness in terms of achieving the stated goal. The ability to evaluate, select, and justify based on the constraints and requirements are assessed as given in the rubric table. The construct validity of the rubric was done by a research scholar having computer engineering background from the educational technology department.

Both the worksheets (no prompts and with prompts) were assessed using the rubric, and the average score of ER skills obtained are shown in table 4.2. In phase1, the students were directed to generate alternative solutions, select and justify the optimal solution. Thus, the problem space sub-skills and pros and cons analysis of solutions are not evaluated for phase1 worksheet, as shown in the second row of table 4.2. In phase 2 worksheet, prompts were given to systematically expand-reduce both problem-solution spaces and were assessed, as shown in the third row of table 4.2. The t-test was done to compare the average scores with and without prompts for each skill (last row in table 4.2).

Table 4.2. Average ER skill scores with and without prompts

N=40	**Problem space**			**Solution space**			
	Understand the problem	**Identify constraints**	**selection of solution component**	**Generated alternative solutions**	**Solution quality**	**Pros and con analysis**	**Justification**
Without prompts (Average)	-	-	-	1.15	1.44	-	0.02
With prompts (Average)	1.78	1.78	1.52	1.71	1.42	0.44	0.47
T-Test (p-value)	-	-	-	0.00	0.88	-	0.00

4. Results and discussion

Results showed significant improvement from phase 1 (without prompts) to phase 2 (with prompts) in generating alternative solutions ($p<0.01$), and justification ($p<0.01$). However, significant improvement was not seen in solution quality ($p=0.88$).

In worksheet without prompts, most of the students directly jumped to solving the problem without an initial exploration of the problem. They could write only one solution in detail in the form of an algorithm. They were not able to abstract data and operations at a higher level of abstraction, as shown in sample worksheet in figure 4.3. The students also had difficulty in writing multiple solutions, which led to weak reasoning and not able to justify how their solution is better.

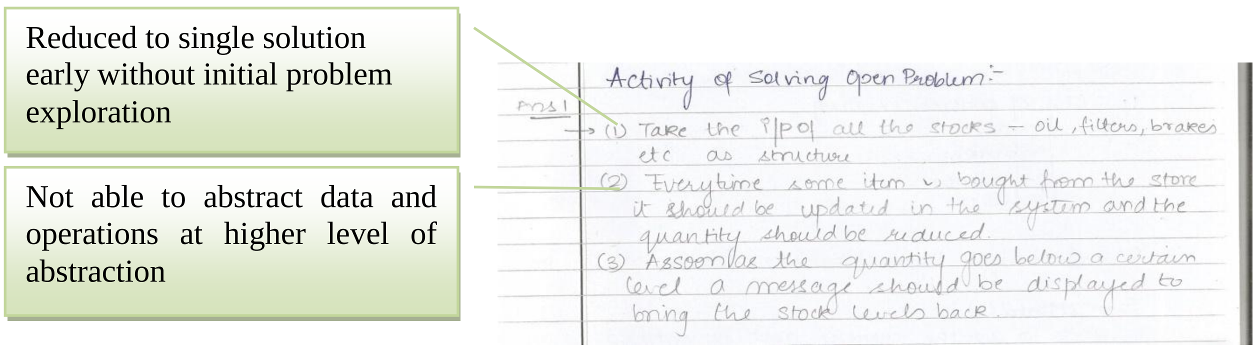


Figure 4.3. Sample worksheet without prompts

In worksheet with prompts, students were able to solve the problem by initially exploring the problem space; formulating the data and operations at an abstract level; exploring the solution space by thinking of alternative solutions; evaluating and justifying the selection as shown in figure 4.4. However, there was no improvement in solution quality in terms of correctness or completeness as the students were not efficiently doing ER skills.

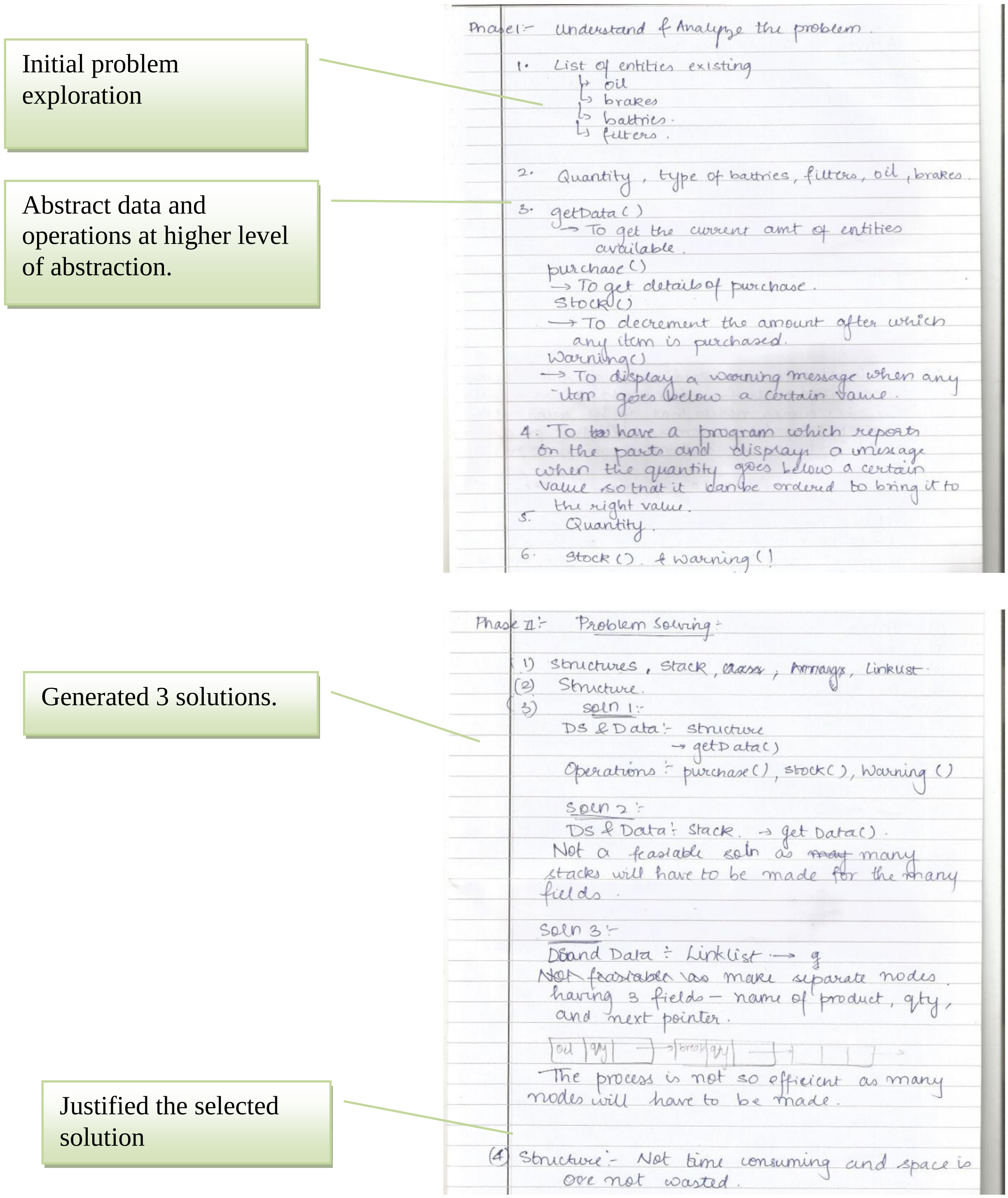


Figure 4.4. Sample worksheet of a student with prompts.

The RQ1-"What is the level of novices in applying ER skills while solving software design problem?" is answered based on the worksheet scores without prompts and with prompts.

The scores in the worksheet without prompts showed that most of the students were generating only one solution (Solution quantity: mean =1.17). They prematurely reduced to a single solution, and solutions written were incomplete (solution quality: mean= 1.44/3). They were not able to justify how the selected data structure or algorithm is better than other alternatives. The low scores on solution quality show that students are not able to apply ER related cognitive processes and one of the reason may be that students at undergraduate level studying in the second year do not have prior experience in solving ill-structured design problems. After solving the problem in phase 1, students were asked to reflect on problem-solving by answering the following questions:

Reflection on problem-solving

1. *How many alternatives did you think of before writing your solutions?*
2. *Could you identify the most efficient solution among multiple solutions?*

For question 1, 27(out of 31) students said that they thought of two or more solutions, but had difficulty in understanding and solving the problem. For the second question, one of the student feedback was, "*I could identify the most efficient solution, but not able to write, so I attempted an easy solution."* Overall, students had difficulty in understanding the problem, breaking the problem into sub-problems, generating alternative solutions, and identify the selection criteria to select an efficient solution.

The worksheet with prompts helped students to understand the problem-solving strategy in terms of applying ER skills in problem and solution space. They were initially exploring the problem space and able to formulate and abstract the solution in terms of data and operations to be solved and were explicitly generating multiple solutions, evaluating and justifying the selected solution. However, no improvement in solution quality and low scores in ER skills show that only prompts as the scaffold is not very effective by itself in triggering the appropriate cognitive processes related to ER skills to solve design problem effectively.

5. **Reflections**

The implications of the novice study are:

1. Students at the undergraduate level do not have prior experience and ER skills to solve the design problem, which is ill-structured.
2. Tend to jump to solution without initial exploration of problem-solution space which affects the problem formulation and solution quality.
3. Prompts are helping in understanding the problem-solving strategy, but are not efficient in triggering appropriate ER cognitive processes to solve the design problem efficiently.

To get insights on how to scaffold novices in doing and learning ER skills efficiently, we did further literature survey. Since ER skills are more relevant in ill-structured problem-solving, we reviewed teaching-learning techniques proposed in ill-structured problem-solving literature across disciplines. The literature survey questions (LQ) asked is:

LQ1.1. What are the scaffolding techniques for teaching-learning of ill-structured problem-solving thinking skills?

We will now discuss the literature towards answering LQ1.1.

4.2 Scaffolding techniques for teaching-learning of ER skills

To achieve our research goal of teaching-learning of ER skills to novices, we have reviewed ill-structured problem-solving literature to identify effective teaching-learning techniques and frameworks for problem-solving thinking skills, followed by a discussion on how these techniques are used for teaching-learning of ER skills.

4.2.1 Technology Enhanced Learning of Thinking Skills (TELoTS)

TELoTS is an initiative of Educational Technology department of IIT Bombay, where the teaching and learning of thinking skills like algorithmic thinking, design thinking, scientific modeling, problem-posing estimation, etc. are promoted.

TELoTS framework (Murthy et al., 2016) recommends steps to be followed by the researchers and designers to design an effective smart learning environment for developing thinking skills; the steps are as shown in figure 4.5.

1. Characterize the thinking skill
a. Identify the competencies of the chosen thinking skill
b. Choose an appropriate topic to develop the thinking skill
c. Create learning objectives
d. Consider assessment measures

2. Design the learning activities
a. Decide appropriate instructional strategies
b. Identify technology features to realize the instructional strategies
c. Create a sequenced set of learning activities

3. Build the smart learning environment
a. Design the modules
b. Design the user interface
c. Implement the system

Figure 4.5. Steps and guidelines of the TELoTS framework (Murthy et al., 2016)

We have followed the TELoTS framework for developing TEL environment for teaching-learning of ER thinking skills. The first step is the characterization of ER thinking skills by identifying the sub-skills, topic, and context to teach ER skills and its assessment. The ER skills are defined in chapter 2, and the context chosen to develop ER skills is software design problem-solving in data structures and algorithm course for undergraduate engineering students. The rubric for the assessment of ER skills was discussed in the previous section 4.2.

Next step in the framework, is to design the technology-enhanced learning environment by designing learning activities using appropriate instructional strategies and technology features. In the next section, we will discuss the learning technologies and instructional strategies suggested in the literature for ill-structured problem-solving.

4.2.2 Cognitive and metacognitive skills in ill-structured problem-solving

The thinking skills used for ill-structured problem-solving are often associated with cognitive and metacognitive skills used during the process of problem-solving (Xun, 2004; James, 1994) and self-regulatory learning (Bennert, 2013).

In the context of problem-solving, cognitive skill is defined as an ability to solve a problem successfully, which involves manipulation and use of the elements of the domain knowledge. Metacognitive skills are higher level skills that enable the use of cognitive skills efficiently to obtain the greatest possible success. It is defined as the ability to evaluate, monitor, control the cognitive skills, and justify the reasoning process (James, 1999) as shown in figure.4.6.

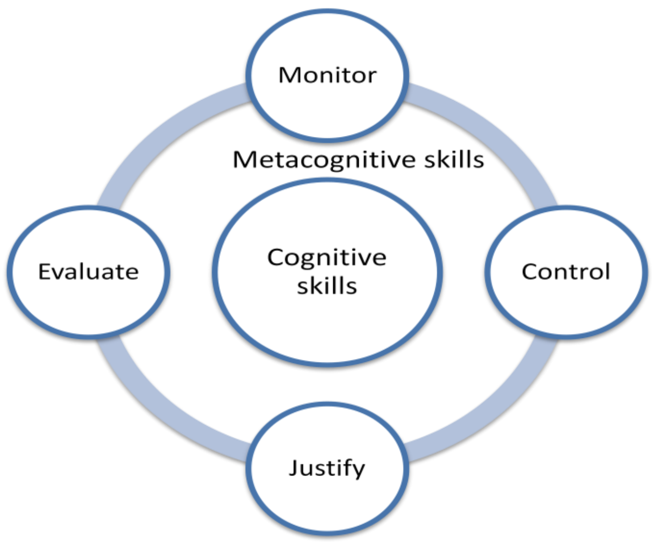


Figure 4.6. Cognitive and metacognitive skills.

The ill-structured problem does not have well-defined goals and sub-goals, has multiple solutions, and the selection criteria for evaluating solutions are unknown (Jonassen, 2000). To solve such problems, the designers engage in the cognitive processes such as understanding the problem-solving domain, planning the problem-solving strategy, identifying problems and sub-problems, generating alternative solutions, and making decisions and justification. The experts are engaged in applying metacognitive skills of evaluating the cognitive processes to monitor the gap between the current goal and the desired goal and take control action plan to improve.

However, the novice usually lacks these metacognitive skills to evaluate and monitor their progress, due to lack of prior knowledge and experience. Research suggests that the novice need to be trained explicitly in using both cognitive and metacognitive skills needed to solve ill-structured problems effectively (James, 1999; Xun, 2004; Redish, 2008; Cooperinder, 2008).

4.2.3 Scaffolding techniques for teaching-learning of thinking skills

In this sub-section, we have reviewed various scaffolding techniques suggested in the literature for teaching-learning of thinking skills.

i. Cognitive and Metacognitive prompts

Prompts are classified into cognitive and metacognitive prompts. Cognitive prompts directly support a student's processing of domain-specific information, for example, recall or elaboration of the concepts. Metacognitive prompts support student's monitoring and control of their information processing through metacognitive and reflective activities like goal setting, orientation, planning, evaluation of solutions, etc. Thus the metacognitive prompts allow learners to apply metacognitive skills adequately during the problem-solving process. Research and empirical evidence shows that cognitive and metacognitive prompts significantly improve cognitive and metacognitive skills using systematic instructions (Bennert, 2013; Xun,2013).

ii. Design principles for effective metacognitive instructions

The design principles derived from research on effective metacognitive instruction are (Bennart, 2013). -

1. The metacognitive activities should be embedded in the domain-specific instructions and should not be taught separately,

2. The application and usefulness of the instructed metacognitive strategies should be explained.

3. Allot sufficient training time to implement and automate the metacognitive activities just learned.

iii. Framework to scaffold ill-structured problem-solving using question prompts

The pedagogical approach to scaffold ill-structured problem-solving using question prompts is based on the constructivist approach stated by Vygotsky's (1978) scaffolding theories on the zone of proximal development(ZPD) (Roblyer, 2012). The ZPD is the difference between novice and expert in performing certain skills, and the novice can learn the skills only when scaffolding is provided by a teacher or peer with a higher skill set (Xun 2004).

Ge and Land (Xun, 2004) proposed a framework to scaffold ill-structured problem-solving using question prompts and peer interactions, mediated with technology. Question prompts include procedural prompts, elaboration prompts, and reflection prompts, each of which serves different cognitive and metacognitive purposes-

1. Procedural prompts are designed to guide learners step by step through the entire processes of a specific problem-solving task (e.g., problem representation, developing solutions, constructing arguments, and monitoring and evaluation). For example, the procedural prompts designed for problem representation are: *identify the problematic situation and define the problem*.
2. The elaborative prompts are designed to prompt students to articulate their thoughts and elicit explanations. For example, prompts used for problem representation are: *what are the facts that suggest the problem, why are the problems occurring?*
3. The reflective prompts help students to engage in self-monitoring and evaluation process during problem-solving. Reflection prompts (e.g., *What is our plan? Have our goals changed? To do a good job on this project, we need to* ..) are designed to encourage reflection on a meta-level that students do not generally consider.

iv. Direct and indirect metacognitive support

The instructions for metacognitive support can be direct or indirect. Direct support teaches metacognitive support directly to the students. The strategies are explained, and students practice the use of strategy. In indirect support, the learning environment is open-ended and promotes the use of strategies without explaining them explicitly. Direct support in the form of extensive training is needed for students lacking metacognitive competence (Friedrich, 1992).

Empirical studies have shown that minimal guidance is ineffective when learners have low prior knowledge and may have negative results when students acquire misconceptions or incomplete or disorganized knowledge (Roblyer, 2012; Kirschner, 2006).

Summary

The summary of instructional strategies for teaching-learning of thinking skills reviewed is shown in figure 4.7.

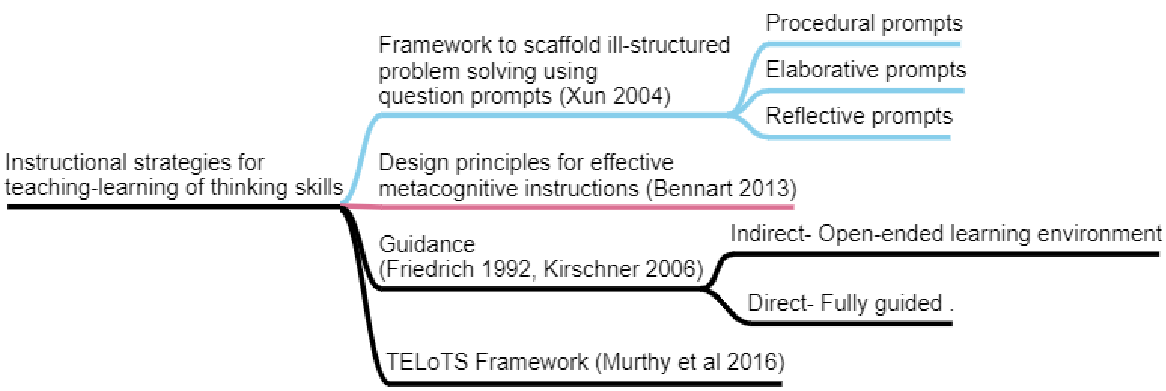


Figure 4.7. Summary of instructional strategies for teaching-learning of thinking skills

We incorporated the appropriate design implications for the design and development of our intervention on developing software design skills and sub-skills identified in the previous section. The design decisions of the intervention are explained in the next sub-section 4.3.

4.3 Design decisions for teaching-learning of ER skills

We have identified various ER cognitive tools used by experts (section 2.3) and various scaffolding techniques for teaching-learning of thinking skills (section 4.2) in ill-structured problem-solving literature. The selection of ER cognitive tools and teaching-learning technique for our design of a learning environment is explained below:

4.3.1 Pedagogical approach for teaching-learning of ER skills

The pedagogical approach of our learning environment is grounded on the constructivist approach stated by Vygotsky's (1978). For novice learners, to learn ER skills which are higher order thinking skills, the externalized support or scaffolding is necessary to trigger and learn cognitive and metacognitive skills (Xun, 2004).

Based on the study1 results (discussed in section 4.1) which show that novices lack the ability to apply ER skills effectively and empirical evidence (Roblyer, 2012; Kirschner, 2006) which states that stronger guidance is necessary for both effective learning and transfer of skills, we base our pedagogical approach on providing fully guided and structured way of teaching ER skills.

The empirical studies by Xun (2004, 2013) shows that framework to scaffold ill-structured problem-solving using prompts is effective in guiding students thinking towards specific aspects of cognitive and metacognitive skills which otherwise they lack to exhibit on their own. We propose to design our learning activities based on the design principles suggested in the framework to scaffold ill-structured problem-solving (Xun, 2004; Bennart, 2013).

4.3.2 Mapping of ER cognitive tools to software design problem-solving

The selection of ER cognitive tools is based on its suitability for solving a software design problem and students prior knowledge. The ER cognitive tools mapped to the software design process in both problem and solution spaces are explained in detail and summarized in figure 4.8.

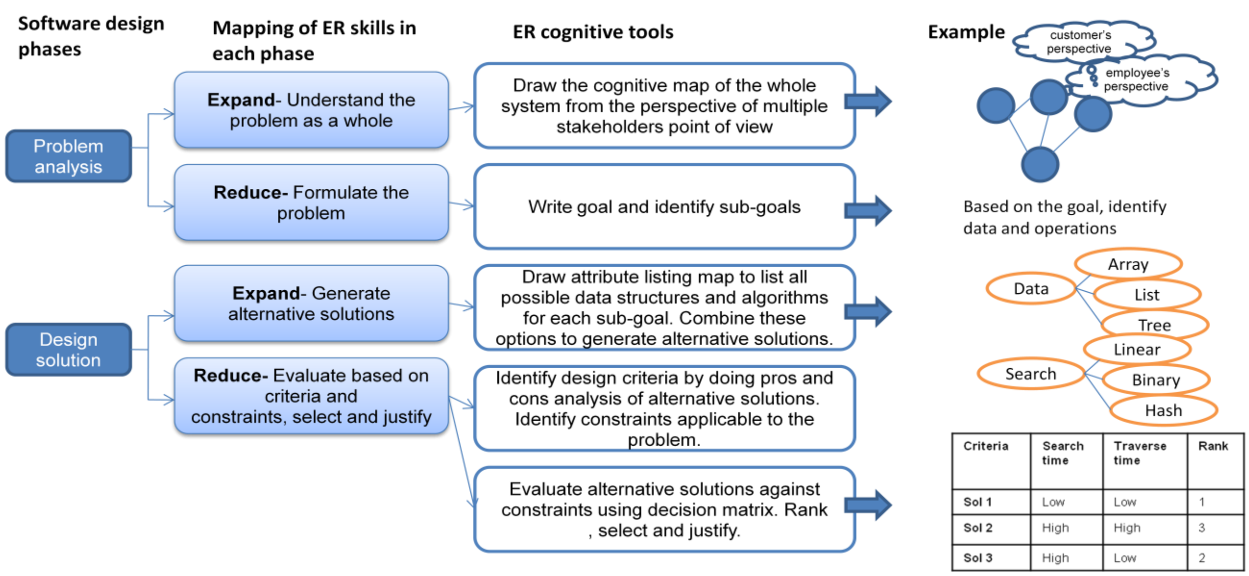


Figure 4.8. Mapping of ER cognitive processes and tools to software design.

ER cognitive tools in problem space:

Drawing cognitive map- The software design problem involves designing software for domains like the bank, library, supermarkets, etc. Such domains consist of multiple stakeholders, entities, and complex interactions among them. To aid students in understanding such a complex problem, drawing of a cognitive map from the perspective of various stakeholders will be more suitable than other techniques. This step will allow the learner to understand and visualize the problem as a whole in terms of different components involved in the problem and interaction among them. Without this step, the learner would tend to focus on few related components in depth rather than think of a broader range of components of the overall problem (Kunene, 2005) which eventually reduces the solution space search.

Identify goals and sub-goals- After the initial problem exploration, based on the goal, learners can abstract the sub-goals (data and operations) to be performed from the components and interactions drawn in the previous step.

ER cognitive tools in solution space:

Attribute listing map- The attribute listing map is used to expand the solution space by listing all possible design options for each sub-goal. The design options are combined to generate alternative solutions. By making learners generate alternative solutions explicitly, the risk of premature selection of solution without considering alternative options will be reduced, and the possibility of designing an optimal solution is increased.

Pros and cons analysis table- The pros and cons analysis table will allow learners to identify design criteria to evaluate alternative solutions. We have seen in study1 that novices tend to select the solution that is easy to implement rather than selecting one based on the constraints applicable to the given problem. This step will help learners to explicitly list all possible design criteria from the alternative solutions generated and identify which one is the important constraint based on the sub-goals to be achieved. This step will allow the learner to think back and forth between solution and problem space.

Decision matrix and justification box- The decision matrix is used to evaluate alternative solutions against the constraints, rank them, and select a solution. The learner has to give an explicit justification of how the selected alternative is better than other alternatives. This will allow the learner to reflect and assess the effectiveness of the solution in achieving the stated goal and constraints and explain the same.

The expert studies discussed in section 2.3 state that professionals may not systematically follow a top-down approach in terms of explicitly generating all problem statements and alternative solutions (Guindon, 1990; Tang, 2010). They frequently go between problem and solution space and constantly assess if the goal and constraints are achieved by simulating the solution using various scenarios. Novices tend to jump to solution space early and are anchored to the initial solution throughout the design process, without considering other alternative options or co-evolving problem-solution spaces (Carman, 2007; Tang, 2010). This behavior of novice is due to the lack of metacognitive skills to evaluate and monitor the current state of the solution against the desired state and take appropriate control action plan. The studies have shown that the ability to evaluate, monitor, and control the cognitive skills depends on the prior experience and expertise of the problem solver in solving similar problems (Adelson 1985; Guindon 1999). However, the process of initial problem exploration, formulation of sub-goals, and explicit generation of alternative solutions improves the quality of design, especially for novices (Adelson 1985; Tang 2008,2010). Hence in our learning environment, pedagogy involves guiding learners through initial problem exploration, formulating a large number of problem statements before moving to solution space and generating all possible alternative solutions before reducing to a single solution.

In the overall DBR cycles shown in figure 4.9, we are in DBR cycle 1- problem analysis (green color). In the next chapter, the detailed solution design and its evaluation are discussed.

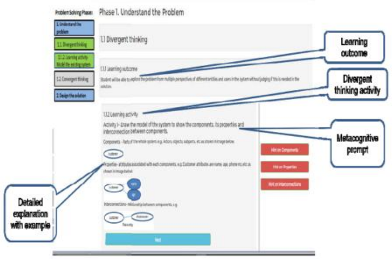
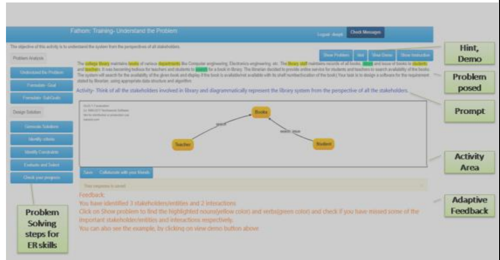
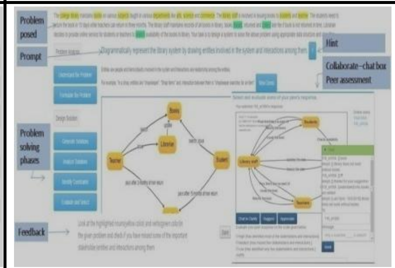
	DBR Cycle 1	DBR Cycle 2	DBR Cycle 3
Problem analysis	1. To assess the level of ER skills in novice. 2. Literature survey to identify the ER cognitive tools and learning techniques, Study 1(N=40), exploratory study	To redesign the TELE with an improved level of scaffolding and introduce reflection activities.	To revise the metacognitive activities for effective monitoring and controlling of ER skills.
Solution	Fathom-Ver1	Fathom-Ver2	Fathom-Ver3
			
Evaluation	**Study 2-** 49 engineering students, exploratory study **Results-** Students were having difficulty in doing ER	**Study 3-** 49 students, experimental pre-posttest. **Results-** Students could do ER skills but faced difficulty in evaluation of ER skills.	**Study 4-** 50 students, experimental pre-posttest. **Results-** Students could do ER skills and monitor ER skills.
Reflection	Prompts are not effective if learners do not have prior experience in ER cognitive skill. Improve the design by adding more scaffolds.	Novice learners are not able to self-evaluate to monitor and control ER skills. Improve metacognitive support.	The pedagogical support in terms of cognitive and metacognitive support and cognitive tools are effective in doing and learning of ER skills.

Figure 4.9. Overall design-based research cycles applied in this thesis

Chapter 5

DBR Cycle 1- Solution Design and Evaluation

In this chapter, we will discuss the solution design of the first version of a technology-enhanced learning environment named Fathom-Ver1, for teaching-learning of ER skills and its evaluation study.

5.1 Design of Fathom-Ver1

The objective of Fathom-Ver1 is to design a learning environment with pedagogical features to support novice learners in performing ER skills. Based on the problem analysis and literature survey discussed in the previous chapter, the design conjectures derived for Fathom-Ver1 are:

1. Enable learning of ER skills in the context of solving a software design problem.

2. Provide structured guidance by designing learning activities to direct learners' thinking towards applying ER skills systematically.
3. Design learning activities using prompts with additional support and ER cognitive tools.

The design features of Fathom-Ver2 are given below-

5.1.1 Software design problem-solving in data structure course

We attempt to develop the ER thinking skills in students through software design problem in data structure course, as shown in figure 5.1. The software design problem posed is a bank problem and the requirements to be achieved are stated at a very broad level as: *"Design software system for a bank to allow customers to check their account balance quickly."* We have scoped the design towards using appropriate data structure and algorithms. The problem is not well-structured as the data items and operations (sub-problems) are not clearly defined in the problem. The solution space consists of searching and selecting appropriate data structure and algorithms for each data item and operations based on the selection criteria.

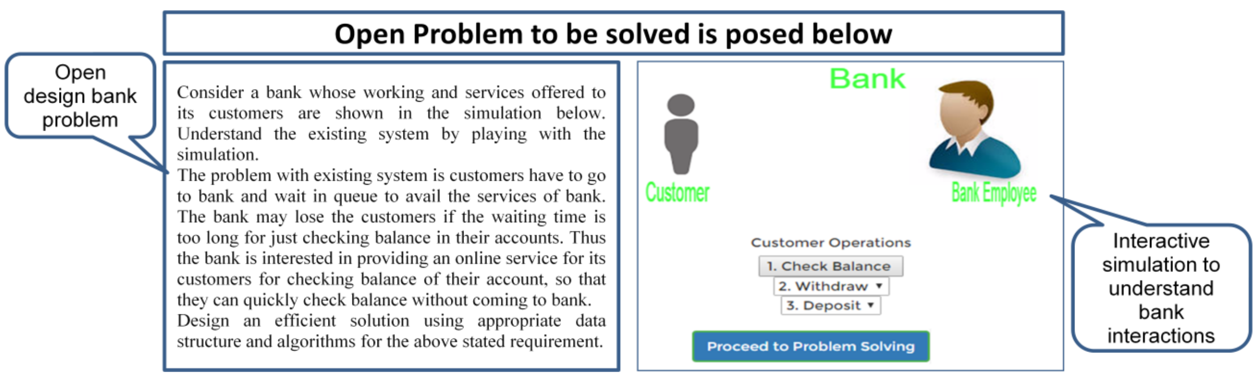


Figure 5.1. The real-life design problem with simulation posed to students in Fathom-Ver1

The design decision of solving the problems in data structures course is based on the principle that it is effective to develop thinking skills in a particular domain rather than teaching the skills in general (Bennart, 2013). Data structures course is selected as choosing appropriate data structure and algorithms is one of the important skills in software design and designer has to make design decisions based on the criteria relevant to the given problem (Tang, 2010).

The real-life scenario posed in the problem is presented in the form of animation, in which the bank scenario is animated to show different operations- withdraw, deposit, and balance inquiry in the bank and the interactions between customer and bank teller. Since the learners lack the experience of real-life problems, the animation will help in expanding the problem space by understanding the problem from the perspective of multiple stakeholders.

5.1.2 Structured guidance and learning activities

After learner reads the problem and plays the animation, as shown in figure 5.1, the activities to be performed are listed on the left-hand side of the screen, as shown in figure 5.2.

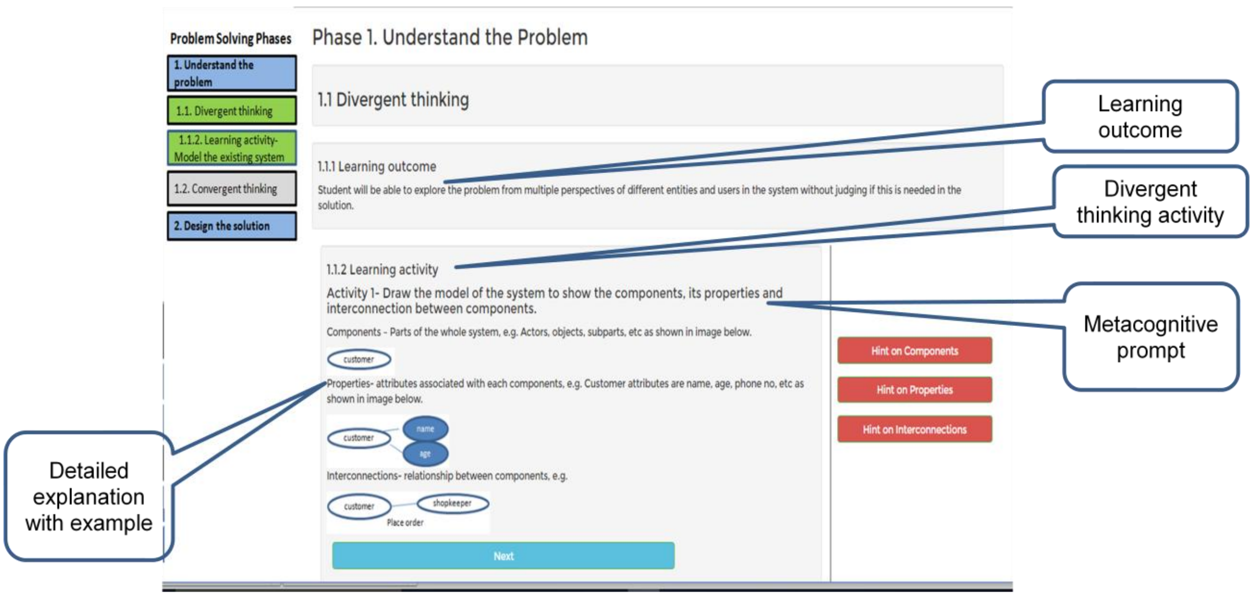


Figure 5.2. Learning activity supported with cognitive prompt and detailed explanation.

The activities listed are:

1. Understand the problem
2. Formulate problem
3. Generate solutions
4. Evaluate solutions
5. Select solution

These learning activities are designed to direct the learner's thinking towards performing ER skills systematically in both problem and solution space. Each activity is explained in detail:

1. **Understand the problem:** In problem space, the first activity: "understand the problem" (figure 5.2.), is designed to explore the problem from multiple perspectives of different

entities and users in the system. The learning outcome is shown first to create the importance of the activity. Then the prompt is displayed -*"Draw the model of the system to show the components, its properties, and interconnection between components"* supported with a detailed explanation of concepts like components, properties, and interconnections and illustrated with examples as shown in figure 5.2.

2. **Formulate problem:** In this activity, the students are prompted to identify the goal and formulate the problem by redrawing the model by adding or deleting some of the components and interconnection. The text box is provided in the system to allow users to write the sub-goals, that is the operations to be performed by the software to achieve the stated goal.
3. **Generate solutions:** In the solution space, the first activity is designed to expand solution space by generating alternative solutions followed by activities to reduce the solution space by evaluating and selecting a solution. The *Generate Solution* activity is supported with animation to illustrate the use of attribute listing techniques to generate multiple designs of a table, e.g., a round table with three legs, square table with four legs, etc. Then learners are prompted to transfer this learning to generate multiple solutions using various data structures and algorithms. The text box is provided to list all the alternative solutions generated by the user.
4. **Evaluate solutions:** After the learner has generated solutions, the next activity is to evaluate solutions. The evaluation can be done efficiently only if the learners can identify the evaluation criteria and constraints in the given problem, a process which is difficult for novices (Jonassen, 2006). To help students in evaluating solutions effectively, the evaluation activity is divided into two steps: first, the pros and cons table pre-populated with the solutions generated by the user is displayed, as shown in figure 5.3, and the learner is prompted to write advantages and disadvantages of each solution and identify the common criteria (e.g., execution time)"; and second step is to identify the constraints, i.e., the mandatory criteria to be achieved in the given problem.
5. **Select solution:** The last activity is *Select Solution using Decision Matrix*. The decision matrix tool is used to allow learners to evaluate solutions against multiple constraints and select the optimal one. In Fathom, the decision matrix is populated with the alternative solutions in the first column and, the constraints in the first row, and the students are

prompted to rank the solutions according to the identified constraints. In the end, the learner has to justify their selected solution by explaining how the selected solution is optimal.

