
This is the author version of the work. There may be differences between this version and the published version. You are advised to consult the published version if you want to cite from it. https://doi.org/10.1109/FIE58773.2023.10342646

https://eprints.gla.ac.uk/306785/

Deposited on 18 September 2023
An Elaboration of Research Led Computer Science Framework for Early Education

1st Noha Abdulkhalig M. Alharbi
Centre for Computing Science Education
University of Glasgow
Glasgow, UK
N.Alharbi.2@research.gla.ac.uk

2nd Quintin I. Cutts
Centre for Computing Science Education
University of Glasgow
Glasgow, UK
Quintin.Cutts@glasgow.ac.uk

Abstract—Teachers face challenges in teaching new subjects at the early primary level. The difficulties vary from subject to subject. Teachers’ first challenge in Computer Science (CS) and Computational Thinking teaching is their technological literacy and competency challenges. Lack of confidence, time, and willingness are significant reasons for the challenge. One of the studies on mathematics teaching shows that teachers need more knowledge tools and models to facilitate the analysis during the teaching process. Researchers assert that educational practice has focused more on technology than on new pedagogies needed to truly use technology in a transformative way. Frameworks are designed to meet high-level objectives set by the authorities. Teachers are provided with teaching materials for developing the course contents. The frameworks act as a connection between high-level objectives and low-level teaching materials. In this paper, we present the elaboration of a framework for teaching CS at the early stages of education. The framework consists of three components based on modelling real-world problems into a machine-world context and then realising the machine solution in the real world. Moreover, the problem-solving component of the framework emphasizes the iterative process of learning that strengthens the previous concepts and provides a logical connection between the previous and new concepts. The framework was evaluated in the context of Saudi Arabia, where the teachers were interviewed individually and then provided with a guide for teaching grade-4 learners. The guide was developed using the material provided by the Saudi Ministry of Education. The teachers taught using this guide over a sequence of lessons. At the end of this time, a second individual teacher interview was conducted to determine teacher experience. The results show that teaching using the framework is effective, and it is not only easier for teachers to understand the teaching method but also for learners to learn the content quickly.

Index Terms—K-12, Computer Science Education, Computational Thinking

I. INTRODUCTION

Computational thinking (CT) development is one of the desired outcomes of today’s education system. It helps in solving real-world problems. Computer science (CS) education is considered a tool to develop the ability among learners. It is argued that CS should be taught from the early learning stage as other subjects, such as mathematics and science [1], [2]. This early developmental activity helps build fundamental problem-solving concepts from early learning stages. Most countries all around the world are making an effort to develop a K-12 CS framework with similar objectives of developing computational thinking. These frameworks set objectives to achieve. The teachers are provided with the necessary materials and contents to develop courses for teaching learners. The framework acts as a guideline for teachers to develop the outline with eyes on meeting high-level objectives. The framework should be able to assist teachers in meeting the objectives while maintaining ease of use and quickly learnable characteristics for teachers. The framework acts as a glue between course contents and materials and the objectives set by a country [3]–[5]. The framework should be easily learnable and have a logical structure that can be easily explained. Meeting the subject’s essence is one of the key characteristics of a framework. A relation between objective, framework, and contents is provided in figure 1. While meeting the objectives, the framework concerns the modelling of real-world problems in a machine-world context and iterating the whole process to further embed the previous concepts.

Teachers are given high-level objectives and low-level classroom materials but do not have the knowledge to make the connection. This is challenging for teachers and ultimately affects the objectives of teaching computational thinking skills among early-stage learners. A framework addressing teaching and learning difficulties is proposed in [6]. This framework identifies three main components of learning, including the problem domain, modelling domain and problem-solving domain. This paper elaborates on the research foundation of the study. It then evaluates the elaborated framework for its performance in the Kingdom of Saudi Arabia (KSA) which is in the initial phase of introducing early-stage CS education. Overall, the core research question is: Can a framework based on CS education research help to bridge the gap for primary teachers between high-level learning objectives and low-level classroom materials?

