UNIVERSIDAD DE COSTA RICA
SISTEMA DE ESTUDIOS DE POSGRADO

VALIDACIÓN EMPÍRICA DEL USO DE PLANTILLAS PARA LA IDENTIFICACIÓN DE REQUERIMIENTOS DE SEGURIDAD DE APLICACIONES DE SOFTWARE

Trabajo final de investigación aplicada sometido a la consideración de la Comisión del Programa de Estudios de Posgrado en Computación e Informática para optar al grado y título de Maestría Profesional en Computación e Informática

ANDRÉS MARTÍNEZ MESÉN

Ciudad Universitaria Rodrigo Facio, Costa Rica

2019

Dedicatoria

A mis padres, por su apoyo incondicional. Sin ellos, jamás lo habría logrado.

Agradecimientos

A mi profesor guía Christian Quesada López y al profesor Marcelo Jenkins Coronas, por la retroalimentación y los consejos brindados durante la elaboración de este TFIA.

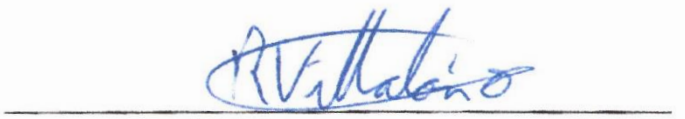
A la profesora Gabriela Barrantes Sliesarieva, por el apoyo en la creación del oráculo y su gentileza al permitirme realizar el estudio en su curso. A los profesores Ricardo Villalón Fonseca y Gustavo Esquivel Quirós, por su apoyo en la creación del oráculo.

A los estudiantes del PCI que participaron en este TFIA.

A mis compañeros Juan Carlos Romero Steller y Ronald Mora Barboza, por acompañarme desde el inicio de la maestría.

A Yorlenny Mora Mesén y la profesora Lidia Arévalo Bravo, por su invaluable ayuda desde que fui aceptado en el PPCI.

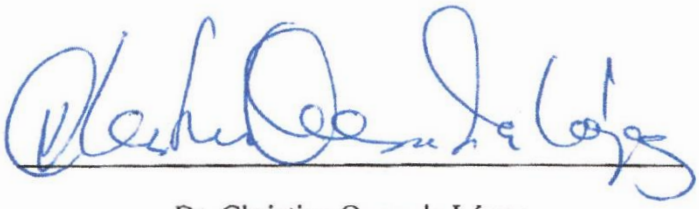
"Este trabajo final de investigación aplicada fue aceptado por la Comisión del Programa de Estudios de Posgrado en Computación e Informática de la Universidad de Costa Rica, como requisito parcial para optar al grado y título de Maestría Profesional en Computación e Informática".



Dr. Ricardo Villalón Fonseca

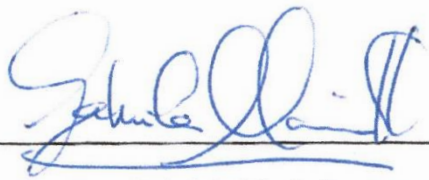
Representante del Decano

Sistema de Estudios de Posgrado



Dr. Christian Quesada López

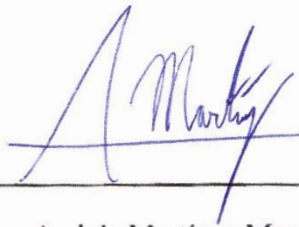
Profesor Guía



Dra. Gabriela Marín Raventós

Directora

Programa de Posgrado en Computación e Informática



Andrés Martínez Mesén

Sustentante

Table of contents

Resumen

La identificación de los requerimientos de seguridad es una tarea esencial que debe de ser abordada en las fases tempranas del ciclo de vida del desarrollo de software (SDLC). Usualmente es encomendada a ingenieros de software y a personal de TI en general, en caso de que no haya un ingeniero de requerimientos en la organización. No obstante, estos profesionales no son expertos en seguridad y podrían fallar al definir cuáles objetivos de seguridad deben estar presentes en un sistema, por lo que podrían generar pocos o inadecuados requerimientos de seguridad. En algunos casos, este proceso de extracción de requerimientos de seguridad no se realiza o se realiza hasta que el sistema está por ser liberado. En este contexto, la seguridad podría ser vista como opcional.

Varios enfoques de ingeniería de requerimientos de seguridad han sido planteados para resolver parte de este problema. Sin embargo, estos enfoques todavía demandan que el profesional sea un experto en seguridad. Algunos investigadores han argumentado que el uso de plantillas de requerimientos podría ayudar a los profesionales a identificar los requerimientos de seguridad implícitos en los requerimientos funcionales. Riaz et al. (2014a) propuso un conjunto de 19 plantillas de requerimientos de seguridad vinculadas a seis objetivos de seguridad: confidencialidad, integridad, disponibilidad, identificación y autenticación, responsabilidad y privacidad. Este método fue diseñado para ayudar en la identificación de objetivos de seguridad implícitos en los requerimientos funcionales.

En esta investigación, se realizaron dos replicaciones diferenciadas y se analizó la efectividad de las plantillas de requerimientos de seguridad para apoyar la identificación de los requerimientos de seguridad implícitos en los requerimientos funcionales. Para esto evaluamos el uso de las plantillas y comparamos los hallazgos obtenidos en estudios previos, en un contexto diferente. En el 2015, la primera replicación del experimento original fue realizada en la Universidad de Costa Rica (UCR). En 2018, realizamos las dos nuevas replicaciones en la UCR. Las respuestas de 17 participantes fueron analizadas en términos de calidad, cobertura, relevancia y eficiencia, y discutimos el impacto de los factores de contexto de las nuevas replicaciones. Los participantes fueron divididos en dos grupos: tratamiento y control. Al primero se le proporcionaron plantillas de requerimientos de seguridad, mientras que el segundo no recibió dicha ayuda. Las respuestas fueron comparadas con las obtenidas en la replicación del 2015 y en total se analizaron las respuestas de 33 participantes.

La prueba de Mann-Whitney fue aplicada y los hallazgos obtenidos confirman algunos de los resultados previos: los grupos de tratamiento tuvieron un mejor desempeño que los grupos de control, en términos de la cobertura de los requerimientos de seguridad identificados. Además, el proceso de extracción de requerimientos tuvo un desempeño significativamente superior en cuanto a la relevancia de una de las dos replicaciones. Las plantillas de requerimientos de seguridad apoyaron a los participantes en la identificación de un conjunto básico de requerimientos de seguridad y los participantes se mostraron anuentes al uso de estas plantillas para la extracción de dichos requerimientos.

Abstract

The identification of software security requirements is an essential and difficult task. This must be addressed in the early stages of the software development life cycle (SDLC). It is often entrusted to software engineers in charge of developing the software system and IT personnel in general, in case there are no requirements engineers in the organization. Nevertheless, these professionals are not security experts and could fail to define which security objectives are or must be present in a system. Therefore, they could elicit few or inadequate security requirements. In fact, this security requirements elicitation process is usually overlooked and not much attention is given to it until the software system is about to be released. In this context, security could be seen as an add-on.

Several security requirement engineering approaches have been proposed to solve this recurrent problem. Nevertheless, these approaches still demand the practitioner to be knowledgeable in security concepts. Various researchers have argued that using security requirements templates could help practitioners to identify implied software security requirements from functional requirements, in the context of a software system. Riaz et al. (2014a) proposed a set of 19 security requirement templates linked to six security objectives: confidentiality, integrity, availability, identification & authentication, accountability and privacy. The method is intended to help in the identification of security objectives in functional requirements. Once these security objectives are identified, applicable templates that support the security goals related to each of the functional requirements can be instantiated.

In this investigation, we conducted two differentiated replications and analyzed the effectiveness of security requirements templates to support the identification of security requirements. Our objective was to evaluate this approach and compare the applicability of the previous findings in a different context. A previous replication of the controlled experiment was conducted at University of Costa Rica (UCR) in 2015. In 2018, we conducted the two additional replications at UCR as well. We evaluated the responses of the 17 participants in terms of quality, coverage, relevance and efficiency and discussed the impact of context factors. Participants were divided into two groups: treatment and control. The first one was provided with suggested security requirement templates, while the latter did not receive such help. The responses were compared to the ones obtained in UCR's 2015 replication. In total, the answers of 33 participants were analyzed.

A Mann-Whitney U test was applied and the derived findings support some previous results: treatment groups performed significantly better than the control groups, in terms of the coverage of the identified security requirements. Besides, the requirements elicitation process performed significantly better in relevance in one of the two latter replications. Security requirements templates supported participants in the identification of a core set of the security requirements and the participants were favorable towards the use of these templates for eliciting security requirements.

List of tables

List of figures

List of abbreviations

IEEE	Institute of Electrical and Electronics Engineers
IT	Information Technology
MOSRE	Model Oriented Security Requirements Engineering
RE	Requirements Engineering
RQ	Research Question
SD	Security Discoverer
SQUARE	Security Quality Requirements Engineering
SRE	Software Requirements Engineering
SRS	Software Requirements Specification

Publication license

UNIVERSIDAD DE COSTA RICA

SEP Sistema de Estudios de Posgrado

Autorización para digitalización y comunicación pública de Trabajos Finales de Graduación del Sistema de Estudios de Posgrado en el Repositorio Institucional de la Universidad de Costa Rica.

Yo, Andrés Martínez Mesén, con cédula de identidad 1-1434-0265, en mi condición de autor del TFG titulado Validación empírica del uso de plantillas para la identificación de requerimientos de seguridad de aplicaciones de software

Autorizo a la Universidad de Costa Rica para digitalizar y hacer divulgación pública de forma gratuita de dicho TFG a través del Repositorio Institucional u otro medio electrónico, para ser puesto a disposición del público según lo que establezca el Sistema de Estudios de Posgrado. SI [X] NO *[ ]

***En caso de la negativa favor indicar el tiempo de restricción: _____ año (s).**

Este Trabajo Final de Graduación será publicado en formato PDF, o en el formato que en el momento se establezca, de tal forma que el acceso al mismo sea libre, con el fin de permitir la consulta e impresión, pero no su modificación.

Manifiesto que mi Trabajo Final de Graduación fue debidamente subido al sistema digital Kerwá y su contenido corresponde al documento original que sirvió para la obtención de mi título, y que su información no infringe ni violenta ningún derecho a terceros. El TFG además cuenta con el visto bueno de mi Director (a) de Tesis o Tutor (a) y cumplió con lo establecido en la revisión del Formato por parte del Sistema de Estudios de Posgrado.

INFORMACIÓN DEL ESTUDIANTE:

Nombre Completo: Andrés Martínez Mesén

Número de Carné: B48843 Número de cédula: 1-1434-0265

Correo Electrónico: andresmartinezm@gmail.com

Fecha: 06/11/2019 Número de teléfono: 8888-2353

Nombre del Director (a) de Tesis o Tutor (a): Dr. Christian Quesada López

FIRMA ESTUDIANTE

Nota: El presente documento constituye una declaración jurada, cuyos alcances aseguran a la Universidad, que su contenido sea tomado como cierto. Su importancia radica en que permite abreviar procedimientos administrativos, y al mismo tiempo genera una responsabilidad legal para que quien declare contrario a la verdad de lo que manifiesta, puede como consecuencia, enfrentar un proceso penal por delito de perjurio, tipificado en el artículo 318 de nuestro Código Penal. Lo anterior implica que el estudiante se vea forzado a realizar su mayor esfuerzo para que no solo incluya información veraz en la Licencia de Publicación, sino que también realice diligentemente la gestión de subir el documento correcto en la plataforma digital Kerwá.

