Proceedings of the Evaluation and Assessment on Software Engineering April 2019, Pages 90–99
https://doi.org/10.1145/3319008.3319017

# Preliminary Evaluation of an Ontology-Based Contextualized Learning System for Software Security

Shao-Fang Wen and Basel Katt

Norwegian University of Science and Technology

Gjøvik, Norway

{shao-fang.wen, basel.katt} @ntnu.no

### ABSTRACT

Learning software security is one of the most challenging tasks in the information technology sector due to the vast amount of security knowledge and the difficulties in understanding its practical applications. The traditional teaching and learning materials, which are usually organized topically and security-centric, have fewer linkages with learners' experience and prior knowledge that they bring to the learning sessions. Learners often do not associate vulnerabilities or coding practices with programs similar to what they were writing in their previous time so that their motivation for learning is hard to be touched by the conventional methods. The purpose of this paper is to demonstrate a contextualized learning system for software security based on an ontological technology. This system facilitates the contextual learning process by providing contextualized access to security knowledge via real software application scenarios, in which learners can explore and relate the security knowledge to the context they are already familiar with. The developed prototype was evaluated by a controlled quasi-experiment in a university. The experiment results show that the prototyped learning system not only yields significant knowledge gain compared to the conventional learning approach but also gains better learning satisfaction of students.

## 1. INTRODUCTION

Software security has been a subject of plethora studies for at least 40 years, and a steady stream of innovations has improved software engineers' ability to secure software development and to protect applications. Improving software security requires many different approaches. One way is to give software engineers or learners the knowledge and skills to resist attacks and handle errors appropriately [5]. To emphasize security, a relatively large number of best practices and vulnerability information have been published by security committees in publications or on the internet. To this extend, the huge amount of information has resulted in a form of information overload to learners. Moreover, the domain of software security is quite context-specific and can be applied in diverse ways [43]. As a result, learning software security becomes a complex and difficult task because learners must not only deal with a vast amount of knowledge about a variety of concepts and methods but also need to demonstrate the applicability of the knowledge through experience in order to understand their practical use.

In traditional software security teaching, little attention is given to what a real-world situation really means to learners, and there is not much content addressing the connection between the security concepts and learner' prior knowledge. In conventional security learning materials, the knowledge content is commonly security-centric and organized topically, which distinguishes two fundamental segments: the white-hat approach, where the main emphasis on security principles and anti-attack mechanisms, and the black-hat, which teaches how to break software and how malicious hackers write exploits. These learning materials are often described in the form of a reference manual or a guide to particular security subjects. The topical knowledge organization is useful for rote memorization of a specific security subject or for information reference later; however, it is difficult for learners to understand the rationale of the topics, and correlate those topics with real software scenarios. Learners usually finish reading such materials with little understanding of the context in which the security knowledge should be applied, or with the feeling that security domain is so extensive and software security is so difficult to achieve that they simply cast it aside.

We argue that the way leaners process security information and their motivation for learning is not touched by the conventional methods. Research indicates that learning is most efficient when it is linked with experience and prior knowledge that students bring to a given learning situation [13, 39]. However, novice learners do not always make connections between new information and prior knowledge or everyday experiences in ways that are productive for learning [37]. In the context of software security learning, learners interpret security knowledge they gain with a range of strongly held personal programming experience. They often do not associate vulnerabilities with programs similar to what they were writing in their previous time. As the suggestion given in the research of engineering education [21], establishing the relevance of learning materials before going into the details can provide the concrete experience that starts the learning process. In order to regulate learning about software security effectively, security knowledge should be contextualized in a meaningful scenario where they can learn security principles and processes with a real-world situation.

Our primary objective is to create conditions for more effective learning for software security that can that can motivate learners and stimulate their interest. This paper is part of an investigation into contextualized learning in the domain of software security, supported by an empirical evaluation. We propose a learning system, which facilitates the contextual learning process by providing contextualized access to security knowledge through real software application scenarios. This learning system is a place where learners can explore and relate the security knowledge to the context they are already familiar with. To develop this kind of learning system, the security knowledge should be modeled and managed in a manner where the software security knowledge can be retrieved taking the context of the application in hand into consideration. Ontologies make it possible to give this kind of purpose since it facilitates capture and construction of domain knowledge and enables representation of skeletal knowledge to facilitate the integration of knowledge bases irrespective of the heterogeneity of knowledge sources [27]. In the paper, an ontology-based web application prototype is presented, which was evaluated by a preliminary experiment in the setting of a university environment. This paper also presents the experimental design and results.

The rest of this paper is organized as follows: In section 2, we introduce the theoretical background of this study. Section 3 reviews the related works on ontology approaches in the software security domain. In section 4, we describe the proposed approach in modeling and presenting software security knowledge. Section 5 presents the detailed design of an underlying ontology of the learning application. Section 6 describes the developed prototype while section 7 summarizes the experimental evaluation of the prototype. Lastly, the discussion and conclusion are presented in section 8.

## 2. THEORETICAL BACKGROUND

The theoretical background of this research is drawn from the field of context-based knowledge and contextualized learning. According to Anind K. Dey [16], context is ‘‘A set of information used to characterize a situation of an entity”. Nonaka [45] indicates that knowledge reflects a particular stance, perspective, or intention in accordance with the characteristics of a specific context, which is different from information. According to Brézillon [6, 7], knowledge comes from a variety of context and it cannot be accurately understood without context. Without proper contextual information, knowledge can be isolated from other relevant knowledge resulting in limited or distorted understanding [23]. Researchers of psychology and education indicate when knowledge is learned in a context similar to that in which the skills will actually be needed, the application of learning to the new context may be more likely [16, 18, 49]. Predmore [50] shows that learning about knowledge content within real-world experience is important because “once [students] can see the real-world relevance of what they’re learning, they become interested and motivated”. Since context can give guidance about when, where and why a piece of knowledge is used, considering the context in knowledge use is very necessary to enhance the applicability of knowledge [5].

Contextualized teaching and learning builds upon a similar concept of putting learning activities into perspective to achieve the best teaching and learning outcomes. Researchers Berns and Erickson define contextualized learning as a practice that endeavors to link theoretical constructs that are taught during learning, to practical, real-world context [4]. The underlying theme behind the contextual learning activities is simple. It recognizes that by embedding instructions in contexts that adult learners are familiar with, learners more readily understand and assimilate those instructions. Contextualized instruction in general, starts with presenting a context from which the concepts are developed on a need-to-know basis [3]. This requires teachers to teach in a more constructivist way, i.e. to position the concepts of the learning subject in contexts recognizable to students and to stimulate active learning of the students [47]. The contextualization of the learning on demand can not only be seen from the point of view of an actual problem or learning situation but also in a longer lasting process of learning activities that are integrated [58].

In computer science education, there is also a broad agreement that teaching units should start from a “real-world” context or phenomenon, aiming to create connections to prior knowledge, to increase the relevance of the material to students or to show application situations of the intended knowledge, thereby increasing motivation[11, 17, 29, 30]. These contrast with more traditional approaches that cover abstract ideas first, before looking at practical applications. Likewise, in software engineering, studying from a context and then abstracting the knowledge gained to be able to use it in a new context is a common way of learning programming that has been observed extensively in both new and experienced programmers [2, 35]. In order to capture and use security knowledge appropriately, it is necessary to first specify which context information is to be handled, and then represent this in a format that is understandable and acceptable to the individuals. Thus, a context for a software security topic includes the circumstances in which its technical content exists. Therefore, to talk about software security in context is to say that knowledge would not only include the basic principles and processes of software security but would consider how security knowledge is used in one or more particular domains or application areas.