II. DIFFERENT PROPOSED SOLUTIONS FOR CS EDUCATION

The logical thinking involved in computer programming and computer science is generally challenging to learn. There are widespread problems with failure rates in university classes [7], as reported most recently in [8]–[10]. The empirical results of previous studies about student programming ability show that most beginners don’t have correct concepts of variables,
parameters and object-oriented programming [11]. A systematic review of failure rates in different countries collected data for 15 countries and 51 institutes and found that the failure rate is over 30% for basic introductory programming courses in general, and it is independent of moderators such as country, programming language, class size and level of education classes [7]. If there are such problems in university classes, where the students take the courses optionally and presumably, therefore, have the motivation and some aptitude, we should be very worried about what may happen in school classes. These problems with learning CS, specifically programming, were identified many years ago. Five major areas of difficulty in learning computing were identified [12]:

- Orientation (understanding the need for programming). Learners do not understand how programming can solve problems and how this skill is helpful. Orientation, also called motivation, is a necessary first element that should be part of CS curricula.
- Notional Machine (understanding the tools of programming language). The operation of programming tools can be captured using the so-called notional machine to provide sufficient explanation to form a sound mental model without unnecessary detail. Sorva [13] interviewed students about their understanding of different types of variables in Java and identified that people create various mental models about programming concepts and how a computer works. Variables are a basic programming concept that students often partially or incorrectly understand, resulting in their failure to develop correct programs [13]. A teacher must structure courses and dedicate sufficient time to properly understanding basic concepts. This is particularly important for high-level programming languages with complex constructs and hidden machines, or students will be overloaded with advanced concepts before understanding the most basic ones. Thus, a firm understanding of the operation of a programming language is required before students can successfully solve problems and implement solutions.
- Mastering syntax of different programming languages. The syntax of a programming language must be rigorously learned and matched to appropriate language semantics. Without this mastery, students will waste effort on consciously translating syntax and easily become mentally overloaded.
- Lack of standardised structure for solving similar problems in different programming languages, referred to as “patterns” or reusable code templates that solve recurring sub-problems [14].
- Lack of problem-solving ability, even if a learner has mastered a programming language, analysed the problem, designed and implemented an appropriate solution, and tested the outcome. Programming constructs should be regularly reinforced through comprehensive exercises in a systematic way. Robins, Rountree, and Rountree [15] identified difficulties in learning the declarative aspect of programming. One of the major challenges in the literature is the focus on programming skills with explicit consideration of the declarative aspect of programming. However, a major limitation of all these knowledge-based theories is that they often fail to consider the way in which knowledge is used or applied [15]. Therefore, Brooks [16] proposed a program comprehension model to address this challenge. According to the theory, knowledge is divided into two domains, which are the original problem domain, in which the problem is formulated in its context and the solution domain, in which available constructs such as algorithms and data structures are used to solve the problem. Program comprehension involves reconstructing knowledge of these domains and serves as the bridge between them. This comprehension process could be top-down, bottom-up, or hypothesis-based initially and later refined using exact information gathered from program text or documentation. The author suggests that program comprehension can be defined as a mapping between the problem domain and the solution or modelling domain with knowledge of the structure of both domains [16].

III. OVERCOMING COMPUTING CHALLENGES

Two contrasting perspectives on learning computing are that it: 1) is an innate ability that learners are born with—with many teachers considering it unlearnable [17], and 2) can be learned, but the way to do so has yet to be determined for every student [18]. Because a body of research exists indicating that computing is learnable, a developmental learning/teaching strategy can be developed and evaluated [18].

Jean Piaget proposed that the ability to learn improvements with biological age and involves four learning phases: sensory-motor, pre-operative, concrete operational, and formal operational [19], [20]. Mixed results have been reported in measuring programming ability among learners. Initial findings by one group show that learning programming cannot improve problem-solving, whereas another group provided opposite results [21]. For example, Kurtz [22] compared the introductory programming performance of 23 students and got a high correlation with Piagetian levels while Barker and
the common-sense claim is that given some target domain of Learning Edge Momentum (LEM) [32]. According to this, dent and independent concepts and proposed the hypothesis improve the code tracing skills of students [30].

correlated with code writing and emphasized the need to through stepwise regression that these constructs are positively code writing, and explanation, was developed. It was found such as basics, sequencing, iterations, exception handling, writing as a dependent skill on code tracing. A set of dif-

ferent programming-related questions were asked of students, making further learning/local repair more difficult. When learning related concepts sequentially from a domain, this simple effect means that the success or failure of learning becomes self-reinforcing over time, and learning acquires momentum toward either a successful or an unsuccessful outcome for the domain as a whole. A further claim of the LEM hypothesis is that not all learning edges are created equal. For some domains, the significance of the LEM effect is particularly strong, such that early success or failure can become strongly predictive of the final learning outcome. LEM varies in strength depending on the properties of the target domain. In particular, momentum varies in proportion to the extent to which the concepts in the domain are either independent or dependent (interrelated/integrated). When the domain consists of tightly integrated concepts, the significance of the edges is increased because they afford multiple constraints on and connections to related concepts. In such cases, the momentum effect (positive or negative) will be vital. Robins' work is important here because programming concepts are highly interrelated, and so great care would be taken with any teaching to ensure that each individual concept is thoroughly understood before moving on to the next one.

