*Advances in Engineering Research, volume 198*
**International Seminar of Science and Applied Technology (ISSAT 2020)**

# Influence of Computational Thinking Framework on User Interface Design

Transmissia Semiawan*

*Department of Computer Engineering and Informatics, Politeknik Negeri Bandung (Bandung State Polytechnics), Indonesia*
**Corresponding author. E-mail: t.semiawan@polban.ac.id*

**ABSTRACT**
Computational Thinking (CT) is a problem-solving approach that consists of four fundamental thinking processes: decomposition, pattern recognition, abstraction, and algorithm. As an improvement to the author's previous study, this paper emphasizes the effects of CT on user interface (UI) design as part of the phases in a software development life cycle (SDLC). An action research approach was used to identify key factors to be considered in UI design regarding the CT process. The result revealed that the abstraction process and pattern recognition were crucial factors in UI design. It shows improvement in the abilities of a number of designers who have succeeded in designing the UI prototypes to increase by approximately 5 times their abstraction and pattern recognition. Hence, the two fundamental processes played an important role in designing the system behavior based on the perspective of user needs and requirements.

***Keywords:*** *computational thinking, user interface design, design thinking, abstraction, decomposition, pattern recognition; algorithm.*

## 1. INTRODUCTION

User interface (UI) design is one of the most important steps in a software development life cycle (SDLC). UI design needs to accommodate user experiences (UX) since it is related to how users interact with the systems, based on the design principles by Jakob Nielsen in Interaction Design Foundation. The success of UI design highly depends on how user requirements are understood based on the following objectives:

a) To analyze the domain of the problem;

b) To identify the best as well as alternative solutions to the problem;

c) To design and implement the selected solution;

d) To evaluate the effects of the implemented solution and to make any necessary revisions.

Considering these objectives, designers need to be able to "read and understand" a large number and variety of data to explore and discover problems and solutions from within the data that relate to the UX toward the systems being designed. This step needs specific thinking process skills that would enable designers to understand ideas embodied through the data. Analytical ability as well as critical thinking skills are therefore significant to signify and identify problems as well as to discover reasonable solutions across the data.

### *1.1. Design Process*

A design process entails systematic design thinking within which creative and innovative skills play a part. Those skills are required as the purpose of the design process is to produce a solution to a problem. The problem–solution correlation should be understood as the process of fulfilling requirements using appropriate deeds of solution. This means that the emphasis is not on the design 'artifact' but more on the way the artifact functions as requested. Therefore, there is no right or wrong in a design except that the design as a solution must meet the requirements that have been defined as solutions to the problems.

In software design, the design artifact as the solution is measured through the 'fitness for purpose' criterion [2]. The measurement is not easy to execute, as the design is a reflection of the designers' thoughts. Moreover, designing software itself is considered difficult because of the software properties, including (F. Brooks in Budgen) [2]:

 

a) *the complexity* of the software components and their processes behavior,

b) *conformity* of the software manners to both users and systems requirements,

c) *ease of changeability* of the software but disregarding the cost of the change,

d) *invisibility nature* of the software as an intangible object making it difficult to describe the design.

In UI design, the design process is more complicated because the artifact or design solution must be developed to reflect the user's needs or preferences, such as ease of use, user experience, or language [3]. This requires a coherent perception between designer and user. This 'connection' is necessary to help designers understand the problem-solution ideas from user perspectives including [4] [5] [6]:

a) identifying users' problems and

b) understanding users' needs and requirements;

c) generating design ideas or design solutions that meet users' needs and requirements; and

d) experimenting and realizing the design concept.

Thus, based on the above perspectives, it is essential to have a critical design approach in UI development.

### *1.2. Computational Thinking*

Computational thinking (CT) indicates a coherent thinking process that emphasizes 'understanding contexts' through data that represents phenomena or facts. As a problem-solving approach, it entails a fundamental thinking process that embraces analytical ability, conceptualizing, describing, interpreting, evaluating, judging, and other human capabilities, which include [3] [7] [8] [9]:

a) *Abstraction*: The ability to describe phenomena in the real-world through a conceptual model based upon provided data. This can be used to formulate and identify problems over the data.

b) *Decomposition*: An ability by which one can define such a description in detail. In terms of problems, they can be divided into some smaller problems, so that they are simpler to be solved.

c) *Pattern recognition*: Ability to find similarities and differences to understand and signify characters over data. This can also be used to formulate problems.

d) *Algorithm*: A logical way of constructing a solution to the problem by producing one by one sequence of processes.