# 3. RELATED WORK

In this section, we describe research works related to this study from the viewpoint of knowledge modeling support for software security based on ontology. According to Gruber [26], an ontology is “an explicit and formal specification of a conceptualization”, that is, a formal description of the relevant concepts and relationships in an area of interest, simplifying and abstracting the view of the world for some purpose [60]. There have been a number of papers published in the area of ontology modeling and applying semantic technologies to software security. Some effort focused on building security ontology to model the security requirements. Salini and Kanmani [54] present an ontology of security requirements for web applications, including concepts of asset, vulnerabilities, threats, and stakeholders. Their work aims at enabling the reuse of knowledge about security requirements in the development of different web applications. Buch and Wirsing [8] present the SecWAO ontology with a focus on a secure web application, which aims to support web developers when specifying security requirements or making design decisions. It distinguishes concepts (classes) between methods, notations, tools, categories, assets, security properties, vulnerabilities, and threats.

Some research works present their ontology to support security design and risk assessment. Gyrard et al. [31] present the STACK ontology (Security Toolbox: Attacks & Countermeasures) to aid developers in the design of secure applications. STACK defines security concepts such as attacks, countermeasures, security properties, and their relationships. Countermeasures can be cryptographic concepts (encryption algorithm, key management, digital signature, and hash function), security tools, or security protocols. Kang and

Liang [33] present a security ontology with the Model Driven Architecture (MDA) approach for the use in the software development process. The proposed ontology shows that the proposed security ontology can be used in modeling and designing security issues and concepts in each phase of the development process with MDA. Marques and Ralha [40] propose an ontology, which is related to the risk management aspect of web-based system development. The model is mainly employed in the design phase of the system development.

Finally, there are some papers focusing on using an ontology to model vulnerabilities and security attacks. Guo and Wang [28] present an ontology-based approach to model security vulnerabilities listed in Common Vulnerabilities and Exposures (CVE). The authors captured important concepts for describing vulnerabilities in the context of software security, providing machine-understandable CVE vulnerability knowledge and reusable security vulnerabilities interoperability. Khairkar et al. [34] present an ontology to detect attacks on web systems. The authors use semantic web concepts and ontologies to analyze security logs to identify potential security issues. This work aims to extract semantic relationships between attacks and intrusions in an Intrusion Detection System (IDS). Razzaq et al. [51] propose an ontology of attacks and an ontology of communication protocols, which provide a construct to improve the detection capability of application-level attacks in web application security. The authors employ the use of semantics in application layer security contrary to tradition signature-based approaches.

## 4. DESIGN APPROACH

To facilitate contextual learning about software security, we propose to model and present software security knowledge with three strategies: (1) Using a meaningful application scenario; (2) Stimulating learners' mental models for software security learning; and (3) Moving from concrete to abstract security knowledge. We describe in details these strategies in the following sections.

### 4.1 A Meaningful Scenario

Contextualized learning often takes the form of real-world examples or problems that are meaningful to the learners personally [52]. Creating the relevance of the learning knowledge before going into the details could provide a stronger foundation for the learning process. Therefore, to begin the process of learning, a meaningful situation for learners must first be established. In our study, the learning situations are created through the use of contextual scenarios in the application context, which utilize some form of anchoring situation events [10] to engage learners with security concepts that are addressed in the software problem or situation. Contextual scenarios refer to different manifestations within a context [19]. We choose a scenario-based approach because scenarios can be easily adapted to the situation of the represented applications and can be easily integrated with the security domain knowledge.

An anchoring event (i.e., the scenario in our study), enabling learners to visualize how the knowledge substance relates to their prior experience [10], could be revisited repeatedly during the learning sessions. For instance, regarding the application functionality of "Generating HTML pages" in web application context there includes a set of scenarios, such as generating static or dynamic pages, and using external data from HTTP requests or data stores. Those scenarios can serve as anchoring events to evoke the learners' memories of programming and draw attention to software events and conditions. Research has shown that using anchoring events in learning promotes memory recall and the subsequent transfer of information to a new setting [10], meanwhile, helps render abstract ideas more concretely and thus provides a cognitive mooring around which newly learned ideas can be linked with learners' prior understandings [9, 57]. The use of anchor evens in our study aims to echo learners' real-world experiences to contextualize security knowledge to help learners apply their emerging understandings about software security to the real software cases, thus helping them see value in their learning sessions.

## 4.2 Mental Models

In order to help learners make sense of complex security knowledge and create a strong and lasting bond among security concepts while they are engaged through various anchoring events, our strategy is to elicit learners' mental models for the navigation of security knowledge. Kenneth Craik [14] suggested that the human mind builds and constructs "small-scale models" to anticipate events. Such mental models allow learners to gain insight regarding their world by building a work scheme [22], which makes it easier for them to access the information needed to understand the knowledge domain, make predictions, and decide upon action to take [53]. This can result in successful learning by engaging students, fostering their concentration, and assisting them in organizing systemic information [55].

Mental models combine a schema or a knowledge structure with a process for manipulating the information in the memory [44], where the knowledge structure interrelates a collection of facts or concepts about a particular topic [1]. To guides learners in approaching personal mental models for learning security knowledge in the domain of software security, there should be a built-in knowledge structure for software security learning. Generally, our intention was to guide learners in answering three questions while dealing with each anchoring event:

- What are the possible attacks?
- Why does it encounter attacks?
- How can these attacks be prevented?

The knowledge structure serves as the basis for both knowledge retention and retrieval, as well as transfer. As learners answer the what–why–how questions, the relationships between the security concepts are revealed in their midst, and thus, their mental model expands.

## 4.3 Concrete-to-Abstract Presentation

To help learners gain a more flexible understanding of the study concept in a range of situations with varying levels of abstraction, we organize security knowledge by blending abstract and concrete perspectives; presenting it with a sequence from concrete to abstract. In our study, abstract knowledge refers to the conceptual security domain knowledge while the concrete knowledge relates to the contextualized scenario-specific security knowledge. Research has shown that presenting knowledge in both concrete and abstract terms are far more powerful than presenting either one in isolation [48]. Lave and Wenger [38] also argued that abstract and generalized knowledge gains its power through the expert's ability to apply it in specific situations. The used concrete-to-abstract approach in knowledge presentation differs from the traditional, where the concepts are of foremost importance and are usually explained first before concrete examples and applications are discussed. Consequently, learners may struggle to finish reading them due to a learning style mismatch. Several studies [20, 41, 42] have shown that the majority of engineering students are sensor-type learners, who like facts, data, and observable phenomena as opposed to theoretical abstractions. Deductive reasoning is facilitated when the domain is familiar and concrete rather than abstract [61]. In such concrete-to-abstract knowledge presentation, learners discover meaningful relationships between practical functions and abstract knowledge in the context of real applications. A method known as concreteness fading [24] has the advantage of initially presenting concepts in a concrete fashion and then, over time, augmenting that initial presentation with progressively more abstract representations of the concepts. Abstract understanding is most effectively achieved through experience with perceptually rich, concrete representations [25], while concrete materials make concepts real and therefore easily internalized [32]. As long as the concrete knowledge and the underlying abstract explanation are understood by learners, learning transfers from one context to another will be more effective.

Proceedings of the Evaluation and Assessment on Software Engineering April 2019, Pages 90–99 https://doi.org/10.1145/3319008.3319017

# 5. THE UNDERLYING ONTOLOGY-BASED KNOWLEDGE MODEL

One of the central ideas embedded within the learning framework is to develop a kernel ontology-based security knowledge model. With this model, the learning application can handle contextualized security knowledge with multiple scenarios in different application-specific contexts and integrates security concepts of security domain knowledge. Figure 1 depicts the component of the ontology.