Code comprehension is an important topic that requires understanding from both programmers and teachers. Brooks' work has already been discussed [16]. Schulte, Clear, Taherkhani, Busjahn, and Paterson [33] evaluated various pro-

gram comprehension models. Mainly, all of these models are composed of two parts: internal and external. The internal parts are related to the programmer's skill, which can be divided into the cognitive structure that he/she used to develop the mental abstraction of a program and the assimilation processes adopted by the programmer to solve the problem. These assimilation processes are categorized by researchers into four different categories; top-down models, bottom-up models, opportunistic models, and integrated models. This study selected six different other studies about program comprehension targeting different assimilation processes. The commonalities in all of these models are different assimilation processes, knowledge bases, and mental representation. Different types of knowledge include programming language semantics, goals, plans, efficiency, domain, and discourse rules. The educational inferences from the study identified three different areas important for program comprehension. The first area is about goals that have to be set for introductory programming courses and contents that are identified by Du Boulay [12]. The second is about teaching and learning sequences. Depending on the requirement of the course contents, any assimilation approach can be adopted for teaching and learning. Finally, the third area is related to the teaching method adopted by teachers for teaching programming concepts [33].

In summary, the Neo-Piagetion theory [18] indicates a
learning trajectory and developmental sequence, Robins [31] suggests taking care that every concept is properly understood, Du Boulay [34] suggests that language and notional machines must be addressed, and Brooks [16] and Shulte et al. [33] point out the importance of program comprehension. All of these works identify that the curriculum should be designed in a way that understanding of problem domain and solution domain, and modelling and problem-solving should be integrated to ensure the re-enforcement of prior concepts.

Fig. 2. Research-led Computer Science Framework (RLCSF).

IV. RESEARCH-LEAD COMPUTER SCIENCE FRAMEWORK (RLCSF)

A Research-led CS Framework (RLCSF) is designed in [6]. The aim of the RLCSF is to make the learning of computational concepts easier. It targets the five areas of difficulties identified by DuBoulay, presented earlier, and works on four main characteristics derived from other subjects such as Science [35] and Math [36]. These characteristics include ease of use, explicit structure, conceptual to practical and meeting the essence of the subject.

As shown in Figure 2, the framework is divided into three main components: the Application Area (also called the problem domain), Building Blocks (Modelling and computing domain) and Problem-solving. The overall design of the framework has the larger objective of developing motivation among learners. This is achieved by presenting complex computing concepts through simpler, unplugged daily life activities. The framework mainly consists of two worlds: the real world and the machine world. The computing concepts are taught and explained through real-world, simpler, unplugged activities. However, understanding these concepts in computing terms in the machine world is called notional machine and is explained by Piaget’s theory of learning [22], [37]. In this theory, the conversion from concrete to abstract transition of learning is actually a transition of concepts from the real world to the machine world. The next stage is implementing the concepts. For this, computer programming languages are used. The lack of standardized programming language structure is a common problem in teaching programming. The RLCSF emphasizes the need to use pattern-oriented instructions in teaching programming so that generalized computing concepts can be taught, thereby reducing the cognitive load at later stages of learning. The third important stage of RLCSF is problem-solving in which the spiral approach is used based on Bruner’s spiral curriculum [38] and the Learning Edge Momentum (LEM) theory [28]. In these theories, each new concept should be taught based on its relation to the previous concepts. This is how the new concepts have logic in the overall problem scenario but also the previous concepts are reinforced.

The three main components of RLCSF are explained as follows:

1) Problem Domain: The problem domain deals with the broad context of a given problem. Learners need an understanding of the context in which a problem is embedded and the kinds of related data and processes involved. Learning about this is a complex task that is taught through unplugged activities. These activities are made part of curriculum material. All of these are in the frame of the real world and computing concepts are implicitly explained through the unplugged activities given in the curriculum material.