The above are the capabilities required by software designers especially when they have to reduce the effects of software properties to produce the implementable design solution that meets users' needs and requirements. Accordingly, the designers must have the capacity to develop ways in which their bonding perception with users is deepened and the users' problems are well understood. In this sense, the designers should not only have the capability to communicate with the users but more on interpreting the various type of data they encounter during the interaction. These skills require creative thinking and imaginations of the designers, particularly in coherently making sense of the data. Here is where the four fundamental CT processes come into play.

Semiawan [10] illustrates the general idea of using CT for problem-solving over the data. It explains how the four fundamental processes of CT are used in the problem-solving process. Figure 1 is an illustration of the way in which people signify phenomena as problems and discover solutions to the problem from within data. This is done by amalgamating analytical ability, creative ability, initiative, persistence, critical thinking to organizing, generalizing, classifying data through abstraction, pattern recognizing, decomposition, and algorithm.

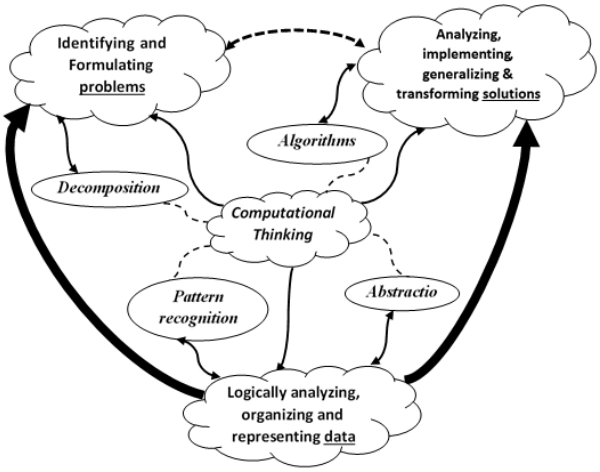


**Figure 1** Detail CT over problem identification and constructing solution using data [10]

There had been several studies on applying the CT framework, in which the framework was analysed and investigated in relation to learning process, such as developing creativity within a group of collaborators [11], implementing CT competencies in learning [12], modelling CT implementation in a class [13], developing CT in higher education through creative programming [14], and understanding CT through creative programming [15]. These studies explored the practicality in which the CT skills could nurture creativity and ability in problem-solving.

A study by Wang, Schneider, & Valacich [11] and Wang & Wang [12] emphasized their analysis of applying CT through the interaction within a collaboration group and between students and teachers. The results of their studies explained that creativity in

problem-solving relied upon the level of difficulty of the problem and the communication ability amongst the members of the group. A study by Curzon, Dorling, Ng, Selby, & Woollard [13] concluded the application of CT for problem-solving over three phases: (a) a 'why' phase to understand the problem; (b) a 'how' phase to solve the problem; and (c) a 'what' phase to present the solution to the problem. The last two studies on CT framework by Romero, Lepage, & Lille [14] and Brennan, Balch, & Chung [15] had been experimental studies that examined the application of CT through creative programming using programming tools. In general, the results of both studies revealed that evaluation and assessment for creativity still requires humans over tool-based automated assessments.

In terms of the effectiveness of UI design, thinking processes that have been discussed above are thought to be suitable approaches in solving UI design problems.

This paper accentuates the way in which the CT process directs the designers' understanding of the domain problems and provide a design solution based on users' expectations. The discussion is based on the results of the previous study by Semiawan [10] that examined the UI designs with the absence of the CT processes.

To contribute to this research, UI designers will agree and understand the importance of the CT as a complement to support their illustration of the way the system should respond to users' demands.