Chapter 1. Introduction

The identification of software security requirements is an essential and difficult task (Riaz, Slankas, King & Williams, 2014b). It is often entrusted to software engineers in charge of developing the software system and IT personnel in general, in case there are no requirements engineers in the organization. Nevertheless, these professionals are not security experts (Hamid & Weber, 2018) and could fail to define which security objectives are or must be present in a system. As a result, they may elicit few or inadequate security requirements to meet the objectives in question. In fact, this SDLC stage is usually overlooked and not much attention is given to it until the software system is about to be released (Riaz et al., 2014b). In this context, security could be seen as an add-on.

Even though many security requirements engineering (SRE) methods had been proposed (Mellado, Blanco, Sánchez & Fernández-Medina, 2010). Most of these still demand the practitioner to have a certain level of security knowledge and expertise. This explains the need for a tool-assisted process that can allow both inexperienced and experienced security professionals to elicit security requirements in a simpler way (Riaz et al., 2014b).

Riaz et al. researched about the topic and concluded that "systems that share common security objectives, such as confidentiality and integrity, often have similar security requirements, thus providing an opportunity to capture such common requirements in the form of reusable templates that can be instantiated in the context of a system" (Riaz et al., 2014b, p.183). Functional requirements in a software requirement specification (SRS) or use case may imply the need of security requirements. Wording in functional requirements may follow a pattern that may lead to the choice of various security requirements templates. Their work suggests that if functional requirements are analyzed and the practitioner concludes they implicitly suggest the need for privacy, to illustrate, privacy templates may be checked to see if they apply in the given context. A total of 19 security requirements templates were developed (Riaz et al., 2014b).

To analyze if this approach truly supports requirements analysts in the identification of security requirements, the authors developed the Security Discoverer process (SD) (Riaz, King, Slankas & Williams, 2014a). It is a method that "suggests applicable templates by automatically parsing individual requirements sentences in the input artifacts and identifying the security implications of the existing functional requirements of a system" (Riaz, King, Slankas, Williams, Massacci, Quesada-López & Jenkins, 2017, p. 2129). A controlled experiment was conducted by Riaz et al., (2014b), in order to evaluate the SD process with computer science graduate students. According to the results obtained in the study, the use of automatically-suggested templates did help participants in the treatment group elicit more software security requirements than students in the control group.

In order to contribute with empirical evidence about the effectiveness of security requirements templates to support the identification of security requirements, we conducted two empirical replications at UCR in 2018. We evaluated the responses of the 17 participants in terms of quality, coverage, relevance (of the elicited security requirements) and efficiency (of the requirements elicitation method) and discussed the impact of context factors. The responses were compared to the ones who participated in UCR's 2015 replication. In total, the answers of 33 participants were analyzed.

Therefore, the objective of this investigation was *to evaluate the effectiveness of the use of security requirements templates for the elicitation of implicit security requirements in functional requirements*.

To achieve this objective, the following specific objectives were carried out:

1. Identify the necessary aspects for an empirical evaluation on the use of security requirements templates.

2. Generate a security requirements specification document for a mobile health application by using the security requirement templates method.

3. Evaluate the effectiveness of the use of templates for the elicitation of security requirements.

This investigation consists of three main phases. First, a literature review was conducted to identify security requirements approaches and empirical evaluations regarding the evaluation of the effectiveness of security requirements templates. Second, the software security requirement specification using security requirements templates was generated for a mobile health application. Finally, two empirical replications were conducted in order to evaluate the effectiveness of the use of templates for the elicitation of security requirements. The detail of the steps followed in each of these objectives and phases is depicted in Figure 1, as well as explained in their corresponding chapters.

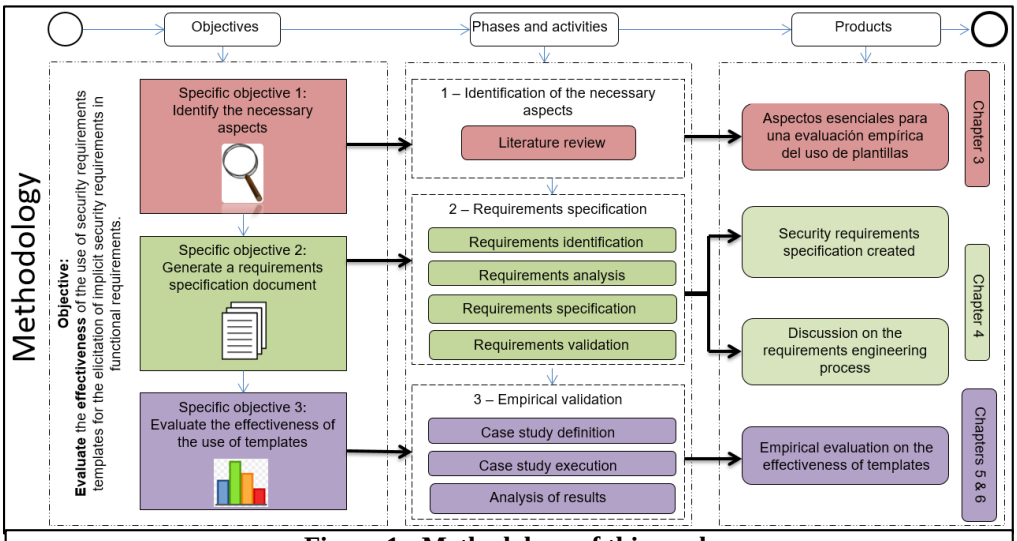


Figure 1 - Methodology of this work

The rest of this document is organized as follows: we present the background in Chapter 2. Related work is shown in Chapter 3. The security requirements specification created for this study is explained in Section 4. The replication process is detailed in Section 5. Section 6 discusses the analysis of results, lessons learned and the scope of SD methodology. Section 7 provides general conclusions and future directions.

Chapter 2. Background

This chapter explains the main concepts related to security requirements templates. Each subsection refers to a different concept.

2.1 - Software requirements

In the context of software engineering, a requirement is defined as “a need, expectation, constraint or interface of any stakeholders that must be fulfilled by the proposed software product during its development” (Chemuturi, 2013, p. 3). It is the main input of a software development project and, as a result, it is reasonable that requirements elicitation is the precursor phase of the software development life cycle (SDLF) (Chemuturi, 2013). A requirement specifies what a system should do and not how it should do it (Riaz & Williams, 2012).

From the point of view of functionality, software requirements can be classified in two categories: “core functional requirements” and “ancillary functionality requirements” (Chemuturi, 2013, p. 13). The former, commonly called "functional requirements", are those that give meaning and purpose to the software, since they address the business processes of an organization. The auxiliary functionality requirements, on the other hand, complement the basic functionality: they include aspects such as usability, data integrity, response times, memory restrictions, industry regulations and security, among others (Chemuturi, 2013).

The need for security arises when stakeholders determine that a resource belongs to the software system, whether tangible (such as money) or intangible (such as confidential information, for example), is of value to the organization (Romero-Mariona, 2009). These resources are called *assets* and stakeholders want to protect them from damage or attacks (Romero-Mariona, 2009). Regardless of the type of software application that is taken into account, each of them usually has the same types of vulnerable assets (data, communications, services, hardware components) (Firesmith, 2003a). Additionally, they “tend to be subject to the same kinds of security threats (e.g., theft, vandalism, unauthorized disclosure, destruction, fraud, extortion, espionage, trespass, etc.) from attacks by the same kinds of attackers (e.g., hackers, crackers, disgruntled employees, international cyber-terrorists, industrial spies, governmental spies, foreign military, etc.)” (Firesmith, 2003a, p. 507). Therefore, unlike functional requirements, security requirements present less variability between one software system and another (Firesmith, 2003a, p. 507), given that they pursue common security goals or objectives (which will be addressed in a later section).

Under this context, security requirements can be defined as “constraints on the functions of the system, where these constraints operationalize one or more security goals” (Haley, Moffett, Laney & Nuseibeh, 2006, p. 37). They should specify what is required to meet the security demand (e.g. indicate the need for a certain level of identification and authentication in some interface), instead of delving into the security mechanisms by which the objective should be achieved (Firesmith, 2003a, p. 507). Once they have been

specified, it is possible to focus on the security controls through which these requirements will be met (Bennaceur, Bandara, Jackson, Liu, Montrieux, Tun, Yu, & Nuseibeh, 2014), but not before.

Also, they must describe the desired security behavior of a system (Riaz, King, Slankas & Williams, 2014a) and "should be expressed as positive statements and not negative statements" (El-Hadary & El-Kassas, 2014, p.464). Often, when there is a need for security, it is "solved" by proving solutions or eliciting requirements that do not define where, when or how they should be executed, nor the asset they must protect. This is because security requirements are difficult to identify and are not usually addressed since the start of the SDLF (Mellado et al., 2010; Riaz et al., 2017). Some authors think of security requirements as high-level security goals or security policies (Haley et al., 2006). An information security policy is a policy that an organization uses to state how its assets must be protected and explains the responsibility each individual has when accessing its technology systems (Al-Hamdani & Dixie, 2009). It does not focus solely on software security, since it encompasses a more general security concept. Security policies mandate security and security requirements help specify these policies. Therefore, *adequate security requirements* must be determined so that security risks can be reduced. "By *adequate security requirements*, we mean requirements that, if respected, lead to a system's security goals being satisfied and by *system* we mean the software and, in addition, people who use the software, and equipment around the software (computers, printers, etc.)" (Haley, Moffett, Laney & Nuseibeh, 2008, p. 133). "They need to be explicit, precise, complete and non-conflicting with other requirements" (El-Hadary & El-Kassas, 2014, p.464).

"A third important aspect of a security requirement concerns the *circumstances* in which it must be satisfied. These describe application conditions of functionality, temporal, or spatial aspects, the social relationships between stakeholders – in general, the 'context' to which the requirement refers" (Fabian, Gürses, Heisel, Santen & Schmidt, 2010, p.12). Figure 2 shows the relationship between the security concepts mentioned so far.