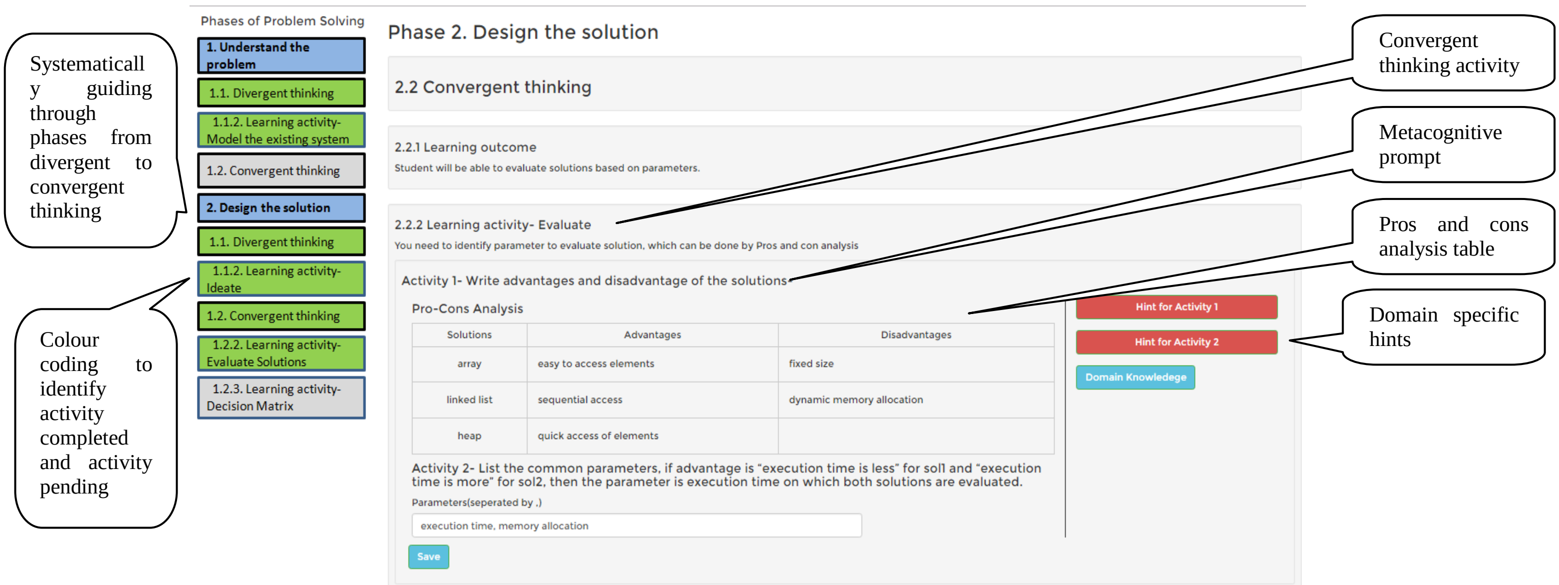


Figure 5.3. The design features- prompts, systematic guidance, color coding, ER tools, and domain-specific hints in Fathom-Ver1.

5.1.3 Domain-specific hints

The domain-specific hints are designed for each activity to assist students in recalling domain-specific concepts required to complete the activity. These hints are designed at various levels, from general to specific to support learners with different levels of prior knowledge. This design decision is based on the principle stating that the cognitive load of the student is reduced by providing flexible domain knowledge (Bennart, 2013).

5.1.4 Overall design of Fathom-Ver1

The overall design features of Fathom-Ver1 are shown in figure 5.4.

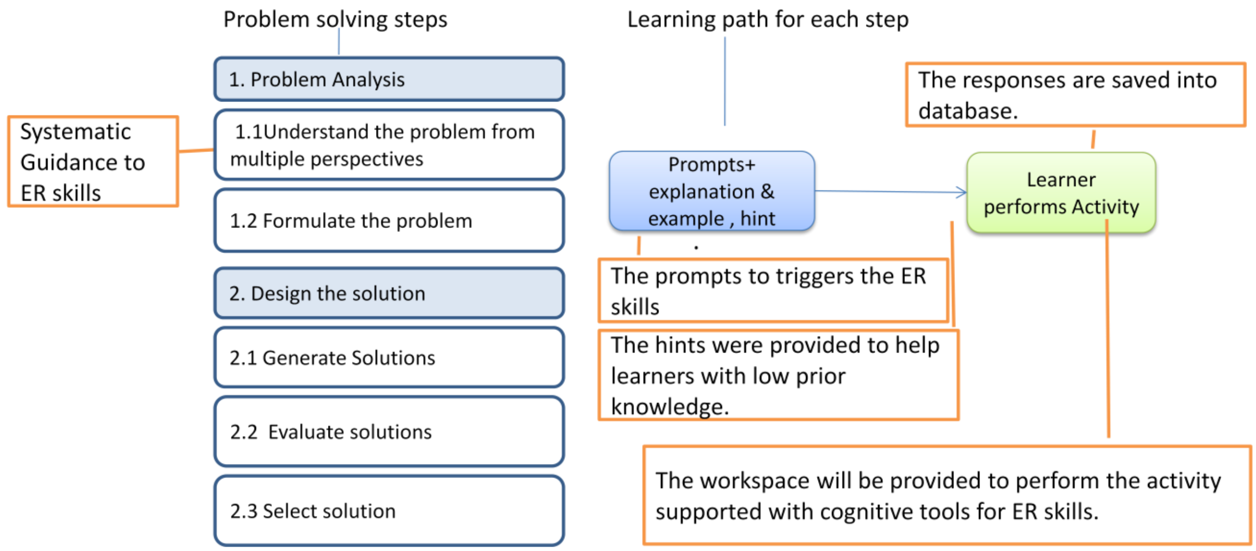


Figure 5.4. Design of learning activities in Fathom-Ver1

The learning environment of Fathom-Ver1 is designed to guide learners to systematically expand-reduce in both problem and solution space. The learning activities are scaffolded with prompts, explanation of new concepts with examples, animations, hints, and ER cognitive tools. The overall structure is to allow learners to expand problem space by drawing a model of the system, followed by problem formulation activity. Next, expand solution space by drawing attribute listing map and generate solutions, followed by evaluation activity with pros and cons analysis tables and decision tables preloaded with solutions. The text boxes are used in activities for writing goals, problem definition, solutions, and justification. The response of the student in each step is stored in the database. These responses are fetched and loaded if the student has already answered and wants to go back to the previous steps to refer to their answers.

In this version, the drawing tools to draw model and attribute listing map are not incorporated into the system, and the students were instructed to draw on paper.

5.2 Evaluation- Study 2

The goal of this study was to investigate the effectiveness of Fathom-Ver1 in performing ER skills. The research question asked was:

> RQ2.1- How effective are the novices able to perform the ER skills using the learning activities designed in Fathom-Ver1?

5.2.1 Study design and participants

We did a field study with 49 undergraduate computer engineering students. The participants were selected based on purposive sampling techniques, as the pre-requisite is to have students who have completed the data structures and algorithm courses. The research study design was a single group exploratory study, as shown in figure 5.5.

Figure 5.5. Research study-2 design.

The single group exploratory study was used for three purposes:

i. To investigate the effectiveness of prompts with additional scaffolds: explanation, example, hint, and animations, in performing the activity.
ii. To understand how students are using the ER cognitive tools while performing the activity.
iii. To assess the performance level of students in ER skills with Fathom-Ver1.

5.2.2 Procedure

The overall steps, followed by the students, were:

1. Register and log in into the system
2. After login, the first opening page provided an introduction to ER skills, learning objectives, and overall learning path in Fathom.
3. Problem-solving
    a. The software design problem (Bank problem) was posed with animation.
    b. They were systematically guided through the steps of problem-solving phases. In each step first, the learning outcome was shown, and then the prompts were shown to perform the activity.

4. At the end of the training, a survey was taken using five points Likert scale to reflect on how well the prompts helped in performing the activities and two open-ended questions were asked to express their likes and dislikes about the system.
5. An interview of four students was taken to elicit what they learned, how the activities helped in applying ER skills, and their confidence in applying the ER skills to solve similar problems.

Students interacted with Fathom-Ver1 for almost 2 hrs.

5.2.3. Data sources

As the purpose of the study was to investigate the effectiveness of Fathom in performing activities and performance level of ER skills, the data collected were:

a. Learner-generated artifacts (responses of each learning activity) during interaction with Fathom-Ver1.

b. Student perception survey on the usefulness of learning activities. The survey questions asked were:

Five-point Likert scale questions:

i. The activity of drawing the model of the existing system helped in understanding the system better.
ii. The activity of writing the goal and remodeling the system helped in formulating the problem.
iii. The activity- ideate, helped in generating multiple solutions.
iv. The activity of identifying parameters based on pros and cons analysis helped in analyzing solutions.
v. The activity of identifying parameters based on pros and cons analysis helped in analyzing solutions.
vi. The activity of identifying constraints and decision matrix helped in evaluating and justifying the selected solution.

Open-ended questions:

i. Write what did you liked and disliked the most in the system?

c. Semi-structured focus group interview of students to elicit how the design features of Fathom-Ver1 were useful in performing and learning ER skills.

5.2.4 Data Analysis

The data analysis was done to analyze the performance of ER skills using rubric, correlation of scores across activities, student perception survey and interview.

1. **Assessment of ER skills using rubric:** The student artifacts generated during the intervention were evaluated using a rubric (table 5.1). The rubric used in study 1 was redesigned to assess the performance of ER skills using the cognitive tools in both problem and solution space, for example, the sub-skill associated with expanding problem space is to draw the complete model of the system and sub-skill associated with reduce is to write a goal and formulate the sub-goals. The scores given are either high (3 marks), medium (2 marks), and low (1 mark) based on the performance of the students.

Table 5.1. Rubric to evaluate ER thinking skills.

Activity rating levels	High(score-3)	Medium(score- 2)	Low(score-1)
Phase 1. Understand the problem			
Expand			
Draw model of the existing system	Most of the actors, entities, properties, and interrelations are shown in the model.	Some of the actors, entities, properties, and interrelations are shown in the model.	The actors, entities, properties, and interrelations are not identified clearly.
Reduce			
Identify the goal	Most of the goals given in the problem are identified and written	Some of the goals are identified and written clearly	The goals that are written are not explained clearly
Formulate the problem	The data items and operations needed to achieve the goal were identified and written clearly.	Some of the data items and operations needed to achieve the goal were identified and written clearly.	The data items and operations were not correctly identified or not clearly written
Phase 2. Design the solution			
Expand			
Generate solutions	Most (4-7)of the possible solutions with data structure and operations were listed.	Some (3-4) of the possible solutions with data structure and operations were listed.	One or two solutions was listed.
Reduce			

Pros and cons analysis	Most of the advantages and disadvantages of the solutions were identified.	Some of the advantages and disadvantages of the solutions were identified.	The advantages and disadvantages of the solutions were not identified correctly.
Identify criteria	Most of the common criteria were identified.	Some of the common criteria were identified.	The criteria were incorrect.
Evaluate solutions	The evaluation table was correctly filled for all parameters and solutions.	The evaluation table was correctly filled for some parameters and solutions.	The evaluation table was not correctly filled.
Identify constraints	The constraints in the given problem were identified correctly.	The constraints were partially correct.	The constraints were not correctly identified.
Decision table	The table was correctly filled for all solutions and constraints and ranked appropriately.	Partially correctly filled for all solutions and constraints and ranked appropriately.	Not filled
Justification	The justification for selecting the optimal solution had good explanation and clarity based on goal and constraints identified.	The justification given had average explanation and clarity based on goal and constraints.	The justification given was not based on either goal or constraints.

2. **Student Perception Survey and interview:** The student perception survey was taken to reflect on how well the cognitive prompts helped in performing the ER thinking activities, on the five points Likert scale (strongly agree, agree, neutral, disagree, strongly disagree). Two open-ended questions were asked to express their likes and dislikes about the system. The mean and standard deviation of the students' rating was calculated to know the tendency of students towards the usefulness of the prompts in doing cognitive activities. Content analysis of the open-ended question's responses was done and summarized.

3. **Correlation Analysis:** The correlation analysis among the scores of each activity was done to find the correlation between activities to understand the relationship between the performances of ER sub-skills. This may help us in taking further design decisions towards improving the learning of ER skills.

5.2.5 Results

The results are discussed as follows:

1. **ER skills performance scores:** The average scores and standard deviation of the ER skills are shown in table 5.2. The scores show that most of the students scored at medium level in the activities: draw model, generate solutions and evaluate solutions, while in all other activities: formulate goal and sub-goal, identify constraints, decision matrix, and justification, the average scores were below medium.

Table 5.2. Mean scores of ER thinking skills

	Phase 1. Understand the problem			**Phase 2. Design the solution**						
	Expand	**Reduce**		**Expand**	**Reduce**					
	Draw model of the system	Identify the goal	Formulate the problem	Generate solutions	Pros and cons analysis	Identify parameters	Evaluate solutions	Identify constraints	Decision table	Justification
Average Score (maximum score-3)	**2.08**	1.85	1.51	1.91	**2.20**	**2.02**	**2.02**	1.88	1.49	1.63
Standard Deviation	0.49	0.91	1.00	0.57	0.58	0.63	0.95	0.88	0.84	0.95

2. **Student perception survey and interview:** The results of student perception survey are shown in table 5.3. The mean and standard deviation of the students' rating against the effectiveness of the prompts in doing the activities are calculated. Most of the students agreed that the learning activities helped perform the ER skills, and the whole process helped in designing a better solution and justify.

Table 5.3. Mean scores of students rating in the perception survey

Survey Question	The activity of drawing the model of the existing system helped in understanding the system better.	The activity of writing the goal and remodeling the system helped in formulating the problem.	The activity-ideate, helped in generating multiple solutions.	The activity of identifying parameters based on Pros and cons analysis helped in analyzing solutions.	The activity of identifying constraints and decision matrix helped in evaluating and justifying the selected solution.	The overall process of problem-solving helped in designing a better solution and justify why the solution is most optimal.
Mean (5-strongly agree to 1-strongly disagree)	4.11	4.06	4.09	4.15	4.09	4.15
Standard Deviation	0.52	0.60	0.65	0.59	0.69	0.47

The student responses to the open-ended questions on likes and dislike about the system are listed with the frequency of students giving the same response in parenthesis:

Likes:

i. Great way to approach any problem (1)
ii. A good/easy way for an optimal decision (3)
iii. Easier to conclude (1)
iv. Helped in understanding the concepts properly (1)
v. Innovative (1)
vi. Easy to use and has a simple UI(4)
vii. Efficient in teaching how to solve a problem(2)
viii. Good learning(1)
ix. Develops analyzing and designing skills (1)
x. Better analysis of problem statement(1)
xi. Helps in understanding the problem definition(1)
xii. Interesting(2)
xiii. Helps in building multiple solutions (2)
xiv. Step by step approach to the given problem(1)
xv. It gave knowledge on the flow of how to go about the problem statement given(1)
xvi. Multiple ideas and solution to a specific problem(1)

Dislikes

i. Time consuming (2)
ii. Very confusing (1)
iii. More simulations required to demonstrate more examples (2)
iv. Lack of problem statement (1)

v. Cannot save our data properly (1)
vi. User interface (2)
vii. Animations can be better (1)
viii. Not everything was clear what was taught to us (1)

The analysis of the open-ended questions and interview data showed that students appreciated that the whole process helped in solving problem, generating multiple solutions, and the activity helped in developing analyze and design skills. Some students said that they wanted animation and examples to demonstrate the use of ER skills in every activity, and some found the whole process to be time-consuming.

3. **Correlation Analysis:** The Pearson correlation between the ER thinking activities showed a significant correlation between activity-generate solutions and activity- pros and cons analysis ($p= 0.002$) and justification ($p= 0.04$). This shows that students who are good at expanding solution space also tend to be good at evaluating and selecting a solution. In our learning environments, we should take measures to ensure that students are doing enough thinking in terms of exploring the current system and generating alternative solutions.

Putting all results together, we now answer the RQ- "How effective are the design features of Fathom-Ver1 in performing the learning activities?"

- The scores of ER skills show that in some of the activities like understand, formulate problem, and evaluating using decision matrix and justification, most of the students scored low; thus these activities have to be redesigned with improved scaffolds. Based on the students interview and our observations during the intervention, we found that students had difficulty in understanding the activity to be performed. For example, the prompt in the activity, *understand the problem:* "*Draw the system model by identifying the components, its properties and interactions among them"* was difficult for a novice to understand the activity to be performed even though explanation and examples were given. One of the reasons may be at that the prompt was at a higher level and not tied to the domain in which the problem is solved.
- The cognitive tools were not effectively used, which resulted in low performance in ER skills. One of the reason was that the drawing of the entity-interaction diagram and

attribute listing map was on paper and students were finding it difficult to keep track of the same in subsequent steps.

- During the interview and survey, students perceived animations and examples as useful in understanding the activity, and they wanted such kind of support in all the activities.

5.3 Reflection

The plan of action for the next cycle of DBR from the aspects of improving the design of the Fathom-Ver1are:

i. Improve the learning activities by redesigning prompts such that they are easy to understand, more specific, and related to the domain in which the problem is solved.
ii. Provide more support, in the form of animations, simulations, explanation, and examples for all activities.
iii. Incorporate assessment of the activities in the form of feedback to allow students to evaluate and improve their skills (Narciss, 2013). These can be in the form of peer interaction, self-evaluation by comparing with expert solution after the activity is performed, system generated feedback or reflective prompts at the end of each activity to help students in monitoring, evaluating and assessment to regulate their thinking (Xun, 2013; Bannert, 2013).
iv. Improve user interface.

Till this chapter, we have completed a discussion of all the phases of DBR cycle 1, as shown in figure 5.6. In the next chapter, the phases of DBR cycle 2 are discussed in detail.

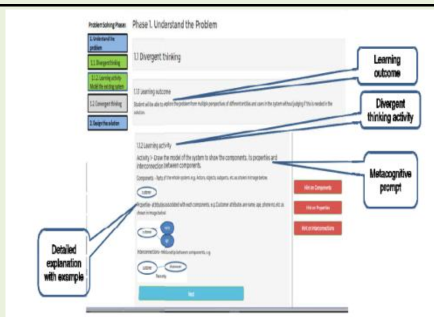
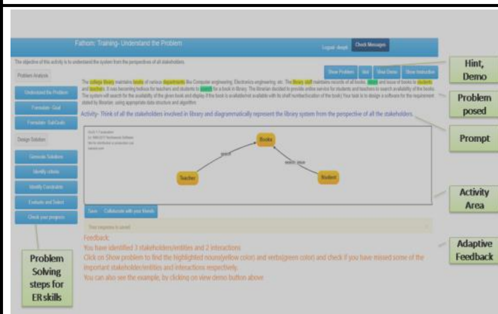
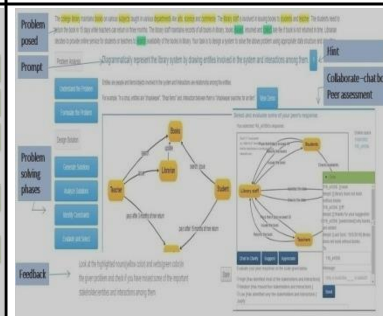
	DBR Cycle 1	DBR Cycle 2	DBR Cycle 3
Problem analysis	1. To assess the level of ER skills in novice. 2. Literature survey to identify the ER cognitive tools and learning techniques, Study 1(N=40), exploratory study	To redesign the TELE with an improved level of scaffolding and introduce reflection activities.	To revise the metacognitive activities for effective monitoring and controlling of ER skills.
Solution	Fathom-Ver1	Fathom-Ver2	Fathom-Ver3
			
Evaluation	**Study 2-** 49 engineering students, exploratory study **Results-** Students were having difficulty in doing ER	**Study 3-** 49 students, experimental pre-posttest. **Results-** Students could do ER skills but faced difficulty in evaluation of ER skills**.**	**Study 4-** 50 students, experimental pre-posttest. **Results-** Students could do ER skills and monitor ER skills.
Reflection	Prompts are not effective if learners do not have prior experience in ER cognitive skill. Improve the design by adding more scaffolds.	Novice learners are not able to self-evaluate to monitor and control ER skills. Improve metacognitive support.	The pedagogical support in terms of cognitive and metacognitive support and cognitive tools are effective in doing and learning of ER skills.

Figure 5.6. Overall design-based research cycles applied in this thesis

Chapter 6

DBR Cycle 2: Improved pedagogical support for teaching-learning of ER skills

In previous DBR cycle, we saw that novices are not skilled to apply ER skills effectively in software design problem-solving. In a learning environment Fathom-Ver1, novices faced difficulty in performing some of the activities in which the pedagogical support was minimal in terms of high-level prompts, example, or animation.

Research suggests that for effective learning of complex problem-solving skills, the learners needs to be engaged at both cognitive and metacognitive level (James, 1999; Xun, 2013; Bannert, 2013). The expert designers exhibit metacognitive behavior in the form of continuously assessing the solution against the test cases and constraints to identify the gaps and take appropriate control action plan to improve (Adelson, 1985; Tang, 2010).

The goal of DBR cycle 2 is to revise the pedagogical features of the activities to improve the performance level in doing of ER skills and regulate the learning of ER skills by engaging novices in both cognitive and metacognitive activities. In the next section, the literature is reviewed to identify the principles for effective scaffolding for novices for doing and reflecting on the ER skills.

6.1 Problem Analysis- Effective cognitive and metacognitive scaffolding techniques

The literature survey was done with the focus to answer the following questions.

LQ 2.1 What are the effective cognitive scaffolds to do ER skills effectively?

LQ2.2 What are the effective metacognitive scaffolds to learn and improve ER skills?

The answers to LQ 2.1 and LQ2.2 are discussed below.

6.1.1 Cognitive scaffolds for problem-based learning

Jonnasen (2011) suggested that learning to solve different kinds of problems from well-structured to ill-structured requires different kinds of instructional methods and cognitive scaffolds.

To support the learner to solve problems, some combination of worked examples, structural analogues, case studies, prior experiences, alternative perspectives, and simulations may be accessed to help learners interpret and solve the problem.

One of the building blocks of problem-based learning environment (PBLE) is cases. A case may be real-life examples or a case study of a solved problem. A case may comprise anything from a sentence level example to a complex, multi-page, or video-based case study. As instructional components, most of these cases are provided to help learners make sense of the problem to be solved.

The main components of Problem-Based Learning Environments (PBLE) are:

i. ***Problem to solve***- The focus of any PBLE is the problem to solve. According to problem-based learning principles, learning is anchored in an authentic problem to solve. The problem provides the purpose for learning. For example, we used a software design problem for learning ER skills.

ii. ***Worked examples***- Worked problems are usually used with well-structured problems to enable learners to model the process of solving the problem. However, it is unknown how effective worked problems are for very ill-structured problems.

iii. ***Structural Analogue***-Learning to solve well-structured problems can also be supported by providing analogous problems for students to compare with the problem to solve. When students compare the problem to solve with structurally similar problems, they gain more robust conceptual knowledge about the problems.

iv. ***Case Studies***- In case studies, students study a problem that was previously experienced. Students analyze the situation and processes and evaluate the methods and solutions. In most case studies, students are not responsible for solving the problems, only analyzing how others solved the problems and engaging in what-if thinking.

v. ***Simulations***-Simulations are environments where components of a problem are manipulated by learners. When learners interact with the simulation, they can change the values of some (input) variables and observe the results on the values of other (output) variables. These exploratory environments afford learners the opportunities to test the causal relationships among factors in the problem. The feedback provided by the system confirms or rejects students' understanding of the relationships as represented by their mental models of the problem.

Jonassen (2011) has suggested that the cognitive scaffolds in the form of studying worked example, structural analogies, case studies, and simulations in relation to the problem to be solved enhances students' understanding of the problem and their abilities to solve ill-structured problems. The majority of research on problem-solving has focused on solving well-structured problems, and less on ill-structured problems. The techniques that work with learning ill-structured problem-solving are not empirically validated.

6.1.2 Feedback in online learning environments

Feedback is important in improving learning in various instructional contexts, including online learning environments. Feedback on learners' response informs the learner about their actual state of learning to regulate the further process of learning in the direction of the learning standards (Narciss, 2013). In tutoring, formative feedback helps the learner to become aware of any gaps that exist between their desired and their current state of knowledge or competencies. Narciss has provided Interactive Tutoring Feedback (ITF) model on providing formative feedback on the gap to regulate learners' learning process.

6.1.3 Reflection in action

In-action reflection is a metacognitive activity in which learner reviews his/her cognitive process or experience. This review process allows learners to identify deficiencies in their performance and specific skills that they need to improve on.

Learners can review their performance by self-assessment against a performance scale. The process of self-assessment is not an easy task, as learners' tend to over-estimate or under-estimate their performance. The self-assessment process can be improved by providing students with scaffolds to help to assess against the scale accurately.

Rivera-Gutierrez et al. (Rivera, 2016) proposed to use in-action reflections to develop interpersonal skills such as empathy towards patients, to medical practitioners. The learners interacted with virtual agents playing the role of a patient. During the interaction, the learners were prompted to self-assess their performance with regards to empathy. The study conducted with third-year dental students showed that the students were relatively accurate in their self-assessment of how empathetic they were to the virtual patient and were significantly more empathetic to the patient after their in-action self-assessment of their empathy.

6.1.4 Intelligent Tutoring System

Intelligent Tutoring Systems (ITS) provide an individualized learning environment for learning concepts in various domains like language, Maths, and Medicine. These are computer programs that model learners' psychological states to provide individualized instruction (Koedinger, 1997).

One of the popular ITS tutoring approach used was to develop a cognitive model of how students should solve problems and use this model to interpret the student problem-solving behavior and to guide the students through the problem-solving process. The cognitive model consists of production rules and is capable of producing any acceptable solution path for a problem. This model is supplemented with errors and misconceptions that students are observed to make. When students make errors during problem-solving, the cognitive model is used to interpret the error and provide appropriate feedback. Studies have shown that these tutors enabled students to acquire problem-solving skills in domains like geometry, algebra, and computer programming language (Anderson, 1992).

ITS model is popularly used to teach well-structured problems in which the solutions path are known and can be easily modeled into the system in the form of rules and misconceptions, for example, maths, physics, languages, etc. However, ITS is not used for tutoring ill-structured problems due to unknown solution path, uncertainty about correct/incorrect solutions, and limited knowledge on the kind of errors or difficulties faced by students.

Summary

In our learning environment, the focus is on learning ER skills in the context of software design problem-solving. The focus of problem analysis and literature review was to identify cognitive support to improve the understanding of the ER skills and metacognitive support to improve the ER skills by monitoring and controlling the cognitive tasks. The summary of the instructional strategies for cognitive and metacognitive support suggested in the literature on problem-based learning is shown in figure.6.1.

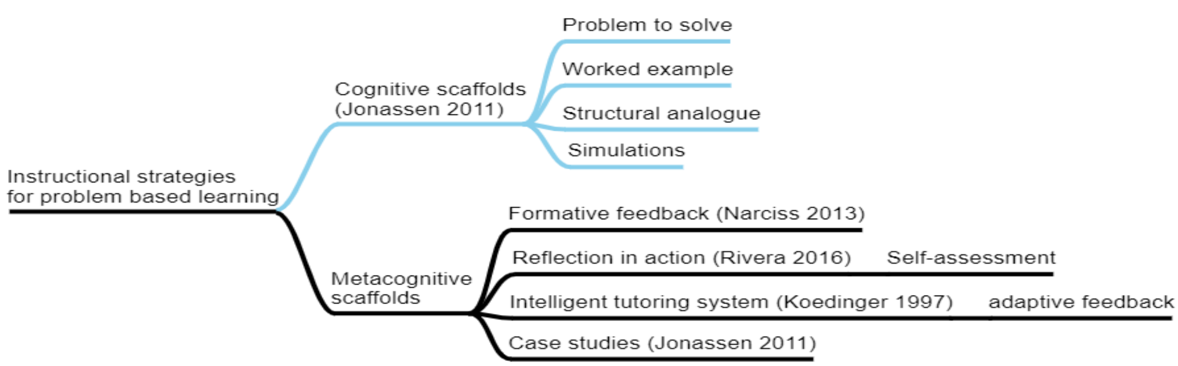


Figure 6.1. Cognitive and metacognitive support for problem-based learning

We have selected worked example as a cognitive scaffold to illustrate the use of ER skills in a similar software design problem, as it will help students to improve the understanding of applying ER skills. It will also aid students to understand the similarity of the two problems (case study and problem being solved) and compare with their problem-solving process.

For effective learning and transfer, it is necessary to engage learners in metacognitive activities towards monitoring and controlling their performance (Bennart, 2013). However, research has suggested that novices are weak at metacognitive activities due to lack of prior experience and skills (Adelson, 1985; Tang, 2008). In Fathom, the metacognitive scaffold is based on the in-action reflective framework (Rivera, 2016) in which we propose to use self-assessment to engage students in metacognitive activities. The detailed design of the revised version of Fathom-Ver2 is discussed in the next section.

6.2 Solution Design- Fathom-Ver.2

The design of Fathom-Ver2 is discussed by presenting the design conjecture, followed by a detailed explanation of design features and learning activities design for learning ER skills in Fathom-Ver2.

6.2.1 Design conjecture

The design conjectures for the revised version of Fathom-Ver2 are:

i. The prompts which are simple and specific to the domain will help learners to understand the activity better than general prompts. For example, the prompt in Fathom-Ver1: "*Draw the system model by identifying the components, its properties, and interactions among them.*"
   is redesigned to:
   "Diagrammatically represent the library system by drawing entities and interactions among them."
ii. Adding more scaffolds in the form of worked example across all activities will help learners to understand the skill better and improve the performance in doing ER skills.

iii. The learning environment equipped with ER cognitive tools and worked examples demonstrating the use of cognitive tools will help learners to use the cognitive tools more effectively.

iv. The system generated feedback and self-evaluation activity will enable students to monitor and control their ER skills during the activity and be able to analyze the case study by assessing and explaining how well the ER skills are applied in a new problem.

6.2.2 Design features

Fathom-Ver2 is designed with increased scaffolds in the form of cognitive and metacognitive activities. The system is structured to solve the software design problem in data structures with following learning activities designed to practice ER skills:

i. Problem Analysis
    a. Understand the problem
    b. Formulate the problem

ii. Design solution
    a. Generate solutions
    b. Analyze solutions
    c. Identify constraints
    d. Evaluate and select

The phases and sub-skills are listed in Fathom-Ver2 on left-side of the screen to scaffold learner to follow the structured learning path. The sub-skills are clickable button, and when selected, the learner is prompted to do the learning activities to perform the targeted sub-skill. The learning activities for each sub-skill consist of:

i. Prompt to guide the thinking process towards the targeted sub-skill and solve the training problem.

ii. Worked example illustrating the use of ER skills and tools to solve the structurally analogous problem. This will enable students to compare the problem to solve with a structurally similar problem and gain more understanding about problem-solving skills such as ER skills.

iii. Hints are provided for a detailed explanation about the skill with example.

iv. Feedback is generated based on the self-evaluation of the ER skill applied.

v. Post activity, reflection prompts are given to allow the learner to reflect on how much the activity helped to achieve the targeted skill. Learners are posed with a new solved problem, and the learner is asked to analyze and rate how well ER skills are applied. This activity ensures that learners reflect on their learning of the targeted sub-skill and apply it to analyze a new problem.

The design of the overall learning activities in Fathom is shown in figure 6.2.

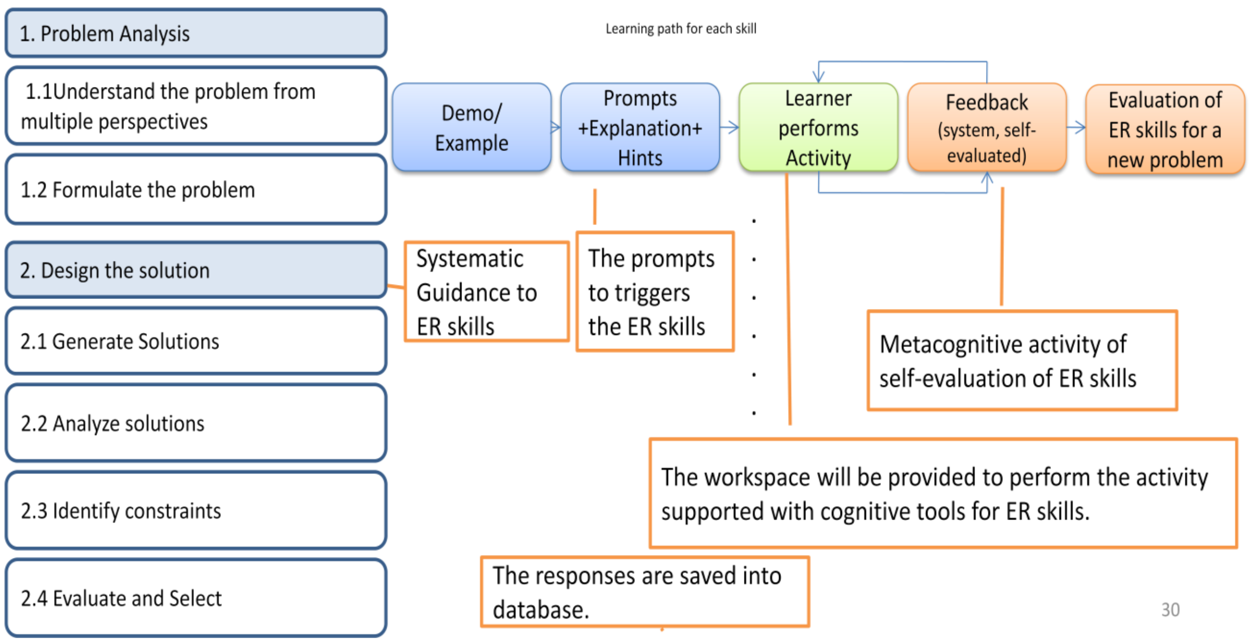


Figure 6.2. Design of learning activities in Fathom

The design enables the learner to see the overall path to be followed to solve the problem as shown in the left-hand side of figure 6.2. The learning activities are explained in detail in next sub-section.

6.2.3 Learning activities

After the learner has logged in to Fathom-Ver2, a training problem is posed. The problem posed is a library management problem, stated as- *"The college library maintains books on various subjects taught in various departments like arts, science, and commerce. The library staff is involved in issuing books to students and teacher. The students need to return the book in 15* *days while teachers can return in three months. The library staff maintains records of all books in the library, books issued, returned and collect a late fee if the book is not returned in time.*

Librarian decides to provide online service for students or teachers to search the availability of the books in the library. Your task is to design a system to solve the above problem using an appropriate data structure and algorithm".

Next, problem-solving steps are displayed on left-hand side of the screen, as shown in figure 6.3. The learning activity designed are:

1. Understand the problem activity

After learner clicks on the step-*understand the problem*, the activity page opens where the demo, hint, prompt, and activity area are displayed as shown in figure 6.3.