2) Modelling and Computing Domain: This component of RLCSF belongs to the machine world and covers learning difficulties such as notional machine, lack of standardized programming structure and learning multiple programming languages. The notional machine is the first problem and relates to programming concepts such as iterations, conditions and sequences. RLCSF tries to provide these concepts and thinking through unplugged activities. The next step is learning a programming language. The teaching of a programming language is achieved through a pattern-oriented teaching method that follows the standardized structure of different constructs in programming. Although teaching and learning particular programming languages is necessary to realise the proposed solution, for a better comprehension of programs, teachers should be aware of the level of the program to teach. For example, instruction level, function level, class level or complete program level [25], and the level of detail in which the computer program needs explanation as provided through the Program Comprehension Model [39], also called the Block Model. Each part of the teaching is designed so that the learner understands its purpose. However, teaching through pattern-oriented instructions reduces learners’ cognitive load at later stages of education while learning other programming languages.

3) Problem Solving: Once the necessary problem and machine domain understandings are learned, problem-solving can be attempted - real problem-solving cannot be achieved until this point. For a given problem, the data and computing processes are extracted in the problem analysis phase. The extracted processes are mapped to the computing concept (notional machine) in the modelling phase. The model is implemented using the given programming language, and the results are validated against the given problem statement.
V. METHODOLOGY

Having described the RLCSF, a study is now presented to investigate the Framework’s value to teachers in bridging the gap between high-level learning outcomes such as “the development of computational thinking skills” and provided classroom teaching and learning materials.

For the study, teachers were recruited from two groups, all teaching computer science to Primary 4 (age 9) and following the same curriculum. Teachers in both groups had already spent a term teaching the first unit of Scratch programming, having received training on using the Scratch application (not on programming). The first group, referred to as the Experienced group, had additionally received instruction using a guide developed by the authors that introduced the RLCSF and explained the provided curricular materials in the context of the RLCSF. The other group, referred to as the New group, had received no additional instruction. All teachers from the Experienced and New groups, with both CS and non-CS backgrounds, are noted in Table I.

The teaching guide used the CS curriculum book’s materials for primary level 4 and RLCSF guidelines to develop sample lessons. These lessons explained the concepts of iterations, conditions and sequencing from the real world and provided an implementation in the Scratch programming tool. The computing concepts from the real world are mapped in a simple way from the problem domain (real world) to the solution domain (machine/modelling world). Then, the complete process of implementing previous concepts with the new concepts is explained to demonstrate the collective importance of all related processes (sequences, conditions and iterations) in a cohesive way.

An interview session was conducted with the Experienced and New teachers one by one, lasting for 15-20 minutes each. A few questions were asked related to the essence of CS subject, ease of use in the provided teaching material, the structure of the curriculum and conceptual to practical implementation. After all the interviews, a guide session was conducted for all to explain different computing concepts in a cohesive way.

TABLE I
TEACHER GROUPS AND THEIR ACADEMIC BACKGROUND

<table>
<thead>
<tr>
<th>No.</th>
<th>Teacher</th>
<th>Group</th>
<th>Background</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Teacher 1</td>
<td>Experimental</td>
<td>CS</td>
<td>T1-Exp-CS</td>
</tr>
<tr>
<td>2</td>
<td>Teacher 2</td>
<td>Experimental</td>
<td>CS</td>
<td>T2-Exp-CS</td>
</tr>
<tr>
<td>3</td>
<td>Teacher 3</td>
<td>Experimental</td>
<td>Non-CS</td>
<td>T3-Exp-NCS</td>
</tr>
<tr>
<td>4</td>
<td>Teacher 4</td>
<td>New</td>
<td>Non-CS</td>
<td>T4-New-NCS</td>
</tr>
<tr>
<td>5</td>
<td>Teacher 5</td>
<td>New</td>
<td>CS</td>
<td>T5-New-CS</td>
</tr>
<tr>
<td>6</td>
<td>Teacher 6</td>
<td>New</td>
<td>CS</td>
<td>T6-New-CS</td>
</tr>
</tbody>
</table>

VI. ANALYSIS OF PRE-GUIDE INTERVIEWS

The audio data from teachers is transcribed into text form—different themes were extracted from the text data through inductive thematic analysis. A description of different themes extracted during the process is given in Table II.