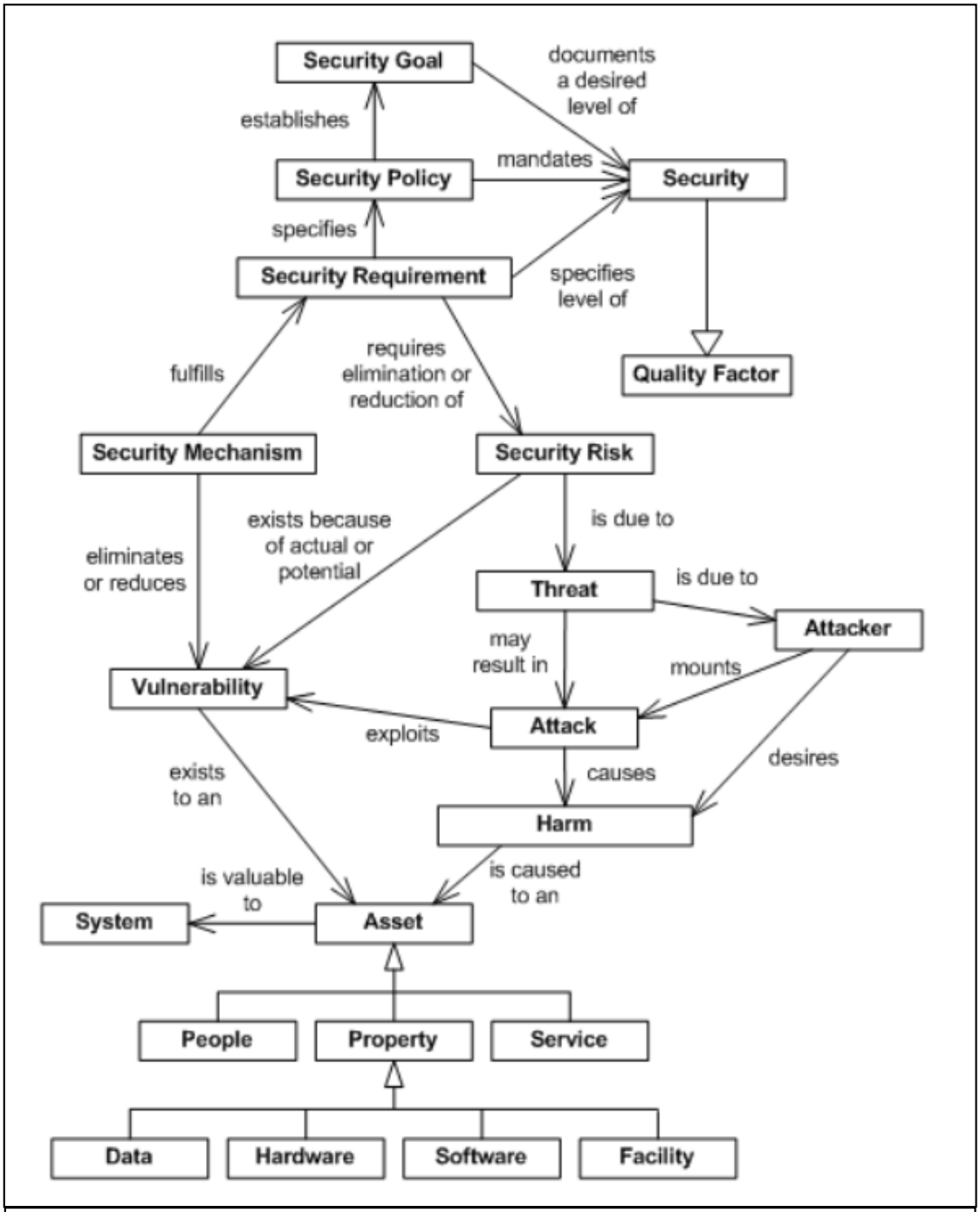


Figure 2 - Context in which security requirements originate

Source: Firesmith, 2004

2.2 – Requirements engineering and software requirements engineering (RE/SRE)

Requirements engineering (RE) is the first major stage of software development (Romero-Mariona, 2009) and existing literature defines it in various ways. It can be seen as "the process of discovering, documenting, and maintaining requirements for software solutions. The key objective of these activities aims at establishing sound goals and constraints for the software system being planned and constructed" (Nguyen, Vo, Lumpe & Grundy, 2012, p. 386). Customers, engineers and users gather together in a joint effort that allows them formulate these assumptions, limitations and goals together (Nguyen, 2012). This effort is carried out along several activities, such as the definition, elicitation, analysis, specification, modeling, validation and management of requirements (Bani-Salameh & Al jawabre, 2015).

Software systems increase in size and complexity, due to the growing need to add more requirements (Tantithamthavorn, Ihara, Hata & Matsumoto, 2014) to meet stakeholders' new demands. In order to ensure that the entire RE process is successful and that software analysts have defined a system that meets the needs of the stakeholders, requirements must be examined, understood and tested by novice and expert stakeholders (Tu, Tempero & Thomborson, 2014). Everyone involved in this process must understand their role in the RE process. However, despite the importance that this stage has, few studies dedicated to the teaching of requirements engineering have been published (Callele & Makaroff, 2006). In fact, requirements elicitation is a difficult task, since meetings and brainstorming with stakeholders must be carried out in a controlled manner in order to be effective (Bulusu, Laborde, Wazan, Barrère & Benzekri, 2017). A challenge in RE, for instance, is to improve communication between users and developers, as it is very bad (Tu, Tempero & Thomborson, 2014). This context explains why the benefits of developing software based on a proper requirements elicitation process are widely recognized in software engineering.

Security requirements engineering (SRE) can be found within RE. "Security requirements engineering is the process of eliciting, specifying and analyzing the security requirements for system. It is concerned with the prevention of harm in the real world and considering security requirements as functional requirements" (Salini & Kanmani, 2012, p. 127).

If requirements elicitation for a software project is difficult and is one of the main causes of why they fail (if not done properly) (Cerpa & Verner, 2009), identifying security requirements presents a higher difficulty: this process is rarely of interest and, most of the time, it is not done in a systematic way. This is because additional security experience is needed to be able to cope with this type of requirements (Mellado et al., 2010). Other studies also share this thinking, since they state that SRE "requires a rare combination of expertise in software security with proficiency in requirements engineering" (Riaz & Williams, 2012, p.1).

Various security requirements engineering approaches have been developed (Mellado et al., 2010; Riaz, Slankas, King & Williams, 2014b). Some of them involve the use

of conceptual frameworks, such as SQUARE (Mead, Hough & Stehney II, 2005), and requirement models (Mellado et al., 2010). Nevertheless, these approaches still demand the practitioner to be knowledgeable in security concepts (Riaz et al., 2014b). "Additionally, these methodologies rarely offer detailed advice on how to contextualize security requirements in the context of their systems" (Riaz & Williams, 2012, p.1), which is a common problem in software projects and continues to be a challenge even for security experts.

2.3- Security objectives

Security objectives are the desired security goals of a system (Schumacher, Fernandez-Buglioni, Hyberston, Buschmann & Sommerlad, 2006). It is "the high-level security-specific goal (such as confidentiality, integrity, availability) defining the outcomes a system must ensure or prevent" (Riaz & Williams, 2012, p. 30) and can be identified from the assets, business goals and organizational principles of an entity (Salini & Kanmani, 2012). Security objectives of software systems are not only technical aspects (from the perspective of systems development), but also operational and administrative aspects (Riaz et al., 2014a). However, this study will only address technical security objectives which are commonly present in software systems.

The research made by Riaz et al. (2014a) concluded that there are six main technical security objectives:

- Confidentiality (C): the degree to which the "data is disclosed only as intended by the enterprise" (Schumacher et al., 2006, p.20).
- Integrity (I): "the degree to which a system or component guards against improper modification or destruction of computer programs or data" (Riaz et al., 2014a, p. 185).
- Identification & Authentication (IA): "the need to establish that a claimed identity is valid for a user, process or device" (Riaz et al., 2014a, p. 185).
- Availability (A): "the degree to which a system or component is operational and accessible when required for use" (Institute of Electrical and Electronics Engineers [IEEE], 1990).
- Accountability (AY): "degree to which actions affecting software assets can be traced to the actor responsible for the action" (Riaz et al., 2014a, p. 185).
- Privacy (P): "the degree to which an actor can understand and control how their information is used" (Riaz et al., 2014a, p. 185).

The idea is that, for each software application you want to develop, all of its security objectives must be identified and materialized through a proper security requirements elicitation process that meet those objectives. "Security requirements are potentially reusable across systems that share same security objectives" (Riaz & Williams, 2012, p. 30). If a security requirement does not contribute to achieving a system security

goal, it would be advisable to modify or delete it. “By identifying the security objectives expressed or implied by a particular sentence within a document, we gain an understanding of the intent of the sentence as well as possible requirements and mechanisms to establish that intent” (Riaz et al., 2014a, p. 185). All input is important to meet security objectives, from meetings with stakeholders to documents that describe the needs of the project.

2.4- Security patterns

A software pattern is defined as “a recurring solution to a standard problem” (Schmidt, Fayad, & Johnson, 1996, p. 37) within a specific context. The first software patterns emerged in the mid-1990s (Schumacher et al., 2006) and, since then, they have been used in various SDLC phases, such as analysis, requirements elicitation, design, architecture, testing, security and configuration management, among others (Riaz & Williams, 2012). A good pattern captures the practice and human experience related to the use of concepts that have been proven successful in the past, so that they can be applied by novices and serve as support for experts in projects of medium and great complexity (Schumacher et al., 2006). Also, they do not describe a particular solution in detail, but they provide the core of the solution to that problem, so that the user can use it at his discretion (Schumacher et al., 2006). “A pattern thus invites its readers to reflect on the problem being presented: to think first and then to decide and act explicitly and consciously” (Schumacher et al., 2006, p. 3).

An interesting characteristic of patterns in general (not limited to software patterns only) is that even though each of them proposes a solution for a certain problem, they can (and sometimes must) work together (Schumacher et al., 2006). More often than not, the solution suggested by a pattern involves the use of more patterns that solve secondary problems related to the original problem (Schumacher et al., 2006). Their generic nature allows them to combine their recommendations to come up with a more accurate global solution.

We can find security patterns within software patterns. Similar to the previous definition, “a security pattern describes a particular recurring security problem that arises in specific contexts and presents a well-proven generic scheme for its solution” (Schumacher & Roedig, 2001, p. 5). These patterns capture the know-how and skills of security experts. This knowledge can come from several sources: individual experiences, common challenges in a domain, common vulnerabilities, standards and best practices, software artifacts and processes used to solve a problem (Riaz & Williams, 2012). Usually, a combination of several of these sources is used to identify problems and offer solutions to them.

It is of great interest to see how all of the concepts of this theoretical background are linked to each other, when studies suggest that “security requirements that help in achieving the same objective across multiple systems may be generalizable in the form of a pattern” (Riaz & Williams, 2012, p. 32). “The more information a pattern has, the more important structure becomes” (Schumacher & Roedig, 2001, p. 5). Not only is it a

challenge to know how to identify and formulate patterns (given that, according to Bloom's taxonomy, the highest levels of intelligence involve the analysis, synthesis and/or restructuring of concepts (Riaz & Williams, 2012), but also keep their structure so that they are uniform and comparable to each other (Schumacher & Roedig, 2001). This is of great relevance when working with multiple patterns. "The merit of a pattern can be assessed in terms of how successfully a practitioner employs a pattern to solve a particular commonly occurring problem" (Riaz, Breaux, & Williams, 2015, p. 15).

To illustrate what has been explained about software patterns, 18 security goal patterns were identified (Riaz, Stallings, Singh, Slankas & Williams, 2016a), which cover the 6 security objectives mentioned by Riaz et al. (2014a) in the third section of this background. They follow this format:

`<p | d | r>-<C | I | A | ID | AY | PR>`

The letters "`p`", "`d`" y "`r`" represent the three possible security actions: prevent (`p`), detect (`d`) y respond (`r`) to a security breach. Therefore, `d-P` means "detect a breach of confidentiality". The complete structure of these patterns can be seen in Figure 3.

Security Action	Security Property	Asset	Actor	Action type
<prevent (p) | detect (d) | respond to (r)> *a breach of*	Confidentiality **(C)**	of **<asset>**	when **<actor>** performs	<read | store | transfer>
	Integrity **(I)**			<create | update | delete | transfer>
	Availability* **(A)**			<create | read | update | delete | store | transfer>
	Id & Authentication ** **(ID)**			
	Accountability **(AY)**			
	Privacy **(PR)**			<read | store | transfer>

* Asset will be a service or system functionality.