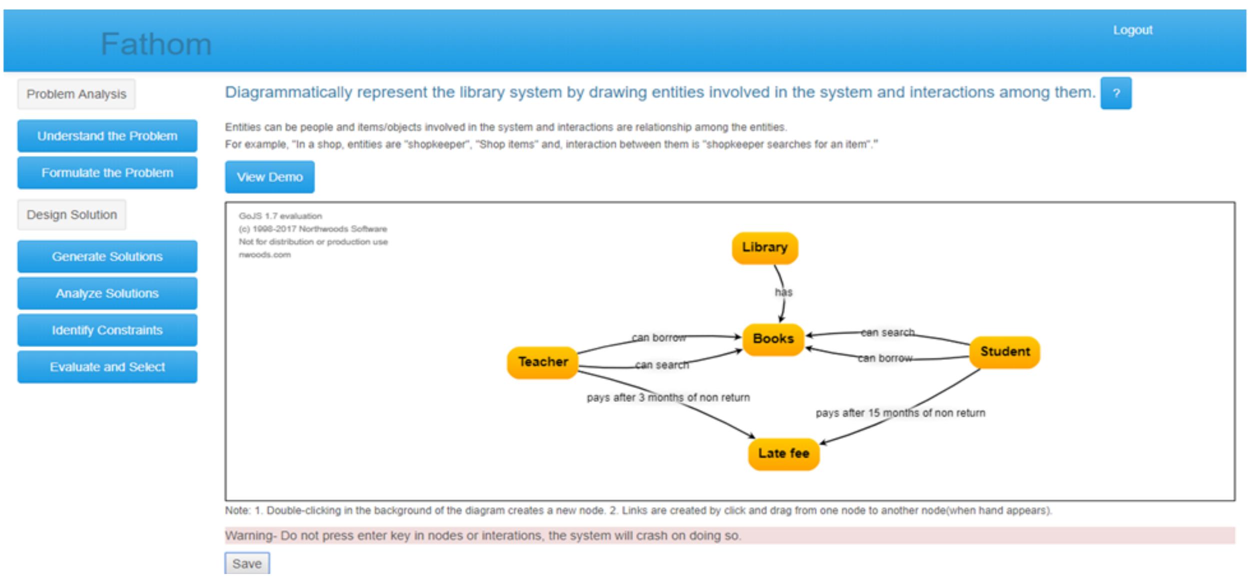


Figure 6.3. Activity page for step- *understand the problem*.

The prompt displayed is: *"Diagrammatically represent the library system by drawing entities involved in the system and interactions among them."* The hint button (?) gives a detailed explanation about the activity with example. The "view demo" button will play a video demonstrating the activity for an analogous problem: shop inventory problem.

The scaffolds are designed to direct the learner's thinking towards understanding the problem from the perspectives of stakeholders and entities involved in the system by drawing the cognitive map of the whole system.

Activity area

The activity area is designed to allow the learner to draw the entity-interaction diagram of the library problem. The nodes represent the entities or stakeholders, and the links between the entities represent the interactions among stakeholders and entities. The interactions are labeled to show the nature of interactions.

Feedback

After the learner clicks on the save button, the system will evaluate the response and generate feedback-

If most of the nodes and interactions have been captured then

the system generates positive feedback as *"Excellent, you have identified six entities and eight interactions,"* and system prompts to go to the next step.

Else if fewer nodes and interactions are captured then

1. the feedback generated is *"Look at the highlighted nouns(yellow color) and verbs(green color)in the given problem and check if you have missed some of the important stakeholder/entities and interactions among them."*

2. The system will highlight the nouns and verbs and will prompt the learner to add them, as shown in figure 6.4.

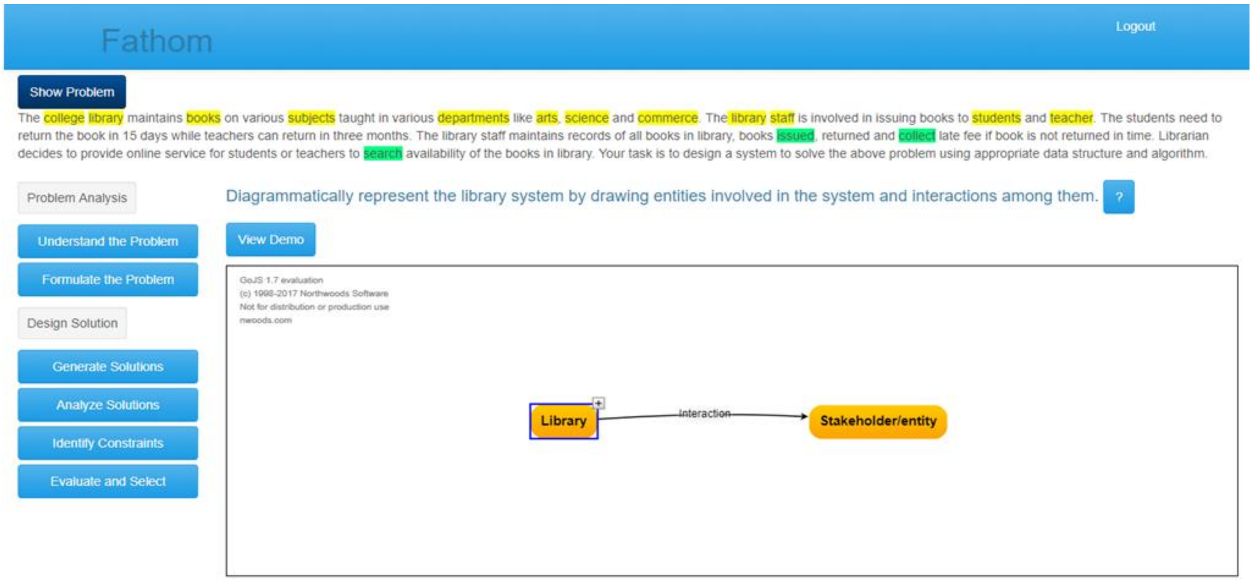


Figure 6.4. Feedback on the understand the problem activity

Reflection

At the end of the activity, the system shows reflection questions (shown below) for the activity: *understand the problem* to reflect firstly on how much the activity helped to achieve the targeted skill and secondly a new problem is posed, and the learner is asked to analyze and rate how the targeted skill is applied.

1. *"Identifying entities and interaction helped me to understand the working of the existing system from the perspectives of multiple stakeholders."*
    - *strongly agree,*
    - *agree,*
    - *neutral,*
    - *disagree,*
    - *strongly disagree.*
2. *If a new problem is given say, "Design banking software for customers to perform online transactions- checking account balance, money transfer, and bill payments." I will start solving this problem by-*

○ *Understanding the working of the bank from the customers' point of view before proceeding to solve the problem.*

○ *Understanding the problem from the perspectives of all stakeholders and entities involved in the bank system before proceeding to solve the problem.*

○ *Directly start solving the problem.*

○ *Others*

Justify why?

The reflection questions are asked to reflect on the skill that they just learned and to involve learners in a metacognitive activity of evaluating and rating a response to a new problem. This process of evaluating response for a new problem may make learner monitor his/her thinking.

2. Formulate the problem

The formulate problem activity page is, as shown in figure 6.5.

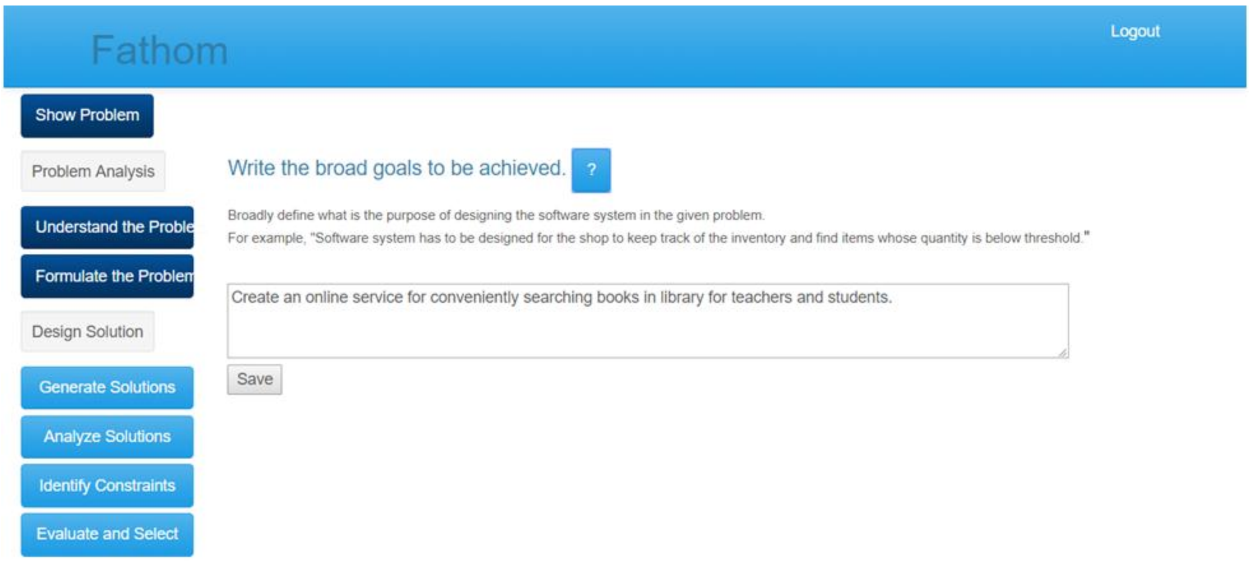


Figure 6.5. Activity- *Formulate the problem-Goal*

The *formulate problem* is divided into two parts: Goal and Sub-goals.

The prompt posed is: *"Write the broad goal to be achieved"* The hint shows explanation with an example, as shown in figure 6.5. The text box without any restriction is provided for the student to answer.

After the learner clicks on save button, the pre-defined list of responses at various levels (bad, average, good, excellent) are shown, and the learner has to self-evaluate by comparing his/her response with ones given in the list. Based on their selection, feedback is generated, which states the corrective measures to be taken to improve the response, as shown in figure 6.6.

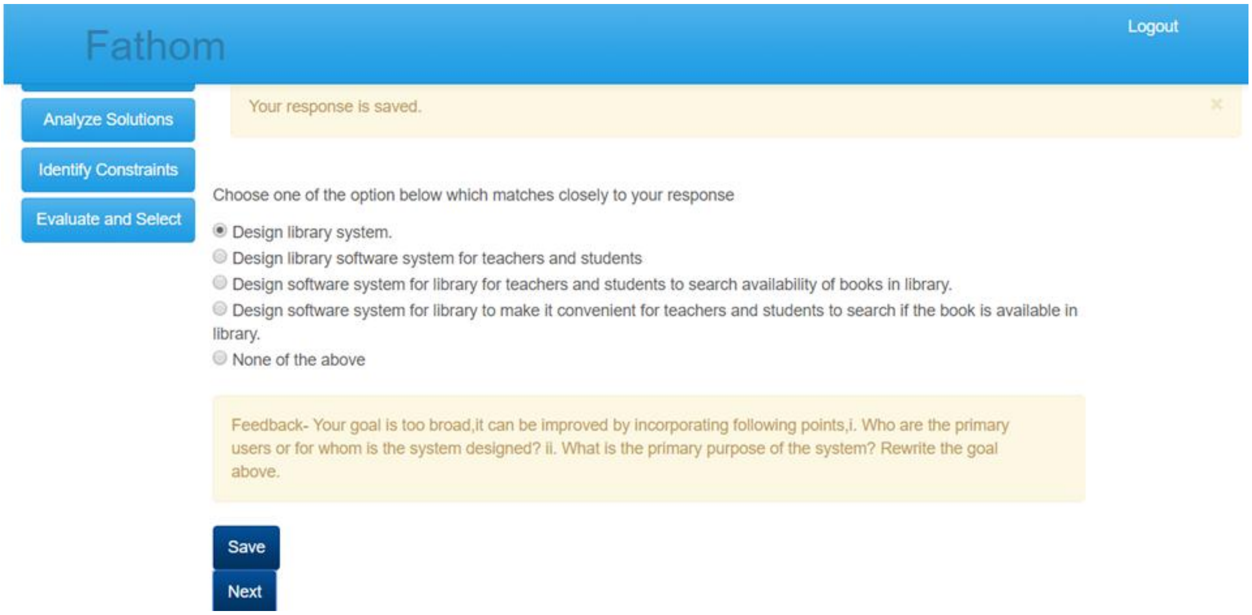


Figure 6.6. Feedback for formulate the problem-Goal

The list of options is based on the previous study's (study 2) responses. We identified the good, medium, and bad responses from the previous study and formulated the list.

Next, the prompt for sub-goal is, *"In the diagram below, identify the entities and interactions for achieving the goal and write as specific sub-goals (functionalities)."* The hint button (?) shows the example of the sub-goal as *"Shopkeeper reduces the quantity of item when the customer buys a new item from the shop."*

The library system's entity-interaction diagram drawn by the student is reproduced as shown in figure 6.7 and learner is prompted to identify the entities and interactions to be included as functionalities in the software system (figure 6.7).

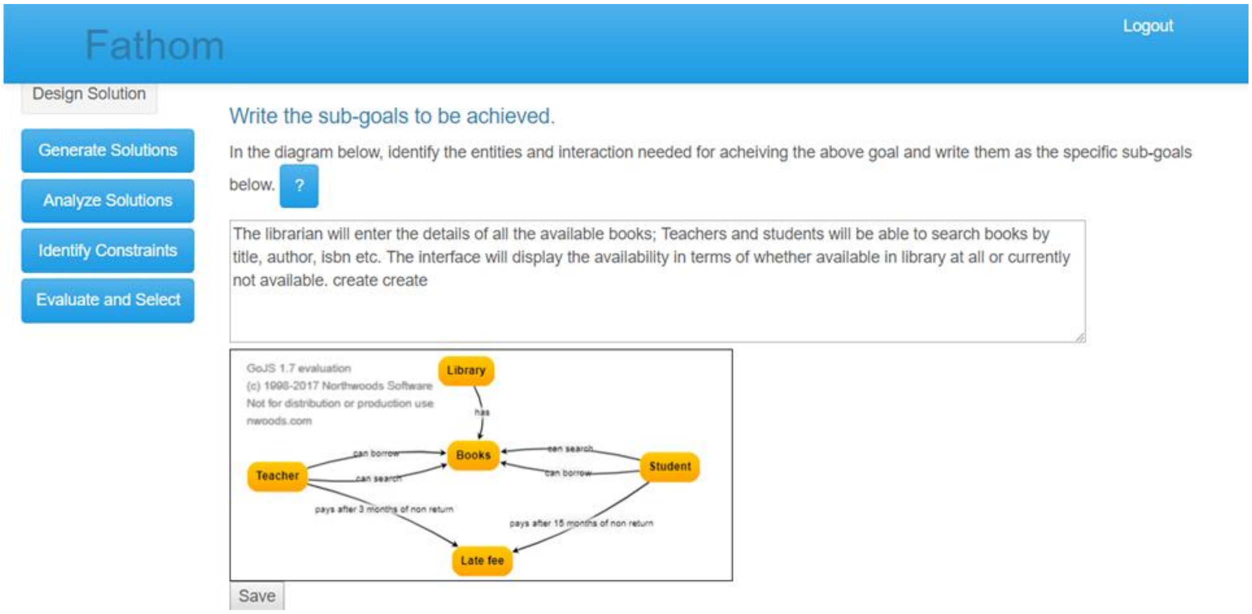


Figure 6.7. Formulate the problem- SubGoal

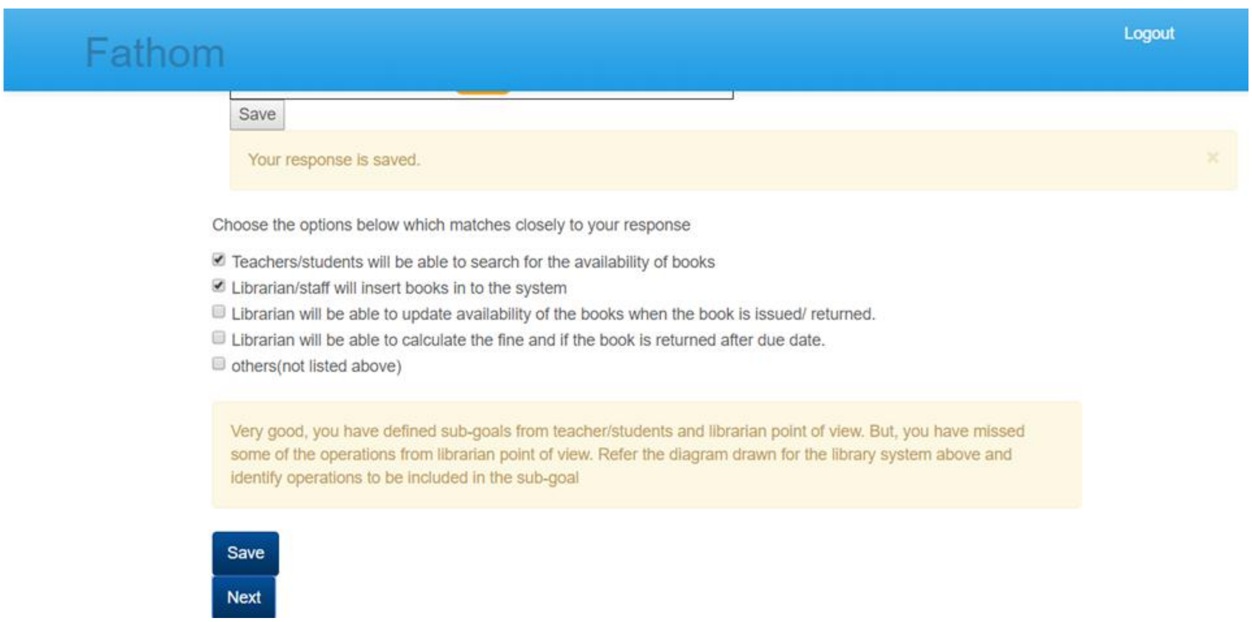


Figure 6.8. Feedback for formulate the problem-Goal

The textbox is provided to the user to write the sub-goals. After learner answers and clicks on save button, the list of sub-goals written from the perspective of students/teacher and librarian and the interactions among them are displayed (figure 6.8). The learner can select one or more options. The feedback is given based on the options selected; if only one option is selected, then feedback prompts to consider other stakeholder's view and rewrite.

Reflection

At the end of the activity, the system shows reflection questions for the activity: *Formulate problem* to reflect on how much the activity helped to achieve the targeted skill and analyze and rate how the targeted skill is applied to a new problem posed as a case study. The bank problem is posed as a case study throughout all the activities, and learners have to analyze and rate how well the targeted ER skill is applied.

i. The process of identifying the entities and interactions from the whole system helped me to define the broad goal and sub-goals to be achieved.

- *strongly agree,*
- *agree,*
- *neutral,*
- *disagree,*
- *strongly disagree.*

ii. For the given bank problem- "Design banking software for customers to perform online transactions- checking account balance, money transfer and bill payments."
Ravi has formulated the problem as given below-
Broad Goal- "To provide an online balance enquiry for customers.
Sub-Goal- "The search operation will find the balance of the given account_no."
Rate Ravi's ability to understand the system from multiple perspectives and define goal and sub-goal.

- ◉ *High [Ravi has identified the data and operations needed to solve the problem from the perspectives of all stakeholders and entities.]*
- ○ *2. Medium [Ravi has identified data and operations from more than one entity, but missed some data or operations needed to solve the problem.]*
- ○ *3. Low [Ravi has identified the data and operations to solve the problem only from the customers' point of view.]*

Justify why?

3. Generate solutions activity

The *generate solution* is the solution space activity, which is divided into two parts: i. Draw attribute listing map, and ii. List solutions.

a. Attribute listing map

The prompt posed is*, "Create the attribute listing map by identifying attributes of the design and its corresponding options."* The prompt is supported with hint and video to illustrate the process of drawing attribute listing map with an example, as shown in figure 6.9.

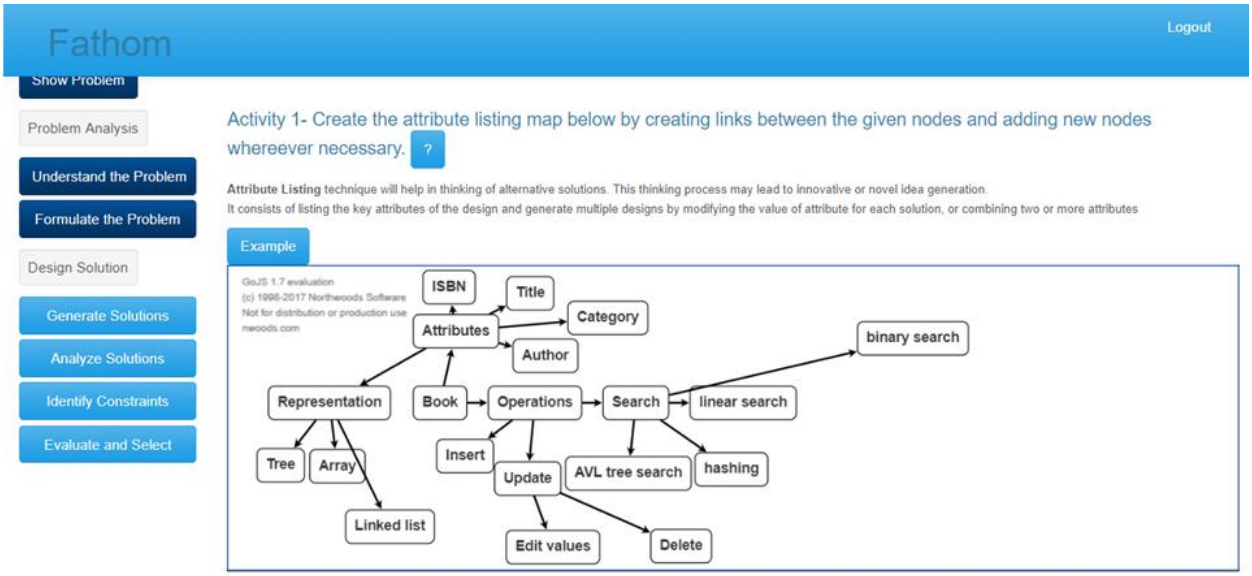


Figure 6.9. Attribute listing map

The drawing area is provided to learners to draw attribute listing map, as shown in figure 6.9. After the learner draws and clicks on save button, the system evaluates the number of nodes and links and generates feedback. If only a few nodes and links are created then the feedback given is *"you have identified very few attributes. See the example on how to generate attribute map by clicking the example button."*

Else, the learner can self-evaluate by comparing the attributes generated as shown in figure 6.10. If any attribute is missing, then feedback is given to add to the map accordingly.

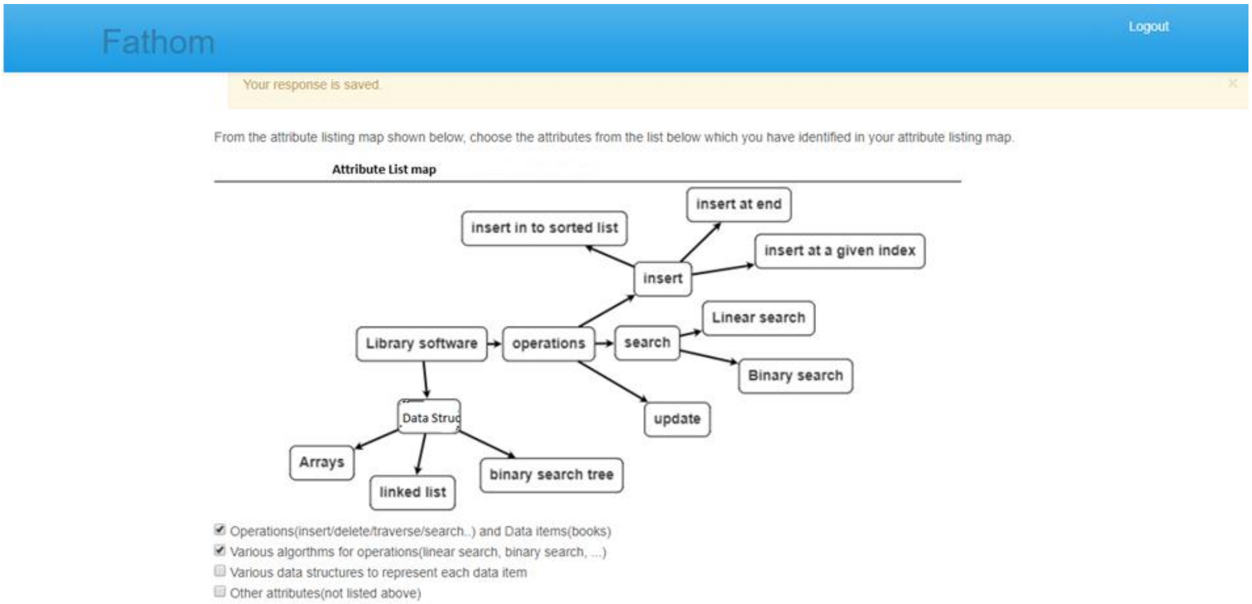


Figure 6.10. Attribute listing map feedback

b. List solutions

After the attribute listing map activity, the next activity is to generate multiple solutions. The prompt, hint with example is shown in figure 6.11. The textbox is provided to write the solution, and the user can add more solutions by clicking on the "*Click to add more solutions*" button.

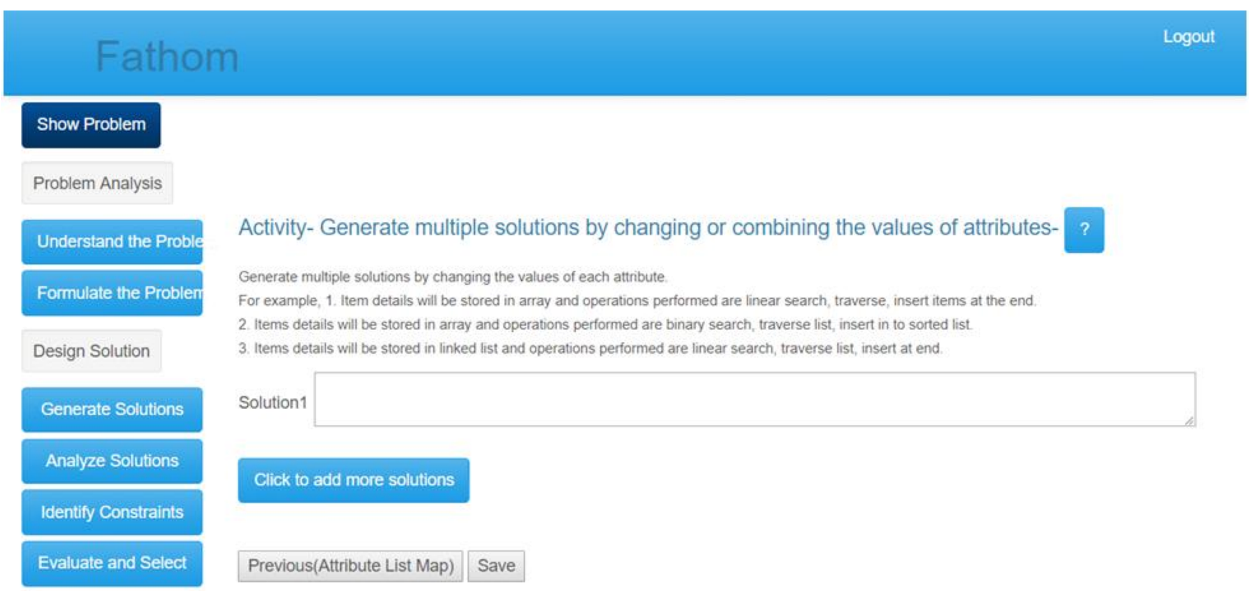


Figure 6.11. Generate solutions activity-prompt, hint

Feedback

If a learner has generated less than three solutions, then the attribute listing map is shown, and feedback is *"You have generated less than three solutions. Refer attribute listing map to generate more valid solutions using various data structures and operations".*

Reflection

The reflection questions in this stage were:

i. The process of identifying various design attributes and their options(data structures, algorithms) in "attribute listing map" activity helped me to generate multiple solutions.

- ○ *Strongly Agree*
- ○ *Agree*
- ○ *Neutral*
- ○ *Disagree*
- ○ *Strongly Disagree*

ii. For the given bank problem- "Design banking software for customers to perform online transactions- checking account balance."
Ravi has formulated the problem as given below-
Broad Goal- "To provide an online balance inquiry for customers.
Sub-Goal- "Search operation will find the balance of the given account_no."
Ravi has generated the following alternative solutions:
a. Arrays are used to store the data, and the linear search will be performed to search the balance of the given account_no.
b. Arrays are used to store the data in sorted order, and the binary search will be performed to search balance of the given account_no.
c. A linked list is used to store the data, and the linear search will be performed to search the balance of the given account_no.
Rate Ravi's ability to generate multiple solutions

- ○ *1.High [Ravi has identified most of the possible solutions.]*
- ○ *2. Medium [[Ravi has missed a few of the possible solutions.]*
- ○ *3. Low [Ravi has identified very few solutions.]*

Justify why?

4. Analyze solutions

After the learner has generated solutions, the next activity is to analyze solutions by identifying criteria to evaluate solutions.

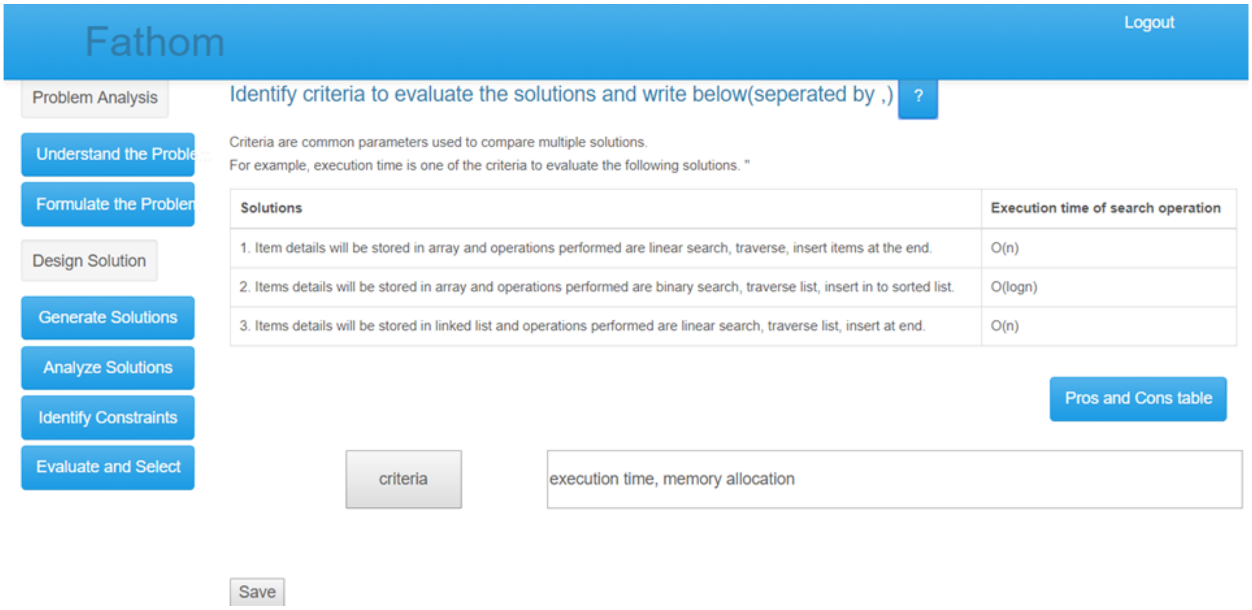


Figure 6.12. Analyze solutions to identify criteria.

The prompt posed is: *"Identify criteria to evaluate the solutions."* The activity is supported with an explanation of the term criteria with a worked example, as shown in figure 6.12.

The textbox is provided to write the answer. After the user enters the answer and clicks on save button, the feedback is generated based on the following rules:

If only 1 criterion is identified then feedback generated is *"You have identified only 1 criterion. Perform pros and cons analysis of the solutions to identify more common criteria"*.

Else the user can proceed to reflection activity.

Reflection

The reflection questions in this stage were-

i. The process of identifying criteria helped me to compare and analyze solutions across various parameters.

Strongly Agree

Agree

Neutral

Disagree

Strongly Disagree

ii. For the given bank problem- "Design banking software for customers to perform online transactions- checking account balance."

Criteria identified are by Ravi are execution time, implementation difficulty, for the solutions listed below- a. Array and linear search b. Array and binary search c. Linked list and linear search d. Binary search tree and search e. Hash table and search.
Rate Ravi's ability to identify criteria to evaluate solutions

- *High [Ravi has identified all the possible criteria]*
- *Medium [Ravi has missed few(1-2) of the criteria]*
- *Low [Ravi has missed many(more than 2) of the criteria]*

Justify why?

5. Identify constraint

This activity is designed to enable learners to think of constraints applicable to the given problem based on the common criteria identified in the previous activity. The prompt given is*: "Identify the constraints based on the criteria,"* and the term constraint is explained with a worked example, as shown in figure 6.13.

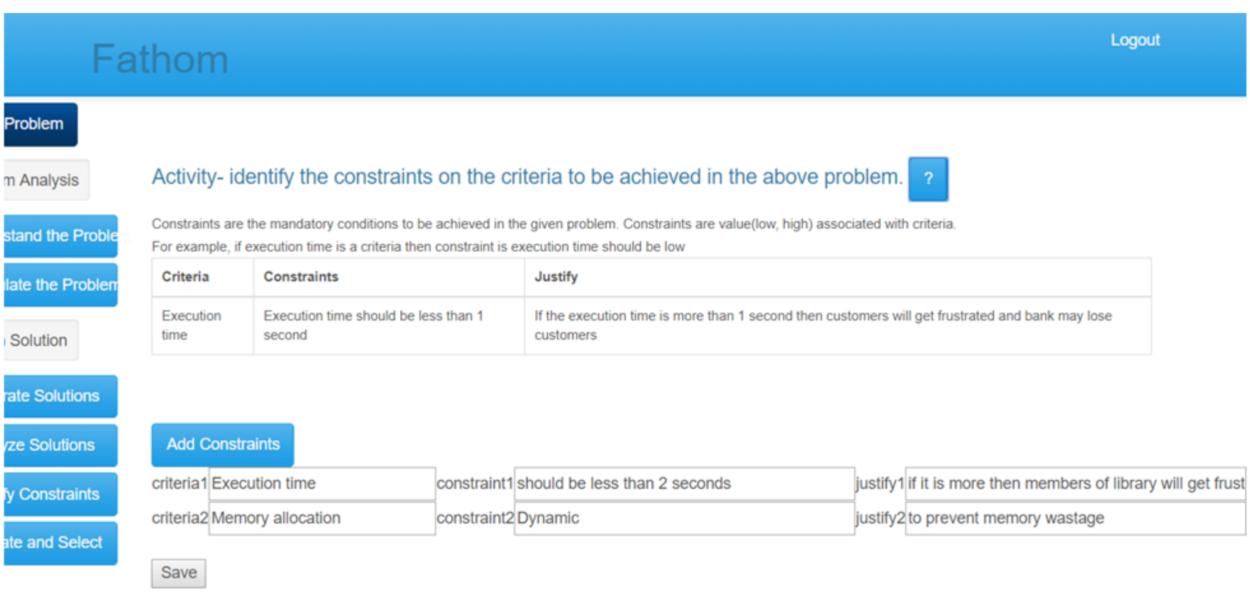


Figure 6.13. Identify constraints activity

The user has to answer by specifying the criteria, its corresponding constraint, and justify why it is important for the given problem. The three textboxes are provided, as shown in figure 6.13. The user can add more constraints by clicking on "*Add Constraints*" button.

After clicking on save button, the user is prompted to self-evaluate their responses by comparing with the list of possible constraints as shown in figure 6.14.