TABLE II
THEMES EXTRACTED FROM TEACHER INTERVIEWS

<table>
<thead>
<tr>
<th>Theme</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The essence of Subject</td>
<td>The reason given by teachers for teaching computer science</td>
</tr>
<tr>
<td>Ease of Use</td>
<td>The level of difficulty faced by teachers in teaching computer science</td>
</tr>
<tr>
<td>Explicit Structure</td>
<td>The comments of teachers about the structure of the curriculum.</td>
</tr>
<tr>
<td>Conceptual to Practical</td>
<td>The ability of the framework to transform theoretical concepts to practical implementation.</td>
</tr>
</tbody>
</table>

A. Essence of Subject

One of the questions by the teachers is about the essence of the subject that they understand related to the framework. The important responses of teachers are as follows:

“Students interact with Scratch and learn programming.” (T1-Exp-CS)

“Acquiring new knowledge and multiple skills when teaching programming and I realized the importance of teaching it to students at young ages. I learned to link the study of programming with the life problems that I encounter and to use programming steps to deal with these problems.” (T3-Exp-NCS)

“I very much enjoyed the positive aspects of the student’s enthusiasm and interaction with regard to the negatives, that the unit was not supposed to be divided, and the programming unit was supposed to be taught in full, even if it took an entire semester in order for the students to benefit and understand more.” (T6-New-CS)

Teachers showed that there is a relationship between computer science education and problem-solving. However, they were not able to identify and benefit from the relationship. This further reduces the motivation.

B. Ease of Use

The opinion of teachers was explored by asking them about the objectives, methodology, timelines and objectives set by the curriculum. The teachers described the difficulty in achieving the objectives set by the curriculum.

“The topics and examples are suitable for teachers in the early stages because they are simple to learn the basics from real life, but we need more training to understand them and link them to programming.” (T3-Exp-NCS)

“The curriculum needs to be rearranged and prepared to suit the age group so that the sequence of topics is from the easiest to the most difficult and from theoretical to practical” (T4-New-NCS)

“The purpose is that the one who cares about the tendencies and needs of the students, their abilities and their preparations and allows the students to carry out various activities that are consistent with these tendencies and work to satisfy those needs. Hence, we can say that the activity curriculum has moved the focus of attention from the study material to..."
the student and made it the focus of the educational and educational process” (T5-New-CS)

Teachers pointed out the need for training about the curriculum and restructuring of the curriculum.

C. Explicit Structure

The role of CS education at the primary level is explored from the teachers’ point of view. The requirement of a framework structure that can be followed is explored. The comments as are as follows:

“The goal is to create a generation capable of harnessing technology for its educational service important role. It is a foundation in computer basics. It needs a clear guide to the objective” (T2-Exp-CS)

“Create a generation that deals with the computer with ease. But, need a guide to do it” (T6-New-CS)

The structure of the intended curriculum was not clear to teachers.

D. Conceptual to Practical

The experimental group teachers showed a better understanding of the three-stage model as follows: “the students are now expressing the aspects of their lives using Scratch codes, such as making a glass of juice and the movement of hands in swimming, following the steps that they studied in the previous guide, so the students apply these steps in any similar aspect of life and its problems” (T1-Exp-CS)

The understanding of teachers in the control group is as follows:

“When understanding the problem, analysing it, applying solutions, and applying algorithms in solving mathematical problems” (T4-New-NCS)

“It helped that they started planning solutions to a problem and it showed up in some of the Scratch clips that were done as study projects.” (T6-New-CS)

VII. POST-GUIDE INTERVIEW DATA

Another session of interviews was conducted with the teacher after the training session. The objective of the second session was mainly to identify teachers’ teaching experience after they teach according to the methodology given in RLCSF. Table III presents the themes extracted from the data.

<table>
<thead>
<tr>
<th>Theme</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Essence of Subject</td>
<td>The reason given by teachers for teaching computer science</td>
</tr>
<tr>
<td>Ease of Use</td>
<td>How much easy to understand and follow the framework.</td>
</tr>
<tr>
<td>Implicit structure</td>
<td></td>
</tr>
<tr>
<td>Understanding of teachers</td>
<td></td>
</tr>
<tr>
<td>Conceptual to Practical</td>
<td>Apply theoretical concepts to practical solutions.</td>
</tr>
<tr>
<td>Practical to Conceptual</td>
<td></td>
</tr>
</tbody>
</table>

A. Essence of Subject

A clear improvement in teachers’ understanding of the objective of the K-12 CS framework is observed in the interviews. The responses of teachers after the guide are as follows:

“I enjoyed teaching programming in class this semester because the main motive behind teaching computer science is to improve learners’ problem-solving skills. Programming is not only about learning to design computer models but to apply them to the problems we face in life.” (T2-New-CS)

“The good thing is that the subject is interesting for students and is useful, important and necessary in the age of technology and thinking. As for me, I can see that computational thinking can be developed using the guide.” (T5-New-CS)

Teachers from both groups, control and experimental, admitted that they better understand the purpose of teaching computers as a way to improve learners’ problem-solving skills.