** Applicable prior to any system access by default. Explicitly indicate the actions or assets that do not need authentication.

Figure 3 - Security goal patterns

Source: Riaz et al., 2016a

2.5- Context-specific security requirements templates

The templates that help in the writing security requirements, from functional requirements, are proposed by Riaz et al. (2014a). The idea is that, for each functional requirement or a statement from a software requirement specification (SRS), you can determine if it can be associated with one or more security objectives (Riaz et al., 2014a; Riaz et al., 2014b). Each security objective is materialized by one or more security requirements templates. In this manner, as the existence of identical security objectives in software applications suggests that they may probably have security requirements in common (Riaz et al., 2014a), the task of raising these security requirements becomes easier. "Unlike typical functional requirements, security requirements can be potentially be highly reusable, especially if specified as instances of reusable templates" (Firesmith, 2003a, p. 507).

As for its structure, these templates offer a generic and incomplete wording of the security requirements proposed, so they must be filled in manually (and under expert judgment) to be adapted to the context of their application. It is up to the user of the templates, also, whether or not he or she wants to use the suggested help.

A total of 19 context-specific templates have been extracted and they are named below, grouped by security objectives (Riaz et al., 2014a). The complete list of these templates is available at http://go.ncsu.edu/securitydiscoverer:

- Confidentiality: C1- authorized access; C2- during storage; C3- during transmission.
- Integrity: I1- read-type actions; I2- write-type actions; I3- delete actions; I4- unchangeable resources.
- Availability: A1- availability of data; A2- appropriate response time; A3- service availability; A5- backup and recovery capabilities; A5- capacity and performance.
- Identification & Authentication: IA1- select context for roles; IA2- unique accounts; IA3- Authentication.
- Accountability: AY1- log transactions with sensitive data; AY2- log authentication events; AY3- log system events.
- Privacy: PR1- usage of personal information.

Figure 4 shows a statement (input sentence) which implicitly expresses the need for confidentiality. To the left, three context-specific security requirements templates about confidentiality are displayed and, below “security requirements”, you can find the generated security requirements from these templates in order to satisfy the mentioned security objective.

	ID	Description	Example
Confidentiality	**C1**	**Authorized access** *Given:* <subject> = user <resource> = sensitive information <action> = create/read/update/delete/share type actions *Add Security Requirements:* • The system shall enforce access privileges that <enable|prevent> <subject> to <action> for <resource>. [see AY1, C2, I1, I2, I4]	**Statement:** The system shall provide a means to edit discharge instructions for a particular patient. **Security Requirements:** • The system shall enforce access privileges that enable authorized users to edit discharge instructions for a particular patient. • The system shall encrypt discharge instructions and store discharge instructions in encrypted format using an industry-approved encryption algorithm. • The system shall monitor the status and location of system components that may contain unencrypted discharge instructions data. • The system shall transmit discharge instructions in encrypted format to and from authorized users.
	C2	**Confidentiality during storage** *Given:* <resource> = sensitive information *Add Security Requirements:* • The system shall encrypt <resource> and store <resource> in encrypted format using an industry-approved encryption algorithm. [see AY3, I2, I4] • The system shall monitor the status and location of system components that may contain unencrypted <resource> data.	
	C3	**Confidentiality during transmission** *Given:* <subject> = user <resource> = sensitive information <action> = send/receive/transmit *Add Security Requirements:* • The system shall transmit <resource> in encrypted format to and from the authorized <subject>. [see C1, I4]	

Figure 4 - Confidentiality templates (left) are filled in and converted into security requirements (right), due to a security need of a functional requirement or use case

Source: Riaz et al., 2014a

2.6- Security Discoverer (SD)

Some studies have proposed methods that facilitate the identification and abstraction of security requirements, based on functional requirements (Lai, Peng & Ni, 2014); Cysneiros & Sampaio do Prado, 2002; Riaz et al., 2017; Riaz et al., 2014a; Yahya, Kamalrudin, Sidek & Grundy, 2015; Riaz et al., 2014b). The presumed hypothesis is that there are implicit security requirements and objectives in functional requirements (Riaz et al., 2014a). Security objectives are defined as the security goals or desired security properties of a system, while security requirements lead to the operation of these objectives (Riaz et al., 2017). Software systems and applications that have common security objectives, such as availability, integrity or data privacy, also have similar sets of security requirements (Firesmith, 2013b). These sets of requirements could be applied in multiple systems to respond to their respective security objectives, using reusable templates instantiated in the context of those systems (Riaz et al., 2014a; Riaz et al., 2014b). In other words, these templates would contain prefabricated security requirements that will eventually be completed with particular data of the system: actors, actions or resources on which the actions are carried out (Riaz et al., 2017).

It is possible to find grammatical similarities in security requirements (Riaz et al., 2014a) and store them in templates. The more similarities are found, the more variety of prefabricated requirements the templates could offer. In this manner, the proposed idea is for a tool to receive and identify the functional requirements or use case scenarios (in natural language) of a system and, based on that input, suggest the use of one or several templates to cover the needs of security corresponding to its context (Riaz et al., 2014a). This tool exists: it is known as Security Discoverer (SD) and works applying machine learning techniques that automatically take the statements of the aforementioned input and assess if these statements are related to one or more security objectives (Riaz et al., 2014a; Riaz et al., 2014b). SD takes into account the six security objectives proposed by Riaz et al. (2014a): confidentiality, integrity, availability, identification and authentication, accountability and privacy.

Figure 5 presents the SD process (Riaz et al., 2017). This tool helps identify and classify security properties immersed in sentences or statements that describe a software system. The goal and main functions of a software system can be understood by experts and non-computer experts by reading and analyzing their functional requirements. These requirements are part of a context: for instance, they make up a banking or health system. Therefore, the wording, the actors and the actions involved in these functional requirements form patterns from which SD can extract useful information to elicit security requirements based on this input.

As mentioned before, Security Discoverer uses several types of machine learning algorithms to identify security requirements from natural language artifacts. To do this, SD must be provided with sets of sentences (already classified by a security expert) to "learn" and "train". Subsequently, the tool would be ready to classify the statements provided by the user.

After being trained, the SD process starts by the tool receiving an "input set of natural language requirements artifacts" (Riaz et al., 2014a, p. 183), as shown in Figure 5. In this case, the input is a sentence indicating "the system shall provide a means to edit discharge instructions for a particular patient" (Riaz et al., 2017, p. 2134). The statement is automatically classified and associated to three different security objectives: confidentiality, integrity and responsibility. In other words, the tool hints there is a need for integrity (when editing), confidentiality (access levels to sensitive data) and an actor responsible for the action (changes and who performed the edit should be logged).

Once the classification is done, SD retrieves from its repository and suggests one or several security requirements templates related to each of the security objectives implied in the input sentence used in the example. In Figure 5, one of the confidentiality templates is recommended ("the system shall enforce access privileges that **<enable | prevent>** **<subject>** to **<action>** **<resource>**") and, from this point onwards, it is up to the security expert if how to fill it in (if needed).

As explained, a tool-assisted process such as Security Discoverer could fill the gap that currently exists in terms of security requirements elicitation. Automatically-suggested templates may help security experts and non-experts "identify a core set of relevant security requirements and focuses critical thinking around security concerns" (Riaz et al., 2014b, p. 1). In addition, these could help them "consider more security requirements than they would have otherwise" (Riaz et al., 2014b, p. 1).

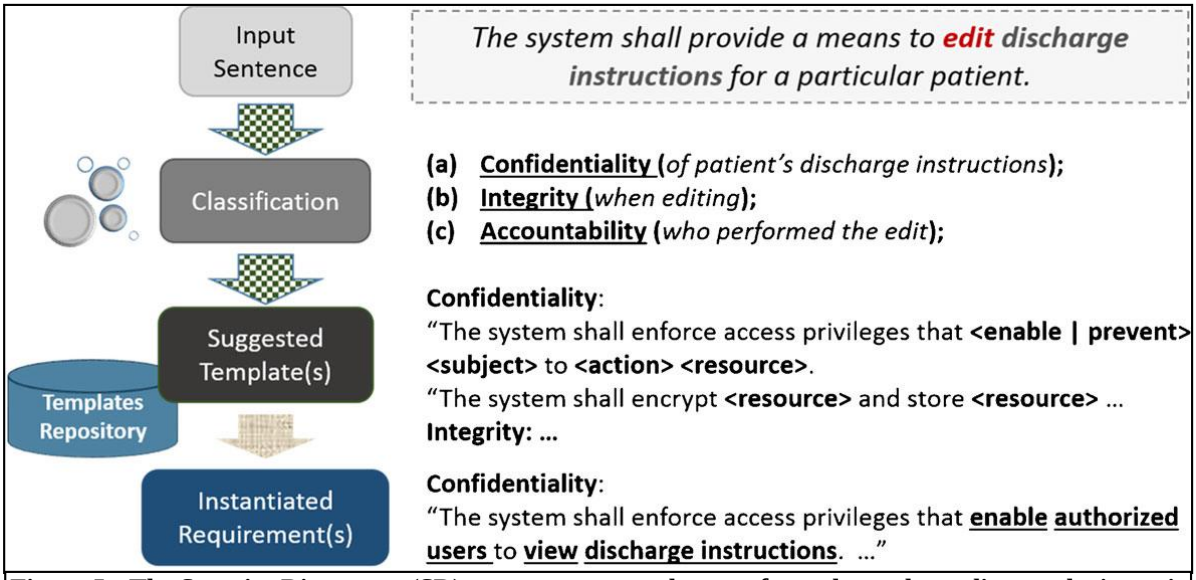


Figure 5 - The Security Discoverer (SD) process suggests the use of templates, depending on the input it receives. The green arrows show the automatized part of the process

Source: Riaz et al., 2014a

Chapter 3. Related Work

In this section, we present the studies related to security requirements engineering approaches, and we detail the essential aspects of an empirical evaluation of these approaches.

3.1 – Security requirements engineering approaches

Mellado et al. (2010) conducted a systematic review of security requirements engineering approaches. Mead et al. (2005) propose the SQUARE methodology. "System Quality Requirements Engineering (SQUARE) is a model developed at Carnegie Mellon by Nancy Mead as part of a research project with Donald Firesmith, and Carol Woody of the Software Engineering Institute" (Mead & Stehney, 2005, p. 2). It helps organizations incorporate security in the early stages of development, through a nine-step process that generates a categorized and prioritized deliverable of security requirements (Mellado et al., 2010). "This model may also be useful for documenting and analyzing the security requirements of fielded systems, and could be used to steer future improvements and modifications to these systems" (Mead, Hough & Stehney II, 2005, p. 3). The nine-step process consists of (Mead & Stehney, 2005, p. 2):

- Step 1: Agree on definitions
- Step 2: Identify security goals
- Step 3: Develop supporting artifacts
- Step 4: Perform risk assessment
- Step 5: Select elicitation techniques
- Step 6: Elicit security requirements
- Step 7: Categorize requirements
- Step 8: Prioritize requirements
- Step 9: Requirements inspections

"The methodology is most effective and accurate when conducted with a team of requirements engineers with security expertise and the stakeholders of the project" (Mead, Hough & Stehney II, 2005, p. 3). It is characterized by classifying security requirements as system level, software level or architectural constraint. In addition, it is important to note that SQUARE does not provide information on how to properly write security requirements, nor does it offer explicit examples of security goals.