## 2. METHODOLOGY

This study used an action research method to explore the influence of four fundamental CT processes on UI design as part of the software design process to understand the gap between the result of 'what is expected to happen' and 'what actually happens' [16]. Through this method, the observer was able to take action while analyzing situations or phenomena throughout the research. Furthermore, Burns [16] describes that the 'what is expected to happen' actually investigates the phenomenon with the intention of signifying and comprehending the problematics over the phenomenon as well as identifying the future-related way out that should be taken regarding the problematics.

Action research is typically designed in several related iterative steps, and each iteration includes planning, action, observation, and reflection to the action [16]. This study was implemented in two iterations, in which the first iteration focused on observing the phenomena without using the CT framework, and the second iteration with the CT framework. In the first iteration, the participants conducted actions naturally with no guidance or directions. In the second iteration, the actions occurred based on the intervention of the researcher to the participants to use a template related to the CT framework. The intervention was expected to help participants conceptualize finding the systems' work patters.

As part of the software development life cycle (SDLC), observation on each iteration emphasized their analysis and design phases. The designers have to be able to relate the design process phase to the analysis phases in the cycle by analysing the existing system (As-Is system) to get users' requirements and linking users' requirements to systems requirements as the groundwork of designing a new system (To-Be system). To achieve these tasks, the designers can use a system model to delineate using four viewpoints: behavioral, functional, structural, and data modelling [2]. In general, these actions include the following:

a) As-Is system analysis: the construction of a domain model and problem identification;

b) To-Be system analysis: the composition of user behavior, data model, and task analysis/process model;

c) Prototyping: the focus on usability goals and aesthetic preferences [1].

Figure 2 illustrates this research design in general.

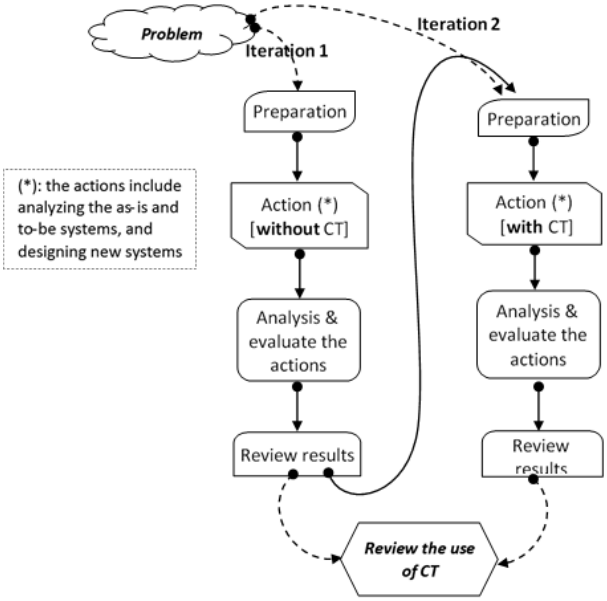


**Figure 2** Action-research based research design

This study involved a total of 200 participants who were divided into 2 iteration intervals, in which the first interval was conducted without CT framework and the second was conducted without CT framework.

Adapting from Semiawan [10], Table 1 shows a mapping of the fundamental processes of the CT framework over the following analysis and design process:

a) *Abstraction and pattern recognition* were investigated through system modelling to delineate the software as the object of the study as well as analyse user and systems requirements. This was

evaluated through the ability in organizing, analysing and representing the data over a domain or conceptual model, and technical or physical model.

b) *Decomposition* was reviewed based on the way the problem was formulated through problem identification, as well as through data model, user behavioural model, and process model.

c) *Algorithms* through which the design solution was calculated were examined during the prototyping design phase over the usability and the aesthetic of the UI design.