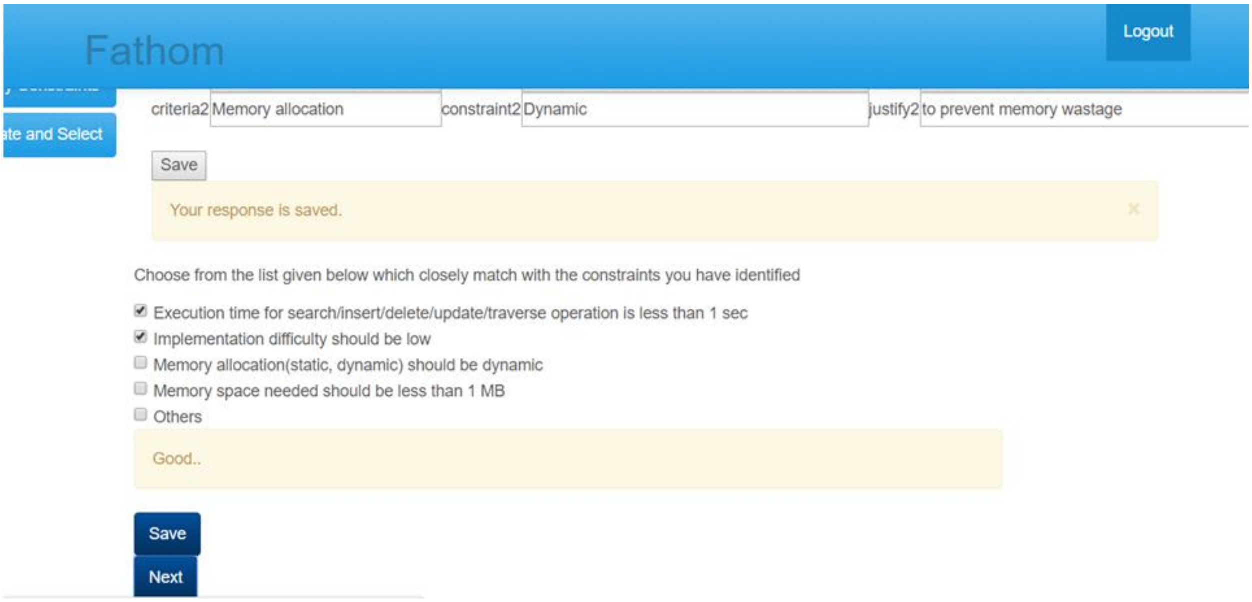


Figure 6.14. Feedback on Identify constraints activity

Reflection

The reflection questions at this stage posed were-

1. The process of identifying constraints helped me to think of the conditions/performance criteria to be achieved in the given problem.

- *Strongly Agree*
- *Agree*
- *Neutral*
- *Disagree*
- *Strongly Disagree*

2. For the given bank problem- "Design banking software for customers to perform online transactions- checking account balance."

Ravi identified the constraints as given below
i. Execution time
ii. Memory usage

Rate Ravi's ability to identify constraints for the given bank problem.

- *High [Ravi has identified constraints correctly]*

○ *Medium [Ravi has identified the criteria applicable for the bank problem, but could not define constraints]*

○ *Low [Ravi could not identify the constraints]*

Justify why?

6. Evaluate and select

The final activity is to evaluate and select a solution. The learner is prompted to evaluate the generated solutions against the sub-goals and constraints using the decision matrix, rank, and justify why the selected solution is better than other alternatives.

The prompt posed is *"Evaluate alternative solutions and select optimal solution using decision matrix,"* supported with an explanation of decision matrix and worked example as shown in figure 6.15.

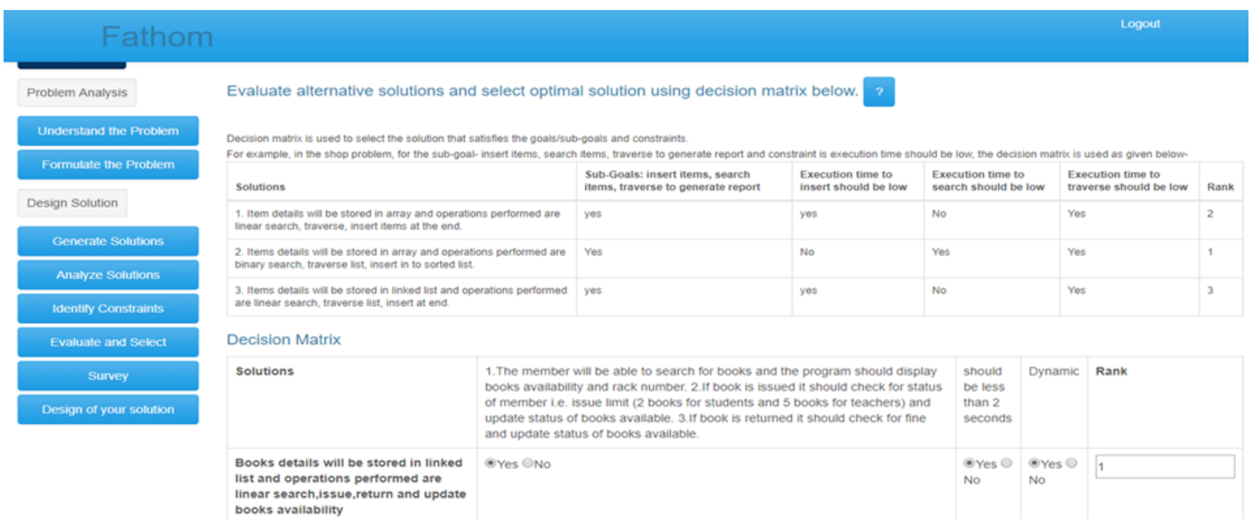


Figure 6.15. Evaluate and select- prompt and hint

The decision matrix is preloaded with alternative solutions generated by the learner in the first column and sub-goals and constraints on the first row. The learner has to enter yes/no in each intersecting cell by evaluating whether the solution achieves the sub-goals and constraints, accordingly rank and justify as shown in figure 6.16.

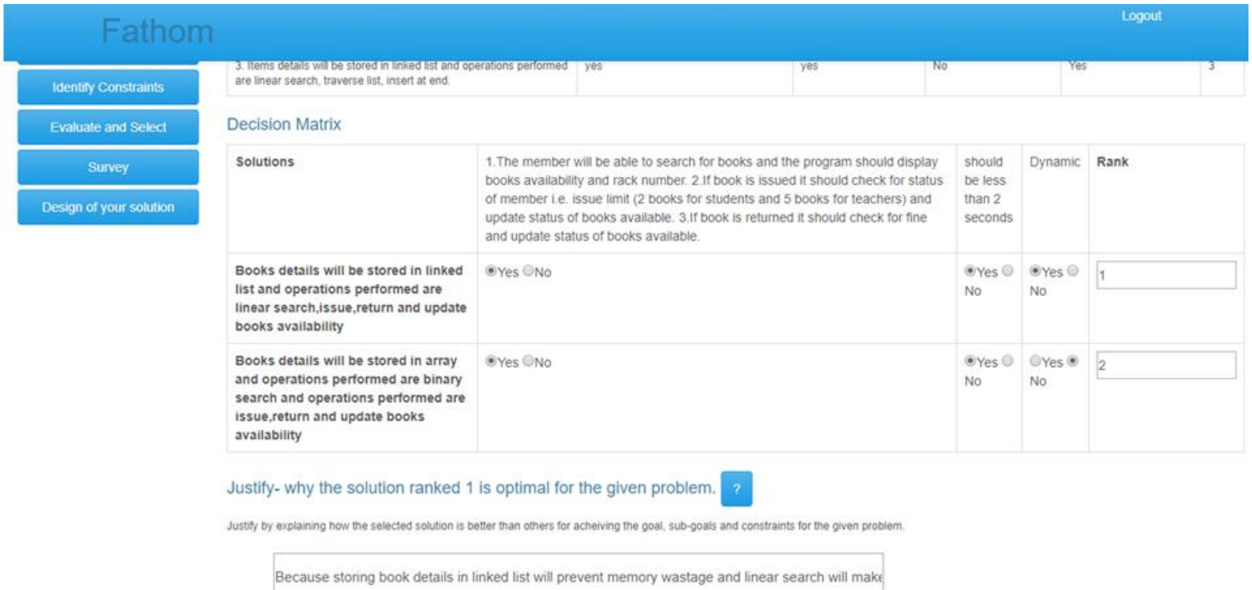


Figure 6.16. Decision matrix and justification box.

After justifying, the learner is asked to self-evaluate based on the options as shown in figure 6.17. The options are designed to allow learners to reflect based on which they have selected the solution: whether the solution is selected because it is easy to implement or because it is optimal than other alternatives in terms of achieving the sub-goals and constraints,

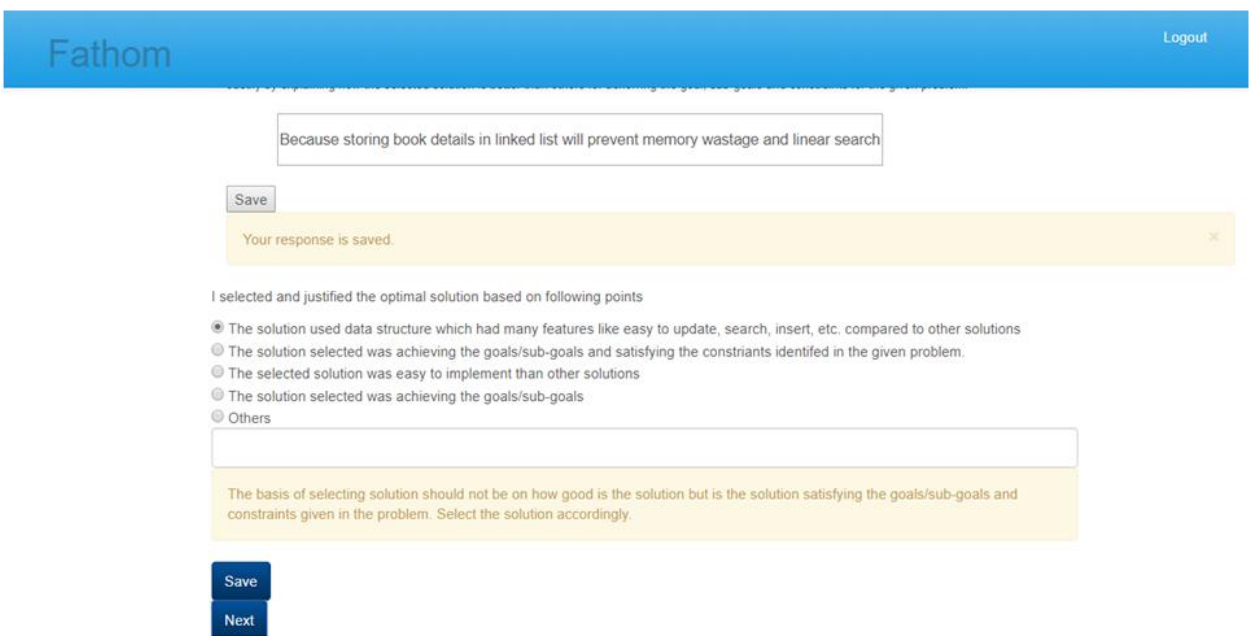


Figure 6.17. Feedback on selection and justification

Reflection

The reflection questions at this stage were-

1. The process of evaluating the solutions based on sub-goals and constraints in the decision matrix helped me to rank and select the solution that is suited for the given problem

- *Strongly Agree*
- *Agree*
- *Neutral*
- *Disagree*
- *Strongly Disagree*

2. The decision matrix helped me to justify how the selected solution is suited for the given problem

- *Strongly Agree*
- *Agree*
- *Neutral*
- *Disagree*
- *Strongly Disagree*

3. For the given bank problem- "Design banking software for customers to perform online transactions- checking account balance."
Ravi had generated the following solutions
i. Customer entity and account details are represented using array and search operation is implemented using linear search
ii. Customer entity and account details are represented using linked list and traverse to search
iii. Customer entity and account details are represented using an array and use binary search
Goal- " The customer should be able to search for the balance of his/her account."
The constraints identified were- execution time for search should be low, efficient memory utilization.
He selected solution ii(linked list) and justified - "linked list is ranked highest because its memory can be allocated during runtime"

Rate Ravi's ability to select and justify.

- *High [Ravi has selected solution and justified based on the goals/subgoals and constraints]*
- *Medium [Ravi has selected solution and justified based on either goals/subgoals or constraints]*
- *Low [Ravi has not selected solution and justified based on the goals/subgoals and constraints]]*

Justify.why?

Table 6.1 shows the comparison between the design features of Fathom-Ver1 and Ver2.

Table 6.1. Design features of Fathom-Ver1 and Fathom-Ver2

Design features	Fathom-Ver1	Fathom-Ver2
Structured guidance	Fully-structured	Semi-structured
Prompts	Domain-independent, e.g., Draw the model of the system to show the components, its properties, and interconnection between components	Domain-dependent, e.g., Diagrammatically represent the library system by drawing entities involved in the system and interactions among them.
Worked example illustrating the process of solving similar software design problem.	No	Yes
Online drawing tool (entity-interaction diagram and attribute listing map)	No	Yes
Self-evaluation activities	No	Yes

It was seen in study 2 that learners had difficulty in performing certain activities in Fathom-Ver1 in which the prompts were at a higher level and not domain-specific. Thus, in Fathom-Ver2, we simplified and supported the prompts with domain-specific information and worked example to aid learners in activating their prior knowledge and thinking towards solving the problem efficiently. To enable learning and transfer of complex problem-solving skills, the learners need to be engaged at both cognitive and metacognitive level. The design of Fathom-Ver2 is modified to facilitate the learning of ER skills at both cognitive (prompts and worked examples) and metacognitive level (self-assessment, feedback, and in-action reflection activities) compared to Fathom-Ver1 in which the focus was only at engaging learners at a cognitive level. In the next section, the evaluation study is discussed.

6.3 Evaluation Study 3- Effectiveness of the design and pedagogy of Fathom-Ver2 in learning ER skills.

The focus of the study is to investigate the effectiveness of Fathom-Ver2 in doing and learning of ER skills for novices. The research question (RQ) investigated is-

RQ 3. How effective is Fathom-Ver2 for novices in doing and learning of ER skills?

To investigate the key design feature of Fathom-Ver2, we generated the following sub RQs:

RQ 3.1. How effective are the pedagogical features of Fathom-Ver2 in helping students to perform and learn ER skills?
RQ 3.2. How effective is structured versus a semi-structured sequence of learning activities in performing ER skills?
RQ 3.3. How effective is the feedback mechanism (system generated, self-evaluated) in monitoring and controlling the ER skills?
RQ 3.4. How effective are in-action reflection activities enabling the student to analyze the ER skills in a new problem solved by a pseudo user?

6.3.1 Participants

Fifty-two students from second-year computer engineering participated in the study. The study was conducted at the end of the data structure course, thus ensuring that students had enough domain knowledge to solve design problems in data structures.

6.3.2 Study Design

The study design was experimental pretest-postest, as shown in figure.6.18. The control group had structured guidance, and the experiment group had semi-structured guidance. In structured guidance, the student strictly followed the linear path in the order- understand; formulate; generate solutions; identify criteria and constraints; evaluate; and select, and not allowed to deviate the order. In semi-structured guidance, the overall learning path was open for the learner to select.

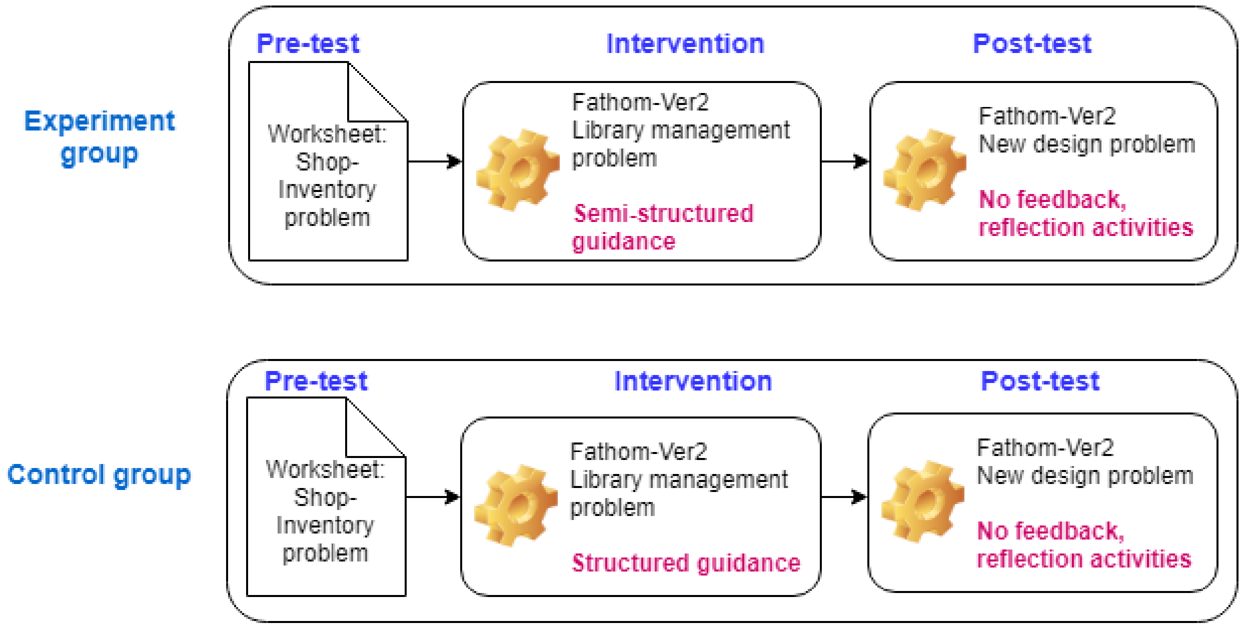


Figure 6.18. Study 3- Study design

The learners were divided into two groups, with approximately 25 students in each group. The division was random.

6.3.3 Procedure

i. Pretest- The students were given a worksheet to solve a shop inventory problem, as shown in figure 6.19. The steps were given to guide the software design process conventionally.

Roll no:

Name:

Pretest

Design a software system for a supermarket, to keep track of items whose quantity is below threshold. The software should generate a report at a end of the day, a list of items whose quantity is below threshold and needs to be ordered.

Your task is to design a software system to solve the above problem using appropriate data structure and algorithm.

Solve by following the steps-

i. Write the broad goal to be achieved.
ii. Write the sub-goals to be achieved.
iii. Design solution using appropriate data structure and algorithms.
iv. Justify why the selected data structure is appropriate for the given problem.

Figure 6.19. Pretest worksheet

ii. **Intervention-** Immediately after pre-test, the learners were divided into experimental and control groups. The experimental group interacted with Fathom-Ver2 with semi-structured guidance and control group interaction with strictly-structured guidance. Both groups took approximately 2 hrs to complete the training.

The training problem posed was the library problem:

The college library maintains books on various subjects taught in various departments like Computer engineering, Electronics engineering, etc. The library staff is involved in issuing books to students and teacher. The students need to return the book in 15 days while teachers can return in three months. The library staff maintains records of all books in the library, books issued, returned and collect a late fee if the book is not returned in time. As it was becoming tedious for teachers and students to search for the book in the library, Librarian decided to provide online service for students or teachers to search availability of the books and display if the book is available with its shelf number (location of the book). Your task is to design a software system for the requirement stated by the librarian, using appropriate data structure and algorithm.

iii. **Posttest-** Posttest was not taken immediately after the intervention due to time constraints. Students were asked to log in into Fathom-Ver2 next day, and select a new problem from the list shown in figure 6.20 to solve. The scaffolds faded were: feedback and reflection activity. The scaffolds retained were: the problem-solving steps; semi-structured guidance, prompts, hints, worked example, videos, and cognitive tools. Total 17 students attempted post-test.

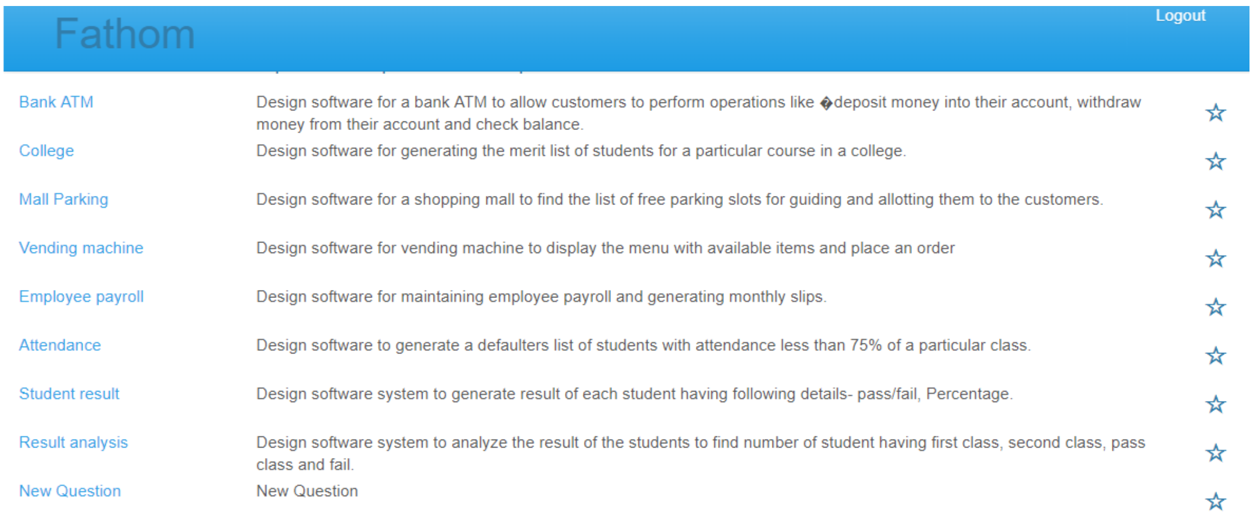


Figure 6.20. Posttest problems

6.3.4 Data sources

The data collected were-

- The pretest worksheets, postest responses of students.
- Learner artifacts (responses of each learning activity) generated during interaction with Fathom.
- Log data of user clicks with a timestamp.
- Student perception rating on the usefulness of activity in learning skills.

6.3.5 Data analysis

The responses of pretest problem, training problem, and post-test problem were evaluated using the rubric given in table 6.2. The rubric used in previous study 2 was designed to assess both ER skills and use of ER cognitive tools (pros and cons table, decision table, etc.). The revised rubric for study 3 is focused only on the assessment of ER skills in more detail as the aim of this study was to measure the learning of ER skills.

Table 6.2. Rubric for rating the ER skills of the software design problem

ER skill	ER activities	Rubric
Problem space		
Expand	Understand the problem	*Score-* ***High*** *(score-3)*- identified all possible entities and interactions identified. ***Medium*** *(score- 2)*- missed a few. ***Low*** *(score- 1)*- only a few identified.
Reduce	Formulate goal	***High*** *(score-3)*- the goal has addressed the following points- requirements broadly defined, primary users and benefits written. ***Medium*** *(score- 2)*- the goal missed one of the above points. ***Low*** *(score- 1)*- the goal missed most of the points.
	Formulate sub-goal (quantity & quality)	***High*** *(score-3)*- identified all the data items and operations for achieving the stated goal. ***Medium*** *(score- 2)*- missed identifying one of the data items or operations that will achieve the goal. ***Low*** *(score- 1)*- missed identifying data item or operation which are important for achieving the goal.
Solution space		
Expand	Generate solution- Quantity	**Score**- count of alternative solutions.
	Solution quality	***High*** *(score-3)*- the solution is complete and correct. In the solution, the data structure is valid, and the operations are mapped to correct algorithms. The clarity is high on how data is stored and how the algorithms are used. ***Medium*** *(score- 2)*- has missed identifying valid data structure or an algorithm for some data items and operations respectively. ***Low*** *(score- 1)*- the solution is incomplete or not correct.
Reduce	Criteria	***High*** *(score-3)*- all possible criteria correctly identified. ***Medium*** *(score-2)* - missed a few.

		Low(*score-1)*- only one identified.
	Constraints	***High*** *(score-3)*- all possible constraints are applicable to the problem. ***Medium*** *(score-2)*- missed a few. ***Low***(*score-1)*- only one identified.
	Select and justify	***High*** *(score-3)*- selected the correct DS and justified how the solution satisfies all the constraints applicable to the problem and is optimal than other alternatives. ***Medium*** *(score-2)*- selected a correct DS and justified without considering all constraints or did not consider other alternatives. ***Low***(*score-1)*- selected an incorrect DS and did not justify based on the constraints applicable to the problem.

Following statistical tests were performed to answer the RQs.

1. The t-test between pre-test and intervention scores for both groups.
2. T-test of pre-test and post-test scores.
3. T-test of intervention scores of both groups
4. Percentage of the rating of student perception survey on the usefulness of activity in learning

6.3.6 Results and discussion

The corresponding data collected, and the analysis done to answer each RQ is shown in table 6.3.

Table 6.3. RQs and corresponding data collection method

Research questions	Data collected and analysis
RQ 3.1. How effective are the pedagogical features of Fathom-Ver2 in helping students to perform and learn ER skills?	1. Comparison of mean scores (t-test) of - pretest to intervention, - pretest to posttest, and - intervention to posttest. 2. Student perception survey
RQ 3.2. How effective is structured versus a semi-structured sequence of learning activities in	Comparison of mean scores (t-test) of experiment and control group

performing ER skills?	
RQ 3.3. How effective is the feedback mechanism (system generated, self-evaluated) in monitoring and controlling the ER skills?	Log data analysis- the percentage of students who have modified their skills after evaluation activity.
RQ 3.4. How effective are in-action reflection activities enabling the student to analyze the ER skills in a new problem solved by a pseudo user?	Percentage of students who have correctly rated the ER skills in a reflection activity.

The average scores and comparison (p-values) of pretest-intervention-posttest of both experiment and control groups are shown in table 6.4, table 6.5, and table 6.6.

Table 6.4. Comparison of pre-test and intervention scores (Control group)

N=27	Understand the problem	Formulate the problem	Generate solutions/ Design solution	Identify criteria and constraints	Evaluate solutions	Select a solution and Justify
Pretest (Average)	-	1.62	1.64	-	-	0.96
Intervention (Average)	2.42	1.88	2	2.33	1.8	1.9
T-test(p value)	-	0.01	0.04	-	-	0.00

Table 6.5. Comparison of pre-test and intervention scores (Experiment group)

N=27	Understand the problem	Formulate the problem	Generate solutions/ Design solution	Identify criteria and constraints	Evaluate solutions	Select a solution and Justify
Pretest	-	1.75	1.5	-	-	0.90
Intervention (Average)	2.55	2.29	2.16	2.31	1.84	1.92
T-test(p value)	-	0.01	0.02	-	-	0.00

Table 6.6. Comparison of pretest, activity and posttest scores

N=17	Understand the problem	Formulate the problem	Generate solutions/ Design solution	Identify criteria and constraints	Evaluate solutions	Select a solution and Justify
Pre-test (average score, N=17)	-	1.5	1.2	-	-	0.7
Intervention (average score, N=17)	2.21	1.94	2.06	2.58	1.94	1.89
Post-test (average score, N=17)	2.41	2.4	2.11	2.4	2.4	2.3
T-test(p value) Pretest-intervention	-	0.00	0.00	-	-	0.00
T-test(p value) Pre-posttest	-	0.00	0.00	-	-	0.00

Student perception survey rating (figure.6.21) showed that 30 to 40% of students strongly agreed and 60-70% agreed that the activities helped in achieving the ER skill and in doing the subsequent activities.

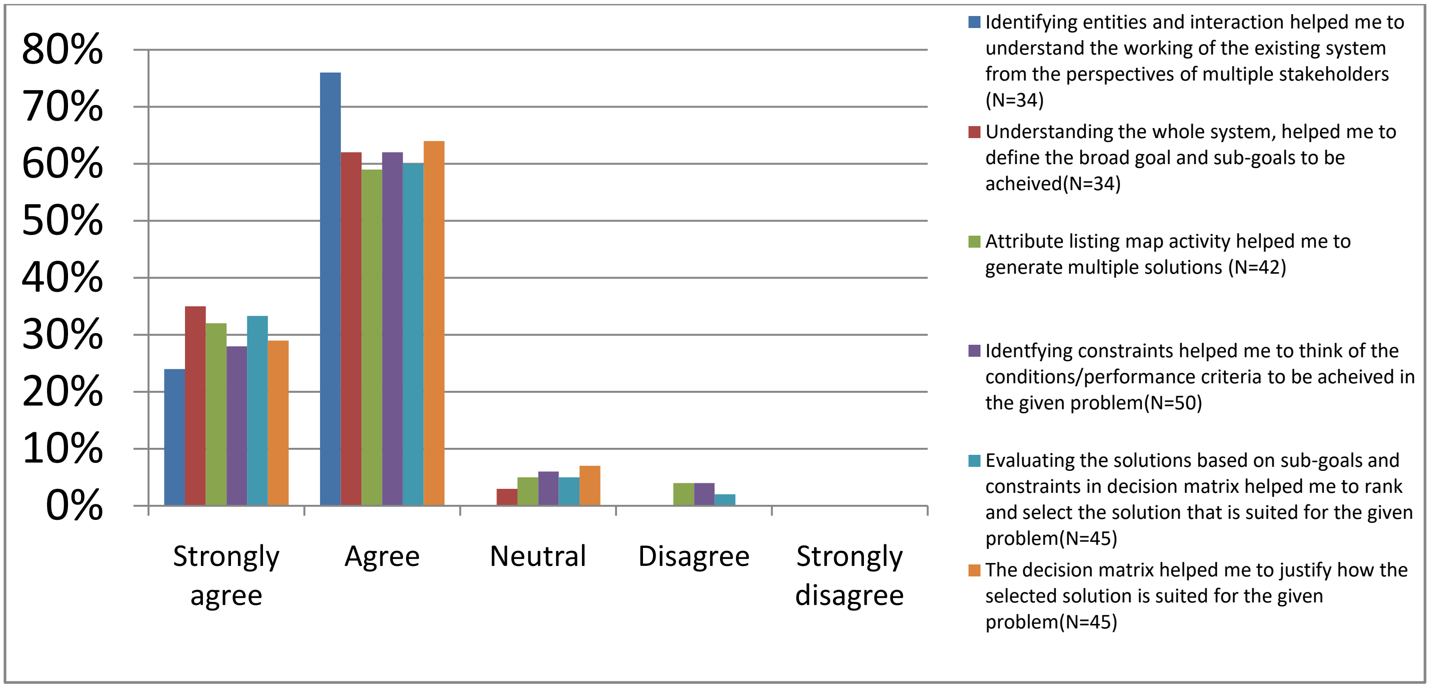


Figure 6.21. Student perception survey rating

Log data had user click's records in the format:

<button name, timestamp, the response of the corresponding text-box>

Putting all these together, the answers to the RQs are discussed below-

RQ3.1- How effective are the pedagogical features of Fathom-Ver2 in helping students to perform and learn ER skills?

For both the groups, significant improvement was seen in the quality of the problem formulation ($p<=0.01$), solution quality ($p<=0.04$) and justification ($p<=0.00$) from pretest to intervention and significant improvement($p<=0.00$) from pretest to posttest scores in quality of problem formulation, solution quality, and justification. The performance during intervention and posttest was not significantly different, which shows that students were able to perform ER skills in the second problem even with no feedback. These scores are triangulated with student perception rating, which shows that above 80% of students perceived that the activities helped them to perform ER skills effectively.

The answer to RQ3.1 is that the redesigned prompts and scaffolds helped to perform ER skills effectively. The prompts were more specific and tied to the domain rather than being general, detailed explanation of new terms with a worked example, video to illustrate the use of cognitive tools using worked example, and hints were helping students to understand the activity and perform ER skills.

RQ3.2. How effective is structured versus a semi-structured sequence of learning activities in performing ER skills?

The average scores of ER skills for experiment group was higher than the control group but were not significantly different. The log data analysis showed that student in the experiment group was initially exploring all activities to understand the problem-solving steps, and later followed a linear path to solve the problem. The semi-structured sequence and structured sequence had no difference in the performance of the ER skills, but semi-structured sequence allowed students to understand better the overall problem-solving process and the freedom to choose the path to solving the problem.

RQ3.3. How effective is the feedback mechanism (system generated and self-evaluated) in monitoring and controlling the ER skills?

The log data was analyzed to check if the learners changed the responses after reading the feedback generated by the system (understand problem activity) and after self-evaluation (all other activities). We found that 80% of students modified their response after reading system-generated feedback in *understand problem* activity, and only 24% (12/50) modified their response after self-evaluation activity. During self-evaluation activity, most of the students overrated or underrated their responses, and some students did the self-evaluation correctly but did not change the response after reading the corrective feedback.

RQ3.4. How effective are in-action reflection activities enabling the student to assess the ER skills in a new problem solved by a pseudo user?

The log data analysis showed that only 10% of students assessed the ER skills correctly. We found that the students were getting confused on which problem are they solving because they were handling two problems: one was the library problem solved during the activity, and another was the bank problem analyzed as a case study during reflection phase. This shows students are not good at handling multiple complex problems at the same time during training, and it adds cognitive load.

6.4 Reflection

The learning activities at various cognitive levels: doing, evaluate, and reflect using case-study, are shown in figure.6.22.

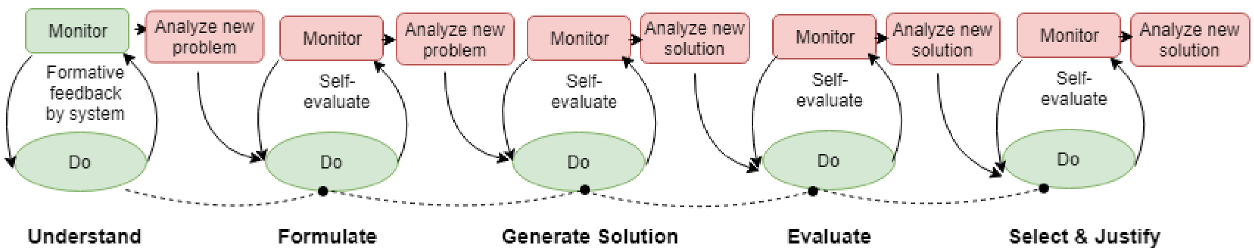


Figure 6.22. Learning activities performed by learners (green- effective, red- not effective)

The overall effectiveness of the learning activities at various cognitive levels is summarized as-

- The learning activities supported with pedagogical feature: prompts, hints, video, worked example helped to practice ER skills effectively (green circles at doing level in figure.6.22).
- The system generated feedback in activity: *understand problem*, helped students to monitor and control their skill (green rectangle in understand in figure 6.22).
- Prior studies show that self-evaluation was effective for interpersonal skills (Rivera, 2016) and learners with professional or prior experience (Adelson, 1985; Tang, 2008). We saw in our study, that for design problems which are ill-structured, the students have difficulty in self-evaluation of the ER skill. They lacked the metacognitive ability to monitor and control their skills on their own (red monitor rectangles in figure 6.22).
- The evaluation of new problem was adding a cognitive load as students had to do context switch between two problems (red analyze rectangles in figure 6.22).

Feedback for the next cycle

- The learning activities and semi-structured guidance will be retained in the next DBR cycle as they were effective in doing and learning ER skills.

- The metacognitive activities of self-evaluation and reflection have to be modified, and metacognitive scaffolds have to be improved to help students to monitor and control ER skills effectively.

Figure 6.23 shows the overall DBR cycles, and till this chapter, we have completed the second DBR cycle. In the next chapter, the phases of DBR cycle 3 will be discussed in detail.

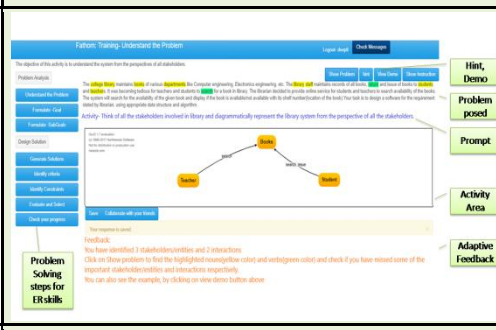
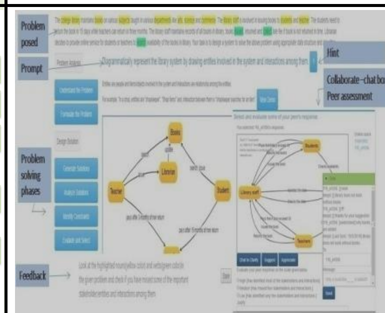
	DBR Cycle 1	**DBR Cycle 2**	**DBR Cycle 3**
Problem analysis	1. To assess the level of ER skills in novice. 2. Literature survey to identify the ER cognitive tools and learning techniques, Study 1(N=40), exploratory study	To redesign the TELE with an improved level of scaffolding and introduce reflection activities.	To revise the metacognitive activities for effective monitoring and controlling of ER skills.
Solution	Fathom-Ver1	Fathom-Ver2	Fathom-Ver3
			
Evaluation	**Study 2-** 49 engineering students, exploratory study **Results-** Students were having difficulty in doing ER	**Study 3-** 49 students, experimental pre-posttest. **Results-** Students could do ER skills but faced difficulty in evaluation of ER skills**.**	**Study 4-** 50 students, experimental pre-posttest. **Results-** Students could do ER skills and monitor ER skills.
Reflection	Prompts are not effective if learners do not have prior experience in ER cognitive skill. Improve the design by adding more scaffolds.	Novice learners are not able to self-evaluate to monitor and control ER skills. Improve metacognitive support.	The pedagogical support in terms of cognitive and metacognitive support and cognitive tools are effective in doing and learning of ER skills.