Another approach is the MOSRE (Model Oriented Security Requirements Engineering) framework (Salini & Kanmani, 2012). It is intended for web applications and covers the following 16 steps (Salini & Kanmani, 2012):

- Step 1: Identify the objective of the web applications
- Step 2: Identify the stakeholders
- Step 3: Identify the assets
- Step 4: Select an elicitation technique
- Step 5: High level of architectural diagram of web applications

- Step 6: Elicit non-security goals and requirements
- Step 7: Generate use cases diagram for the web applications
- Step 8: Identify the security goals
- Step 9: Identify threats and vulnerabilities
- Step 10: Risk assessments
- Step 11: Categorize and prioritize the threats and vulnerabilities for mitigation
- Step 12: Generate misuse cases diagram for the web applications
- Step 13: Identify security requirements
- Step 14: Generate use cases diagram for the web applications considering security requirements
- Step 15: Generate structural analysis models
- Step 16: Develop UML diagrams

As observed, MOSRE not only focuses on security requirements wording. It introduces UML and misuse case diagrams, along with structural analysis models for a better understanding of the context in which the web application operates.

Haley et al. (2006) present "a framework for security requirements elicitation and analysis, based upon the construction of a context for the system and satisfaction arguments for the security of the system". Four major activities are discussed in this framework (Haley et al., 2006):

- Activity 1: Identify functional requirements
- Activity 2: Identify appropriate security goals
    - Identify candidate assets
    - Identify harms (generate threat descriptions)
    - Apply management principles
    - Determine security goals
- Activity 3: Identify security requirements
- Activity 4: Verification of system context
    - Construct satisfaction arguments

In this framework, special attention is paid to the way in which a security requirement must be written. "Security requirements that express what is to happen in a given situation, as opposed to what is not ever to happen in any situation, would facilitate their analysis" (Haley et al., 2006). It goes even further, given that it defines security requirements, security goals, the importance of the context "within which the system and the potential attackers exist" (Haley et al., 2006). The satisfaction arguments serve as a way to determine "whether the security requirements satisfy the security goals and whether the system satisfies the security requirements" (Haley et al., 2006).

Moreover, there are risk analysis-based approaches as well, as stated by another comparison of SRE methods (Fabian et al., 2010). CORAS, for example, is a model-based

method for conducting security risk analysis, inspired by UML diagrams (Braber, Hogganvik, Lund, Stølen & Vraalsen, 2007). Its seven steps are (Braber et al., 2007; Fabian et al., 2010):

- Step 1: Introductory meeting between the representatives of the clients and the analysis team to set the goals, focus and scope of the analysis. Future meetings and workshops are planned.
- Step 2: High-level analysis – "the main purpose is to identify assets and get an overview of the main risks" (Braber et al., 2007, p. 103). Aspects from the introductory meeting are revised.
- Step 3: Approval – "the main goal is to finalize the documentation and characterization of target and assets, and get this formally approved by the client" (Braber et al., 2007, p. 106).
- Step 4: Workshop to identify threats and vulnerabilities.
- Step 5: Risk estimation – "the participants in the workshop provide likelihood estimates for each threat scenario in the threat diagrams" (Braber et al., 2007, p. 111).
- Step 6: Presentation of a first risk analysis to the client to apply corrections (Fabian et al., 2010, p. 30).
- Step 7: Risk treatment – "the risks that are not acceptable are all evaluated in order to find means to reduce their likelihood and/or consequence" (Braber et al., 2007, p. 113).

The formulation of software requirements by means of patterns instantiation has been a practice that has aroused interest among researchers since the last decade. Firesmith (2003a) introduced the idea of using "reusable parameterized templates for specifying security requirements". He argues security requirements must be asset-based and due to the fact assets may vary between software systems, it is of vital importance to apply the following procedure in order to fill the templates (Firesmith, 2003a):

- Step 1: Identify valuable assets
- Step 2: Identify threats
- Step 3: Identify likely attackers
- Step 4: Estimate vulnerability
- Step 5: Determine negative outcomes
- Step 6: Prioritize vulnerabilities
- Step 7: Identify relevant situation
- Step 8: Consider security subfactor
- Step 9: Identify relevant template(s)
- Step 10: Determine security criterion
- Step 11: Determine measure
- Step 12: Determine required level
- Step 13: Specify requirement

It is important to note that Firesmith does not recommend when to apply a particular template. The decision falls on the requirements engineer or the requirements and security teams. It does not provide a tool to instantiate them either.

On the other hand, Barcelos & Penteado (2017) did develop a set of functional requirements patterns and a computational support to specify and manage these patterns, as well as the instantiation of them during the elicitation phase with the client. "The requirements generated in concluded projects are stored in a local repository that allows the reuse of requirements in future projects, with the possibility of changing those requirements if necessary" (Barcelos & Penteado, 2017, p. 12).

In order to evaluate the approach, three case studies were conducted: two of them were carried out with Computer Science and Electrical Engineering undergraduate students at the Federal University of São Carlos (UFSCar) and the last one involved 34 students from an Information System course at the State University of Minas Gerais (UEMG). Participants were distributed into several groups. They were instructed to elicit requirements for a given scenario firstly without the patterns (ad hoc mode) and then redo their work with the computational support (case studies 1 and 3), while case study 2 students worked with the patterns immediately. A questionnaire was given to the students afterwards in which they were asked about their experience using the tool. Basing on the results, the use of the patterns with the aid of computational support had a positive impact on the completeness of the functional requirements the students elicited. Their opinions regarding the use of the tool were favorable.

Another security requirements templates approach (SD) was developed and tested by Maria Riaz and her research team (Riaz et al., 2014b). They conducted a controlled experiment at North Carolina State University (NCSU) with 50 graduate students which were part of a software security course. Students were distributed into control (no template suggestions) and treatment groups (with template suggestions by means of an online tool) and had to elicit security requirements for a software application in the health care domain. "The goal of this experiment is to determine whether the use of automatically-suggested security requirements templates leads to efficient and effective requirements elicitation when compared to a manual approach based on personal expertise" (Riaz et al., 2014b, p.3). Various researchers were involved in the creation of an oracle of security requirements before the start of the study. Significant differences were found in efficiency (requirements/minute) and coverage (related to the identification of requirements in the oracle) at $p < 0.05$. A survey was given to each student, right after finishing the activity, in which they were asked about their experience regarding this security requirement elicitation method.

Following the investigation of Riaz et al., three differentiated replications of her original experiment were conducted in order to evaluate if the original findings could create generalized knowledge across other scenarios and problem domains (Riaz et al.,

2017). Some of them added interesting variations: the suggestions of extraneous templates for treatment group participants and conducting the experiment as a take home activity.

These are the steps that must be followed in order to experiment with the SD methodology:

- Step 1: Identify the functional requirements of the software application.
- Step 2: Identify the security objectives related to each functional requirement.
- Step 3: Select the applicable security requirement templates, for each security objective present in the functional requirement.
- Step 4: Fill in the blank spaces of the template(s).

As for the security objectives, they receive many names in literature. Firesmith (2003a) mentions identification, authentication, authorization, immunity, integrity, intrusion detection, non-repudiation, "privacy (a.k.a., confidentiality)", security auditing, survivability and physical protection as *quality subfactors*. Haley et al. (2006) call them *security goals* and enumerate confidentiality, integrity, availability and accountability. Beckers et al. (2014) refer to them as *domains* and suggest the following for cloud computing systems: confidentiality, integrity, availability, authenticity, security management, non-repudiation, privacy, compliance and transparency.

3.2 – Necessary aspects for an empirical evaluation on the use of security requirements templates

Based on the experimental process of software engineering (Wohlin, Runeson, Höst, Ohlsson, Regnell & Wesslén, 2012) and considering the information gathered from the articles analyzed in the related work, the necessary aspects to conduct an empirical study in security templates are summarized in Table 1 and thoroughly explained in the following subsections.

Table 1 - Necessary aspects to conduct an empirical study in security templates

	Essential aspect	Example
Experiment planning		
Context selection	You should consider a context in which the application demands important security requirements.	Mobile banking and health (Riaz et al., 2017). University services (Barcelos & Penteado, 2017).
Hypothesis formulation	Measure the effectiveness on the use of a treatment that can improve the elicitation of security requirements, in terms of specific metrics.	What is the quality, coverage, relevance and efficiency of the security requirements identified by the use of security requirements templates? (Riaz et al., 2017).
Variables selection	It is important to determine the variables that will be taken into account in the investigation. The	Quality, coverage, relevance and efficiency (Riaz et al., 2017).

	treatment used to elicit security requirements has an impact on these variables.	
Selection of subjects	Security expertise must be considered when selecting subjects. Software/computer science students are not necessarily needed.	Computer science graduate students (Riaz et al., 2017). Computer science and electric engineering undergraduate students (Barcelos & Penteado, 2017).
Choice of design type	Research containing a factor and two treatments are recommended	Factor: security requirements elicitation method (Riaz et al., 2017). Treatment: with/without templates (Riaz et al., 2017).
Instrumentation	It has been observed on literature that instrumentation is prepared for the researcher and his/her participants.	Informed consent, information about the methodology, SRS of the software application, a questionnaire and an oracle (Riaz et al., 2017).
Validity evaluation	The researcher must ensure that the validity threats are evaluated and minimized. Not all threats may be avoided, but most of them can be mitigated at least (Wohlin et al., 2012).	Conclusion validity, internal validity, external validity and construct validity (Wohlin et al., 2012).
Experiment operation		
Preparation	The researcher must obtain consent from participants and let them know their personal information will be kept confidential (if any).	Prepare an informed consent for the participants to sign (Riaz et al., 2017). No personal data was asked (Riaz et al., 2017).
Execution	It is important to choose a place to conduct the experiment and track how much time each participant takes completing the activity.	University classroom (Riaz et al., 2017; Barcelos & Penteado, 2017). Assign the activity as homework (Riaz et al., 2017).
Data validation	The researcher must validate if the responses from participants are valid and complete. Incomplete data should be analyzed to decide if it must be taken out of the sample or not.	Not reported.
Analysis & interpretation		
Descriptive statistics	Descriptive statistics makes it easier to observe how data is distributed and notice abnormal data points (outliers).	Bar graphs, box plots and radar graphs have proven to be suitable to represent data graphically (Riaz et al., 2017).
Data set reduction	The researcher must identify outliers and decide whether or not to include them in the study.	Outliers were observed in the efficiency metric, but they were included as they did not affect the significance of results (Riaz et al., 2017). No data set reductions were observed in literature.
Hypothesis testing	Hypotheses must be tested in order if a certain treatment significantly impacts the independent variables.	Previous studies have used a 2x2 ANOVA for unbalanced groups, to test all of their null hypotheses (Riaz et al., 2017). Results with $p < 0.05$ are considered significant (Riaz et al., 2017).

The necessary aspects presented in Table 1 are explained in more detail in subsections 3.2.1, 3.2.2 and 3.2.3.

3.2.1 – Experiment planning

- Context selection

To carry out a security requirements templates experiment, the researcher must select the context of the software applications which will be objects of study and on which, based on their functional requirements, the explicit security requirements will be elicited. For example, previous studies do not limit security requirements templates experimentation to a specific context. The problem domain in previous investigations regarding the use of software requirements templates and/or patterns have been mobile banking and healthcare (Riaz et al., 2017).