**Table 1.** Analysis and design processes on the CT framework [Adaptation from [10]]

| CT Framework / Design Process | Formulating / Generalizing problems (*decomposition*) | Organizing & analyzing data (*pattern recognition*) | Representing data (*abstraction*) | Implementing solutions (*algorithm*) |
|---|---|---|---|---|
| **As-Is System Analysis:** | | | | |
| Domain Model | | V | V | |
| Problem Identification | V | V | V | |
| **To-Be System Requirements Analysis:** | | | | |
| User Behavioral model | V | V | V | |
| Data Model | V | V | V | |
| Process Model/ Task Analysis | V | V | V | V |
| **Interface Design / Prototyping** | | | | |
| Usability & Aesthetics | | V | V | V |

## 3. RESULT AND DISCUSSION

Figure 2 above shows the research design, Table 1 shows the CT approach, and Table 2 below illustrates the results in resolving the SDLC problem. Table 2 shows the percentage of the number of participants' success rates in each SDLC result ($r$). The success rate ($sr$) is classified as "Good" (successful), "Average" (moderately successful), and "Poor" (less successful). For each SDLC result, the percentage will be calculated according to the success rate ($\forall(r_{sr})$). The percentage is obtained by calculating the ratio between the number of participants at each success rate ($n_{sr}$) and the total participants ($n$) multiplied by 100% (see Formula (1)). This is applied in both iterations, with and without CT. The differences between $\forall(r_{sr})$ when using CT and not using CT are shown in Table 3.

$$\forall(r_{sr}) = \frac{n_{sr}}{n} \times 100\% \quad (1)$$

**Table 2.** Results of resolving SDLC

| SDLC results | # results with no-CT (%) | | | # results with CT (%) | | |
|---|---|---|---|---|---|---|
| | Good | Average | Poor | Good | Average | Poor |
| Domain Model | 26 | 44 | 30 | 60 | 40 | 0 |
| Problem Identification | 0 | 100 | 0 | 33 | 50 | 17 |
| User Requirements | 25 | n/a | 75 | 46 | n/a | 54 |
| Data Model | 25 | 41 | 34 | 46 | 40 | 14 |
| Process Model | 26 | 30 | 44 | 55 | 41 | 4 |
| UI Prototyping | 15 | n/a | 85 | 80 | n/a | 20 |

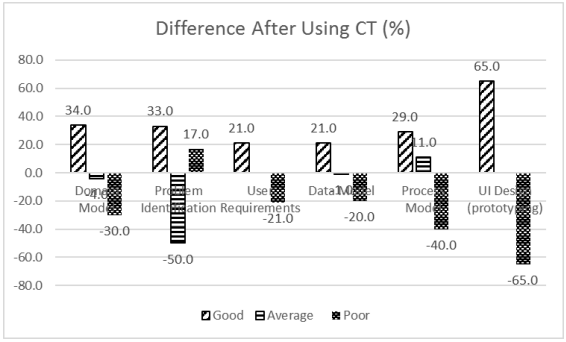


**Figure 3** Percentage of differences after using CT

**Domain Model**: This analysis resulted in a conceptual model of the As-Is system. Through this model, skills of abstraction and pattern recognition were examined using the ability to signify objects and their attributes and figuring out their relationship to characterize the As-Is system. The comprehensibility of the conceptual model was signified in three categories:

a) Domain is clear with well-defined class and attributes and their relationships to conceptualize the system (Good);

b) Domain is understandable with incomplete class and attribute (Average);

c) Domain is unclear (Poor);

In the second iteration, the participants were facilitated to figure out how objects were signified. The result of the activity revealed that the CT approach improved the capability of designers in abstraction and pattern recognition. As shown in Table 2, clear conceptual domains were constructed more than twice greater than those generated without the CT process, and no unclear domain models were produced. However, the percentage of those that produced an understandable domain model, with or without the CT approach (Average condition), was relatively the same (★). This could be concluded that even with the support of the CT approach, some of the designers were not be able to produce a clear conceptual domain.

**Problem Identification**: This was conducted to examine whether or not the participants would be able

to identify and formulate problems over a domain. This is indicated through the percentage of the following conditions:

a) identified the problem clearly (Good);

b) identified the problem but not quite clear (Average);

c) no problem can be identified (Poor).

A very interesting result of this action showed that without any intervention from the researcher, all of the participants fell in the Average condition (Table 2). The participants were able to formulate and identify the problems, although not clearly.

Meanwhile, with facilitation from the researcher, 33% of participants were able to identify and formulate problems thoroughly and clearly (Good), and 17% were not be able to formulate and identify the domain problems (Poor). The remaining 50% of participants were able to moderately formulate and identify problems. The results implied that when CT was not applied, the designers turned to assumptions to identify the problem. Because this activity dealt with reading and understanding domain area, it was concluded that the designers were unable to get an illustration of what the system was, and rather presented their assumptions on the system, which included how the system functioned and with what data the system dealt.