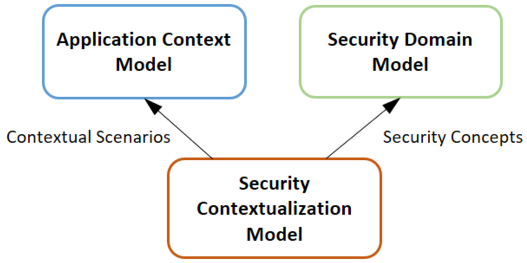


Figure 1. The components of the ontology-based context model

## 5.1 Application Context Modeling

The context model represents a definition of what context is in a specific domain. In our ontology, the context for software security knowledge is supported by the creation of scenarios in different application contexts. The scenario presents a snapshot of possible features and corresponding code fragments in the specific functionality that are included in the *Instruction* class. It also draws on situated security knowledge, that is, understandings particular to the application context in which they generate. Figure 2 represents the application context model used in the ontology. In the context modeling, in additional to scenarios, we focus on characteristics that are highly relevant for retrieval within a software application, concerning three perspectives:

- The functional area (and the corresponding functionalities) that the application associated with.
- The application category that scenario/functionality belongs to,
- The platforms that the scenario functionality used, and

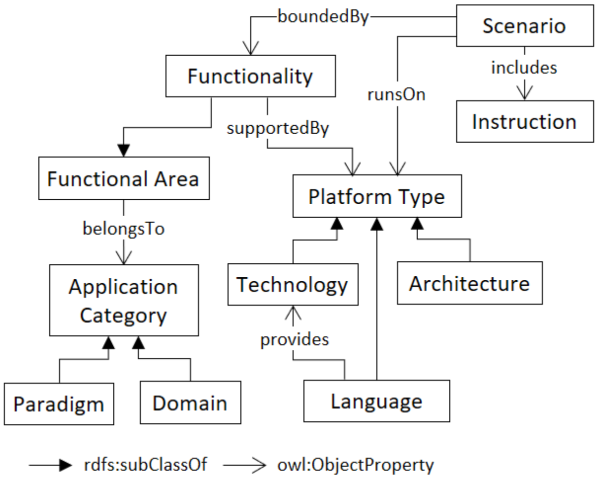


Figure 2. Application Context Model

*Application category*: It is a set of characteristics to categorize software applications, which include two sub-classes: paradigms (e.g., web, mobile, and desktop applications etc.) and the domains (e.g., banking, health, and logistics applications etc.).

*Platform type*: This superclass specifies programming languages, technologies, and architectures that are used to create the software application. Technology can be provided by a certain programming language. For example, Silverlight is the technology that has been implemented in C# language, while J2EE is the subset of Java technologies. Architectures refer to the fundamental system structure to operate the application, such as the MySQL database management system and Android operating system.

*Functional area*: It is a group of application functionalities, which represents an aspect of software applications that can be performed by users or other systems in a particular application category. For example, outputting HTML is a functional area in web applications paradigm, in which generating HTML dynamically using user-supplied data is one the functionalities. A functionality is supported and run on some combinations of platform types.

## 5.2 Security Domain Modeling

The security domain model describes the knowledge that is an object of teaching through a set of concepts (topics to be taught). In the security domain model, we aim to design a security knowledge structure (schema) that is easier to store in the learners' memory for learning. For the purpose, the schema should be simplified and kept to the point for reducing the content load. We, therefore, identify three security concepts that are most widely used throughout the security domain. Ultimately, three classes were incorporated into the security domain model: *Security Attack*, *Security Weakness*, and *Security Practice*. The definitions of the three security concepts are given in the following

*Security Attack*: It represents actions taken against the software application with the intention of doing harm. Examples are SQL injection, Cross-Site Scripting, etc. Security attacks exploit security weakness existed in software applications.

*Security Practice*: It represents methods, procedures or techniques to prevent security weakness.

*Security Weakness:* It represents bug, flaws, vulnerabilities and other errors exist in the software applications.

From a security domain point of view, we only want to indicate which principles or abstract ideas are needed, not their practical implementation. Therefore, we describe security knowledge in this model at a level of abstraction. The instances of these classes specify only the fundamental characteristics of the security concepts, not specific software application aspects. The main advantage of this design is to share a common understanding of the conceptual security knowledge among different security contexts. Furthermore, we adopt an abstract class *Security Domain* as a superclass for all security concepts. In the security domain model, we apply separation of concerns so that only very general descriptions remain as attributes in the class *Security Domain*. Additionally, for convenience, we allow grouping domain knowledge in categories, which themselves can belong to security concepts. Figure 3 illustrates the security concepts and their relationships in the security domain model.

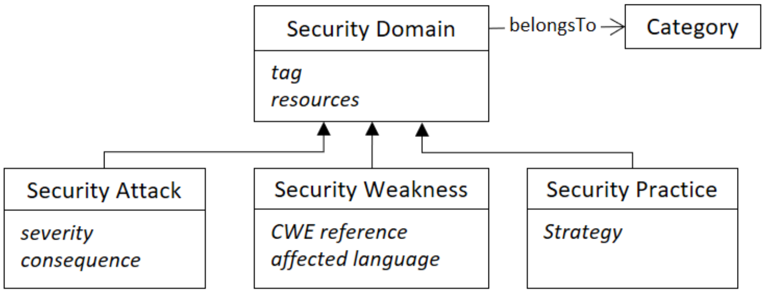


Figure 3. Security domain Model

## 5.3 Security Contextualization Modeling

The term contextualization is used here to describe the process of drawing specific connections between security domain knowledge being taught and an application context in which the domain knowledge can be relevantly applied or illustrated. To this extent, the security contextualization modeling manages security knowledge in the context of specific scenarios and brings together the conceptual knowledge that is described in the security domain model. The including security concepts are aligned with those defined in the security domain model, which are *Security Attack*, *Security Weakness*, and *Security Practice*. However, in order to clearly state the purposes and distinguish them from the security domain model, we use different classes, namely *Concrete Security Attack*, *Concrete Security Weakness*, and *Concrete Security Practice*. Figure 4 illustrates the security contextualization modeling. The abstract class *Contextualized Knowledge* is used from which these three classes inherit common attributes such as tags or external resources. Once the domain knowledge model is defined, each security concept in the contextualized model is able to be connected to the corresponding classes in the security domain model.

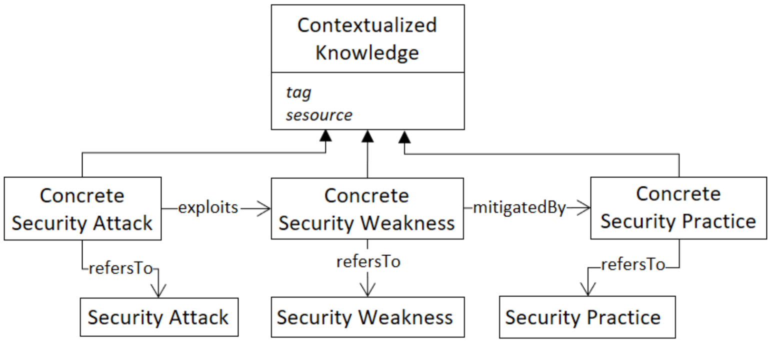


Figure 4. Security contextualization model

Figure 5 shows the completed ontology-based knowledge model including the interrelationships of the components.