- Hypothesis formulation

The researcher must define the hypotheses that he/she wants to test, based on the objective and research questions. An original study and three subsequent replications specify their null hypotheses indicating that the dependent variables of the study are unrelated to the use of security requirements templates (Riaz et al., 2017).

- Variables selection
    - Independent variables: security requirement elicitation technique (recommended templates vs. no recommended templates)
    - Dependent variables: quality, coverage, relevance (of the elicited security requirements) and efficiency (of the security requirements elicitation technique)

- Selection of subjects

The sample population must be representative of the population being studied and for which the researcher will draw conclusions based on the results obtained in the experiment. For the selection of participants, the studies made use of computer science graduate students attending the same university. Participants may also be involved in software development or software security (Riaz et al., 2017). In general, any IT professional would be suitable for this investigation. Having security professionals participate in the study would be the best option, but it is rare to find them. Another option would be to experiment with participants from other disciplines which have initial experience on requirements. This was the case of electrical engineering students who had developed simple systems and were asked to elaborate a requirements specification (Barcelos & Penteado, 2017).

The more students willing to participate in the study, the better conclusions can be withdrawn from the results.

Moreover, it was observed that most participants had entry-level work and academic security experience. During the elicitation process, they may work individually or in groups. For example, there were 42 students divided into 8 groups in one of the three case studies conducted in Brazil (Barcelos & Penteado, 2017). Other studies, on the contrary, were conducted with participants ranging from 16 to 107 graduate students in a single experiment and they all worked individually (Riaz et al., 2017).

- Choice of design type

Experiment designs may range from experiments with one factor with two treatments, to experiments having more than two factors and two treatments. Hypotheses must be formulated according to the design type selected by the researcher. This explains the importance of defining the design of the experiment. Previous studies, for example, have chosen to have one factor with two treatments (Riaz et al., 2017; Barcelos & Penteado, 2017). The purpose of experiments with this design type are to compare the two treatments against each other. Their design type would be set as follows:

  - One factor with two treatments.
    - Factor: security requirements elicitation method.
    - Treatments: with/without recommended templates.

- Instrumentation

Instrumentation plays an important role in security requirements templates investigations because if they are badly designed, the outcomes of the experiment would be impacted negatively. This becomes more important in an investigation in which participants need to learn and apply a new methodology in an unknown context. In this manner, it is recommendable to prepare reference material for the participants to study before the activity, and use during the activity as well. In a case study conducted at State University of Minas Gerais (in Brazil), students were given information regarding requirements concepts and how to obtain them (Barcelos & Penteado, 2017), before the start of the experiment. A similar trend was observed in a controlled experiment conducted at North Carolina State University, in which security requirements objectives and templates documentation was given to students (Riaz et al., 2014b) to revise 1 week before the experiment and they could revise it during the activity as well. They went even further giving the participants a 15-minute explanatory presentation in English, to clarify any doubts before the start of the experiment (Riaz et al., 2014b, Riaz et al., 2017).

Another aspect to take into account is the elaboration of a SRS or use cases for the software application to which security requirements will be elicited, in case there is no access to its original SRS. Riaz et al., (2017) presented use cases to their participants.

An oracle (for the researchers to compare the "real answer" with the responses given by participants) must be prepared before the start of the experiment. The oracle may be a list of previously elicited security requirements for the software application, or the list of applicable security requirements templates (Riaz et al., 2017).

Additionally, there must be a means to collect and process the information from the participants. For example, an online site was prepared for participants to type in their answers which stored them in a database for future analysis (Riaz et al., 2017). Traditional data collection artifacts (forms) may be used as well.

Finally, another trend in some studies is to create a survey to be answered by the participants, after the activity, in order to gather feedback related to the experiment (Barcelos & Penteado, 2017; Riaz et al., 2017).

- Validity evaluation

The researcher must ensure that the validity threats are evaluated and minimized. Not all threats may be avoided, but most of them can be mitigated at least. Four main validity threats categories must be taken into account (Wohlin et al., 2012):

  - Conclusion validity: "Threats to the conclusion validity are concerned with issues that affect the ability to draw the correct conclusion about relations between the treatment and the outcome of an experiment" (Wohlin et al., 2012, p.104).

  - Internal validity: "Threats to internal validity are influences that can affect the independent variable with respect to causality, without the researcher's knowledge. Thus they threat the conclusion about a possible causal relationship between treatment and outcome" (Wohlin et al., 2012, p.106).

  - Construct validity: "Construct validity concerns generalizing the result of the experiment to the concept or theory behind the experiment. Some threats relate to the design of the experiment, others to social factors" (Wohlin et al., 2012, p.108).

  - External validity: "Threats to external validity are conditions that limit our ability to generalize the results of our experiment to industrial practice. There are three types of interactions with the treatment: people, place and time" (Wohlin et al., 2012, p.110).

Two main validity threats (Riaz et al., 2017) may be categorized under conclusion validity. The first one is the number of participants in the study. Reliable conclusions may not be drawn from the study if there is a limited number of participants willing to be part of the experiment. For example, previous studies had a participant count ranging from 16 to 107, per experiment (Riaz et al., 2017). The second one relates to the reliability of measures: ideally, "objective measures (that can be repeated with the same outcome) are more reliable than subjective measures" (Wohlin et al., 2012, p.110). In one replicated experiment conducted at home, time tracking might not have been as efficient as if the activity was done in a classroom and supervised by the researcher.

Moreover, two relevant internal validity threats may take place in security requirements templates experiments (Riaz et al., 2017). The first one is called "diffusion or imitation of treatments" (Wohlin et al., 2012, p.108). Experiments which do not take place at the university campus (without the supervision of a researcher) are subject to this threat. For example, the University of Trento conducted a replicated experiment at home and this could may have led to control students knowing about the suggested templates given to students in the treatment control. The second internal validity threat falls under "interaction with selection" (Wohlin et al., 2012, p.107). Students in treatment groups may just blindly accept all template suggestions given to them (Riaz et al., 2017). One way to mitigate this risk is by using extraneous templates, so that treatment students have to decide whether to apply a template or not. Another internal threat validity falls under "testing and training" (Riaz et al., 2017, p.2170). Control students are in "disadvantage" because they do not receive suggested templates during the activity. In order to reduce this risk, control group participants are allowed to use the reference material during the experiment.

On the other hand, one external validity threat (experimental constraints that limit realism) affects this type of experiments because "participants used a limited amount of time to complete the task, which may affect the quality and coverage of identified security requirements." (Riaz et al., 2017, p.2171). More time may imply better results and it would be ideal to recreate an industry-like requirements elicitation process.

As for the construct validity threats, it is vital to mitigate hypothesis guessing (Riaz et al, 2017). The performance of students may be biased if they start wondering that is the purpose of the study.

3.2.2 – Experiment operation

- Preparation

  It is important to obtain consent from the participants before starting the experiment (Riaz et al., 2017). If personal information is collected, participants

must be told it will be kept confidential. No personal information was collected in various studies (Riaz et al., 2017; Barcelos & Penteado, 2017).

- Execution

The experiments can be executed in different ways. Students can be asked to gather one single time in a university's classroom to conduct the activity (Riaz et al., 2017; Barcelos & Penteado, 2017). Another alternative is to assign the activity as homework (Riaz et al., 2017). The quantity of time given to students to finish the assignment may vary: some researchers gave students 60 minutes, while others gave 180 minutes to finish the task (Riaz et al., 2017). In order to analyze if motivation lead to better performance, one of the experiments was planned as a take-home activity and they had 1 week to finish it (Riaz et al. (2017). In all cases, participants must hand in their work as soon as they are finished, or when time is up.

If the efficiency of the security requirements elicitation method is going to be measured, it is advisable to record the time every participant takes to complete the activity (Riaz et al., 2014b). If the activity uses manual data collection artifacts at home, participants must track their time themselves and write down how much time they took in their answer sheet. If the activity is conducted in a classroom, both the researcher and the student can write down how much they lasted. If the researcher has created a digital data collection artifact (for example, an online site), elapsed time may be automatically recorded as a timestamp (Riaz et al., 2017).

- Data validation

Once the data has been collected, the researcher must validate if the responses from participants are valid and complete. Incomplete data should be analyzed to decide if it must be taken out of the sample or not (Riaz et al., 2017).

3.2.3 – Analysis & interpretation

- Descriptive statistics

Descriptive statistics makes it easier to observe how data is distributed and notice abnormal data points (outliers). Bar graphs, box plots and radar graphs have proven to be suitable to represent data graphically (Riaz et al., 2017). The four metrics used for evaluating participants' responses (quality, coverage, relevance and efficiency) can be represented using these type of graphs.

- Data set reduction

  The researcher must identify outliers and decide whether or not to include them in the study. No data set reductions were observed in literature.

- Hypothesis testing

  “The objective of hypothesis testing is to see if it is possible to reject a certain null hypothesis” (Wohlin et al., 2012, p.132). Previous studies have used a 2x2 ANOVA for unbalanced groups, to test all of their null hypotheses (Riaz et al., 2017). Results with $p < 0.05$ are considered significant.

These identified aspects were considered to propose the empirical study detailed in Chapter 5.

Chapter 4. Security requirements specification

This chapter describes the requirements engineering process activities followed to elaborate the security requirements specification document for NutriMetas health mobile application. The document focuses on the specification of non-functional security requirements. The specification of functional requirements was created as part of the project on which this study is based (Jensen, Quesada-López, Zúñiga, Chinnock & Jenkins, 2015). It was updated, revised and approved by the project stakeholders.

4.1 – Requirements elicitation

Originally, there was a functional requirements specification for NutriMetas, based on the IEEE 830-1998 standard. We took this specification and updated it to the new ISO/IEC/IEEE 29148:2011 standard.

As for the security requirements elicitation process for NutriMetas, it started with a close reading of the updated software requirement specification. It contains 14 functional requirements that explain the functionality of this mobile application. Once all the functional requirements were understood, we started to apply the SD methodology.

Firstly, we identified the security objectives present in all of the 14 functional requirements. Each of the identified security objectives suggests the need to apply one or more security requirements templates to solve the need for security for that specific functional requirement. Therefore, our second step was to select applicable security requirements templates for each functional requirement, based on the previously identified security objectives. These templates include blank spaces that must be filled in with wording related to the functional requirement. Consequently, the third step consisted of filling these templates according to the context of each functional requirement. A total of 93 templates cover the need for security for NutriMetas' 14 functional requirements.

4.2 - Requirements analysis

The new functional requirements specification was revised and approved by NutriMetas' project stakeholders. The elicited security requirements received their approval as well.

This phase allowed us to count with clear and consistent requirements. This functional requirements specification is going to be handed to the participants for them to elicit security requirements following the SD methodology.

4.3 – Requirements specification

In software engineering, a requirements specification refers to "the production of a document that can be systematically reviewed, evaluated, and approved" (Bourque & Fairley, 2014, p.10). It "establishes the basis for agreement between customer and contractors or suppliers (in market-driven projects, these roles may be played by the marketing and development divisions) on what the software product is to do as well as what it is not expected to do" (Bourque & Fairley, 2014, p.11). The requirements elicitation phase generated a security requirements specification for NutriMetas. It applies 93 security requirements templates, as shown on Table 2, to meet the needs of security of its 14 functional requirements. These 93 templates were filled in and 213 security requirements were derived.