Figure 6.23. Overall design-based research cycles applied in this thesis

Chapter 7

DBR Cycle 3: Scaffold novices to regulate the learning of ER skills at the metacognitive level

7.1 Problem Analysis- Principles for effective metacognitive scaffolding

The expert studies (Adelson, 1985; Guindon, 1999; Tang, 2008, 2010) and ill-structured problem-solving literature (Xun, 2013; Jonassen, 2006) has shown that it is necessary for a designer to be involved at both cognitive and metacognitive level to design quality solution. In the context of complex problem-solving, experts are involved at a cognitive level, for example, generating solutions and at the metacognitive level by monitoring and controlling their progress. They evaluate their cognitive processes to assess the gap from current to the desired level of performance and take appropriate action to improve. In DBR cycle-2 study, we found that novices have difficulty in self-evaluating their ER skills and in taking appropriate action to improve. In this DBR cycle, the focus is on scaffolding the metacognitive activity to enable students to monitor their ER skills, identify the gap, and take appropriate action to improve.

In this section, we further did an in-depth analysis of various instructional strategies used for teaching-learning of metacognitive skills in problem-solving literature. The focus is to identify the issues and strategies in regulating learner's behaviors towards monitoring the gaps and take control action plan to improve on the learning task and map it to our solution design.

7.1.1 Interactive tutoring feedback model (ITF)

Feedback is an important factor which informs the learner about the actual state of learning or performance to regulate the learning towards the required standards. Feedback is a post-activity response which may be provided by various external sources, for example by teachers, peers, computer-based agents, etc.

Feedback is not effective if it merely flags the result as correct or incorrect. The feedback should be elaborated enough to allow learners to become aware of any gaps that exist between the desired and their current state of knowledge, understanding, or competencies. Elaborated feedback should have hints, examples, and explanations that are aimed at supporting learners in acquiring the knowledge and competencies necessary for mastering learning tasks. This kind of feedback is known as tutorial feedback (Narciss, 2013).

The interactive tutoring feedback model (ITF) is a validated framework that can be applied to the design and evaluation of tutorial feedback strategies for digital learning environments (Narciss, 2013). The framework considers that when feedback is generated by an external source, two interacting feedback loops exist: the learner feedback loop and the external feedback loop, as shown in figure 7.1.

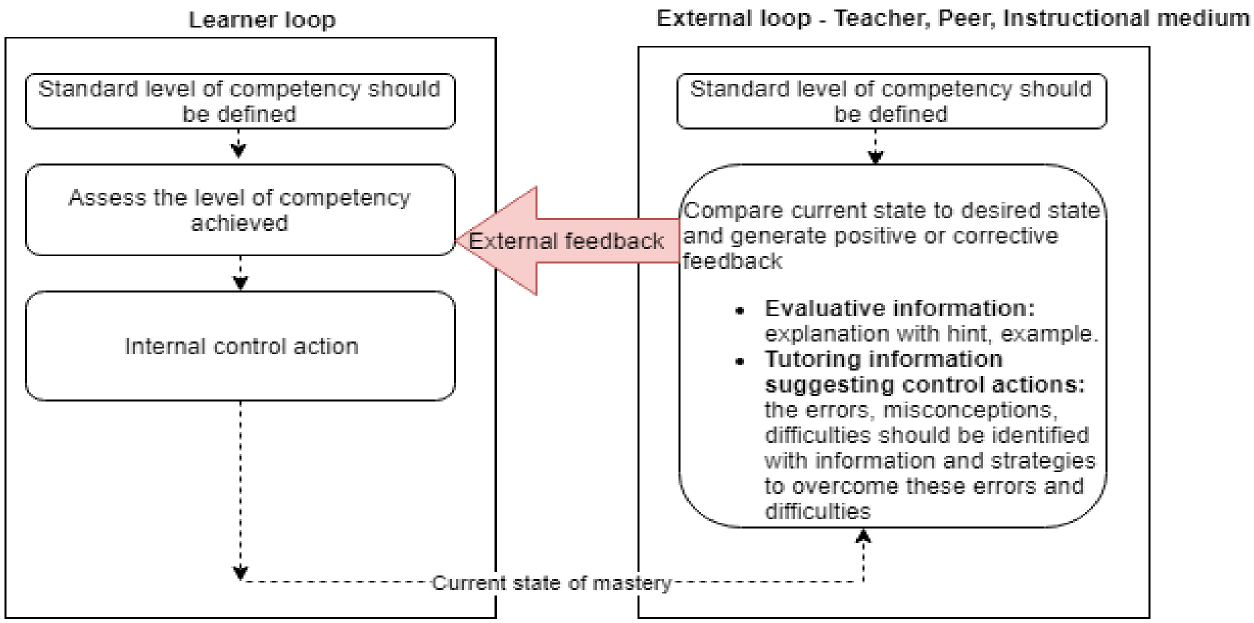


Figure 7.1. Interactive tutoring feedback model (Reproduced from Narciss, 2013)

To regulate and control the acquisition of competencies, both learner and external feedback loops needs to be activated. The external feedback loop is generated by external sources (teacher, peer, instructional medium), by comparing the learner's current state of competencies to the desired state of competencies and accordingly generate positive or corrective feedback. The learner's self-assessment ability plays a major role in creating a learner feedback loop. Learners have to monitor and assess their process of learning or competence acquisition to generate their internal feedback loop. Apart from self-assessment, the learner has to compare the external feedback with internal feedback and generate control actions.

To generate control actions for the regulation of the process of competence acquisition, the errors and difficulties that could arise in connection with mastering task requirements must also be identified, as well as the information and strategies that are needed to overcome these errors or difficulties. If a learner has selected adequate control actions, their competence acquisition process should be improved.

Overall, the nature and quality of feedback message are determined by the following factors:

i. **Feedback scope and function**: Feedback can affect the learning process at different levels: cognition, metacognition, and motivation. It can acknowledge, confirm, or reinforce the correct response, and in doing so, promote the acquisition of the knowledge and cognitive processes for accomplishing the learning task. Feedback can also contribute to correct errors or misconceptions. It can prompt the metacognitive strategies or encourage students in maintaining their efforts and persistence.

ii. **Feedback content**: The feedback content may contain

- Knowledge of the performance- providing learners with the formative feedback on the accomplishment of the complex task (number of correct responses or task completed, number of errors, time taken, grade)
- Knowledge of the result- feedback on the correctness or quality of the response (correct/incorrect, flag errors, good job)
- Knowledge of the correct response, providing the correct response or sample solution.
- Elaborated feedback, providing additional information besides knowledge of result or knowledge of the correct response (e.g., hints, guiding questions, explanations, worked examples).

3. **Feedback presentation**: The feedback can be communicated at various timing (immediate, delayed), and allow single or multiple tries.

The study (Narciss et al., 2013) was done to evaluate the effects of interactive feedback tutoring model in the context of fraction tasks provided by the web-based mathematics education system, ActiveMath. The adaptive feedback was generated based on the typical errors identified through a review of empirical fraction error analyses. The feedback was supported with hints and explanation, providing students with a tutoring hint after the first failure, and a tutoring explanation after a second failure. The study was done with sixth and seventh graders. The log-data analysis was done to explore the behavioral process variables; namely succeeding vs. failing in correcting the error of a task, and skipping vs. trying to correct an error after feedback was provided. The results were variable across task types and task steps. Considering gender as one of the variables, the male student's knowledge gain was significantly lower under all feedback strategies, especially under providing first a conceptual hint. Moreover, after conceptual feedback hints or explanations, students skipped more frequently and succeeded less often to correct the error. This result indicates that it is important to gain further insights into the dynamic interplay among the individual learner characteristics and situational feedback conditions.

Summary

To summarize the ITF-model implications for effective feedback design, the instructional designers and researchers should have complete understanding and representations of the relevant competencies, and analyze the conditions and characteristics of the learners as well as available feedback sources. The feedback should address the typical errors made by the learners and generate positive or corrective feedback, elaborated with hints, explanation, and information and strategies that are needed to overcome these errors or difficulties. However, the studies have shown mixed results and need to be further investigated.

7.1.2 Collaboration: Peer review

Peer-review is another technique used to engage learners in metacognitive activity during ill-structured problem-solving. It enables students to see multiple perspectives from peers' responses to the problem solutions and help them notice things they might not have thought. By compelling learners to examine their thinking after reviewing the peer's responses, learners get engaged in metacognitive activity and self-regulation during problem-solving. Previous studies (Ge, X., Planas, 2010) have shown that learners perceived that peer review is useful to see different perspectives towards problem-solving. However, the peer review mechanism did not show any advantage for the treatment group over the control group. It was suggested that to improve the learning via peer review; the learners should be given the opportunity to make comments and suggestions to each other, instead of just viewing each other's responses (Xun, 2013).

To engage learners to actively evaluate their peer's responses, we referred to effective collaborated learning methods to allow participants to actively demand explanation and justification from their peers. The Collaborative Learning Conversation Skills Taxonomy (Soller, 1999) shown in Figure7.2, illustrates the conversation skills which are key to collaborative learning and problem-solving. The taxonomy suggests various subskills (request, inform, motivate) of active learning skills with its attributes and its corresponding sentence opener (figure 7.2). Active Learning (AL) conversation skills, such as Encourage, Explain, Justify, and Elaborate, describe the core communication activities of effective learning groups.

The three subskill categories encompassing Active Learning are Inform, Request, and Motivate.

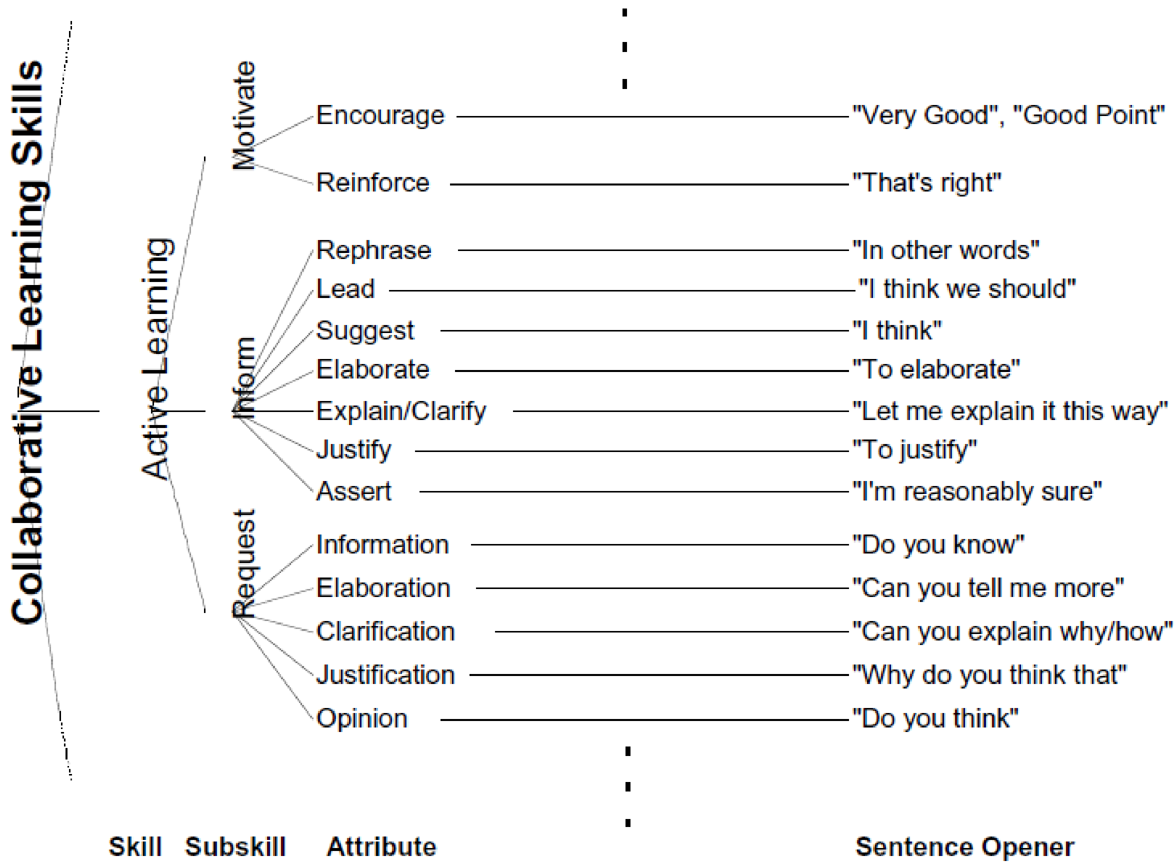


Figure 7.2. The active learning section of the Collaborative learning conversation skill taxonomy (Soller, 1999)

Summary

Students solving an open-ended problem which has no single correct solution must explain their viewpoint to peers and justify their opinion. The feedback or support should be provided to students to engage in active learning conversation for effective learning and problem-solving.

To encourage students in effectively collaborating with peers at the metacognitive level, we have integrated the collaborative learning conversational skill taxonomy into our design.

7.1.3 Implications for our TELE design

The studies (Narciss et al., 2013) on evaluating various feedback strategies were mostly done for improving conceptual knowledge in the context of solving well-structured problems and generating feedback based on typical errors made during the task. Some of the challenges to adopt these feedback strategies for ill-structured problem-solving are that the type of errors is not known, the problem-solution space is not well-structured, has undefined sub-problems and constraints, has many alternative solutions, and the process of solving the problem is not linear. All these issues make it difficult to automate the assessment of the task and provide feedback.

In the previous version, Fathom-Ver2, we designed a feedback mechanism based on the reflective framework (Rivera, 2016). After the activity, the learners had to self-evaluate by comparing their response with the given set of pseudo-responses and select the ones which closely match. Based on the selection, the positive or corrective feedback was generated by the system to allow the learner to monitor his/her response by comparing the given options with their performance, identify the gap and take control action plan to improve. However, we found that only 24 % of students were able to do this metacognitive activity efficiently, while most of the students either over-rated or under-rated their performance and failed to take control action plan to modify their responses. This shows that novices are not good at self-evaluation task, and the need to scaffold the evaluation task is necessary.

In this version of Fathom-Ver3, we aimed to scaffold learners to effectively evaluate and control their ER skills by providing external feedback before the self-evaluation activity. After learners perform the activity, the system will evaluate the level of the ER skills of the learner and accordingly provides positive or corrective feedback based on the ITF model (Narciss, 2013). The cognitive biases that novice exhibit in applying ER skills are identified based in our previous studies and the research studies in ill-structured problem-solving, as shown in table 7.1.

Table 7.1. Cognitive biases of novices in applying ER skills

ER skills	**Novice cognitive biases**	**Outcome/error**
Understand the problem from multiple stakeholders' perspectives.	Tend to think only from a developers' perspective and does not visualize the system as a whole.	Identify one or two entities and interactions that are salient.
Formulate problem	Not able to formulate the goal from user-point of view.	The goal is broadly written, e.g., design library management system, without addressing who are the users, broad requirement, and

		purpose.
	Not able to abstract a complete set of sub-goal at a higher level of abstraction.	Incomplete problem formulation, reduces the solution space search and solution quality is affected.
Generate alternative solutions	Generate single solution using design options that are recently studied or easy to use.	The solution may be sub-optimal.
Evaluate and select a solution	Face difficulty in identifying criteria and constraints.	Select and justify without thinking of selection criteria, thus leading to weak justification.

In our system, we attempt to design the feedback generation mechanism to address the cognitive biases shown in table 7.1. The evaluation is done by counting or matching the learner's response with pre-defined keywords or phrases and accordingly the gaps are identified and specified in the feedback with appropriate explanation to help learners in identifying the gap and take appropriate action. In corrective feedback, the learners are guided to see hints, examples, refer related task, and review peer's responses to help them understand the desired level of performance and monitor the gaps and take appropriate action to improve.

Collaboration with peers is a new feature added in Fathom-Ver3 and is designed based on Collaborative Learning Conversation Skills Taxonomy (Figure7.3). It enables learners to understand how their peers have performed, evaluate peers' responses, and chat with them to ask clarification questions, appreciate or give suggestions on peer's responses. Later, the learner is given the option to compare their work with peers by reflecting on how their work is different from their peer's work. Reflection prompts are given to scaffold this metacognitive activity to monitor and take action to improve effectively.

Figure 7.3. shows the redesigned feedback generation feature of Fathom-Ver3.

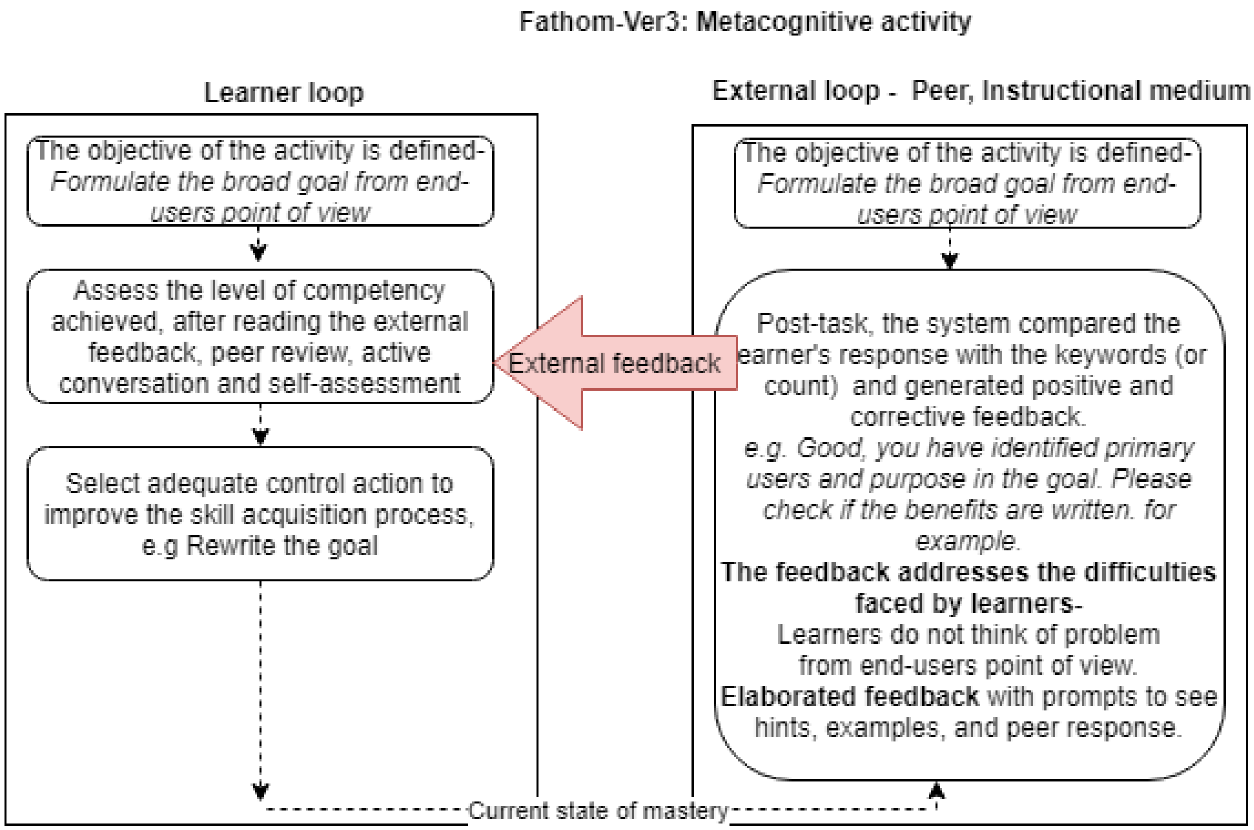


Figure 7.3. Feedback generation mechanism in Fathom-Ver3.

Table 7.2 shows the comparison between the design features of Fathom-Ver2 and Ver3.

Table 7.2. Design features of Fathom-Ver1 and Fathom-Ver2

Design features	**Fathom-Ver2**	**Fathom-Ver3**
Cognitive support		
Semi-structured guidance	✔	✔
Prompts, Explanation of new terms	✔	✔
Worked example, and videos illustrating the use of cognitive tools	✔	✔
ER cognitive tools.	✔	✔
Metacognitive support		
Self-evaluation activities	✔	✔
Analysis of case study (new problem)	✔	x
System-generated feedback	Only one activity	For all activities
Peer-review	x	✔
Chat facility with peers for active collaboration	x	✔

The design of Fathom-Ver3 is discussed in detail in the next section.

7.2 Solution Design- Fathom-Ver.3

The goal of the third cycle of DBR was to enhance the scaffolding mechanism for a metacognitive activity to enable learners to monitor and control the cognitive tasks related to ER skills.

7.2.1 Design conjecture

The design conjecture is that by scaffolding learners in metacognitive activity, the learners will be able to evaluate their ER skills and devise a control action plan to improve the skill. The design features are based on the principles of Interactive Tutoring Feedback Model (ITF-model), peer review, and collaboration skills for effective ill-structured problem-solving (discussed in section 7.1).

7.2.2 Design features

The pedagogical features for doing the cognitive tasks related to ER skills are the same as those in the previous version of Fathom- Ver2. The metacognitive activities of self-evaluation and reflection are supported with system-generated feedback, peer review, and active learning based collaboration conversation. The pedagogical design of Fathom-Ver3 is shown in figure 7.4.

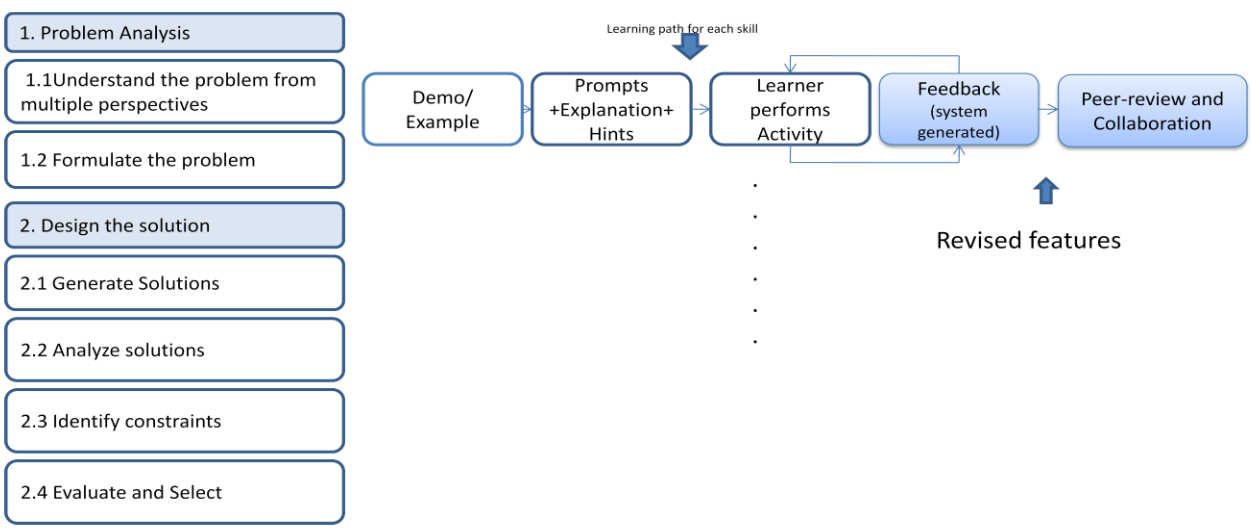


Figure 7.4. Pedagogical features of Fathom-Ver3 with revised metacognition activity

The design features retained in Fathom-Ver3 from the previous version of Fathom-Ver2 are:

Problem posed: The learner is posed with a library design problem:

"The college library maintains books of various departments like Computer engineering, Electronics engineering, etc. The library staff maintains records of all books, return and issue of books to students and teachers. It was becoming tedious for teachers and students to search for a book in the library. The librarian decided to provide online service for students and teachers to search the availability of the books. The system will search for the availability of the given book and display if the book is available/not available with its shelf number(location of the book).

Your task is to design software for the requirement stated by the librarian, using appropriate data structure and algorithm."

Problem-solving phases*:* The learner is guided through the ER activities in both problem and solution space. In problem space, the expand activity is to understand the problem from multiple perspectives and reduce activity is to formulate the goal and data and operations for achieving the goal. In solution space, the expand activity is to generate alternative solutions and reduce activity is to identify the criteria and constraints to evaluate alternative solutions and, select one and justify.

Prompts and additional scaffolds: For doing ER activity, the scaffolds are: prompts, explanation of new terms, worked examples and videos to demonstrate the use of cognitive tools using example, hints and domain-related notes.

Redesigned activities:

In *formulate problem* activity, the prompt in previous version-Fathom-Ver2 was: *"In the diagram below, identify the entities and interactions for achieving the goal and write as sub-goals."* We observed during study-3 that students were having difficulty in understanding the prompt as the term sub-goals was ambiguous and not specific to the domain.

The *formulate problem* activity is improved to help learners in abstracting the data and operations by choosing from the entities and interactions drawn in previous step-*understand the problem.*

In current version-Fathom-Ver3, the activity is divided into two sub-activities with revised prompt and design, as shown in figure 7.5. The prompt for activity1 is: *"from the library diagram, choose entities that are relevant for solving the problem and for each entity write the data attributes in the format given below"*, supported with explanation, example and a dropdown list preloaded with entities and interactions (figure 7.5), instead of a single textbox given in Fathom-Ver2. The two drop-down list is populated with entities and interactions separately, and learners can select and represent them into data items and operations.

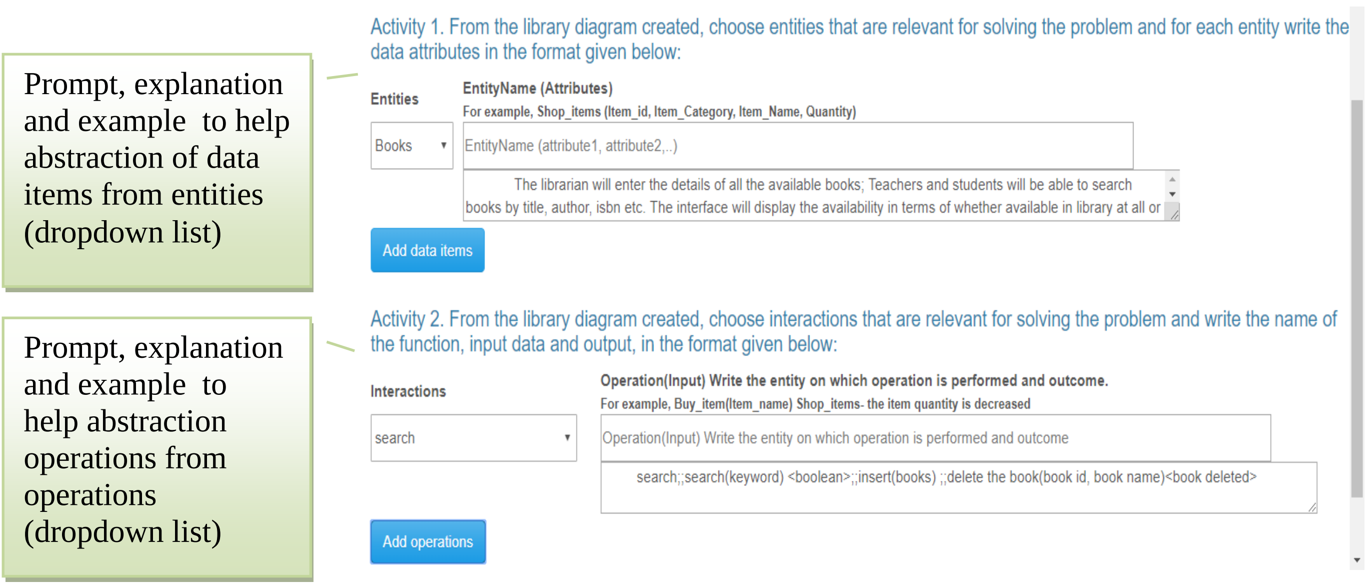


Figure 7.5. Improved design of Formulate-sub-goal activity

The new features added are:

System-generated feedback is generated by evaluating the gaps and providing an elaborated explanation on what are the gaps and the action plan to monitor and revise the skill. The positive or corrective feedback is generated at the end of each activity after learner saves his/her response. Figure 7.6 shows the feedback generated at the end of step- *Formulate goal.*

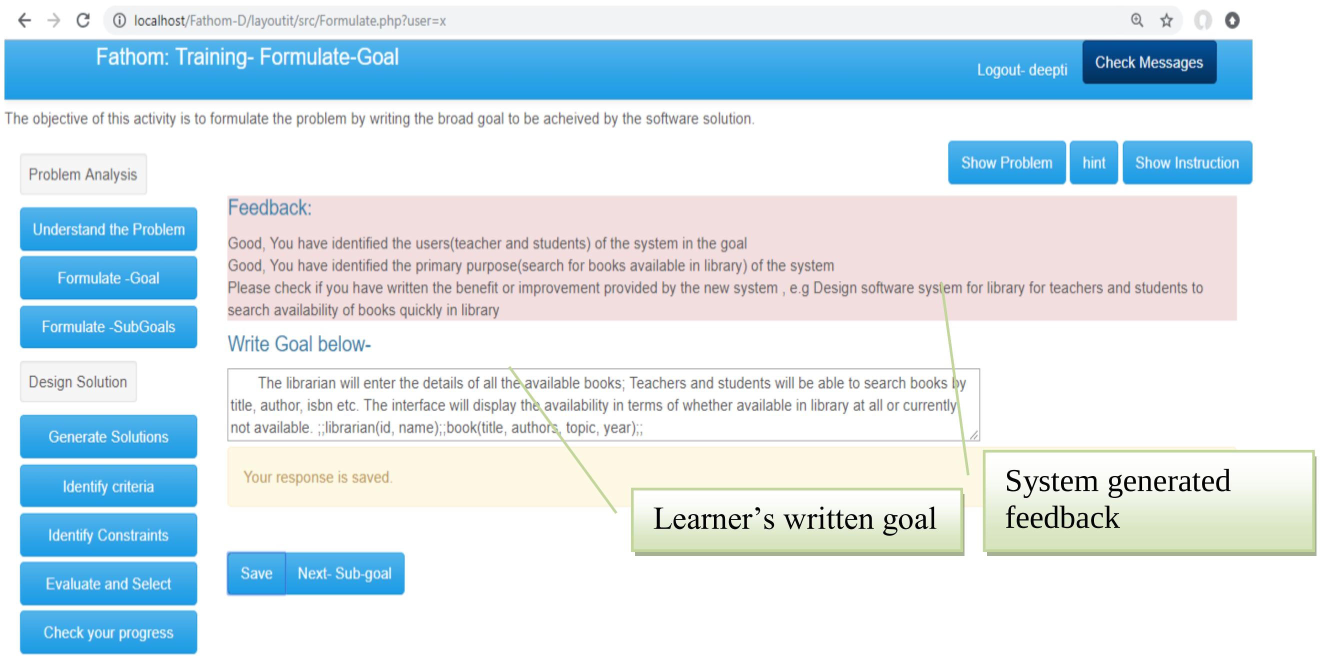


Figure 7.6. Fathom-Ver3: Formative feedback in step Formulate: Goal

As shown in figure 7.6, the feedback message is generated after the system has evaluated the goal written by the learner in the text-box. The goal is evaluated against the rubric: primary users (teachers, students, librarian) identified, and the broad requirement (search for availability of books) and benefits (saves time) of the software system are written. The system matches the goal written by the learner with pre-defined keywords and phrases and based on the keywords present or missing, the formative feedback is generated. It addresses the competency level achieved by the learner, to help in reassuring the learner about the cognitive processes used correctly, and lacunas, if any, are specified by explaining the element that is missing, elaborated with an example or hint. This allows the learner to monitor and evaluate his response by comparing against the given pointers and example and take corrective action.

Peer-review: This is a new feature added, which allows collaborating with peers during each activity. On selecting this option, the list of online users is listed, and the learner can select one of the peers to collaborate. Collaboration aim is to allow learners to see how their peers have performed the activity, evaluate their responses against the given scale of high-medium-low, and justify the evaluation. This helps the learner to see other's perspectives of doing, think, and evaluate the gap.

Since the peer review is an asynchronous activity, the situation may arise where none of the peers may have reached the activity in which the learner is currently performing. To avoid empty peer list, we added three pseudo-peers with preloaded responses representing peers at different levels: low, medium, and high level. The learner can also chat with these pseudo-users by asking clarification questions, appreciate and suggest changes. Based on the questions asked, the system compares it with preloaded keywords and fetches the corresponding chat message from the database.

Active collaboration conversation: Based on the empirical findings (Xun, 2013) which suggest that by just seeing peer's responses, the learner does not benefit much, we have incorporated chat facility to allow learners to explain, clarify or justify their opinions. The chat facility is based on active collaboration conversation taxonomy (Soller, 1999). Just below the peer's response, as shown in figure 7.7, the chat buttons are provided to guide the conversation in the direction of request to clarify, inform suggestions, and motivate. The clarify button is used to ask a peer to clarify any queries regarding the response, suggest button is used to suggest any changes in peer's response and appreciate button is used to motivate the peer for their good work.

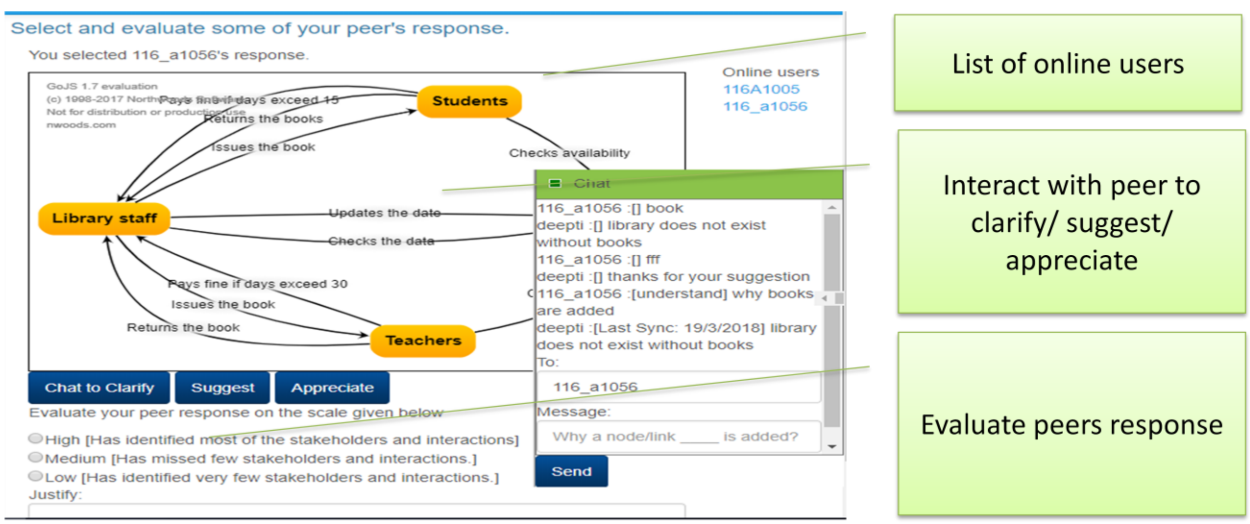


Figure 7.7. Peer review and collaboration conversation

The overall design features of Fathom-Ver3 are shown in figure 7.8. In the next section, the evaluation study done to investigate the effectiveness of Fathom-Ver3 is discussed.