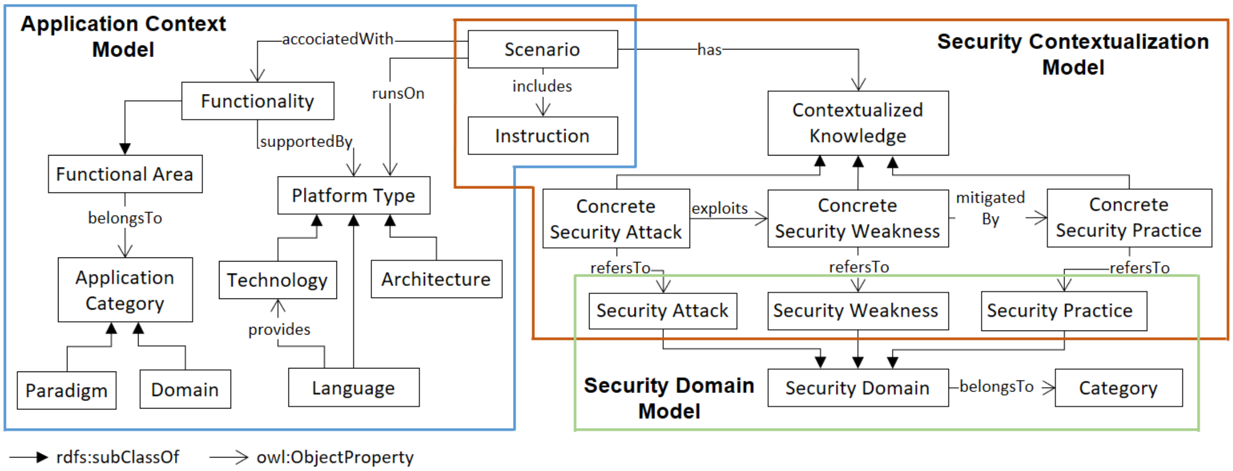


Figure 5. The ontology-based security knowledge model

# 6. THE DEVELOPED PROTOTYPE

We have developed a proof-of-concept prototype to demonstrate the proposed approach. The high-level system architecture diagram is presented in figure 6. The front-end was designed as web-based user interface and through it, learners can access the knowledge content. The backend was implemented in Java and access to the ontology repository was provided through the Jena API[1], a Java framework for building semantic web applications. Jena provides extensive Java libraries for helping developers develop code that handles RDF, OWL, and SPARQL in line with published W3C recommendations[2].

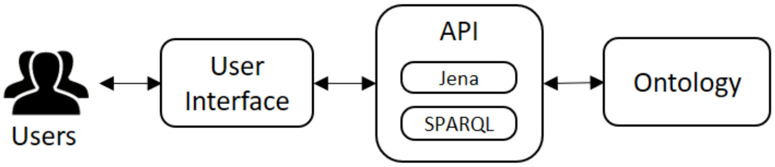
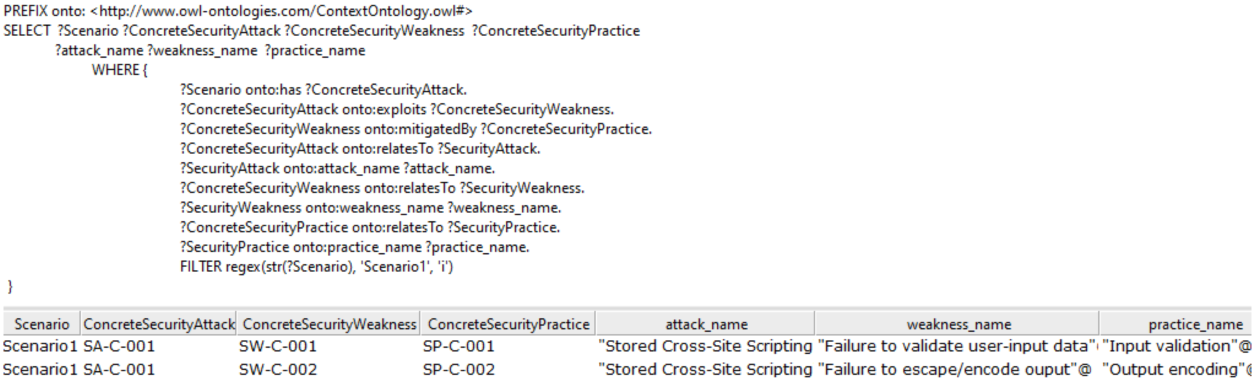


Figure 6. High-level system architecture diagram

## 6.1 Construction of the Ontology

To construct the ontology, we used Protégé and OWL[3] Editor because of its simplicity and popularity [59]. When searching the ontology, we use SPARQL protocol to extract information from the RDF graph. An example of SPARQL and the executed result is presented in Figure 7. The objective of this query is to return the instances of contextualized security knowledge of a specific scenario, and the short names of related abstract domain knowledge.



```
PREFIX onto: <http://www.owl-ontologies.com/ContextOntology.owl#>
SELECT  ?Scenario ?ConcreteSecurityAttack ?ConcreteSecurityWeakness  ?ConcreteSecurityPractice
          ?attack_name ?weakness_name  ?practice_name
                WHERE {
                                ?Scenario onto:has ?ConcreteSecurityAttack.
                                ?ConcreteSecurityAttack onto:exploits ?ConcreteSecurityWeakness.
                                ?ConcreteSecurityWeakness onto:mitigatedBy ?ConcreteSecurityPractice.
                                ?ConcreteSecurityAttack onto:relatesTo ?SecurityAttack.
                                ?SecurityAttack onto:attack_name ?attack_name.
                                ?ConcreteSecurityWeakness onto:relatesTo ?SecurityWeakness.
                                ?SecurityWeakness onto:weakness_name ?weakness_name.
                                ?ConcreteSecurityPractice onto:relatesTo ?SecurityPractice.
                                ?SecurityPractice onto:practice_name ?practice_name.
                                FILTER regex(str(?Scenario), 'Scenario1', 'i')
 }
```

| Scenario | ConcreteSecurityAttack | ConcreteSecurityWeakness | ConcreteSecurityPractice | attack_name | weakness_name | practice_name |
|---|---|---|---|---|---|---|
| Scenario1 | SA-C-001 | SW-C-001 | SP-C-001 | "Stored Cross-Site Scripting | "Failure to validate user-input data" | "Input validation"@ |
| Scenario1 | SA-C-001 | SW-C-002 | SP-C-002 | "Stored Cross-Site Scripting | "Failure to escape/encode ouput"@ | "Output encoding" |

Figure 7. An example of SPARQL and the executed result

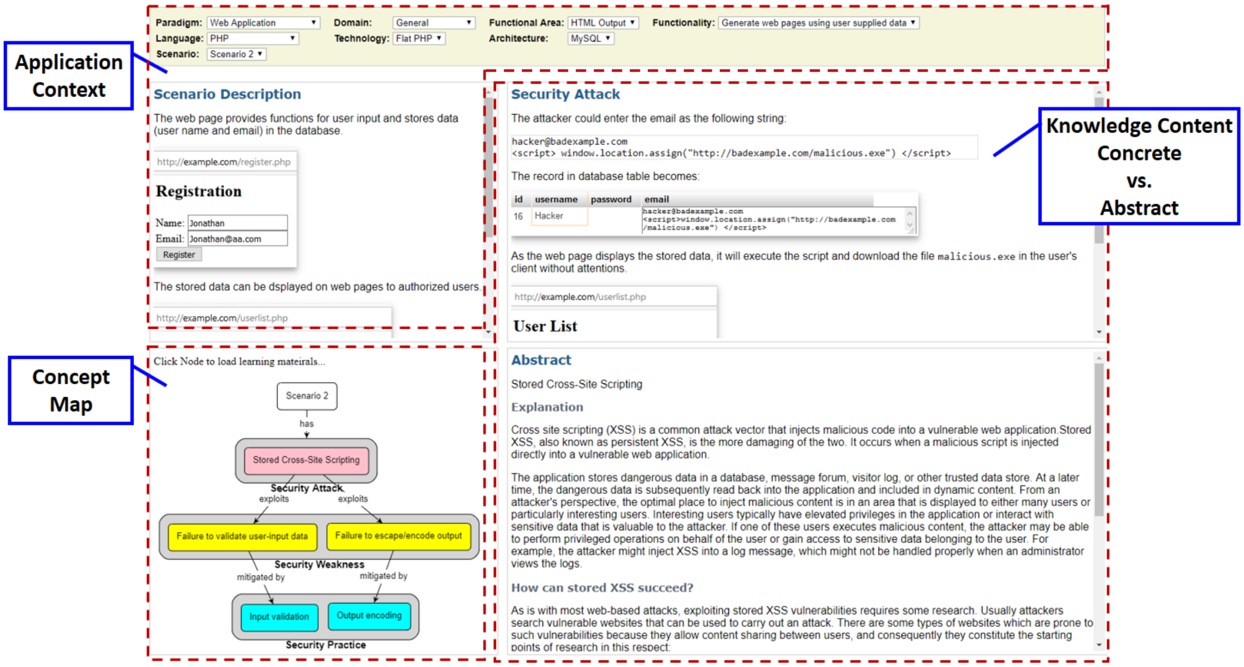
## 6.2 The process of Contextual Learning

In the prototype, we set up the learning environment in a web application paradigm, using a pure PHP technology and MySQL as the architectural database. To prepare for learning materials based this specified context, the author developed a preliminary set of functionalities to operate a real web-based application, including a login module, data input/output features, data processing, and database access. Three critical application scenarios were created for each function within the scope of the application context. The user interface of the prototyped system is presented in figure 8.