Table 2 – Recommended security requirements templates for NutriMetas' functional requirements

Functional requirement	Recommended templates
R0	C1 – authorized access, C2 – confidentiality during storage, C3 – confidentiality during transmission, I2 – maintaining integrity during write-type actions, IA2 – unique accounts, AY2 – logging authentication events
R1	C1, C2, C3, I1 – maintaining integrity during read-type actions, I2
R2	C1, C2, C3, I1, I2
R3	C1, C3, I1, A2 – maintaining appropriate response time, A3 – maintaining service availability, A5 – limit resources to ensure capacity and performance, AY1 – logging transactions with sensitive data, PR1 – usage of personal information
R4	C1, C2, C3, I1, I2, AY1
R5	C1, C2, C3, I1, I2, AY1
R6	C1, C3, I1, A2, A3, A5, AY1, PR1
R7	C1, C2, C3, I2, A3

R8	C3, I1, A3
R9	C1, C3, I1, A2, A3, A5, AY1, PR1
R10	C1, C3, I1, A2, A3, A5, AY1, PR1
R11	C1, C3, I1, A2, A3, A5, AY1, PR1
R12	C1, C3, I1, A2, A3, A5, AY1, PR1
R13	C1, C2, C3, I1, I2, I4, A3, AY1, PR1

4.3.1 – Security requirements specification document for NutriMetas' mobile application

This subsection contains the security requirements specification for NutriMetas. For each of the functional requirements, the identified security objectives are mentioned and under each objective, the security requirements generated with each template are displayed. The bold text represents what the researcher completed in each template.

(R0) - User activation

Security objective: confidentiality

1. C1 - authorized access
    - The system shall enforce access privileges that **enable the nutritionist** to **create** a **new patient-type user**.
2. C2 - confidentiality during storage
    - The system shall encrypt **user activation information** and store **user activation information** in encrypted format using an industry-approved encryption algorithm.
    - The system shall monitor the status and location of system components that may contain unencrypted **user activation information** data.
3. C3 - confidentiality during transmission
    - The system shall transmit the **information regarding the activation of a patient-type user** in encrypted format to and from the authorized **user**.

Security objective: integrity

4. I2 - maintaining integrity during write-type actions
    - The system shall ensure that all mandatory information is provided for the **activation of a patient-type user** before **its creation**.
    - The system shall protect against loss of information during **the activation of a patient-type user**.
    - The system shall have provision to report errors in **the activation of a patient-type user** during **its creation**.

- The system shall have provision to report errors in **the activation of a patient-type user** after **its creation**.
- The system shall have provision to correct errors in **the activation of a patient-type user** if errors are detected.
- The system shall ensure synchronization of **the activation of a patient-type user** if multiple **nutritionists** can perform **the activation** on **user information** simultaneously.

Security objective: identification & authentication

5. IA2 - unique accounts

   - Each user should be assigned a unique identifier that can be used for the purpose of authentication.

Security objective: accountability

6. AY2 - logging authentication events

   - The system shall log every time the patient logs into and logs out of the system.
   - At a minimum, the system shall capture the following information for the log entry: **user** identification, timestamp, **activation**, **the patient-type user**, and identification of the owner of **the patient user**.

(R1) - Nutritional record configuration (configuración del expediente)

Security objective: confidentiality

1. C1 - authorized access

   - The system shall enforce access privileges that **enable the nutritionist** to **create** a **new patient-type user**.

2. C2 - confidentiality during storage

   - The system shall encrypt **user activation information** and store **user activation information** in encrypted format using an industry-approved encryption algorithm.
   - The system shall monitor the status and location of system components that may contain unencrypted **user activation information** data.

3. C3 - confidentiality during transmission

   - The system shall transmit the **information regarding the activation of a patient-type user** in encrypted format to and from the authorized **user**.

Security objective: integrity

4. I1- maintaining integrity during read-type actions

   - The system shall ensure consistent understanding of **the patient's nutritional record** by the **user** during **its configuration**.
   - The system shall have provision to report errors in **the patient's nutritional record** during **its configuration**.

5. I2 - maintaining integrity during write-type actions

    - The system shall ensure that all mandatory information is provided for the **patient's nutritional record** before **its configuration**.
    - The system shall protect against loss of information during **its configuration**.
    - The system shall have provision to report errors in **the patient's nutritional record** during **its configuration**.
    - The system shall have provision to report errors in **the patient's nutritional record** after **its configuration**.
    - The system shall have provision to correct errors in **the patient's nutritional record** if errors are detected.
    - The system shall ensure synchronization of **the patient's nutritional record** if multiple **nutritionists** can perform the **configuration** on **the patient's nutritional record** simultaneously.

(R2) - Health goals configuration (configuración de metas de salud)

Security objective: confidentiality

1. C1 - authorized access

    - The system shall enforce access privileges that **enable** the **nutritionist** to **configure** the **patient's health goals.**

2. C2 - confidentiality during storage

    - The system shall encrypt **the patient's health goals** and store **the patient's health goals** in encrypted format using an industry-approved encryption algorithm.
    - The system shall monitor the status and location of system components that may contain unencrypted **patient's health goals** data.

3. C3 - confidentiality during transmission

    - The system shall transmit **the patient's health goals** in encrypted format to and from the authorized users.

Security objective: integrity

4. I1- maintaining integrity during read-type actions

    - The system shall ensure consistent understanding of the patient's health goals by the user during its configuration.
    - The system shall have provision to report errors in **the patient's health goals** during **their configuration (insertion)**.

5. I2 - maintaining integrity during write-type actions

    - The system shall ensure that all mandatory information is provided for the **patient's health goals** before **their configuration (insertion)**.
    - The system shall protect against loss of information during the **configuration (insertion)**.
    - The system shall have provision to report errors in the **patient's health goals** during the **configuration (insertion)**.

- The system shall have provision to report errors in the **patient's health goals** after the **configuration (insertion)**.
- The system shall have provision to correct errors in **the patient's health goals** if errors are detected.
- The system shall ensure synchronization of **the patient's health goals** if multiple **nutritionists** can perform configurations on **the patient's health goals** simultaneously.

(R3) – Patient access to his/her nutritional record

Security objective: confidentiality

1. C1 - authorized access

    - The system shall enforce access privileges that **enable the patient** to **access** his/her **nutritional record**.

2. C3 - confidentiality during transmission

    - The system shall transmit the **patient's nutritional record** in encrypted format to and from the authorized **patient**.

Security objective: integrity

3. I1- maintaining integrity during read-type actions

    - The system shall ensure consistent understanding of **the patient's nutritional record** by the **patient** when **viewing**.
    - The system shall have provision to report errors in **the patient's nutritional record** when **viewing**.

Security objective: availability

4. A2- maintaining appropriate response time

    - The system shall respond to user access **to his/her nutritional record** within **5 seconds**.
    - The system shall provide monitoring capabilities to ensure that response time falls within **5 seconds**.

5. A3- maintaining service availability

    - The system shall ensure **nutritional record viewing** to user **access request** 24 by 7.
    - The system shall provide monitoring capabilities to ensure that **nutritional record viewing** is available to users / other system components 24 by 7.

6. A3- limit resources to ensure capacity and performance

    - The system shall limit the number of records presented to the user for **its nutritional record** when the request limit of **100 results** is reached.
    - The system shall notify the user that the maximum number of records was returned.

Security objective: accountability

7. AY1 - logging transactions with sensitive data

    - The system shall log every time the **user (patient or nutritionist) checks the nutritional record**.
    - At a minimum, the system shall capture the following information for the log entry: **user** identification, timestamp, **check**, the **nutritional record**, and identification of the owner of **the nutritional record**.

Security objective: privacy

8. PR1 - usage of personal data

    - The system shall allow the owner of **the nutritional record** to be notified of potential authorized uses of **the nutritional record**.
    - The system shall allow the owner of **the nutritional record** to be notified when the **nutritional record** is accessed by **the user (patient or nutritionist)**.
    - The system shall have the ability to get consent from the owner of **the nutritional record** before accessing **the nutritional record** for authorized use.
    - The system shall allow access to **the nutritional record** for authorized use without prior consent only under exceptional circumstances as defined by applicable privacy policy.
    - The system shall have provision for the owner of **the nutritional record** to withdraw consent for accessing **the patient's nutritional record**.
    - The system shall have provision for de-identifying the **nutritional record** before access by **nutritionists**, if required.

(R4) - Modification of patient's personal information

Security objective: confidentiality

1. C1 - authorized access

    - The system shall enforce access privileges that **enable the patient** to **modify** his personal information.

2. C2 - confidentiality during storage

    - The system shall encrypt **the patient's personal information** and store **the patient's personal information** in encrypted format using an industry-approved encryption algorithm.
    - The system shall monitor the status and location of system components that may contain unencrypted **patient's personal information** data.

3. C3 - confidentiality during transmission

    - The system shall transmit **patient's personal information** in encrypted format to and from authorized **users**.

Security objective: integrity

4. I1- maintaining integrity during read-type actions

    - The system shall ensure consistent understanding of **the patient's personal information** by the **user (patient and nutritionist)** when **viewing**.
    - The system shall have provision to report errors in **the patient's personal information** when **viewing**.

5. I2- maintaining integrity during write-type actions

    - The system shall ensure that all mandatory information is provided for the **patient's personal information** before **modification**.
    - The system shall protect against loss of information during **the modification**.
    - The system shall have provision to report errors in **the patient's personal information** during **the modification**.
    - The system shall have provision to report errors in **the patient's personal information** after **the modification**.
    - The system shall have provision to correct errors in **the patient's personal information** if errors are detected.
    - The system shall ensure synchronization of **the patient's personal information** if multiple **users** can perform **modifications** on **the patient's personal information** simultaneously.

Security objective: accountability

6. AY1 - logging transactions with sensitive data

    - The system shall log every time **a user (patient or nutritionist) modifies the patient's personal information**.
    - At a minimum, the system shall capture the following information for the log entry: **user (patient or nutritionist)**, identification, timestamp, **modification**, **patient's personal information**, and identification of the owner of **the personal information**.

(R5) - Add nutritional data along with the nutritionist

Security objective: confidentiality

1. C1 - authorized access

    - The system shall enforce access privileges that **enable the patient** to add **his/her nutritional data**.

2. C2 - confidentiality during storage

    - The system shall encrypt **the patient's nutritional data** and store **the patient's nutritional data** in encrypted format using an industry-approved encryption algorithm.
    - The system shall monitor the status and location of system components that may contain unencrypted **patient's nutritional data**.

3. C3 - confidentiality during transmission

    - The system shall transmit **patient's nutritional data** in encrypted format to and from the authorized **user**.

Security objective: integrity

4. I1- maintaining integrity during read-type actions

    - The system shall ensure consistent understanding of **the patient's nutritional data** by the **user** during **sending**.
    - The system shall have provision to report errors in **the patient's nutritional data** during **sending**.

5. I2 - maintaining integrity during write-type actions

    - The system shall ensure that all mandatory information is provided for the **patient's nutritional data** before **its insertion**.
    - The system shall protect against loss of information during the **insertion**.
    - The system shall have provision to report errors in **the patient's nutritional data** during **its insertion**.
    - The system shall have provision to report errors in <resource> after **insertion**.
    - The system shall have provision to correct errors in **the patient's nutritional data** if errors are detected.
    - The system shall ensure synchronization of **the patient's nutritional data** if multiple **users** can perform **updating** on **the patient's nutritional data** simultaneously.