Nevertheless, looking at condition (b) Average, the research assumed the capabilities of the designers and participants to be the same, whether or not the CT had been applied. This would be a topic of interest for further investigation (★).

**User Behavioural Model**: This model was used in the study to examine an exercise in which whether or not participants could define user requirements as the foundation of system. This activity required the designers to shift their perspectives as users on the As-Is systems that included the lack of the systems and the way the systems were expected to function. Based on this shifted perspective, designers worked on correlating the To-Be systems with the users' needs and requirements. As illustrated in Table 2, the number of those that used the CT approach and able to define user requirements were almost double compared to those not using CT. The study noted that even through the introduction of the CT approach, the percentage of inability to identify user requirements remained quite large (more than 50%). This was slightly higner by only 21% (Table 3) of those without CT. This indicates the lack of designers' capability in capturing what was needed and required in designing new systems. Following the two previous activities (developing domain model and identifying problems), as the study showed insufficient abstraction and pattern recognition skills, designers had trouble in depicting users' needs and requirements. Due to this issue, the participants/designers were likely to attain an in-depth analysis of systems requirements based more on their perspectives rather than on users' perspectives.

**Data Model**: This model used an Entity-Relationship (E-R) Diagram and was intended to depict a new system (To-Be system) by recognizing entities and their relationship from the provided data. The analysis was conducted based on the level of E-R completeness in abstracting the system. As shown in Table 2, the comparison of those who were able to abstract the system with and without CT process is as follows:

a) Complete E-R (Good): There were almost double domain models developed by those with the CT process compared to those without the CT process;

b) Incomplete E-R (Average): The models built by those working with and without CT process were nearly the same, with only a 1% difference between using CT and not using CT (★);

c) Unclear E-R (Poor): Only 14% of the participants who worked using the CT process came out with unclear E-R compared to 34% without CT.

The above results showed that CT can be used to improve the capability of the designer participants in modelling the system, particularly from the data view point. The study compared these results to the domain model results and discovered that no designer/participants who used CT skills produced unclear domain models but continued to produce unclear data models. The assumption for this condition could be due to the characteristics of domain model that provides a more conceptual description of the system and the data model more technical, therefore it gave the impression that detailed work with data is not easy. This is proof of the challenges that designers experience when they need to describe the system using entities/attributes and their relationships while identifying entities and their relationships at the same time.

**Process Model**: The purpose of this model is to see the ability to break down a system into several sub-systems based on its functional area. The analysis emphasized what and how the system should function according to its requirements. This involves examining the skills of abstraction and decomposition as well as pattern recognition by logically identifying, analysing, generalizing, and representing data. As shown in Table 2,-the process models developed by those who worked using the CT approach compared to those who did not is represented by the completeness of the functional models in three categories: a) model with complete functions (Good), b) model with incomplete functions (Average), and c) model with unclear functions (Poor). The result showed that those that used the CT approach was two times greater than those that did not use the CT approach in concluding the model with thorough functional systems. The result showed extreme gap (almost 40%) between those that resulted with unclear

functions, wherein those that used the CT approach resulted much lower than those that did not use the CT approach. This provided evidence that CT is able to improve the designers' capabilities. However, along with the improvement, it also disclosed the evidence that there were 11% more (★) more designers/participants who used the CT approach and ended up with incomplete functional systems (Average result) than those that did not use the CT approach. This pointed out that system modelling from the functional point of view is still a system that needs to be considered. As with data modeling, this also indicates that working with system modeling in detail still needs to be taken into consideration.

**UI design**: This activity was intended to see the correlation between UI design with process and data designs. It means that every single interaction taken by users should be related to the data that was involved and how the systems respond. The result of this activity revealed that, without the CT approach, most of the UI designs (85%) were constructed based on how user interactions rather than the way the system worked, which entailed the usability aspects. In this sense, the designs referred to the designers'/participants' self-perspectives rather than to the users' needs and requirements (see the users' requirements captured). Accordingly, the interface designs did not take into account the way the system should easily and unambiguously respond because the designs were not supported with data and process models (see the data model and process model without CT process). On the contrary, with the CT approach, about 80% of the 'Good' interface designs were much easier to use, learn, and understand (about 5 times compared to non-CT). In this sense, when completing a task, users were provided with easy step-by-step instructions to improve the usability of the interface.