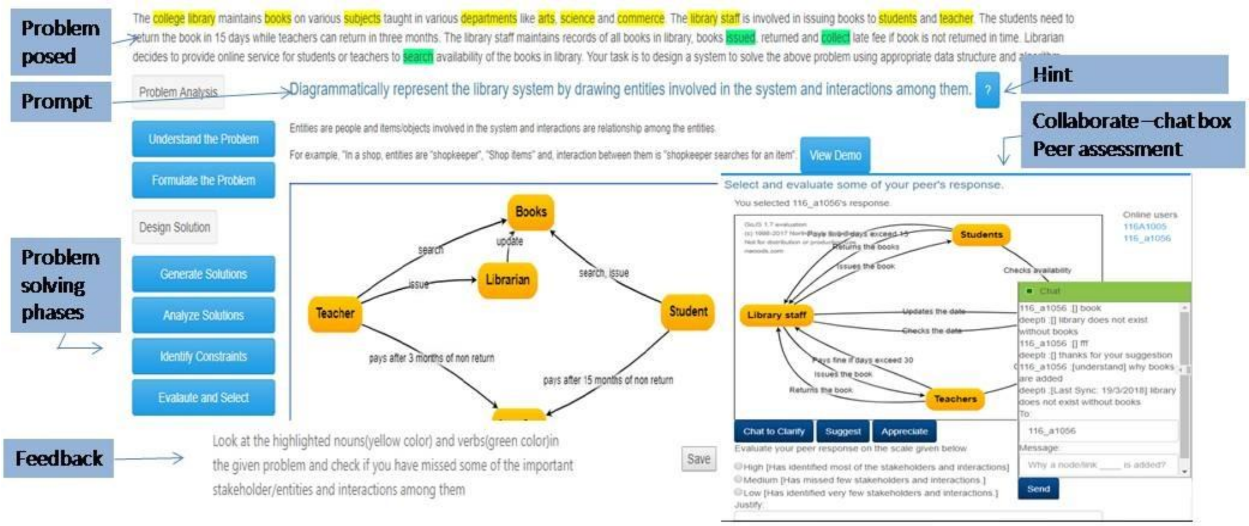


Figure 7.8. Overall design features of Fathom-Ver3

7.3 Evaluation: Study 4- Effectiveness of the design and pedagogy in learning ER skills.

In the evaluation phase, our goal was to study how novices learn ER skills in Fathom-Ver3, equipped with the cognitive tools, pedagogical features, and revised metacognitive activities.

To study the effect of each variable on learning of ER skills by novices, we have devised the following three research questions, which we will be investigating in study 4:

- RQ 4.1-How effective are the cognitive tools of Fathom-Ver3 in learning of ER skills?
- RQ 4.2- How effective are the pedagogical features in Fathom-Ver3 in learning ER skills?
- RQ4.3-How effective are the metacognitive activities in Fathom-Ver3: formative feedback and collaboration activity, in enabling students to monitor and evaluate their ER skills?

7.3.1 Study design and participants

We did a field-study with 50 undergraduate computer engineering students. The participants were selected based on purposive sampling techniques as the pre-requisite is to have students who have completed the data structures and algorithm courses. The research study design was experimental pretest-posttest. The design of the study is shown in figure 7.9.

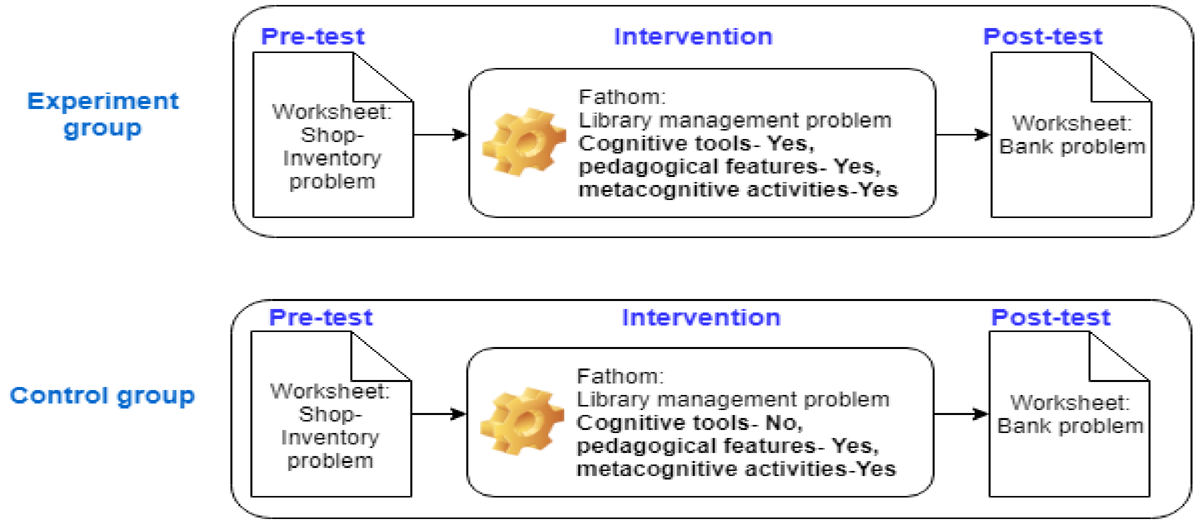


Figure 7.9. Research study design

The experimental pretest-posttest design was used for two purposes:

i. The experiment-control group design was used to measure the difference in the performance of ER skills with and without cognitive tools.

ii. The pretest-posttest design was used to measure the effectiveness of the Fathom (with all features) in learning ER skills.

7.3.2 Procedure

The procedure of the study is as follows:

Pretest: First, the participants were given a shop-inventory problem to be solved on a worksheet, as shown in figure 7.10.

Roll no:

Pretest

The supermarket has of items of various categories- grocery, bakery, cleaning, vegetables, etc. Everyday hundreds of customers visit the supermartket and the shopkeeper want to ensure that none of the item goes out of stock. The shopkeeper wants a software to be designed to keep stock of all items and track items whose quantity is below the threshold. The software should generate a report at the end of the day, of items whose quantity is below threshold and needs to be ordered.

Your task is to design a software system to solve the above problem using appropriate data structure and algorithm.

Solve by following the steps-

v. Write the broad goal to be achieved, e.g. Design a software system for...
vi. Write the data and operations to be performed by the system
vii. Design solution by choosing appropriate data structure and algorithm, for storing data and performing operations respectively.
viii. Justifywhy the solution is good for the given problem and better than other alternative solutions.

Figure 7.10. Pretest worksheet

The participants took 30 minutes to solve the problem, and after completing the worksheets were collected from them.

Intervention- Immediately after pretest, the students were divided into two groups. The students were assigned to two groups according to their enrolment-ids. The first 30 students were assigned to the experiment group, and the next 30 students were assigned to the control group. The equivalence of both groups in terms of data structures conceptual knowledge was verified based on the quiz (appendix 1) taken before the study. No significant difference (p=0.47) was seen. The experiment group interacted with the Fathom-Ver3 with all features: cognitive tools, pedagogical features, and metacognitive activities. The control group interacted with Fathom-Ver3 with no cognitive tools; other features were retained as it is. The cognitive tools removed were drawing of the entity-interaction diagram in problem space, drawing of attribute listing map in solution space, pros and cons table, and decision matrix. Both the groups solved library management problem. The participants interacted with Fathom-Ver3 for almost 2 hours.

Posttest- After the intervention, during posttest, the participants were given bank problem to be solved on a worksheet as shown in figure 7.11. The participants were told to apply the ER skills learned during the intervention. The participants took almost 30 minutes to solve the posttest problem.

Roll no:

Post test

A new branch of bank XYZ is opened in a city. The bank provides following services to customers: maintaining customer accounts, withdraw amount, deposit amount, and check balance in their account. The bank manager wants a software system to be designed for its customers to provide a service of checking their account balance, while other services will be offline. The system should take customers' account_no as input and display the balance in the account.

Your task is to design software for the above requirement using appropriate data structure and algorithm

Figure 7.11. Posttest problem

7.3.3 Data sources

As the purpose of the study was to measure the effectiveness of Fathom-Ver3 in learning ER skills, we collected data from multiple sources.

The data sources are:

1. The pretest, posttest worksheets of students.
2. Learner artifacts generated during interaction with Fathom.
3. Log data- time-stamped user clicks
4. Screen recording-using Camtasia software during the intervention
5. Focus group Interview – Four students were interviewed at the end, to know their perceptions on how they learned ER skills using Fathom-ver3, how each feature helped in

doing the ER activity during the intervention and in the posttest, and the difference in solving the problem from pretest to posttest.

6. Student perception survey on usefulness and usability.

The corresponding data sources used to answer each RQ 4.1 to RQ 4.3 is shown in table 7.3.

Table 7.3. RQ to data source mapping

RQ	**Data Sources**
RQ4.1- How effective are the cognitive tools of Fathom-Ver3 in learning of ER skills?	Pretest and Posttest scores for both experiment and control group
RQ4.2- How effective are the pedagogical features effective in learning ER skills?	The data sources are pretest-intervention-posttest scores, log data analysis, and interview data, student perception survey.
RQ4.3-How effective is the metacognitive activities: formative feedback and collaboration activity in enabling students to monitor and evaluate their ER skills?	Log data analysis, interview data, and student perception survey.

7.3.4 Data Analysis

The student artifacts generated during pretest, intervention, and posttest was evaluated using a rubric (table 7.4). The rubric designed in DBR cycle 2 is used with minor enhancements to assess the ER skills in both problem space and solution space. For example, in problem space, the learner's ability to expand is assessed by counting the number of distinct entities and interactions identified. In solution space, the ability to expand is assessed by counting a number of alternative solutions generated. The reduce skill is the ability to formulate a goal and abstract data and operations and select an optimal solution based on the constraints. The quality of abstraction of a module are measured based on: coupling (independent modules); cohesion (well bounded behaviour); sufficiency (enough operations defined to achieve the desired behaviour); completeness (all operations defined to achieve the desired behaviour); and primitiveness (low level operation) (Booch, 2009). We have adopted these quality factors to assess the quality of problem formulation and solution. In our rubric, the quality of problem formulation is assessed based on two factors: sufficiency and completeness of data items and operations defined in terms of achieving the stated goal. The quality of the solution is measured in terms of completeness (all data items and operations are designed using data structure and algorithm) and correctness (valid data structure and algorithm used) in achieving the stated goal.

Table 7.4. Rubric to assess ER skills

ER skill	**ER activities**	**Rubric**
Problem space		
Expand	Understand the problem	***Score-*** Count of entities and interactions identified
Reduce	Goal	***High*** *(score-3)***-** The goal has addressed the following points- requirements broadly defined, primary users, and benefits written. ***Medium*** *(score- 2)***-** The goal missed one of the above points ***Low*** *(score- 1)*- The goal missed most of the points
	Sub-Goal (quantity & quality)	***High*** *(score-3)*- Identified all the data items and operations for achieving the stated goal. ***Medium*** *(score- 2)*- missed identifying one of the data items or operations that will achieve the goal ***Low*** *(score- 1)*- missed identifying data item or operation which are important for achieving the goal.
Solution space		
Expand	Generate solution- Quantity	**Score-** Count of alternative solutions identified
	Solution quality	***High*** *(score-3)***-** The solution is complete and correct. In the solution, the data structure is valid, and the operations are mapped to correct algorithms. The clarity is high on how data is stored and how the algorithms are used. ***Medium*** *(score- 2)*-has missed identifying valid data structure or an algorithm for some data items and operations. ***Low*** *(score- 1)*- the solution is incomplete Or incorrect.
Reduce	Criteria	***High*** *(score-3)*- all possible criteria correctly identified, ***Medium*** *(score-2)* - missed a few,

		Low*(score-1)*- identified few.
	Constraints	***High*** *(score-3)*- all possible constraints are correctly applicable to the problem, ***Medium*** *(score-2)* - missed a few, ***Low****(score-1)*- identified few.
	Select and justify	***High*** *(score-3)*- selected the correct DS and justified how the solution satisfies all the constraints applicable to the problem and is optimal compared to other alternatives, ***Medium*** *(score-2)* - selected a correct DS and justified without considering all constraints or did not consider other alternatives. ***Low****(score-1)*- selected an incorrect DS and did not justify based on the constraints applicable to the problem.

The final rubric was validated with an expert instructor with more than ten years of teaching experience, who has taught data structures course more than three times to undergraduate engineering students. Both of us independently assessed artifacts, produced during the intervention. Cohen's Kappa was run to determine if there was inter-rater reliability between the problem formulation and solution quality scores of both the raters. There was good agreement between the two raters, $\kappa = .634$, $p < .046$.

The log data consists of log records: <userid, button clicked, time stamp>. The log data and screen recordings were analyzed to calculate the time spent in each ER activity and total time, the sequence of interactions of each student, for example, student 1: understand phase- viewed hint, perform an activity, read feedback, view video, redo activity, collaborated with a peer, etc. These analyses will give us insights into how students interact with different features in the environment, time spent, and its effect on learning ER skills.

The interview data were transcribed, and the content analysis was done to identify meaningful units of analysis. These units were mapped to its corresponding codes in terms of how the students perceived the activities to be useful in learning ER skills, pedagogical features, the difference in problem-solving from pretest to posttest, and challenges.

The student survey was taken to measure the usability of the tool using System Usability Scale (SUS) tool. Total 43 students took the survey and the SUS score calculated was 53.

7.3.5 Results

Student artifacts and corresponding ER scores

i. The average scores of pretest-intervention-postest scores are compared for the experiment group based on t-test (p-value) and effect size (Cohen's d value), as shown in table 7.5. During pretest, the students were asked to solve the problem in a conventional way using four steps: (i) write a goal, (ii) write sub-goals (data items and operations), (iii) design solution using appropriate data structure and algorithm and (iv) justify. Thus in table 7.5, in pretest column, the sub-skills: understand the problem, generate solutions (only one solution generated), and identify criteria/constraints are not evaluated.

The scores show significant gain from pretest to posttest in

- quality of problem formulation (p=0.05, effect size= 0.66) and
- solution quality (p=0.00, effect size= 1.23)
- justification (p=0.01, effect size 1.24)

Table 7.5. Pretest-intervention and posttest average score comparison.

N=24	**Average scores**			**T test- (P-value)**	**Effect size Cohen's d value**
	Pretest	**Posttest**	**Intervention**	**Pre-Posttest**	
Understand the problem (Count)	NA	5.20	9.92	NA	NA
Goal (max score=3)	1.80	2.21	2.68	**0.03**	0.85
Sub-Goal (max score=3)	1.42	1.88	2.08	**0.05**	0.66
Generate solution-Quantity	1.00	2.47	1.86	NA	NA
Solution quality (max score=3)	1.25	1.88	1.75	**0.00**	1.23
Criteria (count)	NA	1.92	1.93	NA	NA
Constraints (count)	NA	1.50	1.62	NA	NA
Select and justify(max score=3)	1.33	2.00	2.06	**0.01**	1.24

ii. The comparison of average scores of experiment and control group is shown in table 7.6. The equivalence of both the groups in solving design problem is established as there is no significant difference in each sub-skills during the pretest. The experiment group showed a significant gain over control group during posttest for skills: understand the problem (p<=0.00), goal (p<=0.00), number of solutions generated (p<=0.03) and solution quality (p<=0.00).

Table 7.6. Experiment-control group score comparison.

	Average-Pretest			**Average- Activity**			**Average- Posttest**			
	Experiment Group	**Control Group**	**P-Value**	**Experiment Group**	**Control Group**	**P-Value**	**Experiment Group**	**Control Group**	**P-Value**	**Effect size-Cohen's d value**
Understand the problem (Count)	-	-	-	9.92	11.04	0.17	5.20	2.95	**0.00**	0.99
Goal (max score=3)	1.80	1.96	0.14	2.68	2.62	0.35	2.21	1.69	**0.00**	1.01
Sub-Goal (max score=3)	1.42	1.27	0.19	2.08	2.19	0.25	1.88	1.56	0.07	0.48
Generate solution-Quantity	-	-	-	1.86	1.65	0.26	2.47	1.75	**0.03**	0.61
Solution quality (max score=3)	1.25	1.08	0.05	1.75	1.58	0.20	1.88	1.09	**0.00**	1.81
Criteria (count)	-	-	-	1.93	1.68	0.16	1.92	1.50	0.06	0.56
Constraints (count)	-	-	-	1.62	1.68	0.39	1.50	1.50	0.50	0.00
Select and justify(max score=3)	1.33	1.36	0.44	2.06	1.92	0.22	2.00	1.89	0.26	0.24

Log data analysis

The log data analysis was done to calculate the time taken in each ER activity and the sequence analysis of the interactions of learners during the intervention.

i. Analysis of time spent on each activity

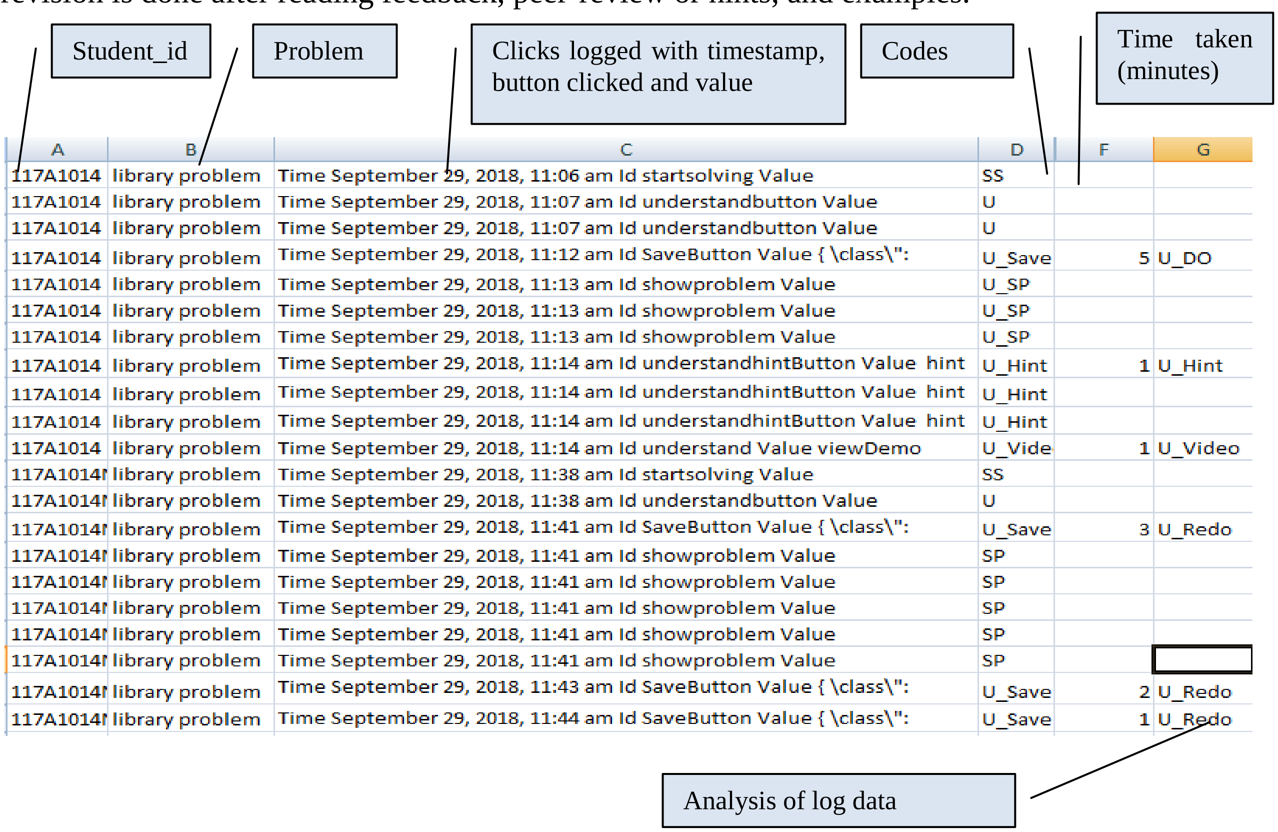
Average time (minutes) taken by learners to complete each ER activity, and a comparison of both groups are shown in table 7.7. The experiment group took significantly more time in understand problem activity than the control group, and the control group took significantly more time to evaluate solutions than the experiment group. However, there is no significant difference in total time spent to solve the problem by both groups.

Table 7.7. Comparison of time taken to complete activities by experiment and control group.

	Experiment group-Average time (minutes)	**Control group Average time (minutes)**	**T-test (P value)**
understand problem	27.6	10.2	0.00*
formulate problem	28.7	33.9	0.21
Generate solutions	18.1	12	0.11
evaluate solutions	3.86	9.2	0.00*
select solution and justify	2.48	2.8	0.45
Total time	**74.21**	**68.1**	**0.22**

ii. Sequence analysis

The log data was further analyzed to find the sequence of actions performed by the students during intervention and time taken in each activity. The log data captured the triplet at each click as shown in figure 7.12 (first three columns): <student_id, problem, timestamp, and name of the button clicked>. These were analyzed to identify the sequence of activities performed by students in each learning activity, as shown in the next three columns in figure 7.12. The codes were assigned to each record based on the action performed. If the learner has clicked understand problem button, then the code assigned is “U,” later if the learner has clicked on save button, then the code assigned is "U_save" and so on. For each save button clicked, the time spent on doing the activity is recorded. Further analysis is done to find if the learner has revised the activity. This is done by checking if the learner has clicked the save button second time after a window of one minute, which means that the learner has revised his/her responses and we code this action as: U_Redo, where the first letter indicates the activity. Also, we checked if the revision is done after reading feedback, peer-review or hints, and examples.



A	B	C	D	F	G
117A1014	library problem	Time September 29, 2018, 11:06 am Id startsolving Value	SS		
117A1014	library problem	Time September 29, 2018, 11:07 am Id understandbutton Value	U		
117A1014	library problem	Time September 29, 2018, 11:07 am Id understandbutton Value	U		
117A1014	library problem	Time September 29, 2018, 11:12 am Id SaveButton Value { \class\":	U_Save	5	U_DO
117A1014	library problem	Time September 29, 2018, 11:13 am Id showproblem Value	U_SP		
117A1014	library problem	Time September 29, 2018, 11:13 am Id showproblem Value	U_SP		
117A1014	library problem	Time September 29, 2018, 11:13 am Id showproblem Value	U_SP		
117A1014	library problem	Time September 29, 2018, 11:14 am Id understandhintButton Value hint	U_Hint	1	U_Hint
117A1014	library problem	Time September 29, 2018, 11:14 am Id understandhintButton Value hint	U_Hint		
117A1014	library problem	Time September 29, 2018, 11:14 am Id understandhintButton Value hint	U_Hint		
117A1014	library problem	Time September 29, 2018, 11:14 am Id understand Value viewDemo	U_Vide	1	U_Video
117A1014I	library problem	Time September 29, 2018, 11:38 am Id startsolving Value	SS		
117A1014I	library problem	Time September 29, 2018, 11:38 am Id understandbutton Value	U		
117A1014I	library problem	Time September 29, 2018, 11:41 am Id SaveButton Value { \class\":	U_Save	3	U_Redo
117A1014I	library problem	Time September 29, 2018, 11:41 am Id showproblem Value	SP		
117A1014I	library problem	Time September 29, 2018, 11:41 am Id showproblem Value	SP		
117A1014I	library problem	Time September 29, 2018, 11:41 am Id showproblem Value	SP		
117A1014I	library problem	Time September 29, 2018, 11:41 am Id showproblem Value	SP		
117A1014I	library problem	Time September 29, 2018, 11:41 am Id showproblem Value	SP		
117A1014I	library problem	Time September 29, 2018, 11:43 am Id SaveButton Value { \class\":	U_Save	2	U_Redo
117A1014I	library problem	Time September 29, 2018, 11:44 am Id SaveButton Value { \class\":	U_Save	1	U_Redo

Figure 7.12. Log data captured and analysis (codes and time taken)

iii. Interview data

The interview data analysis was done to identify student's perception on how the features of Fathom-Ver3, were useful in learning ER skills, how each feature helped in doing the ER activity during the intervention and in the posttest, and the difference in solving the problem from pretest to posttest.

The overall learning of ER skills perceived by the students during the training with Fathom-Ver3 are listed below:

1. Students got a basic idea of how to approach a problem after the intervention. The corresponding quote – "*I got a basic idea to how to construct, how to approach the problem because the first time (pretest) I did not know how to approach itself."*
2. During the problem analysis phase, the drawing of the diagram in "understand the problem" activity helped to visualize all entities present and how they are linked to each other and identify basic requirements that need to be satisfied by the software. The corresponding quote of one of the student is: *"it helped to visualize the problem, and it helps me to see that all the other entities are linked so that it simplified the problem."*
3. The understand activity helped to break the problem into sub-problems. The corresponding quote: *"In pretest, I could not think of alternative solutions because did not know how to break the problem and think of each sub-problem as a separate unit which has different solution options."*
4. In the solution design phase, the attribute listing map helped to think of each entity as a separate unit to branch out for more sub-branches in terms of generating alternative options. Generating solutions helped them to think of evaluating based on various criteria and select, as per the quote *"When we think of more solutions, then we decide which one is better, easier or presentable for the user who is using it. It is done at the same time as working with single solutions."*
5. The additional support-explanation of new concepts with an example, hint and additional recourses (pros and cons analysis of DSA, execution time of each operation), helped to understand the activity to be performed.
6. Examples were helping to understand in what direction to think.
7. The feedback helped to understand the gap and was easy to understand.

The challenges faced by students in pretest were-

1. Students did not know how to approach the problem.
2. They tend to write programs instead of understanding the dynamics of the problem. They usually got lost and were not clear what to do and what not to do.
3. They start solving without considering all the entities and later when they realize, it's too late to come back and modify. The quote – "*I knew some entities were there. But I did not consider all the entities. So I work on some entities. And then after that, I let it struck me*

that I need to add more entities. So at the moment I already proceeded half the way, so I could not add or incorporate in pretest".

4. Thinking more about low-level functions to be used in the program.
5. Could not think of alternative solutions because did not know how to break the problem and think of each sub-problem as a separate unit which has different solution options.
6. Students select the data structure or solution that they are more comfortable with or easy to use. The quotes were-

> ***Researcher****[00:15:52] In pretest were you thinking of alternative solutions.*
>
> ***Student:*** *[00:15:57] No no no. I just stuck to one and worked on that.*
>
> ***Researcher:*** *[00:16:01] How do you select*
>
> ***Student:*** *[00:16:07] It comes to me first. Or the one which I am very used to and I can do better.*

iv. Student survey for measuring Usability:

The SUS score of 53 shows that the system is not very good in terms of usability. The design has to be improved to make it more user-friendly and usable.

7.3.6. Discussion of RQs

Next, we will answer our RQs based on the results-

- RQ4.1- How effective are the pedagogical features of Fathom in learning ER skills?
- RQ4.2- How effective are the cognitive tools of Fathom in learning ER skills?
- RQ4.3-How effective is the metacognitive activities: formative feedback and collaboration activity in enabling students to monitor and evaluate their ER skills?

RQ1- How effective are the pedagogical features of Fathom in learning ER skills?

The significant gain from pretest to intervention in quality of problem formulation, solution quality, and justification shows that the pedagogical features were effective in doing ER skills.

In *understand the problem* activity, after reading the prompt, hint, and watching the video, students were expanding the problem space effectively by drawing entities and interactions. Based on the scores in Table 7.5, students were able to draw on average ten entities and interactions and average five in the posttest. The log data analysis showed that students were able to understand the activity to be performed by reading the prompt, watching the video, using a hint and collaborating with peers to do or improve the performance. Based on interview analysis, students perceived that the drawing of the diagram in *understand the problem* activity helped to visualize all entities present and how they are linked to each other.

In *problem formulation* activity, after reading the prompt and explanation, the students were able to reduce the problem space by writing the goal to be achieved from the user point of view and abstract the data items and operations from the entities and interactions during the intervention. The quality of formulation of the goal, and sub-goals (abstraction of data items and operations) in terms of completeness, significantly improved from pretest to intervention ($p<-0.00$) and pretest to posttest ($p<=0.05$). During the interview, students perceived that in problem formulation activity, the goal was used to identify the variables for each entity.

In *generate solution* activity, as per log data analysis, it was seen that students were able to draw the attribute listing map and generate alternative solutions with the help of prompts, hints, and example videos. As per scores in table 7.3, on average, two (max-6, min-1) alternative solutions were identified during the intervention and three in the posttest. The quality of the solution significantly improved from pretest to intervention ($p<=0.00$) and from pretest to posttest ($p<=0.00$). During the interview, students perceived that the attribute listing map helped to think of each entity as a separate unit to branch out for more sub-branches in terms of generating alternative options.

In *evaluate* activity, students identified 2 to 3 criteria and constraints. During the interview, students perceived that they had difficulty in understanding criteria, but hint and additional domain-knowledge related recourses (pros and cons analysis of DSA, execution time of each operation) helped to understand. This activity also helped students to go back to problem space from solution space to identify the constraints applicable to the problem. Students perceived that generating solutions helped them to think of evaluating alternatives based on various criteria and select, as per the quote *"When we think of more solutions, then we decide which one is better,* *easier or presentable for the user who is using it. It is done at the same time as working with single solutions."*

In *select* and *justify* activity, as per log data analysis, the prompt, hint, and decision table was helping to rank and select the solution. This is also evident based on the significant improvement in reasoning and justifying how the selected solution satisfies the requirements in the given problem, from pretest to posttest ($p<=0.01$).

Overall, the pedagogical activities supported with a systematic guidance, prompts, explanation of new terms with an example, worked example, videos, and hints, helped students to understand the activity to be performed. Students perceived that prompts, hints, examples helped to understand in what direction to think. The significant gain from pretest to intervention implies that the pedagogical features were efficient in scaffolding students in doing ER skills.

The significant gain from pretest to posttest implies that students were able to learn and transfer ER skills in a new problem context when all the scaffolds were withdrawn. We will illustrate the learning of ER skills from pretest to posttest using a sample worksheet of a student.

During pretest, the problem given was: *"The supermarket has items of various categories- grocery, bakery, etc. Every day hundreds of customers visit the supermarket, and the shopkeeper wants to ensure that none of the items go out of stock. Design software to generate a report at the end of the data, of items whose quantity is below the threshold and needs to be ordered. Design using an appropriate data structure and algorithm"*.

The students were asked to solve the problem by following the steps:

1. Write the broad goal to be achieved, e.g., Design a software system for..
2. Write the data and operations to be performed by the system
3. Design solution by choosing appropriate data structure and algorithm, for storing data and performing operations, respectively.
4. Justify why the solution is good for the given problem and better than other alternative solutions.

In the sample worksheet of student S1 (figure 7.13), we will discuss the responses in each step 1-4.

1. Write the broad goal to be achieved, e.g., Design a software system for..

The goal is written at a very broad level as *stock management*, as shown in figure8.2. The student is not carefully reading the problem statement and understanding the main goal.

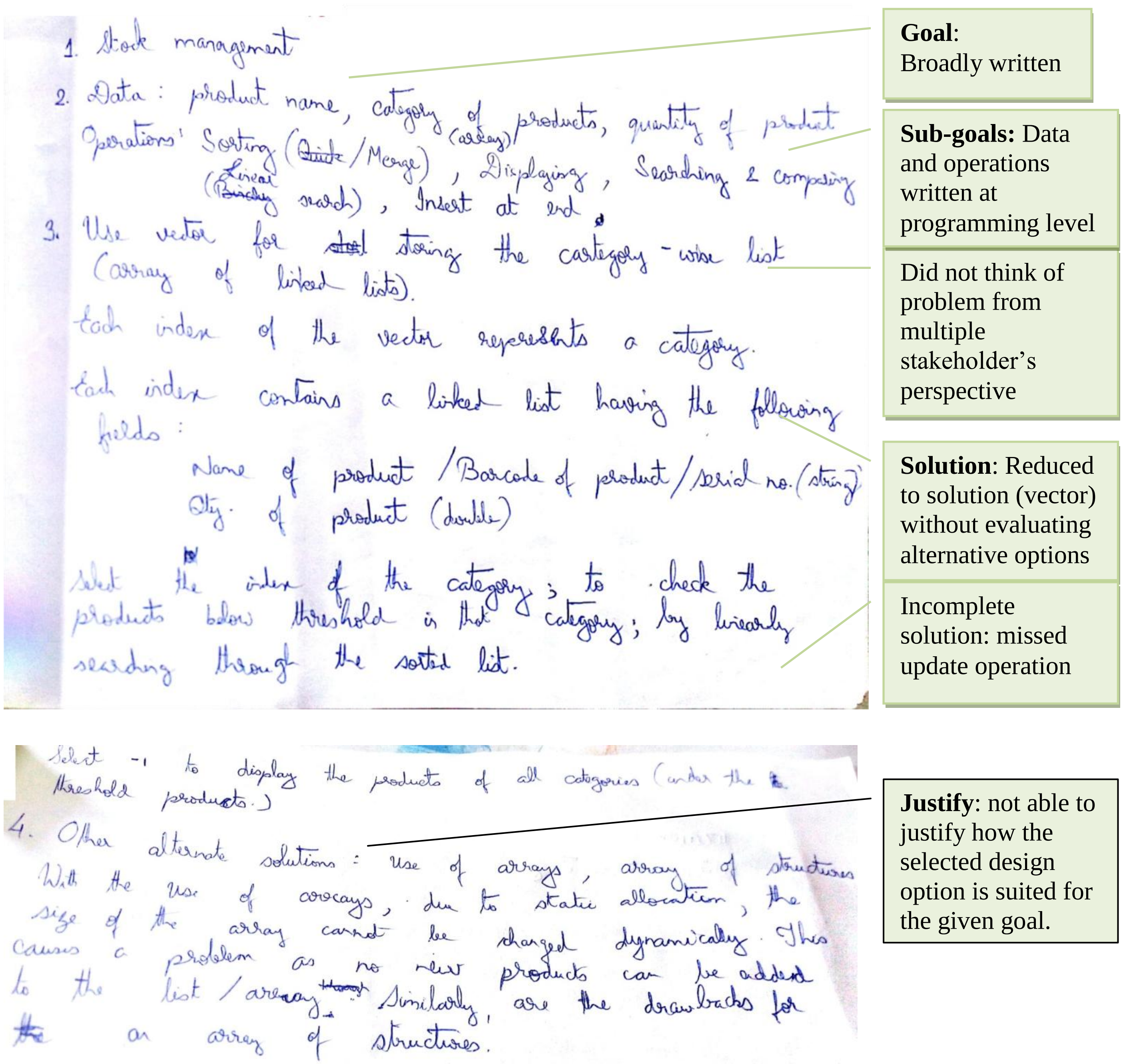


Figure 7.13. Pretest – solved worksheet of a student-S1.

2. Write the data and operations to be performed by the system

The problem formulation is done at a programming level *(Data: product name, category, and quantity, Operations: merge-sort, display, search & compare using linear search and insert at the end).* The students are not able to formulate at a higher level of abstraction and capture the dynamic functions that will change the values of the data items, for example, Item_sold (decrease the quantity), item_ordered (increase the quantity). This showed that students were converging early to the solution space without spending much time in understanding the problem from the perspective of stakeholders involved in the system, e.g., customer, shopkeeper, etc.

3. Design solution by choosing an appropriate data structure and algorithm.

In solution space, the student made a decision choice of selecting vector data structure for storing data. The student's selection criterion was based on selecting the solution that is easy to implement and not on the constraints in the given problem or on how well the solution achieves the goal or important sub-goals (search for an item and insert new items). The student justified his/her selection by stating the advantages of the selected data structure, which are not related to problem goals and important sub-goals.