[1] https://jena.apache.org/

[2] https://www.w3.org/2001/sw/

[3] Web Ontology Language (OWL), a markup language based on Resource Description Framework/Extensible Markup Language (RDF/XML).


Application Context
Knowledge Content Concrete vs. Abstract
Concept Map
Register
Click Node to load learning mateirals...
Scenario 2
has
Stored Cross-Site Scripting
Security Attack
exploits
exploits
Failure to validate user-input data
Failure to escape/encode output
Security Weakness
mitigated by
mitigated by
Input validation
Output encoding
Security Practice

Paradigm: Web Application ▾ Domain: General ▾ Functional Area: HTML Output ▾ Functionality: Generate web pages using user supplied data ▾
Language: PHP ▾ Technology: Flat PHP ▾ Architecture: MySQL ▾
Scenario: Scenario 2 ▾

**Scenario Description**

The web page provides functions for user input and stores data (user name and email) in the database.

http://example.com/register.php

**Registration**

Name: Jonathan
Email: Jonathan@aa.com

The stored data can be dsplayed on web pages to authorized users.

http://example.com/userlist.php

**Security Attack**

The attacker could enter the email as the following string:

```
hacker@badexample.com
<script> window.location.assign("http://badexample.com/malicious.exe") </script>
```

The record in database table becomes:

| id | username | password | email |
|---|---|---|---|
| 16 | Hacker | | hacker@badexample.com <script>window.location.assign("http://badexample.com /malicious.exe") </script> |

As the web page displays the stored data, it will execute the script and download the file `malicious.exe` in the user's client without attentions.

http://example.com/userlist.php

**User List**

**Abstract**

Stored Cross-Site Scripting

**Explanation**

Cross site scripting (XSS) is a common attack vector that injects malicious code into a vulnerable web application.Stored XSS, also known as persistent XSS, is the more damaging of the two. It occurs when a malicious script is injected directly into a vulnerable web application.

The application stores dangerous data in a database, message forum, visitor log, or other trusted data store. At a later time, the dangerous data is subsequently read back into the application and included in dynamic content. From an attacker's perspective, the optimal place to inject malicious content is in an area that is displayed to either many users or particularly interesting users. Interesting users typically have elevated privileges in the application or interact with sensitive data that is valuable to the attacker. If one of these users executes malicious content, the attacker may be able to perform privileged operations on behalf of the user or gain access to sensitive data belonging to the user. For example, the attacker might inject XSS into a log message, which might not be handled properly when an administrator views the logs.

**How can stored XSS succeed?**

As is with most web-based attacks, exploiting stored XSS vulnerabilities requires some research. Usually attackers search vulnerable websites that can be used to carry out an attack. There are some types of websites which are prone to such vulnerabilities because they allow content sharing between users, and consequently they constitute the starting points of research in this respect:

Figure 8. The user interface of the developed prototype

In the learning application, the learning process begins with the concrete in a context familiar to learners and then gradually leads to an understanding of the abstract. First of all, a meaningful situation for learners must first be established. The access to learning contents in the learning application mainly happens scenario-oriented. We use the scenario as the starting point for learning security concepts on a need-to-know basis while presenting the modeled security knowledge. Based on the desired knowledge the learner selects relevant criteria from the application-context menu to scope the learning scenario. The instructional part of the scenario is made up of practical demonstrations of the pre-described application functionality and the code fragments behind it that bridge the corresponding security knowledge. As described previously, the selected scenario served as an anchoring event that can be view throughout the learning session to anchor learning in the learners' personal experience.

To guide learners navigating through the contextualized knowledge efficiently, it is necessary to illustrate the relationship between the security concepts. On the one hand, it must be transparent for learner about, which causes and effects relevant to the learning content he (or she) is studying. On the other hand, this is essential for learners in order to integrate the semantical impact of the knowledge structure into the mental models for efficient learning. For the purpose, we outline the learning contents in a graphical *Concept Map*, which shows in the left-corner part of the screen. Concept Map is a visual representation of different concepts and their relationships. With the use of concept mapping, the learning arena can be virtualized in a learner's mind [56]. The design of our ontology is able to provide the basis for the development of the concept map of the relationship between these concepts. As learners navigate through the concept map and figure out the answers for the what-why-how questions, their mental model also expands consequently.

While a node is clicked on the concept map, the relevant knowledge content is displayed in the right half of the screen, where the upper part is the contextualized knowledge and the lower part is an abstract explanation, following the concrete-to-abstract presentation strategy. By concrete representations, we include perceptually detailed and rich materials, such as demonstrating security attacks with different exploits, identifying mistakes in the source code, and showing the secure coding practices to fix the mistakes. After experiencing the facts, learners then move on to abstract knowledge, where the conceptual explanation is presented. Therefore, dynamic, e.g., situational application scenario is integrated together with the security domain knowledge.

## 7. PROTOTYPE EVALUATION

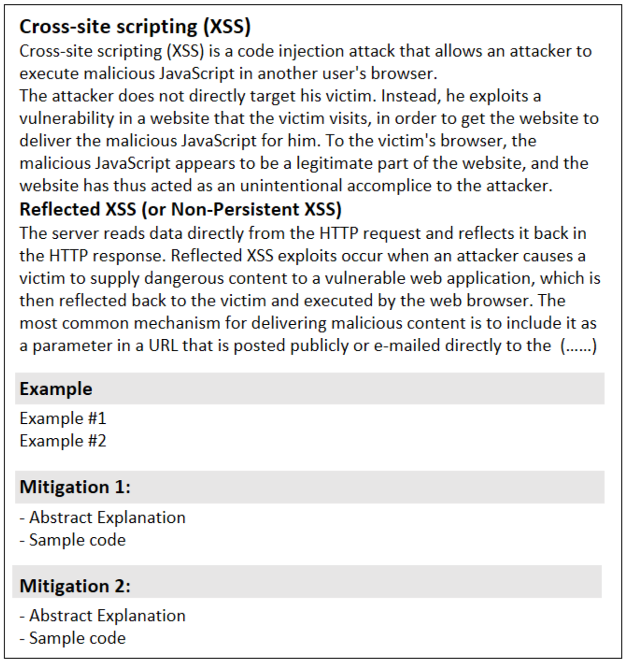
In this section, we describe the evaluation of the proposed approach as well as the developed prototypes. A non-equivalent pre- and post-test group based on the quasi-experiment was designed and executed in the setting of a university learning environment. The research design is presented in table 1. The participants were 36 Bachelor students from two main study programs, Bachelor in IT operations information security, and Bachelor in Programming of Norwegian University of Science and Technology.