B. Ease of Use

A question is asked about their experience of teaching computer science curriculum in previous and current teaching sessions. The difference in teaching is explained by different teachers as follows:

“Learning has changed due to new content such as using repetition in Scratch, pausing and timing of pauses, colouring in Scratch, creating 2D drawings using Scratch blocks and Building an object’s character. There are many things to do in CS education and the guide helped to easily plan lessons.” (T2-Exp-CS)

“Due to the easy description of concepts, the students became aware and interested in the importance of programming and the expansion of its use by creating and designing different projects.” (T5-New-CS)

The teachers appreciated the difference before and after teaching the CS concepts.

C. Explicit Structure

A major question was about teachers’ experience with the effectiveness of teaching guides. The answers of teachers show positive responses about the RLCSF structure.

“The guide is understandable its style is simplified and it is in line with the examples of curricula. In each model, the guide presents the steps for solving the problem, which is the first stage: Understanding the scope of the problem. The learner must understand the context of the problem and give the expected inputs, processes and outputs. • Given inputs: number of steps, waiting time • Operations: repetition and rotation • Expected output: movement of the cat Phase Two: Learning computing tools Scratch tool (Iteration with waiting Repetition: Repeating a task a certain number of times Phase Three: Solving the problem. This is a three-step process. Now, the learners will solve the problem given in the first stage. The first step is to analyze and understand the problem. The learner will identify the inputs, processes and outputs using the knowledge gained from the previous examples. The second step
is to write the algorithm in a simple English/Arabic language, explaining the steps point by point.” (T1-Exp-CS)

“The guide is an understandable and clear structure, and this guide is clearly in line with the established curriculum, and the reason is the presence of clear images and the way to explain the paragraphs in a clear and very easy way.” (T4-New-NCS)

The teachers explained the effectiveness of the teaching guide in terms of their understanding of the framework structure. The teachers showed confidence in the guide, hence the RLCSF for its structure.

D. Conceptual to Practical

The basics of RLCSF is the three-stage model. Teachers’ experience with the effectiveness of the three-stage model is asked of the teachers. The response is as follows:

“The three-stage methodology can help learners develop computational thinking skills. When identifying the problem and understanding its context, inputs and outputs, then learning computing tools and applying the steps of the problem from identifying the problem, analyzing the problem, and then writing an algorithm and programming to express that problem, this will be useful. Helps develop computational thinking skills” (T1-Exp-CS)

“The introduction of a computer course in the primary stage helps to learn the basic skills of using computers, including them in the traditional method based on paper transactions.” (T5-New-CS)

VIII. DISCUSSION

The interview sessions are part of evaluating the RLCSF in the context of Saudi Arabia’s K-12 CS education. The teachers who participated in the study attended a pre- and a post-guide/post-teaching interview session. However, the study’s objective is to determine teachers’ experience with different teaching pedagogy provided in the RLCSF. In the pre-interview session, there is less motivation from teachers observed about CS education at an early level. The teachers found teaching CS difficult and recommended restructuring the existing curriculum. This pre-interview session was followed by a teaching guide developed based on RLCSF teaching pedagogy. The post-interview from teachers revealed the generality of RLCSF, including its explicit structure, ease of use, conceptual to practical support and meeting the objective set by the RLCSF.

IX. CONCLUSION

The paper elaborates on the research foundation of the existing research-led CS framework. Related teaching and learning difficulties are discussed, and the framework is evaluated based on four desired characteristics in any education framework. Teachers in Saudi Arabia perform the evaluation in the form of interviews. These interviews are conducted in two sessions, and a teaching guide based on the RLCSF is provided to teachers after the first interview session. The results show that the RLCSF is practical and improves learners’ performance, and increases teachers’ ease of use. However, an extended study covering a larger number of teachers and data collected through surveys and interviews can help better elaborate the results.

REFERENCES