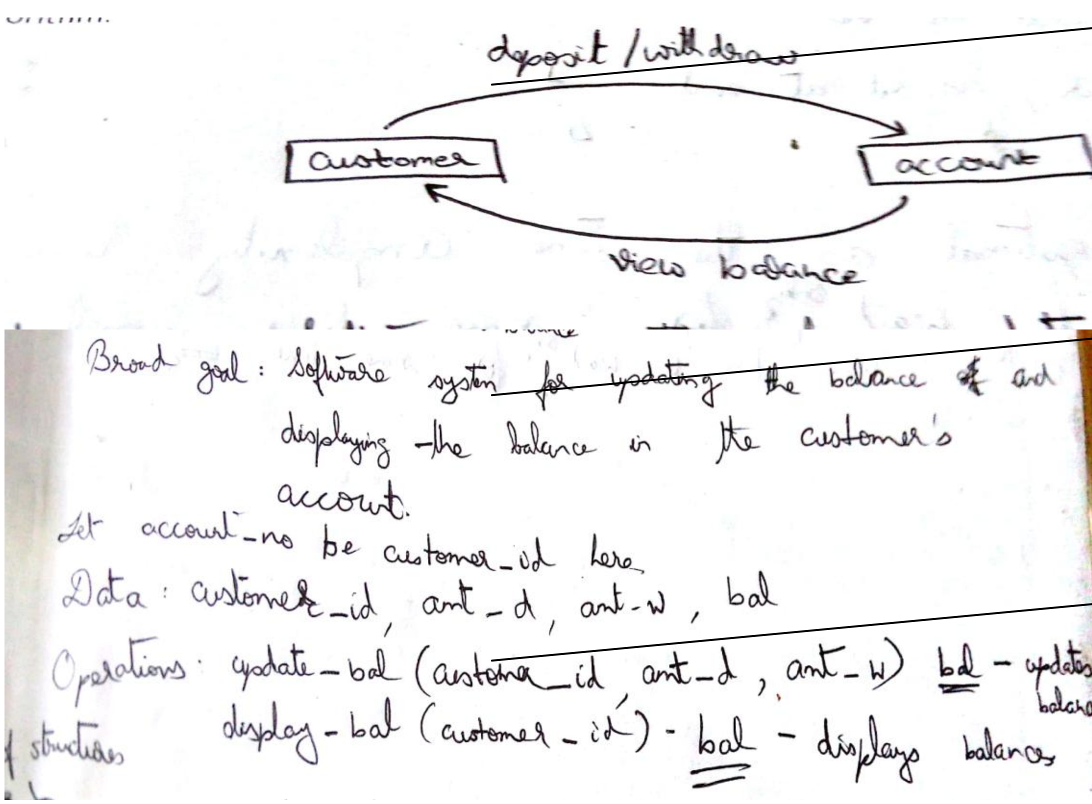
4. Justify why the solution is good for the given problem and better than other alternative solutions.

The student S1 justification (figure.7.13) shows that s/he selected vector over the array, as vector allows dynamic memory allocation for inserting a new product. The student is fixed to the initial solution and trying to fit its advantages to one of the not so important sub-goal in the problem.

This shows that students tend to take a solution-based approach (Carman, 2007) in which they reduce early to a solution without exploring other options and not selecting against the important sub-goals or constraints. Thus lack of ER skills leads to only a satisficing solution (not an optimal solution) and incomplete solution with weak justification.

The sample posttest worksheet of the same student S1 is used to illustrate the ER skills performed in posttest (figure 7.14).

The new problem posed was a bank problem – "*A new branch of bank XYZ is opened in a city. The bank provides the following services to customers: maintaining customer accounts, withdraw amount, deposit amount, and check the balance in their account. The software system* *has to be designed for its customers to provide a service of checking their account balance, while other services will be offline. Design software using appropriate data structure and algorithms".*



Problem space: Explored problem space using Entity-interaction diagram from customer and bank perspective.

Goal: written goal from user's perspective and addressed the specific requirements.

Sub-goals: able to abstract data and complete set of operations to achieve the stated goal.

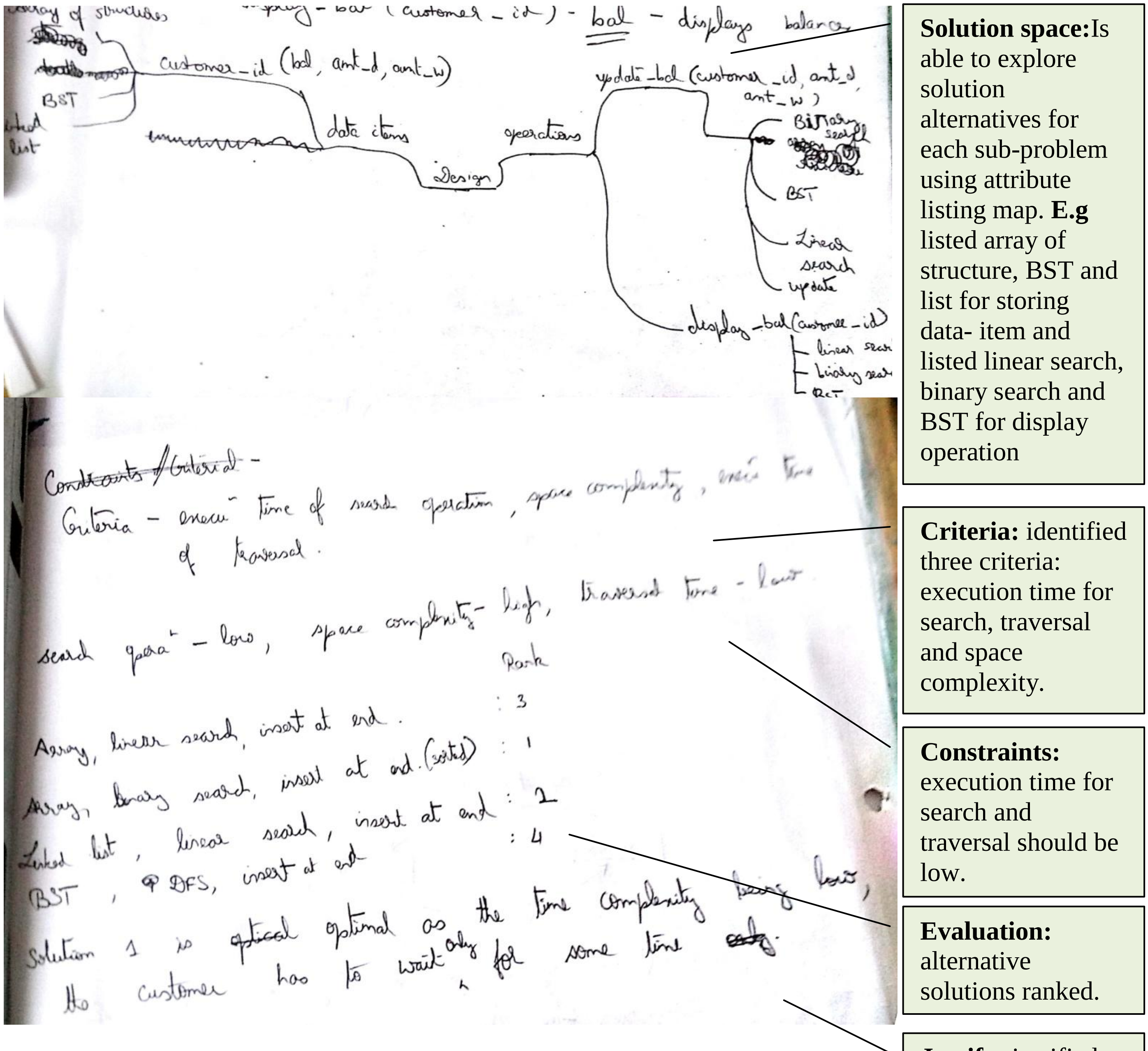


Figure 7.14. Posttest – solved worksheet of a student-S1.

ER Skills in problem space: During posttest s/he visualized the whole system by drawing entities and interaction from the customer and bank perspective. The learner was able to formulate the goal and sub-goals by capturing all the data items and well defined independent operations: update_bal (cust_id, amt_withdraw, amt_deposit), and display (cust_id). The learner was able to abstract at higher-level than at programming level and quality of the abstraction in terms of coupling (loose and independent), cohesiveness and completeness was maintained. The student was able to break the problem into sub-problems.

ER skills in solution space: In solution space, the student expanded the solution space by listing alternative options for each data item and operation using the attribute listing map. The diagram helped to visualize different design attributes of the solution and think of various options for each attribute. After listing all the alternatives (data structures and algorithms for each data item and operations respectively), the design criteria to evaluate solutions (search time, space complexity, etc.) and constraints applicable to the given problem (search time should be low for quickly displaying the balance) are identified. Based on the constraints, the optimal solution is selected by ranking the solutions and justifying the selection.

As shown in figure 7.14, the student S1 followed the process of expand-reduce in solution space by generating alternative options using attribute listing map, identifying constraints applicable to the problem and evaluating, selecting, and justifying. The quality of the solution is better in terms of completeness and correctness. S/he could justify based on important constraints in the problem.

RQ4.2 How effective are the cognitive tools of Fathom in learning ER skills?

The experiment and control group average scores, p-value (t-test), and effect size are shown in table 7.5. The pretest scores for experiment and control group were not significantly different, which establishes equivalence in both the groups. During the intervention, the performance of both the groups was the same in ER skills. However, during posttest, the experiment group showed a significant gain over the control group in activities: understand the problem, generate more alternative solutions (quantity) and quality solutions.

This implies that cognitive tools are effective as explained below:

i. Drawing of the entity-interaction diagram is more effective in expanding the problem space than simply listing the entities and interaction. It enabled learners to visualize the complex problem better and in terms of entities and interactions and capture the dynamics of the system. The activity of drawing entity-interaction diagram is a transferable skill as it is retained in posttest than simply listing.

ii. Drawing of attribute listing map is more effective than listing data structures and algorithms. The visual representation aids learners in exploring more diverse options for each data item and operations. During the interview, the control group students said that they were generating alternative solutions by referring the worked example and thus in posttest, the students were not able to perform as the scaffolds were not present. The attribute listing map drawing is a transferable skill as experiment group was able to draw and generate alternative solutions in the posttest.

iii. The quality of the solution is significantly better in experiment group than the control group. This implies that by drawing diagrams in problem space and solution, space leads to generating more diverse solutions and improves the quality of the solution.

iv. The pros and cons analysis and decision matrix in experiment group did not help in selecting and justifying as the scores were not significantly different in both the groups. Only prompts were enough to identify criteria and constraints, evaluate solutions, and select and justify.

RQ 4.3- How effective are the metacognitive activities: formative feedback and collaboration activity in enabling students to monitor and evaluate their ER skills?

The log data was analyzed to find student interaction in terms of monitoring and controlling their cognitive processes. The system-generated feedback, peer-review, and active collaboration conversation were used for scaffolding metacognitive activity. To analyze the student's metacognitive activity from log data, we considered that student is involved in metacognitive activity if he/she is revising the ER skill after reading the feedback or collaborating with a peer. It was seen that 80 % of students were evaluating their response with the help of feedback and peer review and taking control action plan to improve. However, it was seen that very few students were engaged in higher level activities like self-evaluation, peer-evaluation, and chat facility to seek clarification or justification from peers.

Further analysis of interactions of high and low performers in each activity was done to get more insights. Figure 7.15 and 7.16 shows the interactions performed by high and low performers, respectively. The figure shows learner's interactions at two levels: (a) Doing level: The ovals represent the sequence of interactions done while performing the activity with time spent. (b)Evaluate level: The rounded rectangles show the activities done after reading the feedback provided by the system.

The high performers were-

i. Actively seeking the help provided in the learning environment. As shown in figure 7.15, while doing the activity, the high performers saw solved examples (demo videos), hints, referring data structure notes, and collaborating with peers.

ii. In most of the activities, they were modifying their activities after reading feedback and were self-evaluating and reviewing peer's responses by engaging in justifying and explaining their point of view. This shows that they were able to monitor and regulate their skills.

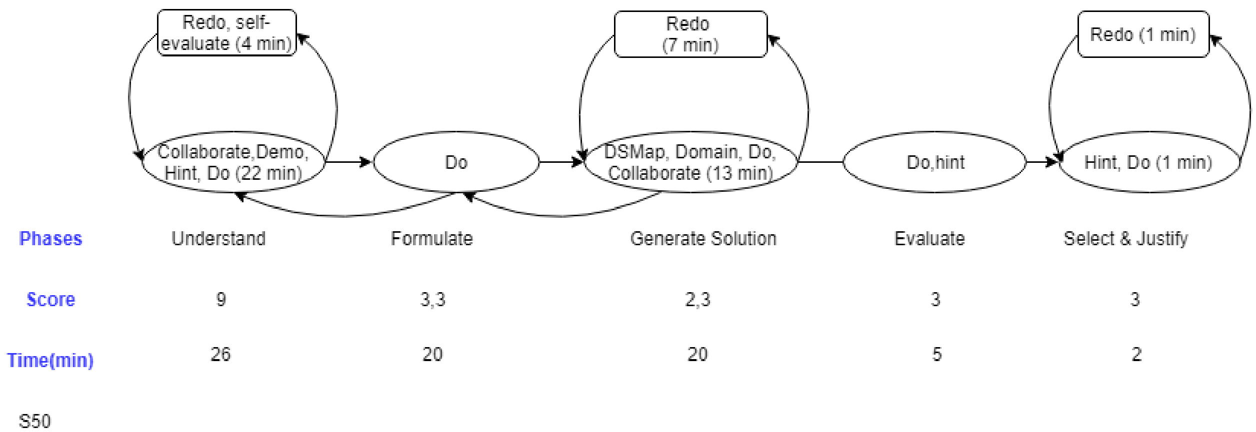


Figure 7.15. High Performer (High score in solution quality).

As shown in figure 7.16, the low performers were-

i. Not actively seeking the help provided in the environment in the form of hints, demo videos, or examples.

ii. Superficially seeing what other peers have performed

iii. Not actively monitoring and taking control action plan to revise the activity performed.

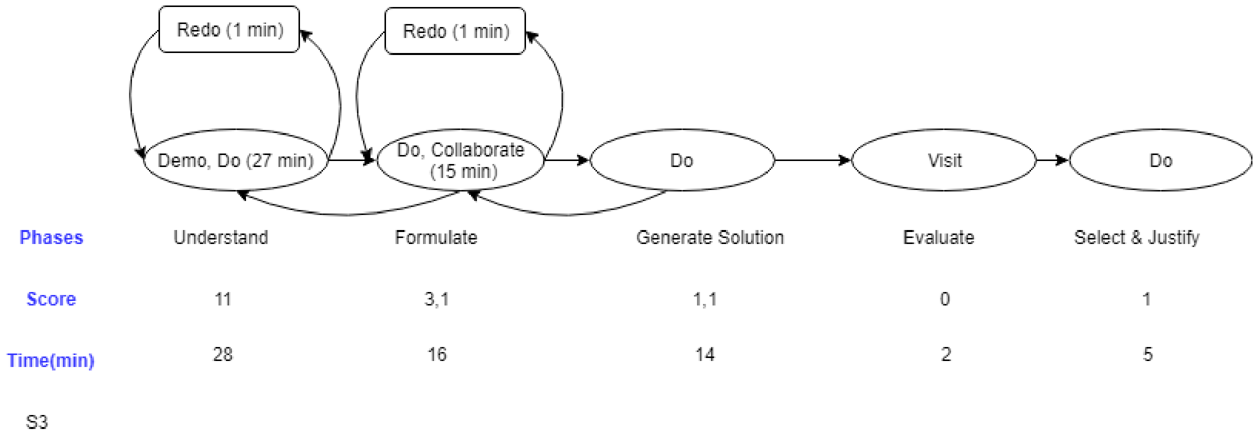


Figure 7.16. Low performer (Low score in solution quality).

7.4 Reflection

The broad RQ- "How effective is Fathom-Ver3, in learning ER skills", was investigated using three sub-RQs. For answering RQ 4.1(pedagogical tools) and RQ 4.3(metacognitive activities), the results of log data analysis and intervention scores show that pedagogical features (prompts, hints, videos, examples) and metacognitive activities (feedback and peer review) in Fathom-Ver3 were effective in doing and monitoring ER skills during the intervention. For RQ4.2- the results of the experimental pretest-posttest study showed significant gain in quality of the problem formulation and solution from pretest to posttest when ER cognitive tools are used.

The significant gain from pretest to posttest implies that students learned ER skills, and use of ER skills resulted in significant improvement in the quality of problem formulation and solution design, and the ability to reason the design decisions concerning constraints applicable to the problem. However, from the log data analysis, it was seen that most of the students were revising their skills on reading system-generated feedback and very few students were engaged in self-evaluation, peer-evaluation and active conversation with peers. One reason could be that novice learners tend to seek expert feedback to regulate the learning of complex problem-solving skills and less on higher-level evaluation activities like self-evaluation and peer-evaluation.

Chapter 8

Discussion

In this chapter, we review first the overall research path, followed by the answers to research questions. Then, we present the claims, generalizations from this thesis, and the limitations of this research.

8.1 Overview of the research

In this thesis, the research was aimed to characterize expand-reduce skills in the context of software design problem-solving, identify the challenges faced by novices in doing and learning ER skills and design the technology-enhanced learning environment for teaching-learning of ER skills.

We started with the research problem: "Teaching-learning of expand-reduce(ER) skills to undergraduate engineering students in the context of software design." Design based research

(DBR) methodology was used to design-evaluate–redesign the technology-enhanced learning environment: Fathom, for learning ER skills in the context of software design.

The overall research questions formulated and answered during our research were:

RQ1- What are the challenges faced by novices in ER skills while solving a software design problem?

RQ2- How effective is Fathom in learning ER skills?

RQ3- How did the design features of Fathom help novices to do and learn ER skills?

In this thesis, we have reported three DBR cycles and refinement of pedagogical features in three versions of Fathom. In the first cycle of DBR, the research goal was to understand the challenges faced by novices in doing ER skills and design a technology-enhanced learning environment (named Fathom), equipped with the ER cognitive tools, and support novices with effective scaffolding techniques to learn ER skills in the context of the software design problem. In the second cycle of DBR, the goal was to improve the pedagogical features in Fathom to do and learn ER skills at both cognitive and metacognitive level. In the third cycle, the goal was to improve the scaffolds at a metacognitive level to enable novices to monitor and control their ER skills.

8.2 Answers to research questions

In this section, we will discuss the answers to our research questions based on the four studies done across three DBR cycles.

RQ1- What are the challenges faced by novices in ER skills while solving a software design problem?

RQ1 is answered based on pretest analysis of student's artifacts of study 1, study 3, and study 4.

The findings in pretest are that students tend to take a solution based approach (Carman, 1999), and converge early on a single solution without explicitly doing ER in problem and solution space. The effects of not doing ER in problem space and solution space on design outcome are based on our research studies and research in ill-structured problem-solving (Ackoff, 1979;

Adelson, 198; Volkema, 1983; Guindon, 1999; Carman 200; Tang, 2008; 2010) is summarized in figure 8.1.

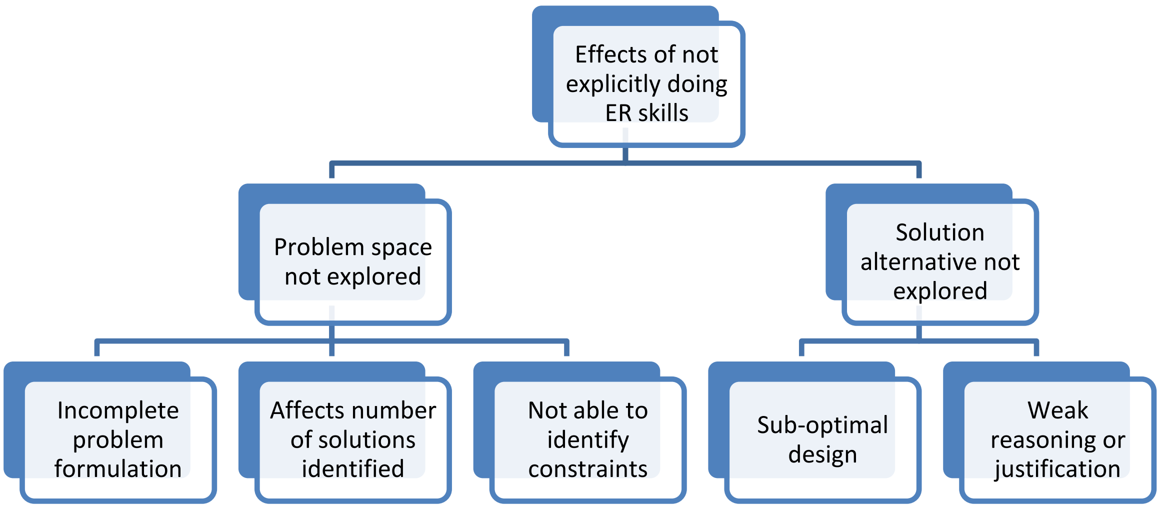


Figure 8.1. Effects of not explicitly doing ER skills

The initial problem space exploration is crucial as it may have cascading effects on problem formulation and solution design (Ackoff, 1979; Adelson, 198; Volkema, 1983). Further, if the solution alternatives are not explored then one may end up with a satisficing solution (not optimal) (Guindon, 1999) and weak justification on how the solution is better than other alternatives (Carman, 1999; Tang, 2008).

Based on a pretest in study 3 and 4, the design process followed by the students and its outcomes is summarized in table 8.1.

Table 8.1. Pretest: Novices' design process and its outcome

Method used	**Design process**	**Outcome**
Early convergence/ solution based approach	1. Problem space- Are not thinking of the problem as a whole, not able to visualize the dynamics involved in the system as they do have a mental model of the system.	1. Incomplete problem formulation 2. Not able to abstract data and operations at a higher level.
	1. Solution space- The alternative design options are not explicitly considered initially. 2. The solution design was anchored over a selected data structure at the programming level and not able to abstract the data and operations at a higher level. 3. Not able to identify the selection criteria in the given problem, as not able to go back and forth to problem and solution space. 4. Not able to evaluate and select alternative solutions based on the selection criteria.	1. No expansion in solution space before reduce led to suboptimal design. 2. The design decisions were taken without considering all alternatives, and thus, the reasoning of selection is weak. Students were trying to justify the advantages of the selected solution over other alternative and not on the criteria appropriate to the problem.

RQ2- How effective is Fathom in learning ER skills?

The RQ2 is answered based on posttest analysis of study 4. After the students were trained in ER skills using Fathom, the posttest was administered to know how well students can transfer the ER skills in a new problem context.

The design process followed by the students and design outcomes are summarized in table 8.2.

Table 8.2. Posttest: Novices' design process and its outcome

Method used	Design process	Outcome
Explicit ER-Problem-based approach	1. Problem space- Are thinking of the problem from the perspective of multiple stakeholders, able to visualize the system by drawing entities and interaction among them.	1. Able to abstract data and operations at a higher level.
	1. Solution space- The alternative design options are explicitly considered initially. 2. Are able to identify the selection criteria in the given problem, are able to go back and forth to problem and solution space. 3. Able to evaluate and select solutions based on the selection criteria.	1. The abstraction of data items and operations in problem space helped to improve the quality of the solution design. 2. Explicit expansion in solution space led to selecting an optimal design. 2. The design decisions were taken considering all alternatives, thus able to reason the selection w.r.t. selection criteria.

In problem space, the students were expanding their thinking by drawing entities and interaction diagram from the perspectives of various stakeholders. This drawing process was helping them in visualizing the complex system and the operations performed between various entities. The broad goal was carefully written and addressed the requirements of the software system. The sub-goals were formulated by defining the data and operations at a higher level of abstraction. The quality of abstraction of data and operations were maintained in terms of defining operations that are independent(loosely coupled), the behavior of each operation is well defined, and completeness was achieved as all the operations are defined to achieve the stated goal.

RQ3- How did the design features of Fathom help novices to do and learn ER skills?

RQ3 is answered based on log data analysis and assessment of student artifacts generated during the intervention in study 3 and study 4 and students' interview.

In Fathom, the activities are designed at two levels:

1. To scaffold the learners to practice ER skills in the context of solving software design problems.
2. To regulate the learning of ER skills by enabling learners to monitor their learning of ER skills, evaluate the gap, and take appropriate control action plan to improve the ER skills.

Fathom evolved based on the learner's feedback from version-1 to version-3 in terms of using cognitive tools for practicing ER skills, and pedagogy support for learning ER skills at a cognitive and metacognitive level. We will discuss the effectiveness of various design features of Fathom: pedagogical features in doing and learning of ER skills, metacognitive features, and cognitive tools.

1. Pedagogical features

i. ***Systematic guidance****:* The systematic guidance to expand-reduce in problem and solution space improved the quality of problem formulation and solution design. This was evident based on the pretest-posttest scores of study3 and study4, which showed significant gain in problem formulation and solution quality from pretest to posttest.

ii. ***Prompts with explanation and worked example*****:** Based on study2 feedback, it was evident that general prompts at a high level, for example, *think of alternative solutions*, are not effective in triggering appropriate ER related cognitive processes as learners don't have prior experience in solving design problems. Based on the results of study3 and study4, we saw that the elaborated prompts which were specific to the domain and supported with an explanation of new terms with worked example were more effective in triggering the thinking. The worked example or case studies in relation to the problem to be solved enhances students‘ understanding of the problem and their abilities to solve it (Jonassen, 2006). In Fathom, the worked example is divided into six steps to illustrate the use of the ER skill and cognitive tool applicable to the activity. During the interview, the students said that the example was helping them to think in the correct direction. Overall, the students were able to perform the activity as prompted with the help of the cognitive support provided in the learning environment.

2. Metacognitive features

For effective learning of a complex problem-solving skill, the learners need to be engaged at both, cognitive and metacognitive level (James 1999, Xun 2013, Bannert 2013). The expert designers exhibit metacognitive behavior in the form of continuously assessing the solution against the test cases and constraints to identify the gap and take appropriate control action plan to improve (Adelson, 1985; Tang, 2010). In Fathom, the scaffolds were designed to regulate the metacognitive skills using self-evaluation, formative feedback, peer review, and active chat facility. The finding of each feature is explained below:

i. System-generated feedback: In Fathom, based on the rubric and cognitive biases of novices, the rules are written in the system to generate the feedback with both positive and corrective messages elaborated to motivate the cognitive processes correctly performed and the corrective action plan addressing the gap. The study4 log data analysis showed that the system generated feedback was regulating the student metacognitive skill in terms of evaluating their performance and taking appropriate action to improve their skills. The screen capture video (study 4) showed that during the *understand problem* activity, initially, students were adding one or two entities and interactions, e.g., student (entity)->search for(interaction) -> books(entity). The feedback that prompted them to look at the nouns and verbs in the problem statement enabled learners to think and add other related entities (librarian, teachers, etc.) and interaction (issue, return books, update, etc.) in the library system.

ii. Peer review and collaboration**:** The peer review activity is a metacognitive activity designed to allow learners to see how the peers have performed, evaluate peer response, compare their answers with peers' answers and engage in collaborative active learning conversation. The log data analysis showed that almost 50% of learners were seeing their peer's response, evaluating and improving their skills. However, very few students were using the chat facility to seek or give explanation or justification to peers.

iii. Self-evaluation and peer-evaluation: Based on study 3 results, we saw that novice problem solvers are not good at self-evaluating their ER skills. They were not able to evaluate their level of performance against the given set of responses or with the help of rubric. Also, in study 4, it was seen that very few students were engaging in self-evaluation or peer-evaluation activity.

3. ER cognitive tools

In study 4, the experiment group equipped with ER cognitive tools performed significantly better in posttest than the control group in the activities: understand problem, problem formulation, generate solutions, and solution quality. The ER cognitive tools used were:

i. ***Entity-interaction diagram:*** The cognitive tools of drawing entity-interaction map and attribute-listing map were effective in expanding and reducing the problem and solution space than listing the entities-interactions in problem space or design options in solution space. In study 4, the experiment group equipped with entity-interaction diagram significantly performed better in exploring the problem space than the control group in the posttest. The entity-interaction diagram enables the learner to visualize the problem space from the perspective of various stakeholders involved and expand by branching out from one entity to other related entities by identifying the interactions between them. It helps to simplify the understanding of the complex problem and identify the requirements to be addressed in the solution.

ii. ***Writing goals and sub-goals*** is used to reduce the problem space to sub-problems by explicitly stating the broad goal to be achieved from the perspective of the end users. Based on the goal, the entities-interactions are reduced to the data items and operations at a higher level of abstraction.

iii. ***Attribute listing map***: Drawing attribute listing map is more effective in expanding the solution space than simply listing the various design options in solution space. This was seen in study 4, where an experiment group equipped with attribute listing map generated significantly more number of alternative solutions than the control group in the posttest. It enables the learner to represent sub-problems (data items, operations) of the solution design as attributes and branch out each sub-problem to various alternative design options. Next, multiple solutions are generated by selecting valid combinations of design options for each sub-problem.

iv. ***Pros and cons and decision table:*** The experiment and control group had no significant difference in evaluating and selecting solutions with and without pros and cons or a decision table.

8.3 Local learning theories

One of the aims of design-based research is to generate contextually relevant design principles that inform the local context or factors that led to learning effects (Wang, 2005).

In this section, we present the local learning theories (Plomp and Nieveen, 2010), which informs how the learner's interaction with Fathom learning environment at both cognitive and metacognitive level led to the learning of ER skills. These theories are extracted from study 4 based on the analysis of log data, screen capture videos, and student's interview data.

We will discuss the local learning theories explicated based on our research studies done to assess the learning of ER skills at both cognitive and metacognitive level:

Cognitive level:

Drawing entity-interaction diagram aids novices in expanding problem space: In study 4, the experiment group equipped with drawing tools significantly performed better than the control group. During the interview (study 4), students said that the activity of drawing entities and interactions enabled them to visualize the problem as a whole in terms of entities involved and how they are linked to each other. They were able to expand by starting with one of the entity systematically and subsequently added related entities.

The drawing of the entity-interaction diagram was providing the affordance to systematically expand by starting from one or two entities and eventually adding related entities and interactions among them.

Entity-interaction diagram enables novices to formulate sub-goals: Entity-interaction diagram enabled novices to visualize the whole system at an entity-interaction level and use it to structure the problem into sub-problems. During the interview (study 4), students said that the *formulate problem* activity the entity-interaction diagram enabled towards identifying the variables for each entity and operations for each interaction.

Drawing attribute listing map enables novices to expand and reduce the solution space: The attribute list map was providing novices the affordance to visualize various design options for each sub-problem and think of the evaluation criteria to reduce to the optimal solution. During the interview (study 4), students perceived that the process of breaking the problem into sub-problems was aiding to expand the solution space for each sub-problem independently which they were not able to do during the pretest.

Identifying criteria and constraints are necessary to evaluate and select a solution: The criteria identification based on pros and cons analysis table enabled learners to identify the common criteria against which the solutions can be evaluated. Next, the learners were prompted to select the criteria that can be a constraint applicable to the problem and justify. This allowed learners to think of the constraints and justify how they are important for the given problem. The activity of ranking the solutions by evaluating against the sub-goals and constraints using the decision table allowed learners to think of the optimal solution which satisfies the sub-goals and select and justify.

Metacognitive level

Feedback is necessary for novices to regulate the learning of ER skills - Based on log data analysis (study 4) it was seen that 80 % of learners were able to monitor and take control action plan to revise the ER skill on reading the feedback. The feedback system designed based on ITF model proposed by Narciss (2013) was effective in generating an internal feedback loop, which is necessary to monitor and improve the performance. The feedback was given immediately after the activity and was addressing both positive and corrective aspects to motivate students by appreciating them for good work and direct their attention towards areas of improvements. The feedback was elaborated with an explanation on the level of performance achieved and lacuna if any, supported with information to guide learners to see hints, examples, peers' responses, and domain-related notes to improve. During the interview, students said that feedback was helping them to identify the gaps in their performance.

Peer review enables learners to see alternative ways of solving the problem - Based on log data analysis (study 4) almost 50% of learners were checking the responses of peers, but very few were using it to self-evaluate or review the performance level of ER skills of the peer's response. While, in some cases, learners were not focussing on their work but skimming through all peer's responses without much learning happening but was affecting the performance at the task.

Peer-evaluation, self-evaluation and active collaboration are advanced features to enhance the acquisition of ER skills- Based on log data analysis (study 4) it was seen that high performers were actively using self-evaluation, peer-evaluation and active chat facilities to monitor and control their ER skills. This facility was designed to allow learners to regulate the metacognitive thinking skills by comparing their performance with peer's performance, evaluate the gap, use chat facility to seek or give explanation, challenge the peer's answers or appreciate other's work. However, most of the students were not actively using this facility during the training period which suggests that during training the learners spend most of their efforts in understanding and learning the skills at a cognitive level and external feedback is important to scaffold their metacognitive skills. The novices have difficulty in self-regulating the metacognitive behavior using self-evaluation or peer-evaluation features for learning complex problem-solving skills like ER skills.

8.4 Effective pedagogical design principles for the learning of ER skills

We have summarized the overall pedagogical design principles for the learning of ER skills based on the various studies done in this research work, in table 8.3. We have explained how our local learning theories support existing theories and some of the new findings that can be tested by other researchers in a similar context. These are useful for educators and educational technology researchers to design a learning environment for teaching-learning of ER skills in the context of solving design problems to novices.

Table 8.3. Effective pedagogical design principles for learning ER skills

Design principles (Pedagogy, ER cognitive tools)	**Evidence**
1. The training of ER skills in problem and solution space using technology-enhanced learning environment equipped with cognitive tools and pedagogical support improves the quality of problem formulation and solution design.	Significant gain from pretest to posttest in study 3 and study4.
2. The following ER cognitive tools were effective for novices in performing and learning of ER skills are: i. The drawing of diagrams enables novices to visualize and expand the problem space and eventually aids to break the problem into sub-problems. The previous findings in support to this principle state that external representations in the form of diagrams help to visualize and expand the problem space and reduce to sub-problems effectively (Adelson, 1985; Guindon, 1990; Ellspermann, 2007; Wang, 2018). ii. The attribute listing map drawing enables to expand solution space by visualizing each sub-problem as a separate unit which can be branched into different alternative design options. This is a new finding as the use of attribute listing map to expand solution space was suggested (Liu, 2004) but not validated in any of the previous studies. iii. Identify criteria and constraints: Previous studies have suggested that experts implicitly and quickly perform pros and cons analysis of alternative solutions based on some selection	Significant gain in experiment group equipped with cognitive tools over the control group (study 4)

criteria (Guindon, 1990). However, novices face difficulty in identifying the selection criteria (Jonassen, 2006; 2008). The pros and cons analysis table helps novices in identifying design criteria by evaluating the alternative design options and identifying the criteria used for comparison. These criteria can be converted into constraints that apply to the given problem. This process will enable novices to explicitly identify selection criteria and constraints which they are not able to do on their own.

iv. The decision table enables novices to rank solutions by evaluating alternative solutions against the constraints and select the optimal solution. This is a weak finding as in study 4; it was seen that there was no significant difference between experimental and control group scores on selecting and justifying with or without decision table.

v. The justification box to justify how the selected solution is better than other alternatives improves the reasoning capability as learners have to reflect and explain how the selected solution is better for the given problem compared to other alternative solutions. The findings of the expert studies also state that explicit reasoning improves the decision-making process and solution quality (Tang, 2008).

The following pedagogical design features were effective in performing and learning of ER skills:

1. Workspace to practice ER skills supported with cognitive and metacognitive scaffolds enables novices to perform and learn ER skills. This supports the design principles stating that for learning complex problem-solving skills, novices need to be coached to practice and reflect at the meta-level which they generally are not able to do on their own (Bannert, 2013; Xun, 2004; Schon, 1987).	The scores of ER skills during the intervention and posttest triangulated with log data analysis and interview data of study 4 show that the activities in Fathom were effective in performing and learning ER skills.
2. Cognitive scaffolds	
i. Learning activity supported with domain-specific prompts elaborated with an explanation of new terms, hints, worked example, and domain-related notes are effective in performing and learning ER skills. The prior research work also states that the absence of specific-domain knowledge, question prompts were futile in activating a learner' prior knowledge or relevant schema (Ge, Chen, & Davis, 2005) The suggestion made by Jonassen (2006) that the additional support in terms of worked example is essential to help learners understand the complex problem-solving task is validated based on our studies.	In study 1, 2 and 3, based on student interview and scores, we found that for novice learners, higher level prompts (e.g., generate alternative solutions) are not effective in performing ER related activities.
3. Metacognitive scaffolds	
i. The self-evaluation activity is not effective in regulating the metacognitive behavior, as students tend to either over-estimate or under-estimate their performance and not able to take appropriate	In study-3 and study 4, students were not effectively using self-

action plan to improve the skill. This does not support the findings of the previous study (Rivera, 2016), which showed that self-evaluation activity helped dental students to be significantly more empathetic towards patients in the virtual agent learning environment.	evaluation feature.
ii. External formative feedback addressing both the positive and corrective action plan is necessary to regulate the learning of ER skills for novices. The feedback should be generated based on the difficulties or cognitive biases of the learners identified in novice studies. This confirms the effectiveness of the design principle suggested in interactive tutoring feedback model (Narciss, 2013) for ill-structured problem-solving.	In study 4, 80 % of students had revised their skills in reading feedback.
iii. The peer-review, self-evaluation, and peer-evaluation with active chat facility allow high performers to elicit explanation, question or appreciate the design decisions taken by the peers.	Log-data analysis in study 4 showed that only high performers were self-evaluating their responses.