Based on the above results, it appears that CT skills as thinking competencies could be developed as a complement to one's natural abilities by exercising the CT processes. Comparing both iterations, the results of analysis and design processes in iterations 1 lead to insufficient system models with lack of conceptualization, pattern recognition, problem formulation, and common-sense in thinking process. As shown in Table 2, problem identification and conceptual as well as detail technical models within which abstraction, decomposition, and pattern recognition skills that had been acquired could not be established properly across the analysis and design processes. This also occurred in the process of establishing an algorithm to discover the UI design solution. Meanwhile in iteration 2, it was shown that by and large CT framework could produce quite significant results. This is in terms of improving the capabilities of the designers in signifying systems during the analysis phase throughout formulating problems as well as modelling the systems and turning out with design solutions during design phases.

Reviewing the above explanation, abstraction and pattern recognition in design thinking seem to be the most crucial processes in the CT approach as both processes had been applied in analysis as well as design phases (see Table 1). When both of these competencies were weak, by which domain problem could not be depicted and system models (both conceptually and technically) could not be be represented, then the design process would be at risk. As shown in Table 2, this condition occurred for those designer participants who worked without the CT framework. They were not able to comprehend domain problems clearly by conceptualizing the system domain and formulating system problems based on the users' perspective. Besides abstraction and pattern recognition, decomposition skill also plays an important role, particularly when one needs to break down into more specific domain problems to see the case from a more in-depth point of view. As seen in Table 2, the constructed data and process models were also insubstantial. This occurred as the designers were insufficiently not able to delineate the systems in detail. Consequently, the UI designs that had been developed were not as they had been expected. On the contrary, the UI designs developed by those who worked using the CT as interventions were somewhat relevant in terms of systems requirements that answered users' needs and requirements. This showed that the UI designs were presented with a clear conception of the system which will be easier for the users to operate the system. Infusing this skill into the analysis and design phases, if given the entire system, could increase the capability of the designers, particularly in problem-solving and system modelling.

Nonetheless, albeit with the intervention of the researcher, some of the results remained intriguing. These were mostly observed in the Average condition of modeling the system (these were marked with (★) symbol). The (★) symbol indicated that the results were not significantly different between those that had and had not used the CT. In this sense, the provided guideline or directions would not utterly improve the abilities for abstraction, pattern recognition, and logical thinking process, since there had been participants who were still able to complete with the trivial designs. Furthermore, concerning decomposition skills, the study also revealed that a considerable number of designers failed to deal with detailed system modelling. This implied the lack of designers' capability in reflecting the complexity of the system. These cases may have been a consequence of a limited number of iterations that had been taken, and as such putting the CT approach into practice would not have been optimal. Accordingly, further in-depth analysis of the Average (★) condition is required as it presented doubtful cases.

Hence, it was brought to light that the effectiveness of UI design depends strongly on the conception of the system domain Understanding the system from the users' perspectives also become crucial and extremely

essential. This is true as the designers need to figure out and to match both user needs and the way the systems respond to the user needs in order for it to present as little resistance as possible to a flow of logical interaction between systems and users.

## 4. CONCLUSION

Based on the results of the study, the CT approach can somehow be put into practice throughout the SDLC phase, particularly in the analysis and design phases. This study showed that, by not using the CT approach, a clear domain could not be established and domain problems had been inadequately understood (referring to analysing As-Is system). This affects the lack of deriving important requirements, and subsequently, a clear model of the design solution (To-Be system) could not be thoroughly established. Based on the study, it appears that the CT processes that had been somewhat difficult to be carried out were abstraction and pattern recognition. This was indicated by the insufficient conception by the participants to model the As-Is and To-Be systems. By infusing CT skills into the analysis and design phases, it appeared that CT could be used to some extent to support designers in improving their design process, mostly in delivering a good solution to the problem that has been identified from a clear conception of the system.