Table 1. Experiment Design

| Group | Number of participants | Treatment |
|---|---|---|
| Experiment | 18 | $X_1$ |
| Control | 18 | $X_2$ |

Remark:
$X_1$: The proposed learning system
$X_2$: Conventional learning material

The participants were randomly assigned to either control or experimental groups. The students in the experimental groups were treated the proposed learning system ($X_1$) while the control group adopted a conventional learning approach, which was a hard-copy document ($X_2$). The learning subject was focused on a common security attack in web applications: Cross-Site Scripting. According to OWASP's Top 10 Application Security Risks – 2017 [46], it is the third most risky web applications' vulnerability and the most widespread. To construct the learning material for the control group, the authors extracted information from textbooks and resources on the internet, combing with the authors' teaching experience in the domain of software security. The knowledge content was organized in the order of abstract-to-concrete where the conceptual description of the vulnerability subject was described in the first place, followed by examples with code fragments of exploits. Mitigations for the vulnerabilities were explained in the last section. Figure 9 shows a simplified view of the learning material $X_2$ for the control group.



**Cross-site scripting (XSS)**

Cross-site scripting (XSS) is a code injection attack that allows an attacker to execute malicious JavaScript in another user's browser.
The attacker does not directly target his victim. Instead, he exploits a vulnerability in a website that the victim visits, in order to get the website to deliver the malicious JavaScript for him. To the victim's browser, the malicious JavaScript appears to be a legitimate part of the website, and the website has thus acted as an unintentional accomplice to the attacker.

**Reflected XSS (or Non-Persistent XSS)**

The server reads data directly from the HTTP request and reflects it back in the HTTP response. Reflected XSS exploits occur when an attacker causes a victim to supply dangerous content to a vulnerable web application, which is then reflected back to the victim and executed by the web browser. The most common mechanism for delivering malicious content is to include it as a parameter in a URL that is posted publicly or e-mailed directly to the (......)

**Example**

Example #1
Example #2

**Mitigation 1:**

- Abstract Explanation
- Sample code

**Mitigation 2:**

- Abstract Explanation
- Sample code

Figure 9. A sample of the learning materials for the control group

## 7.1 Data Collection

To collect data and measure the dependent variables, two types of instruments were used: knowledge test sheets and the survey questionnaires. Knowledge test sheets, differentiated by pre-test ($T_1$) and post-test ($T_2$), were developed to measure the knowledge gain (i.e., $T_2$ to $T_1$), in which items were created across two type of security knowledge—theoretical and practical. Theoretical items focused on recalling and understanding of conceptual security knowledge. Practical items require students identifying possible attacks in a given software context, marking coding errors in code fragments, applying knowledge to different situations. The pre- and post-tests were similar except for the formulation of some questions, their order, and the answer options. In each test sheet, there were 12 questions and the value for each question was five points.

We designed two survey questionnaires ($S_1$ and $S_2$) to collect students' perception of the two learning approaches. Questionnaire $S_1$ was developed to measure the learning satisfaction of students in the experimental group. Two major sections with five questions for each were designed in $S_1$, which are "System operation" and "Learning attitude". In this questionnaire, all respondents were required to choose the answer that reflects their own views and stance on the statements that are administered in accordance with the 5-point Likert scale, ranging from "strongly disagree" to "strongly agree". Questionnaire $S_2$ was created to collect all students' perception about the two approaches (i.e., $X_1$ vs. $X_2$) in order to understand their learning preferences. In this questionnaire, students were asked to indicate their preferred learning approach that best fits the statement of each question.

## 7.2 Experimental Procedure

The detailed experimental procedure is presented in table 2. The students were randomly assigned into two groups (experimental and control group) while they entered the classroom. They were first introduced to the main objectives of the experiment and informed of the procedure. After completing the pre-test sheets, students went through and studied the learning materials using the treatments given to them. At the end of the learning session, all students took the post-test exam where students of the experiment group filled out questionnaire $S_1$ additionally. In the last 30 minutes, students were asked to experience the learning approaches that were different from the previous one they practiced, and completed questionnaire $S_2$ afterward. This ended the experimental procedure.

Table 2. The experimental procedure

| Step | Activity | Duration (minutes) | Treatment | |
|---|---|---|---|---|
| | | | Experimental group | Control Group |
| 1 | Pre-test | 15 | $T_1$ | $T_1$ |
| 2 | Learning session | 60 | $X_1$ | $X_2$ |
| 3 | Post-test | 15 | $T_2$ | $T_2$ |
| 4 | Survey I | 5 | $S_1$ | -- |
| 5 | Experiencing session | 25 | $X_2$ | $X_1$ |
| 6 | Survey II | 5 | $S_1$ | $S_1$ |

## 7.3 Experimental Analysis

### 7.3.1 *Knowledge Gain Analysis*

The students' knowledge gain on the different type of treatment were determined using the compare means analysis. Table 4 reveals the mean analysis of students' performance on the pre- and post-test, including the mean scores and standard deviation. The results of the statistical analysis show that there was a positive knowledge gain (i.e. post-test to pre-test score) for both groups. However, the experimental group had higher achievement levels than the control group, as shown in Figure 4. The average knowledge gain in the control group was 10.28 whereas was 16.11 in the experiment group.

Table 4. Compared means analysis of students' performance on the pre- and post-test

| | | N | Mean | Std. Deviation |
|---|---|---|---|---|
| Control Group | Pre-Test | 18 | 30.00 | 11.757 |
| | Post-Test | 18 | 40.28 | 9.922 |
| Experimental Group | Pre-Test | 18 | 30.83 | 11.789 |
| | Post-Test | 18 | 46.94 | 7.503 |

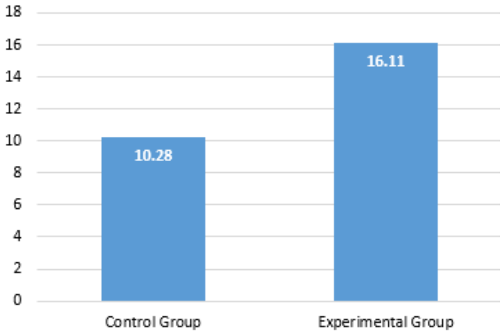


Figure 10. Knowledge gain for the control and experiment groups

To determine whether there was a significant difference between the pre-test performances of the experimental and control groups, an independent sample t-test was used. Table 5 shows the t-test analysis for pre-test results. The significant level (0.519) of Levine's test for equal variance was greater than 0.05, indicating "Equal variance assumed". Following the value indicated in Levine's test, we got "Sig. (2-tailed)" value of 0.833, which is above 0.05. Therefore, the null hypothesis of the independent sample t-test was rejected (*p* > 0.05). This implies that there was no significant difference between the two groups in terms of pre-test scores (i.e., the initial security knowledge). Therefore, the significance of the knowledge gain can be concluded,

Table 5. Independent sample t-test for pre-test score

| | | Levene's Test | | t-test for Equality of Means | | | | |
|---|---|---|---|---|---|---|---|---|
| | | F | Sig. | t | df | Sig. (2-tailed) | Mean Difference | Std. Error Difference |
| Pre-Test | Equal variances assumed | 0.425 | 0.519 | -0.212 | 34 | 0.833 | -0.833 | 3.924 |
| | Equal variances not assumed | | | -0.212 | 34 | 0.833 | -0.833 | 3.924 |

We also performed an independent sample t-test for the post-test mean scores. As can be seen in table 6, the difference between the post-test mean score of the two groups was significant (2-tailed Sig. = 0.029, *p* < 0.05). This indicated that the experimental treatments have resulted in a significant difference in security knowledge gain between the two groups of students.