8.5 Limitations

We acknowledge the limitations of the thesis work from the learners, instructors, and domain perspective, to help in identifying the future research direction.

1. Learner characteristics

The findings of our research studies are limited to learners with the following characteristics-

i. Computer engineering students.
ii. Completed data structures and algorithms course.
iii. Proficient in English and in using a computer.
iv. Belong to an urban area.

2. Instructional design

The instructional design of Fathom was focused on designing activities and scaffolds to facilitate the learning of ER skills by novice learners via an online learning environment. The course instructor or teacher has no active role in authoring the instructional design in terms of changing the problem, scaffolds, assessment, etc.

3. Topic and domain

The research work is carried out in the context of software design problem-solving in data structures course for computer engineering students. This is not tested for design problems in other similar courses or other engineering domains.

4. Near vs. far transfer

The research studies were done to test the near transfer of ER skills in the same course. We did not test the far transfer as the studies were not longitudinal.

8.6 Generalizability

We have designed and validated Fathom for learning ER skills in the context of software design problem-solving in data structures and algorithm course. Here we attempt to generalize the use of ER skills for other software design problems and pedagogical changes for solving problems in other domains and examine possible concern.

8.6.1 ER skills in other software design problems

In our research, we focussed on developing ER skills in the context of software design problems using appropriate data structures and algorithms. The learning activities and cognitive tools were designed to suit the characteristics of software design problems in data structures course and based on prior experience of computer engineering students.

To solve software design problems related to final year project or real project, for example, "design an intelligent traffic signal control system," the ER skills are still applicable. Here are the guidelines on revising the model for such a case:

a. The cognitive tool of drawing the model of the system by identifying entities (vehicles, maps, traffic lights, sensors) and interactions (vehicles on the road, traffic lights change based on the traffic, etc) is useful to expand the problem space by thinking of entities involved and simulating the possible roles and interactions with other entities. This process helps to form a working mental model of the whole system.
b. Reduce the problem by identifying the requirements and sub-problems (calculate the vehicles in each road, change the duration of the signal) to be solved based on the goal and the entity-interaction diagram.
c. In solution space, attribute listing map will help to branch out alternative design options for each sub-problem and generate solutions by combining the design options.
d. Reduce the solution space by identifying the design criteria (reliability, cost) and constraints (high reliability in calculating the number of vehicles) important for the given problem and evaluate and select the optimal solution.

We believe that the learning activities in Fathom have to be redesigned with minor changes in contextualizing prompts, examples, and feedback for a new software design problem in other courses other than data structures.

8.6.2 ER skills in other domain

ER skills are applicable to problems in other domains like design a computer network, design electronic circuit, design control systems, troubleshoot problems, etc, which are ill-structured and complex. The cognitive tools designed in Fathom to expand-reduce are applicable to solve problems in which-

- The problem space has to be explored and visualized from the perspectives of various stakeholders or to model the complex system in terms of sub-components and interactions among them.
- In solution space, each sub-problem can be represented as an attribute, and each attribute has alternative design options. The criteria and constraints are used to reduce the solution space to select the optimal solution.

Overall, to generalize the learning activities in Fathom for other domains, as mentioned earlier, the cognitive tools and processes applicable to solve the problem in the domain have to be identified. For example, if a lift control system has to be designed, then the appropriate cognitive tool to expand problem space may be a state transition diagram instead of the entity-relation diagram. The learning activities should be redesigned with cognitive and metacognitive scaffolds to help the learner to use appropriate ER cognitive tools applicable to the domain and based on prior experience of the learners. For example, if the learner is experienced then only prompts are enough to trigger ER skills, while for inexperienced learner, more scaffolding in terms of examples and expert feedback is needed.

Chapter 9

Conclusion

This chapter summarizes the contributions of this thesis work, discusses the future research directions coming out of this thesis and the final reflections.

9.1 Contributions of the thesis

We report the contributions of this thesis in five categories: 1. Characterization of ER skills in the process of software design, 2. Providing insights about the cognitive biases of a novice, 3.Identification of ER cognitive tools, 4. Identification of effective cognitive and metacognitive scaffolds in technology-enhanced learning environment for doing and learning of ER skills to the novice, and 5. Research studies and outreach. These findings can be used by educational technology researchers and instructional designers to design and evaluate similar learning environments for teaching-learning of thinking skills in the context of complex problem-solving.

1. Characterization of ER skills in the process of software design

ER skills are characterized as the ability to expand and reduce both problem and solution space systematically. In the problem space, expand skill is the ability to understand the system as a whole and reduce to sub-problems based on the goal to be achieved. Similarly, in solution space, expand by branching out all possible design alternatives for each sub-problem, and generating alternative solutions and then reduce by evaluating and selecting a solution based on selection criteria. The use of ER skills in problem space and solution space in software design improves the quality of problem formulation and solution design. These implications are useful to novice software designers in applying the underlying ER cognitive processes in the process of software design before drawing formal design artifacts.

2. The cognitive bias of novice in applying ER skills

The novice tends to reduce early to the solution that they are more comfortable with or is easy to implement without exploring alternative design options suitable to the problem. This results in sub-optimal design and weak reasoning. These implications are useful for engineering educators and engineering education researchers.

3. Identification of ER cognitive tools and processes

The cognitive tools effective in doing ER skills are-

i. Entity-interaction diagram – It helps to explore the problem space and simplify the understanding of the complex problem. It also enables to visualize the dynamics of the problem and identify the requirements to be addressed in the problem.

ii. Writing goal and sub-goals- It helps to write the goal to be achieved from an end-user point of view and formulate the data items and operations at a higher level of abstraction.

iii. Attribute listing map- It helps to expand the solution space by branching out design options for each attribute (data and operations). Generate alternative solution by using valid combinations of each attribute.

iv. Identify criteria and constraints- It helps to interleave problem-solution space by identifying the design criteria using pros and cons analysis of alternative solutions and identifying the important constraints on design criteria that apply to the given problem.

v. Decision table- It helps to evaluate alternative solutions against the constraints, select the optimal solution, and justify.

4. Identification of effective cognitive and metacognitive scaffolds in technology-enhanced learning environment for doing and learning of ER skills to novice

Cognitive scaffolds for novices in doing and learning ER skills:

i. Only prompts are not effective in directing novice learners' thinking towards triggering the underlying cognitive processes, especially if learners do not have prior experience in solving design problems. The prompts should not be general but specific to the domain and supported with additional support to aid in understanding the activity to be performed.
ii. A worked example is effective in gaining more clarity in how the skills are applied in solving a similar problem and helps to compare learner's performance with the solved example. The process of using cognitive tools is best illustrated with videos or animations as they explicate the thinking process and skills in the given context.
iii. Hints are useful for learners with low prior knowledge or have difficulty in understanding the skill to be performed in the given problem.

Metacognitive scaffolds to monitor and take control action plan to improve:

i. Novice learners are not good at self-evaluating and taking control action plan immediately after practicing the skill for the first time.
ii. Formative feedback is an important factor in enabling novices to monitor the gap and take appropriate corrective action plan.
iii. The peer review and chat facility to seek or give an explanation on the evaluation process also helps to improve on metacognitive skills, especially for high performers.

These findings can be used by engineering educators and educational technology researchers who aim to design instructional strategies for teaching ER skills in the context of solving ill-structured problems to the novice.

5. Research studies and outreach

a. Studies to test the effectiveness of Fathom: The four research studies (N=200) were done to study the effectiveness of pedagogical features and cognitive tools in Fathom in learning ER skills. The research studies carried out were exploratory studies (study 1, study 2) and pretest-posttest experimental studies (study 3 and 4). The studies contribute to the research space in the field of technology-enhanced learning of thinking skills (TELoTS) in terms of replicable research methods, data collection methods (quantitative and qualitative), assessment instruments (rubric to assess ER skills) and findings.

b. Implications of using Fathom to engineering students and instructors: The Fathom has pedagogical features to enable novices to learn ER skills in the context of software design problem-solving. Novices will benefit in terms of learning complex problem-solving skills which they can transfer for solving other similar software design problems, for example, final year project or workplace design problems. Engineering educators can also directly use Fathom in a lab setting for teaching-learning of ER skills in data structures course to their students.

9.2 Future work

In this section, we present the future work to further the research towards the teaching-learning of ER skills in various directions. Most of the future work has emanated from the limitations stated in the last chapter and some towards achieving generalizability.

- The pedagogy of Fathom to be tested and validated for another type of design problems.

  In our studies, we have focussed on teaching-learning of ER skills in the context of software design problem-solving. The ER cognitive tools, prompts, hints, worked example, and feedback are contextualized towards solving software design problem using appropriate data structures and algorithms. One of the directions for future work is to validate the pedagogy of Fathom for solving another type of design problems (General software, Network, Electronic circuit) by contextualizing the pedagogical features in the new domain based on the prior knowledge of learners. One can start with by identifying the suitable ER cognitive tools and redesign of the prompts worked example, and feedback.

- The longitudinal studies to be conducted over one year on the same set of students to test far transfer of ER skills.

    In our studies, the students were trained on ER skills using single problem only, and the learning was tested by giving similar problem in posttest immediately after the intervention. As future work, longitudinal studies can be planned with a span of one or two years in which rigorous training on ER skills can be given in one semester (data structure course), and test if the students can far transfer these skills in a new course (computer network course).a

- The instructor authoring feature can be incorporated to allow teachers to create a problem-solving environment for learning ER skills in their courses.

    The current design of Fathom does not allow the instructor to design or modify the learning environment. The instructor authoring feature can be incorporated into Fathom learning environment to modify the problem, prompts, worked example, domain related notes, ER cognitive tools, add/remove an activity, change the structure of the activity, etc.

- The learning patterns of the students can be further analyzed to identify the low performers during the intervention and take timely actions to motivate them towards learning.

    Based on log data analysis, the learning pattern of high and low performers can be identified and incorporated into the learning environment to predict the low performers and provide timely feedback to the instructor or the learner. For example, in study 4, we saw that low performers were not doing the activity on their own but copying from peers. This kind of pattern can be detected, and the learners can be motivated to perform the activity.

- Qualitative studies to get insights related to self-regulation of metacognitive skills for the learning of ER skills.

In our studies, we found that students were not actively using peer review, self-evaluation, and chat facility to regulate the metacognitive skills. It was also seen that the peer review activity was used to copy from the peer rather than doing the activity on their own, which was detrimental towards the learning of ER skills. The research question that emerges from this observation is- "why the novice learners are not able to regulate their skills via self-evaluation, peer-evaluation or communicate with their peers to seek explanation, clarify or appreciate peer's responses via the learning environment's chat facility?"

Qualitative studies have to be done to get a detailed insight into what are the learner's difficulties in self-regulating the metacognitive skills to evaluate, identify the gap, and take control action plan.

- The feedback mechanism based on keyword checking can be improved by incorporating sophisticated NLP algorithms.

  The current feedback mechanism is not very robust as it is based on matching the keywords to generate feedback. The feedback mechanism can be improved by using NLP algorithms to find the semantics of the sentences written by the learners to generate feedback.

9.3 Final Reflection

This thesis work is the amalgamation of efforts towards identifying the level of a novice in applying ER skills, challenges faced by them in learning ER skills and designing a learning environment to scaffold novices in learning ER skills. The design of Fathom was evolved through successive evolutions and involved the use of ER skills. During each iteration, we enumerated all the learner difficulties, various scaffolding methods, cognitive tools, etc. and choose based on the feedback from previous studies which resembles ER. The major driving force for doing this research was to contribute towards the engineering education research in teaching-learning of thinking skills. The dual role of engineering educator and educational researcher helped me to achieve the possible research outcomes and contributions in the field of engineering education for teaching-learning of complex problem-solving skills. Personally, this research work has helped me to hone my skills towards becoming a good researcher, thinker, and teacher.

Appendix

Appendix 1

Data structures and Algorithms (DSA) Quiz

* Required

1. **Email address** *

2. **Name** *

3. **Roll no** *

Quiz Questions

4. **What are the advantages of arrays?**

   Mark only one oval.

   - Used to implement other data structures like stack and queue
   - Quick and easy to access elements
   - All of the mentioned
   - Easier to store elements of same data type

5. **What are the disadvantages of arrays?** *

Mark only one oval.

- All of the mentioned
- Insertion and deletion becomes tedious
- We must know before hand how many elements will be there in the array
- There are chances of wastage of memory space if elements inserted in an array are lesser than than the allocated size

6. **Consider an implementation of unsorted singly linked list. Suppose it has its representation with a head pointer only.Given the representation, which of the following operation can be implemented in O(1) time? 1. Insertion at the front of the linked list 2. Insertion at the end of the linked list 3. Deletion of the front node of the linked list 4. Deletion of the last node of the linked list**

Mark only one oval.

- 1 and 2
- 1 and 3
- 1, 2, and 3
- 1, 2 and 4

7. **Linked lists are not suitable for the implementation of?**

Mark only one oval.

- Insertion sort
- Radix sort
- Polynomial manipulation Binary
- search

8. **Linked list is considered as an example of__________________type of memory allocation.**

Mark only one oval.

- Dynamic Static
- Compile time
- None of the mentioned

9. Array is considered as an example of________________type of memory allocation.

Mark only one oval.

- Dynamic Static
- Compile time
- None of the mentioned

10. Which of the following points is/are true about Linked List data structure when it is compared with array

Mark only one oval.

- It is easy to insert and delete elements in Linked List
- Random access is not allowed in a typical implementation of Linked Lists All of the mentioned

11. The data structure required to check whether an expression contains balanced parenthesis is?

Mark only one oval.

- Stack
- Queue
- Graph Tree

12. Which data structure can be used to store records of 100 students with data items roll no, name, course, marks 1. Array 2. Array of Structure 3. Linked list 4. Binary Search Tree using lists

Mark only one oval.

- 1
- 1 and 2
- 1, 2 and 3
- 2, 3 and 4

13. In given data structure struct customer { char name [100]; int age; char phone[10];} cust[100]; which operation is used to find and print customers whose age is below 25.

Mark only one oval.

- Search
- Traverse
- Insert Delete
-

14. In data structure below, struct customer { char name [100]; int age; char phone[10];} cust[100]; which operation is used to find customer with given phone number.

Mark only one oval.

- Search
- Traverse
- Insert Delete
-

15. Advantages of linked list representation of binary trees over arrays?

Mark only one oval.

- dynamic size
- ease of insertion/deletion
- ease in randomly accessing a node
- both dynamic size and ease in insertion/deletion

16. Disadvantages of linked list representation of binary trees over arrays?

Mark only one oval.

- Randomly accessing is not possible
- Extra memory for a pointer is needed with every element in the list Difficulty
- in deletion
- Random access is not possible and extra memory with every element

17. How to travel a tree in linkedlist representation?

Mark only one oval.

- using post order traversing using
- pre order traversing using post
- order traversing all of the
- mentioned

18. What is a hash table?

Mark only one oval.

- A structure that maps values to keys A
- structure that maps keys to values A
- structure used for storage
- A structure used to implement stack and queue

19. If several elements are competing for the same bucket in the hash table, what is it called?

Mark only one oval.

- Diffusion
- Replication
- Collision
- None of the mentioned

20. What is the search time complexity in hash table ?

Mark only one oval.

- O(n)
- O(logn)
- O(nlogn)
- O(1)

21. What is a hash function?

Mark only one oval.

- A function has allocated memory to keys

A function that computes the location of the key in the array A

function that creates an array

None of the mentioned

22. The worst case occur in linear search algorithm when …….

Mark only one oval.

Item is somewhere in the middle of the array Item is

the last element in the array

Item is not in the array at all

Item is the last element in the array or item is not there at all

23. If the number of records to be sorted is small, then …… sorting can be efficient.

Mark only one oval.

Merge Heap

Selection

Quicksort

24. Which of the following is not true in binary search algorithm?

Mark only one oval.

must use a sorted array

requirement of sorted array is expensive when a lot of insertion and deletions are needed there must be

a mechanism to access middle element directly

binary search algorithm is not efficient when the data elements more than 1500.

25. How many comparisons are needed to find a element 10 in following array using binary search. a[]={ 1,2,3,4,5,6,7,8,9,10}

Mark only one oval.

3

10

5

1

26. How many comparisons are needed to find a element 10 in following array using linear search. a[]={ 1,2,3,4,5,6,7,8,9,10}

Mark only one oval.

3

10

5

1

Appendix 2

Study consent form

STUDY TITLE: Study of student use of Fathom technology-enhanced learning environment.

You have been asked to participate in a research study conducted by Deepti Reddy from the Inter-Disciplinary Program in Educational Technology at the Indian Institute of Technology Bombay (IITB). The purpose of the study is to understand how students learn software problem solving skills using the Fathom technology-enhanced learning environment. The results of this study will be included in the Ph.D. thesis of Deepti Reddy. You were selected as a possible participant in this study because you are a BE-Information Tech/BE-Computer Engg student in the Mumbai University engineering college.

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

- In this study you will be asked to solve real-life engineering problems using and without the Fathom technology-enhanced learning environment.

- Your solutions will be used for research purposes only by the investigators of this study.

- 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 2-3 hours.

- You will not be compensated for the participation.

- We will not use your name in publications; however we may need to use your academic qualification details if you give us permission.

- We would like to record the audio of your interview so that we can use it for reference while proceeding with this study. If you grant permission for this interview to be recorded, you have the right to revoke recording permission and/or end your participation at any time. If we use your voice anywhere it will not be identified by name.
- We would like to capture your computer screen using CamStudio software as you solve the problems so that we can use it for reference while proceeding with this study. If you grant permission for this screen capture, you have the right to revoke recording permission and/or end your participation at any time. If we use this screen capture anywhere, we will blank out your personal information.

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 given a copy of this form.

I give permission for the following information to be included in publications resulting from this study ((*Please check all that apply*)

[ ] my academic qualification details [ ] direct quotes from my audio recordings
[ ] screenshots from my computer screen

Your name:

Your signature ____________________________________ Date ____________

Signature of Investigator ________________________Date _________

Please contact Deepti Reddy (deeptir@iitb.ac.in) or Prof. Sridhar Iyer, IDP ET IITB (sri@iitb.ac.in) with any questions or concerns.

References

Açar, A. E., and D. S. Rother. "Design Thinking in Engineering Education and its Adoption in Technology-driven Startups." *Advances in Sustainable Manufacturing*. Springer, Berlin, Heidelberg, (2011): 57-62.

Ackoff, Russell L. "The future of operational research is past." *Journal of the operational research society*, (1979): 93-104.

ABET, "Criteria for accrediting engineering programs", 2014.

Adelson, Beth, and Elliot Soloway. "The role of domain expenence in software design." *IEEE Transactions on Software Engineering* 11 (1985): 1351-1360.

Amálio, Nuno, and Christian Glodt. "A tool for visual and formal modelling of software designs." *Science of Computer Programming* 98 (2015): 52-79.

Amiel, Tel, and Thomas C. Reeves. "Design-based research and educational technology: Rethinking technology and the research agenda." *Journal of educational technology & society*11.4 (2008): 29-40.

An, Yun-Jo, and Li Cao. "Examining the effects of metacognitive scaffolding on students' design problem solving and metacognitive skills in an online environment." *Journal of Online Learning and Teaching* 10.4 (2014): 552.

Azevedo, Roger, and Allyson F. Hadwin. "Scaffolding self-regulated learning and metacognition–Implications for the design of computer-based scaffolds." *Instructional Science*33.5 (2005): 367-379.

Basadur, Min, Mitsuru Wakabayashi, and George B. Graen. "Individual problem-solving styles and attitudes toward divergent thinking before and after training." *Creativity Research Journal* 3.1 (1990): 22-32.

Basadur, Min, et al. "Collaborative problem solving through creativity in problem definition: Expanding the pie." *Creativity and Innovation Management* 9.1 (2000): 54-76.

Bannert, Maria, and Christoph Mengelkamp. "Scaffolding hypermedia learning through metacognitive prompts." *International handbook of metacognition and learning technologies*. Springer, New York, NY, (2013): 171-186.

Bengtsson, Mariette. "How to plan and perform a qualitative study using content analysis." *NursingPlus Open* 2 (2016): 8-14.

Booch, "Object-Oriented Analysis And Design With Applications", 3/E, Pearson Education India, 2009.

Brown, Tim. "*Change by design: How design thinking creates new alternatives for business and society"*. Collins Business, 2009.

Cassel, Sofia, and Björn Victor. "A structured approach to training open-ended problem solving." *2015 IEEE Frontiers in Education Conference (FIE)*. IEEE, 2015.

Cooperrider, Bryan. "The importance of divergent thinking in engineering design." *Proceedings of the 2008 American Society for Engineering Education Pacific Southwest Annual Conference*, 2008.

Coyle, Geoff. "The Analytic Hierarchy Process (AHP). Practical Strategy: Structured Tools and Techniques. Open Access Material." 2004.

Dennis, Alan R., et al. "Structuring time and task in electronic brainstorming." *MIS Quarterly* 23.1, 1999.

Dorst, Kees, and Nigel Cross. "Creativity in the design process: co-evolution of problem–solution." *Design studies*22.5 (2001): 425-437.

Dym, Clive L., et al. "Engineering design thinking, teaching, and learning." *Journal of engineering education* 94.1 (2005): 103-120.

Eden, C., and F. Ackermann. "SODA—The principles. J. Rosenhead, J. Mingers, eds. Rational Analysis for a Problematic World Revisited." (2001): 21-41.

Ellspermann, Susan J., Gerald W. Evans, and Min Basadur. "The impact of training on the formulation of ill-structured problems." *Omega* 35.2 (2007): 221-236.

Felder, Richard M., and Rebecca Brent. "Designing and teaching courses to satisfy the ABET engineering criteria." *Journal of Engineering Education* 92.1 (2003): 7-25.

Ge, Xun, Ching-Huei Chen, and Kendrick A. Davis. "Scaffolding novice instructional designers' problem-solving processes using question prompts in a web-based learning environment." *Journal of educational computing research* 33.2 (2005): 219-248.

Ge, Xun, Lourdes G. Planas, and Nelson Er. "A cognitive support system to scaffold students' problem-based learning in a web-based learning environment." *Interdisciplinary Journal of Problem-based Learning* 4.1 (2010): 4.

Clarebout, Geraldine, et al. "Metacognition and the use of tools." *International handbook of metacognition and learning technologies*. Springer, New York, NY, (2013): 187-195.

Guilford, Joy Paul. *A Revised Structure of Intellect, Studies of Aptitudes of High-level Personnel. JP Guilford*. University of Southern California, 1957.

Guindon, Raymonde. "Knowledge exploited by experts during software system design." *International Journal of Man-Machine Studies* 33.3 (1990): 279-304.

Howard, Thomas J., Stephen J. Culley, and Elies Dekoninck. "Describing the creative design process by the integration of engineering design and cognitive psychology literature." *Design studies* 29.2 (2008): 160-180.

Jagadish, Iyer, Sridhar. "A method to construct counterexamples for greedy algorithms." *Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education*. ACM, 2012.

Janitor, Jozef, František Jakab, and Karol Kniewald. "Visual learning tools for teaching/learning computer networks: Cisco networking academy and packet tracer." *2010 Sixth International Conference on Networking and Services*. IEEE, 2010.

James Geiwitz. "Training Metacognitive Skills for Problem Solving", *U.S. Army Research Institute for the Behavioral and Social Sciences*, 1994.

Jonassen, D. "Supporting Problem Solving in PBL. Interdisciplinary Journal of Problem-based Learning." (2011).

Jonassen, David H., and Woei Hung. "All problems are not equal: Implications for problem-based learning." *Interdisciplinary Journal of Problem-Based Learning*, (2008): 4.

Jonassen, David, Johannes Strobel, and Chwee Beng Lee. "Everyday problem solving in engineering: Lessons for engineering educators." *Journal of engineering education*, (2006): 139-151.

Jonassen, David H. "Toward a design theory of problem solving." *Educational technology research and development48.4*,(2000): 63-85.

Kim, Kyung Hee, and Robert A. Pierce. "Convergent Versus Divergent Thinking." *Encyclopedia of Creativity, Invention, Innovation and Entrepreneurship*. Springer New York, (2013): 245-250.

Kirschner, Paul A., John Sweller, and Richard E. Clark. "Why minimal guidance during instruction does not work: An analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching." *Educational psychologist* 41.2, (2006): 75-86.

Koedinger, Kenneth R., et al. "Intelligent tutoring goes to school in the big city." *International Journal of Artificial Intelligence in Education (IJAIED)* 8 (1997): 30-43.

Kothari, Chakravanti Rajagopalachari. *Research methodology: Methods and techniques*. New Age International, 2004.

Kunene, K. Niki. "A Field Experiment on the Effect of Task Decomposition on the Quality of Decisions in a Group Support Environment." *Int. J. Comput. Syst. Signal* 6.2 (2005): 13-25.

Lai, Justin Y., et al. "Prompt versus problem: Helping students learn to frame problems and think creatively." *Third International Conference on Design Computing and Cognition*. (2008).

Le, Nguyen-Thinh, Frank Loll, and Niels Pinkwart. "Operationalizing the continuum between well-defined and ill-defined problems for educational technology." *IEEE Transactions on Learning Technologies* 6.3, (2013): 258-270.

Liu, Zhiqiang, and Dieter J. Schonwetter. "Teaching creativity in engineering." *International Journal of Engineering Education* 20.5, (2004): 801-808.

Madhuri Mavinkurve, " Development and assessment of engineering design competencies" (Doctoral thesis), (2015).

Mangano, Nicolas, et al. "How software designers interact with sketches at the whiteboard." *IEEE Transactions on Software Engineering* 41.2, (2015): 135-156.

Nada , Dabbagh, and Anastasia Kitsantas. "Using learning management systems as metacognitive tools to support self-regulation in higher education contexts." *International handbook of metacognition and learning technologies*. *Springer New York,* (2013): 197-211.

Narciss, Susanne. "Designing and evaluating tutoring feedback strategies for digital learning." *Digital Education Review* 23 (2013): 7-26.

National Academy of Engineering, U. S. “The engineer of 2020: Visions of engineering in the new century”. *Washington, DC: National Academies Press*, 2004.

Norese MF. “Macrame: a problem formulation and model structuring assistant in multiactorial contexts.”, *European Journal of Operational Research*, (1995): 25–34.

Öllinger, Michael, et al. "Does visualization enhance complex problem solving? The effect of causal mapping on performance in the computer-based microworld Tailorshop." *Educational technology research and development* 63.4, (2015): 621-637.

Organisation for Economic Co-operation and Development (OECD). "PISA 2012 results: Creative problem solving: Students’ skills in tackling real-life problems (Volume V)." (2014).

Oslapas, Arunas. "Beyond brainstorming: creative problem-solving techniques." *Proceedings of IEEE Frontiers in Education Conference-FIE'93*. IEEE, 1993.

Pugh, S, “Total design: integrated methods for successful product engineering”, *Addison-Wesley Publishers Ltd*, Strathclyde, 1991.

Pressman, Roger S. “Software engineering: a practitioner's approach.”, *Palgrave Macmillan*, Seventh edition, 2010.

Quintana, Chris, et al. "A scaffolding design framework for software to support science inquiry." *The journal of the learning sciences* 13.3, (2004): 337-386.

Ramon Lawrence, “Teaching Data Structures Using Competitive Games”, *IEEE Transactions On Education*, Vol. 47, No. 4, November 2004.

Reddy, Patil Deepti, et al. "Thinking, Pairing, and Sharing to Improve Learning and Engagement in a Data Structures and Algorithms (DSA) Class." *International Conference on.Learning and Teaching in Computing and Engineering (LaTiCE)*, IEEE, 2015.

Redish, Edward F., and Karl A. Smith. "Looking beyond content: Skill development for engineers." *Journal of Engineering Education* 97.3, (2008): 295-307.

Reeves, Thomas C. "Design research from a technology perspective." *Educational design research* 1.3, (2006): 52-66.

Rivera-Gutierrez, D., Kleinsmith, A., Childs, G., Pileggi, R., & Lok, B. “Self-Assessment Through Interactive In-Action Reflections to Improve Interpersonal Skills Training.*”, IEEE 16th International Conference in Advanced Learning Technologies (ICALT)*, IEEE, (2016): 143-147.

Roblyer, Margaret D., and Aaron H. Doering. "Integrating educational technology into teaching.", 2012.

Runco, Mark A., and Selcuk Acar. "Divergent thinking as an indicator of creative potential." *Creativity Research Journal* 24.1, (2012): 66-75.

Saaty, Thomas L. "Decision making with the analytic hierarchy process." *International journal of services sciences* 1.1, (2008): 83-98.

Sahana Murthy, Sridhar Iyer, and Madhuri Mavinkurve., “Pedagogical Framework for Developing Thinking Skills Using Smart Learning Environments In 'Learning, Design, and Technology”, An International Compendium of Theory, Research, Practice, and Policy'; Michael J Spector, Barbara B Lockee and Marcus D. Childress (eds). *Springer International Publishing* Switzerland, 2016.

Samson D. “Managerial Decision Analysis”, Irwin, Homewood, IL, 1988.

Sandoval, William A. "Developing learning theory by refining conjectures embodied in educational designs." *Educational psychologist* 39.4, (2004): 213-223.

Sandoval, William. "Conjecture mapping: An approach to systematic educational design research." *Journal of the learning sciences* 23.1, (2014): 18-36.

Schon, D. A. “Educating the Reflective Practitioner. Toward a New Design for Teaching and Learning in the Professions.”, The Jossey-Bass Higher Education Series. *Jossey-Bass Publishers*, 350 Sansome Street, San Francisco, CA 94104, 1987.

Sharp, James J. "Methodologies for problem solving: An engineering approach." *The Vocational Aspect of Education*42.114, (1991): 147-157.

Sheppard, Sheri, et al. "Examples of freshman design education." *International Journal of Engineering Education*13.4, (1997): 248-261.

Shields, Patricia M., and Nandhini Rangarajan. “A playbook for research methods: Integrating conceptual frameworks and project management.”, *New Forums Press*, 2013.

Simon HA. “The new science of management.”, *New York, NY: Harper & Row*; 1960.

Soller, Amy, et al. "What makes peer interaction effective? Modeling effective communication in an intelligent CSCL." *Proceedings of the 1999 AAAI fall symposium: Psychological models of communication in collaborative systems*. Cape Cod, 1999.

S. Pugh, “Concept selection: a method that works.” In: Hubka, V. (ed.), Review of design methodology. *Proceedings international conference on engineering design*, Rome. Zürich: Heurista, (1981):497-506.

Tang, Antony, et al. "What makes software design effective?" *Design Studies* 31.6, (2010):614-640.

Tang, Antony, et al. "Design reasoning improves software design quality." *International Conference on the Quality of Software Architectures*. Springer, Berlin, Heidelberg, 2008.

Timothy A Budd, “An Active Learning Approach to Teaching the Data Structure Course”, *Proceedings of the Thirty-Seventh SIGCSE Technical Symposium on Computer Science Education*, 2007.

Treffinger, Donald J. "Creative problem solving: Overview and educational implications." *Educational Psychology Review* 7.3, (1995): 301-312.

Tselios, Nikolaos K., Nikolaos M. Avouris, and Maria Kordaki. "Student task modeling in design and evaluation of open problem-solving environments." *Education and Information Technologies* 7.1, (2002): 17-40.

Velázquez-Iturbide, J. Ángel. "Refinement of an experimental approach tocomputer-based, active learning of greedy algorithms." *Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education*. ACM, 2012.

Volkema, Roger J. "Problem formulation in planning and design." *Management science* 29.6 (1983): 639-652.

Wang, Minhong, et al. "Using cognitive mapping to foster deeper learning with complex problems in a computer-based environment." *Computers in Human Behavior*, 2018.

Warendorf, Kai, and Colin Tan. "ADIS-An animated data structure intelligent tutoring system or Putting an interactive tutor on the WWW." *Proceedings of Workshop" Intelligent Educational Systems on the World Wide Web"* at AI-ED. Vol. 97. 1997.

Willis, Jerry W., ed. *Constructivist Instructional Design (CID): Foundations, Models, and Examples*. IAP, 2009..

Xun, G. E., and Susan M. Land. "A conceptual framework for scaffolding Ill-structured problem-solving processes using question prompts and peer interactions." *Educational Technology Research and Development* 52.2, (2004): 5-22.

Xun Ge., "Designing Learning Technologies to Support Self-Regulation During Ill-Structured Problem-Solving Processes." *International Handbook of Metacognition and Learning Technologies, Springer* New York, (2013): 213-228.

Carmen, Zannier, Mike Chiasson, and Frank Maurer. "A model of design decision making based on empirical results of interviews with software designers." *Information and Software Technology* 49.6, (2007): 637-653.

Publications

1. Reddy, Deepti, Sridhar Iyer, and M. Sasikumar. "Technology Enhanced Learning (TEL) Environment to Develop Expansionist-Reductionist (ER) Thinking Skills through Software Design Problem Solving.", *Tenth International Conference on Technology for Education (T4E)*. IEEE, 2018.

2. Patil Deepti Reddy, Sridhar Iyer, Sasikumar M, “FATHOM: TEL environment to develop divergent and convergent thinking skills in software design”, *17th International Conference on,Advanced Learning Technologies (ICALT),* IEEE, 2017.

3. Reddy Patil Deepti, Sridhar Iyer, M. Sasikumar, "Teaching and Learning of Divergent and Convergent Thinking through Open-Problem Solving in a Data Structures Course", *International Conference on Learning and Teaching in Computing and Engineering (LaTiCE),* IEEE, 2016.

4. Patil Deepti Reddy, "Teaching and Learning of Divergent and Convergent Thinking through Open-Problem Solving in a Data Structures Course“, *Doctoral Consortium of LaTiCE,* 2016.

5. Reddy, Patil Deepti, et al. "Thinking, Pairing, and Sharing to Improve Learning and Engagement in a Data Structures and Algorithms (DSA) Class." *International Conference on Learning and Teaching in Computing and Engineering*. IEEE, 2015.

Acknowledgement

I thank Prof. Sridhar Iyer and Prof. Sasikumar M., for their constant support and guidance at every stage of this research. I am grateful for their time, expertise, and encouragement given throughout, which kept me focused and motivated towards achieving the research goals. I express my gratitude and thanks to Prof. Sahana Murthy for being my teacher and Research Progress Committee member, helped in developing the research aptitude for educational technology research. I also thank Prof. Ganesh Ramakrishnan, Research Progress Committee member, for his feedback and comments, which helped in improving my thesis work from domain aspect.

I thank my fellow researchers in Educational Technology department, IITB, for always being friendly, lively and vibrant that helped to keep my spirits and motivation high in doing research, discussions, and any other extra-curricular activities. I thank Kavya for her input in system development and also helping in doing a pilot test of the system, which was useful in improving the usability. I thank Shitanshu, Aditi, and Lakshmi for their feedback on the pedagogical aspects, Rwito for helping in analyzing data, Gargi ma'am for her feedback on research methods, Anurag and Soumya for being open for discussions, and Prajish for his help in resolving system related errors during development. I am grateful to Rumana and Ulfa for being my interns and developing the initial version of Fathom. I thank Navneet and Pankaj for their help in conducting the research study with students. Last but not the least, I thank Ajit for his backend support in hosting the system onto the web server and his willingness to help even during holidays.

All this would not have been possible without family support. I am grateful to my husband for the unconditional support in every aspect, whether it is emotional or technical. I appreciate my kids, Vamshi, and Trisha for their encouraging words, and love them for exhibiting responsibility in handling things on their own in my absence. I thank my parents, mother-in-law, and father-in-law for being my backbone which made this journey smooth. I thank my father for his constant motivation and instilling in me the courage to take up challenging task, hard-work and not to give up attitude. I imbibed the inner strength of my mother, who taught me to deal with problems in a calm and simple way.

I once again express my gratitude towards each and everyone who supported me directly or indirectly throughout my academic career.

I thank the Almighty for giving me this opportunity, the strength and the ability for this journey in the pursuit of knowledge.

Date:

IIT Bombay, India P. Deepti Reddy