However, this study also showed that some 'midpoint' outcomes (Average) produced by those who practiced with the CT approach turned up with nearly the same percentage by those who worked without the CT approach. This situation is recommended for further study to prove that the extent of CT affects the design process.

Nonetheless, in general, the CT approach can be taken into account to underpin the design process. Particularly referring to abstraction and pattern recognition that had entailed all stages of analysis and design processes, the practice of these two CT processes must be seriously considered. Otherwise, the design would be at risk, particularly concerning the system usability.

## ACKNOWLEDGMENTS

This research was supported by the Overseas Conference Funding Program from the Directorate General of Research Strengthening and Development, Ministry of Research, Technology and Higher Education (Kemenristekdikti), the Republic of Indonesia [grant number B/460/E5.3/ KI.03.01/2019].

## REFERENCES

[1] Interaction Design Foundation (IDF), "Design Principles," 2016. [Online]. Available: https://www.interaction-design.org/literature/topics/design-principles. [Accessed 2018].

[2] D. Budgen, "Software Design: An Introduction," in *Software Engineering, Vol 1: The Development Process, 2nd Edition*, Washington, Brussels, Tokyo, The Institute of Electrical and Electronics Engineers, Inc., 2002, pp. 197-208.

[3] I. S. f. T . i. E. (ISTE), "Operational Definition of Computational Thinking for K–12 Education," 2011. [Online]. Available: www.iste.org/docs/ct.../computational-thinking-operational-definition-flyer.pdf. [Accessed 2018].

[4] J6-Design-Australia, "The Principles of Design," 2015. [Online]. Available: http://www.j6design.com.au/6-principles-of-design. [Accessed 2018].

[5] U. D. o. H. &. H. S. (HHS), "Usability Guidelines," 2010. [Online]. Available: http://webstandards.hhs.gov/guidelines/#. [Accessed 2018].

[6] Interaction Design Foundation (IDF), "The Basic of User Experience (UX) Design," Interaction-Desgn.Org, 2002.

[7] BBC, "Introduction to computational thinking," 2018. [Online]. Available: https://www.bbc.com/bitesize/guides/zp92mp3/revision/1. [Accessed Oct 2018].

[8] J. M. Wing, "Computational Thinking—What and Why?," *The Link News Letter, the School of Computer Science, Carnegie Mellon University, Issue 6.0 / Spring 2011*, pp. 20-23, 2011.

[9] J. M. Wing, ""Computational Thinking" – Viewpoint.," *COMMUNICATIONS OF THE ACM*, Vols. Vol. 49,, no. No. 3,, pp. pg. 33-35., 2006.

[10] T. Semiawan, "User interface design analysis pertaining to computational thinking framework," in *IEEA '19 Proceedings of the 8th International Conference on Informatics, Environment, Energy, and Applications*, Osaka, Japan, 2019.

[11] X. Wang, C. Schneider and J. S. Valacich, "Enhancing creativity in group collaboration: How performance targets and feedback shape perceptions and idea generation performance," *Computers in Human Behavior*, vol. 42, pp. 187-195, 2015.

[12] M. Wang and Y. f. Wang, "A Study on Computer Teaching Based on Computational Thinking," *International Journal of Emerging Technologies in Learning (iJET)*, vol. 11, no. 12, 2016.

[13] P. Curzon, M. Dorling, T. Ng, C. Selby and J. Woollard, "Developing computational thinking in the classroom: a framework," Creative Commons

Attribution-NonCommercial - Computing At School, United Kingdom, 2014.

[14] M. Romero, A. Lepage and B. Lille, "Computational thinking development," *International Journal of Educational Technology in Higher Education,* vol. 14, no. 42, 2017.

[15] K. Brennan, C. Balch and M. Chung, CREATIVE COMPUTING, Harvard: Harvard Graduate School of Education, 2015.

[16] A. Burns, "Action Research," in *The Cambridge Guide to Research in Language Teaching and Learning*, Cambridge, Cambridge University Press., Editors: J. D. Brown & C. Coombe, 2015, pp. 99-104.