Table 6. Independent sample t-test for the post-test score

| | | Levene's Test | | t-test for Equality of Means | | | | |
|---|---|---|---|---|---|---|---|---|
| | | F | Sig. | t | df | Sig. (2-tailed) | Mean Difference | Std. Error Difference |
| Post-Test | Equal variances assumed | 0.142 | 0.709 | -2.274 | 34 | 0.029 | -6.667 | 2.932 |

| | | | | | |
|---|---|---|---|---|---|
| Equal variances not assumed | -2.274 | 31.651 | 0.030 | -6.667 | 2.932 |

Then in order to see whether the treatment given to the experimental group had caused a statistical difference in students' performances; a paired sample t-test was performed as well. Table 7 shows that there was a significant average difference between pre-test and post-test scores ($t_{17}$ = 7.734, $p < 0.05$) in the experiment group

Table 7. Paired sample t-test of pre- and post-test for the experimental group

| | | Paired Differences | | | | | |
|---|---|---|---|---|---|---|---|
| | | Mean | Std. Deviation | Std. Error Mean | t | df | Sig. (2-tailed) |
| Experiment Group | Post-test - Pre-test | 16.111 | 8.838 | 2.083 | 7.734 | 17 | 0.000 |

### 7.3.2 *Questionnaire Analysis*

Table 8 presents the evaluation of students' learning satisfaction (questionnaire $S_1$) in the experimental group. As shown in the table, the satisfaction degree achieved 4.07 in terms of system operation and 4.09 regarding the learning attitude.

Table 8. The evaluation of student' learning satisfaction in the experimental group

| Category | Question | Mean |
|---|---|---|
| System Operation | I agree that the applying learning technique in the system is novel and it can assist my learning. | 4.11 |
| | I am very clear about the learning procedure embedded in the system. | 4.00 |
| | The system organizes security knowledge in a structured and collected manner. | 4.21 |
| | The knowledge content provided by the system is easy to understand. | 4.00 |
| | I think that the system is useful for learning security knowledge. | 4.05 |
| | *Average* | 4.07 |
| Learning Attitude | The system helps me deepen the memorized impression on the learning subject. | 4.11 |
| | The system helps me relate security knowledge to what I knew or experienced before. | 4.16 |
| | The system reduces the difficulty in learning secure programming. | 4.11 |
| | I find that at times studying the learning materials gives me a feeling of personal satisfaction. | 4.05 |
| | The system helps me foster a positive attitude towards learning security knowledge. | 4.00 |
| | *Average* | 4.09 |

Table 9 summarizes the result of students' learning preferences evaluation (questionnaire $S_2$) for the two learning approaches. It indicates that among the 36 students, 77.78% of students agreed that the learning system organized security knowledge that fit their learning preferences. Meanwhile, 88.89% of students considered the contextualized learning system can promote their learning interest much more than the conventional materials. The most important, all students thought that the proposed learning system can ease information overload on learning security subjects.

Table 9. The evaluation of student' learning preferences

| Question | Proposed Learning System (%) | Conventional Material (%) |
|---|---|---|
| The approach organizes security knowledge in a way that fits my learning preference. | 77.78 | 22.22 |
| The approach can promote my learning interest much more. | 88.89 | 11.11 |
| The approach eases information overload on learning security subjects. | 100 | 0 |
| The approach can make my security knowledge progress more. | 72.22 | 27.78 |
| The approach can benefit most people in learning software security. | 83.33 | 16.67 |

## 8. DISCUSSION AND CONCLUSION

In this study, an ontology-based contextualized design of the software-security learning system is proposed with three strategies. The first is to establish meaning scenarios to create a meaningful situation for learners. The design of the application context aims to activate the learner's prior knowledge of software programming and anchors the learning about security knowledge. The second strategy is to organize underlying security knowledge in a structured manner that can stimulate learners' mental models to support more efficient learning in the specified context. The third is to guide learners to engage with concrete knowledge before studying abstract knowledge. This strategy assists learners in discovering meaningful concepts and relationships between practical functions and abstract knowledge when working in this context.

The developed prototype was evaluated by a controlled quasi-experiment with 36 bachelor students. We used pre-test/post-test to measure students' security knowledge gain, and questionnaires to evaluate their learning satisfaction. The result of pre-test/post-test experiment indicates an increase in students' level of security knowledge for both learning approaches; the experimental group yielded more knowledge gain in average than the control group. According to the statistical t-test analysis result, there is no significant difference between the two participating groups of students in term of initial security knowledge (Table 5). However, there resulted in a statistical difference in security knowledge gain between the two groups of students after applying the treatments (Table 6). Additionally, the average difference between pre-test and post-test scores for the experiment group is also proved significant (Table 7). This concludes that students using the proposed learning system yielded significantly better knowledge gain than those using conventional learning materials.

On the other hand, the evaluation of students' satisfaction about the two learning approaches shows positive result, as the respondents expressed their higher learning satisfaction with the learning system using contextualized security knowledge than conventional learning materials. The survey results also show that most students were very interested in the proposed learning system and all agreed that this approach could ease the information load effectively. Our approach attempts to place security learning in the context of real application scenarios. The benefits of this contextualized approach can also be explained by the effective mechanism of intrinsic motivation, where a learner is drawn to engage in a task because it is perceived as interesting, enjoyable, and/or useful [12, 15, 36]. Since the given context is connected and relevant to their prior knowledge and life experiences in software development, security learning can then be related to a similar programming topic that they want to learn about or a problem to be solved. We believe this implies a direct effect of the contextualized learning approach on higher overall learning satisfaction, which motivates students to learn.

In conclusion, our proposed approach to establishing a contextualized learning system does provide a sounder basis for software security learning than conventional methods. It is recommended that curriculum developers of software security materials should use the context-based approach as one of the teaching strategies to improve students' performance in security knowledge. As part of our future work, we plan to improve the usability of the user interface and to enrich the knowledge content with a variety of application scenarios, meanwhile, to conduct detailed evaluations, analyzing the benefits in various application domains.

## REFERENCE

[1] "Psychology Wiki"; Available from: http://psychology.wikia.com/wiki/Knowledge_structure.

[2] Apvrille, A. and M. Pourzandi (2005), "Secure software development by example". IEEE Security & Privacy, volume 3, issue 4, pages 10-17.

[3] Bennett, J., F. Lubben, and S.J.S.e. Hogarth (2007), "Bringing science to life: A synthesis of the research evidence on the effects of context-based and STS approaches to science teaching". volume 91, issue 3, pages 347-370.

[4] Berns, R.G. and P.M. Erickson (2001), "Contextual Teaching and Learning: Preparing Students for the New Economy. The Highlight Zone: Research@ Work No. 5". volume, issue, pages.

[5] Bishop, M. (2010), "A Clinic for" Secure" Programming". IEEE Security & Privacy, volume 8, issue 2, pages.

[6] Brézillon, P. (2002), "Modeling and using context: Past, present and future", Rapport de recherche interne LIP6: Paris. pages.

[7] Brézillon, P. and J.-C. Pomerol (1999), "Contextual knowledge sharing and cooperation in intelligent assistant systems". Le Travail Humain, volume, issue, pages 223-246.

[8] Busch, M. and M. Wirsing (2015), "An Ontology for Secure Web Applications". Int. J. Software and Informatics, volume 9, issue 2, pages 233-258.

[9] Cognition and T.G.a.V.J.E. Psychologist (1992), "The Jasper series as an example of anchored instruction: Theory, program description, and assessment data". volume 27, issue 3, pages 291-315.

[10] Cognition, et al. (1992), "Anchored instruction in science and mathematics: Theoretical basis, developmental projects, and initial research findings". volume, issue, pages 244-273.

[11] Cooper, S. and S.J.A.I. Cunningham (2010), "Teaching computer science in context". volume 1, issue 1, pages 5-8.

[12] Cordova, D.I. and M.R.J.J.o.e.p. Lepper (1996), "Intrinsic motivation and the process of learning: Beneficial effects of contextualization, personalization, and choice". volume 88, issue 4, pages 715.

[13] Council, N.R. (2000), "How people learn: Brain, mind, experience, and school: Expanded edition". volume: National Academies Press.

[14] Craik, K.J.W. (1967), "The nature of explanation". volume 445. CUP Archive.

[15] Dean, R.J., L.J.C.C.J.o.R. Dagostino, and Practice (2007), "Motivational factors affecting advanced literacy learning of community college students". volume 31, issue 2, pages 149-161.

[16] Dey, A.K.J.P. and u. computing (2001), "Understanding and using context". volume 5, issue 1, pages 4-7.

[17] Diethelm, I., P. Hubwieser, and R. Klaus (2012), "Students, teachers and phenomena: educational reconstruction for computer science education". in Proceedings of the 12th Koli Calling International Conference on Computing Education Research. ACM.

[18] Dolmans, D.H., et al. (2005), "Problem-based learning: Future challenges for educational practice and research". Medical education, volume 39, issue 7, pages 732-741.

[19] Errington, E.P.J.I.J.o.L. (2009), "Being there: closing the gap between learners sand contextual knowledge using near-world scenarios". volume 16, issue, pages 585-594.

[20] Felder, R.M. and L.K.J.E.e. Silverman (1988), "Learning and teaching styles in engineering education". volume 78, issue 7, pages 674-681.

[21] Felder, R.M., et al. (2000), "The future of engineering education II. Teaching methods that work". volume 34, issue 1, pages 26-39.

[22] Gentner, D. and A.L. Stevens (2014), "Mental models". volume: Psychology Press.

[23] Goldkuhl, G. and E. Braf (2001), "Contextual knowledge analysis-understanding knowledge and its relations to action and communication". in Second European Conference on Knowledge Management Proceedings.

[24] Goldstone, R.L. and Y.J.C.p. Sakamoto (2003), "The transfer of abstract principles governing complex adaptive systems". volume 46, issue 4, pages 414-466.

[25] Goldstone, R.L. and J.Y.J.T.J.o.t.L.S. Son (2005), "The transfer of scientific principles using concrete and idealized simulations". volume 14, issue 1, pages 69-110.

[26] Gruber, T.R. (1993), "A translation approach to portable ontology specifications". Knowledge acquisition, volume 5, issue 2, pages 199-220.

[27] Gruber, T.R. (1995), "Toward principles for the design of ontologies used for knowledge sharing?". International journal of human-computer studies, volume 43, issue 5, pages 907-928.

[28] Guo, M. and J.A. Wang (2009), "An ontology-based approach to model common vulnerabilities and exposures in information security". in ASEE Southest Section Conference.

[29] Guzdial, M.J.A.I. (2010), "Does contextualized computing education help?". volume 1, issue 4, pages 4-6.

[30] Guzdial, M.J.J.o.C.S.i.C. (2006), "Teaching computing for everyone". volume 21, issue 4, pages 6-6.

[31] Gyrard, A., C. Bonnet, and K. Boudaoud (2013), "The stac (security toolbox: attacks & countermeasures) ontology". in Proceedings of the 22nd International Conference on World Wide Web. ACM.

[32] Kamina, P. and N.N. Iyer (2009), "From concrete to abstract: Teaching for transfer of learning when using manipulatives". volume, issue, pages.

[33] Kang, W. and Y. Liang (2013), "A security ontology with MDA for software development". in Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2013 International Conference on. IEEE.

[34] Khairkar, A.D., D.D. Kshirsagar, and S. Kumar (2013), "Ontology for detection of web attacks". in Communication Systems and Network Technologies (CSNT), 2013 International Conference on. IEEE.

[35] Ko, A.J. and B.A. Myers (2008), "Debugging reinvented: asking and answering why and why not questions about program behavior". in Proceedings of the 30th international conference on Software engineering. ACM.

[36] Kozeracki, C.A.J.N.d.f.c.c. (2005), "Preparing faculty to meet the needs of developmental students". volume 2005, issue 129, pages 39-49.

[37] Land, S.M.J.E.T.R. and Development (2000), "Cognitive requirements for learning with open-ended learning environments". volume 48, issue 3, pages 61-78.

[38] Lave, J., E. Wenger, and E. Wenger (1991), "Situated learning: Legitimate peripheral participation". volume 521423740. Cambridge university press Cambridge.

[39] Leach, J., P.J.S. Scott, and Education (2003), "Individual and sociocultural views of learning in science education". volume 12, issue 1, pages 91-113.

[40] Marques, M. and C.G. Ralha (2014), "An ontological approach to mitigate risk in web applications". the Proceedings of SBSeg, volume, issue, pages.

[41] McCaulley, M.H., et al. (1983), "APPLICATIONS OF PSYCHOLOGICAL TYPE IN ENGINEERING-EDUCATION". volume 73, issue 5, pages 394-400.

[42] McCaulley, M.H.J.E.E. (1976), "Psychological Types in Engineering: Implications for Teaching". volume 66, issue 7, pages 729-736.

[43] McGraw, G. (2006), "Software security: building security in". volume 1. Addison-Wesley Professional.

[44] Merrill, M.D. (2000), "Knowledge objects and mental models". in Advanced Learning Technologies, 2000. IWALT 2000. Proceedings. International Workshop on. IEEE.

[45] Nonaka, I. and N. Konno (1998), "The concept of" ba": Building a foundation for knowledge creation". California management review, volume 40, issue 3, pages 40-54.

[46] OWASP, "OWASP Top 10 Application Security Risks - 2017"; Available from: https://www.owasp.org/index.php/Top_10-2017_Top_10.

[47] Parchmann, I., et al. (2006), "“Chemie im Kontext”: A symbiotic implementation of a context-based teaching and learning approach". volume 28, issue 9, pages 1041-1062.

[48] Pashler, H., et al. (2007), "Organizing Instruction and Study to Improve Student Learning. ". IES Practice Guide. NCER 2007-2004. National Center for Education Research., volume, issue, pages.

[49] Perin, D. (2011), "Facilitating student learning through contextualization: A review of evidence". Community College Review, volume 39, issue 3, pages 268-295.

[50] Predmore, S.R.J.T.C.E. and Careers (2005), "Putting it into Context". volume 80, issue 1, pages 22-25.

[51] Razzaq, A., et al. (2014), "Ontology for attack detection: An intelligent approach to web application security". computers & security, volume 45, issue, pages 124-146.

[52] Rivet, A.E. and J. Krajcik (2008), "Contextualizing instruction: Leveraging students' prior knowledge and experiences to foster understanding of middle school science". Journal of Research in Science Teaching: The Official Journal of the National Association for Research in Science Teaching, volume 45, issue 1, pages 79-100.

[53] Rouse, W.B. and N.M.J.P.b. Morris (1986), "On looking into the black box: Prospects and limits in the search for mental models". volume 100, issue 3, pages 349.

[54] Salini, P. and S. Kanmani (2013), "Ontology-based representation of reusable security requirements for developing secure web applications". International Journal of Internet Technology and Secured Transactions, volume 5, issue 1, pages 63-83.

[55] Seel, N.M., S. Al-Diban, and P. Blumschein (2000), "Mental models & instructional planning", in Integrated and holistic perspectives on learning, instruction and technology, Springer. pages 129-158.

[56] Shambaugh, N.J.J.o.V.L. (1995), "The cognitive potentials of visual constructions". volume 15, issue 1, pages 7-24.

[57] Sherwood, R.D., et al. (1987), "Some benefits of creating macro-contexts for science instruction: Initial findings". volume 24, issue 5, pages 417-435.

[58] Specht, M. (2008), "Designing contextualized learning", in Handbook on information technologies for education and training, Springer. pages 101-111.

[59] Tudorache, T., et al. (2013), "WebProtégé: A collaborative ontology editor and knowledge acquisition tool for the web". Semantic web, volume 4, issue 1, pages 89-99.

[60] Wand, Y., V.C. Storey, and R. Weber (1999), "An ontological analysis of the relationship construct in conceptual modeling". ACM Transactions on Database Systems (TODS), volume 24, issue 4, pages 494-528.

[61] Wason, P.C. and D.J.T.Q.J.o.E.P. Shapiro (1971), "Natural and contrived experience in a reasoning problem". volume 23, issue 1, pages 63-71.