Security objective: accountability

6. AY1 - logging transactions with sensitive data

    - The system shall log every time the **user updates the patient's nutritional data**.
    - At a minimum, the system shall capture the following information for the log entry: **user** identification, timestamp, **update**, **the patient's nutritional data**, and identification of the owner of **the nutritional data**.

(R6) – View patient's healthy eating goals

Security objective: confidentiality

1. C1 - authorized access

    - The system shall enforce access privileges that **enable the patient** to access **his/her healthy eating goals**.

2. C3 - confidentiality during transmission

    - The system shall transmit the **healthy eating goals** in encrypted format to and from the authorized **users (patient or nutritionist)**.

Security objective: integrity

3. I1- maintaining integrity during read-type actions

    - The system shall ensure consistent understanding of **the healthy eating goals** by the **user** when **viewing**.
    - The system shall have provision to report errors in **the healthy eating goals** when **viewing**.

Security objective: availability

4. A2- maintaining appropriate response time

    - The system shall respond to user **access to his/her healthy eating goals** within **5 seconds**.
    - The system shall provide monitoring capabilities to ensure that response time falls within **5 seconds**.

5. A3- maintaining service availability

    - The system shall ensure **access to healthy eating goals** to the user **24 by 7**.
    - The system shall provide monitoring capabilities to ensure that **the access to healthy eating goals** is available to users / other system components **24 by 7**.

6. A5- limit resources to ensure capacity and performance

    - The system shall limit the number of records presented to the user for **the healthy eating goals** when the request limit of **25 results** is reached.
    - The system shall notify the user that the maximum number of records was returned.

Security objective: accountability

7. AY1 - logging transactions with sensitive data

    - The system shall log every time the user **reads his/her healthy eating goals**.
    - At a minimum, the system shall capture the following information for the log entry: **user** identification, timestamp, **read/access**, **healthy eating goals**, and identification of the owner of **these healthy eating goals**.

Security objective: privacy

8. PR1 - usage of personal data

    - The system shall allow the owner of **the healthy eating goals** to be notified of potential authorized uses of **the healthy eating goals**.
    - The system shall allow the owner of **the healthy eating goals** to be notified when the **healthy eating goals** is **accessed/read** by **the user (patient or nutritionist)**.

- The system shall have the ability to get consent from the owner of **the healthy eating goals** before accessing **the healthy eating goals** for authorized use.
- The system shall allow access to **the healthy eating goals** for authorized use without prior consent only under exceptional circumstances as defined by applicable privacy policy.
- The system shall have provision for the owner of **the healthy eating goals** to withdraw consent for **accessing healthy eating goals**.
- The system shall have provision for de-identifying the **healthy eating goals** before **access** by **nutritionists**, if required.

(R7) - Record daily activity of healthy eating goals

Security objective: confidentiality

1. C1 - authorized access

    - The system shall enforce access privileges that **enable the patient** to **add/record his/her daily activity of healthy eating goals**.

2. C2 - confidentiality during storage

    - The system shall encrypt **the daily activity of healthy eating goals** and store **the daily activity of healthy eating goals** in encrypted format using an industry-approved encryption algorithm.
    - The system shall monitor the status and location of system components that may contain unencrypted **daily activity of healthy eating goals** data.

3. C3 - confidentiality during transmission

    - The system shall transmit **the daily activity of healthy eating goals** in encrypted format to and from the authorized **users**.

Security objective: integrity

4. I2 - maintaining integrity during write-type actions

    - The system shall ensure that all mandatory information is provided for the **daily activity of healthy eating goals** before **their insertion**.
    - The system shall protect against loss of information during **the insertion**.
    - The system shall have provision to report errors in **the daily activity of healthy eating goals** during **their insertion**.
    - The system shall have provision to report errors in **the daily activity of healthy eating goals** after **their insertion**.
    - The system shall have provision to correct errors in **the daily activity of healthy eating goals** if errors are detected.
    - The system shall ensure synchronization of **the daily activity of healthy eating goals** if multiple **users** can perform <action> on **the daily activity of healthy eating goals** simultaneously.

Security objective: availability

5. A3- maintaining service availability

- The system shall ensure **daily activity of healthy eating goals insertion** to **the patient 24 by 7**.
- The system shall provide monitoring capabilities to ensure that **the daily activity of healthy eating goals insertion** is available to users / other system components **24 by 7**.

(R8) - Reminder for the patient to register daily goal achievements

Security objective: confidentiality

1. C3 - confidentiality during transmission

- The system shall transmit **the reminder for the patient to register his/her daily goals** in encrypted format to and from the authorized **patient**.

Security objective: integrity

2. I1- maintaining integrity during read-type actions

- The system shall ensure consistent understanding of **the reminder to register daily goals** by the **patient** when **viewing**.
- The system shall have provision to report errors in **the reminder to register daily goals** when **viewing**.

Security objective: availability

3. A3- maintaining service availability

- The system shall ensure **the displaying of the reminder to register daily goals** to **the patient 24 by 7**.
- The system shall provide monitoring capabilities to ensure that **the displaying of the reminder to register daily goals** is available to users / other system components **24 by 7**.

(R9) – View health goals achievement report

Security objective: confidentiality

1. C1 - authorized access

- The system shall enforce access privileges that **enable the patient** to **view his/her health goal achievement report**.

2. C3 - confidentiality during transmission

- The system shall transmit **the health goals achievement report** in encrypted format to and from the authorized **user (patient and nutritionist)**.

Security objective: integrity

3. I1- maintaining integrity during read-type actions

    - The system shall ensure consistent understanding of **the health goals achievement report** by the **user** when **viewing**.
    - The system shall have provision to report errors in **the health goals achievement report** when **viewing**.

Security objective: availability

4. A2- maintaining appropriate response time

    - The system shall respond to user **request to view the health goals achievement report** within **5 seconds**.
    - The system shall provide monitoring capabilities to ensure that response time falls within **5 seconds**.

5. A3- maintaining service availability

    - The system shall ensure **the displaying of the health goal achievement report** to **the user 24 by 7**.
    - The system shall provide monitoring capabilities to ensure that **the health goal achievement report** is available to users / other system components **24 by 7**.

6. A5- limit resources to ensure capacity and performance

    - The system shall limit the number of records presented to the user for **health goal results** when the request limit of **25 results** is reached.
    - The system shall notify the user that the maximum number of records was returned.

Security objective: accountability

7. AY1 - logging transactions with sensitive data

    - The system shall log every time **the user views the health goal achievement report**.
    - At a minimum, the system shall capture the following information for the log entry: **user** identification, timestamp, **view/access**, **health goal achievement report**, and identification of the owner of **the health goal achievement report**.

Security objective: privacy

8. PR1 - usage of personal data

    - The system shall allow the owner of **the health goal achievement report** to be notified of potential authorized uses of **the health goal achievement report**.

- The system shall allow the owner of **the health goal achievement report** to be notified when the **health goal achievement report** is **accessed/viewed** by **the user (patient or nutritionist)**.
- The system shall have the ability to get consent from the owner of **the health goal achievement report** before accessing **the health goal achievement report** for authorized use.
- The system shall allow access to **the health goal achievement report** for authorized use without prior consent only under exceptional circumstances as defined by applicable privacy policy.
- The system shall have provision for the owner of **the health goal achievement report** to withdraw consent for **accessing the health goal achievement report**.
- The system shall have provision for de-identifying the **health goal achievement report** before **access** by **nutritionists**, if required.

(R10) – View report of patient's obtained medals by goal accomplishments

Security objective: confidentiality

1. C1 - authorized access

    - The system shall enforce access privileges that **enable the patient** to **view his/her report of obtained medals by goal accomplishments**.

2. C3 - confidentiality during transmission

    - The system shall transmit **the report of obtained medals by goal accomplishments** in encrypted format to and from the authorized **patient**.

Security objective: integrity

3. I1- maintaining integrity during read-type actions

    - The system shall ensure consistent understanding of **the report of obtained medals by goal accomplishments** by **the patient** when **viewing**.
    - The system shall have provision to report errors in **the report of obtained medals by goal accomplishments** when **viewing**.

Security objective: availability

4. A2- maintaining appropriate response time

    - The system shall respond to user **access to the report of obtained medals by goal accomplishments** within **5 seconds**.
    - The system shall provide monitoring capabilities to ensure that response time falls within **5 seconds**.

5. A3- maintaining service availability

    - The system shall ensure user **access to the report of obtained medals by goal accomplishments** to **the patient 24 by 7**.
    - The system shall provide monitoring capabilities to ensure that **the access to the report of obtained medals by goal accomplishments** is available to users / other system components **24 by 7**.

6. A5- limit resources to ensure capacity and performance

    - The system shall limit the number of records presented to the user for **the obtained medals by goal accomplishments** when the request limit of **25 results** is reached.
    - The system shall notify the user that the maximum number of records was returned.

Security objective: accountability

7. AY1 - logging transactions with sensitive data

    - The system shall log every time the **users view the report of obtained medals by goal accomplishments**.
    - At a minimum, the system shall capture the following information for the log entry: **user** identification, timestamp, **view/access**, **the report of obtained medals by goal accomplishments**, and identification of the owner of **the report of obtained medals by goal accomplishments**.

Security objective: privacy

8. PR1 - usage of personal data

    - The system shall allow the owner of **the report of obtained medals by goal accomplishments** to be notified of potential authorized uses of **the report of obtained medals by goal accomplishments**.
    - The system shall allow the owner of **the report of obtained medals by goal accomplishments** to be notified when the **report of obtained medals by goal accomplishments** is **viewed/accessed** by **the user (patient and nutritionist)**.
    - The system shall have the ability to get consent from the owner of **the report of obtained medals by goal accomplishments** before accessing **the report of obtained medals by goal accomplishments** for authorized use.
    - The system shall allow access to **the report of obtained medals by goal accomplishments** for authorized use without prior consent only under exceptional circumstances as defined by applicable privacy policy.
    - The system shall have provision for the owner of **the report of obtained medals by goal accomplishments** to withdraw consent for **accessing the report of obtained medals by goal accomplishments**.
    - The system shall have provision for de-identifying the **report of obtained medals by goal accomplishments** before **access** by **nutritionists**, if required.

(R11) – View report of patient's last registered anthropometric data

Security objective: confidentiality

1. C1 - authorized access

    - The system shall enforce access privileges that **enable the patient** to **view his/her report of last registered anthropometric data**.

2. C3 - confidentiality during transmission

    - The system shall transmit **the report of last registered anthropometric data** in encrypted format to and from the authorized **user**.

Security objective: integrity

3. I1- maintaining integrity during read-type actions

    - The system shall ensure consistent understanding of **the report of last registered anthropometric data** by the **users** when **viewing**.
    - The system shall have provision to report errors in **the report of last registered anthropometric data** when **viewing**.

Security objective: availability

4. A2- maintaining appropriate response time

    - The system shall respond to **the user's request to access the report of last registered anthropometric data** within **5 seconds**.
    - The system shall provide monitoring capabilities to ensure that response time falls within **5 seconds**.

5. A3- maintaining service availability

    - The system shall ensure **the access to the report of last registered anthropometric data** to **the user** 24 by 7.
    - The system shall provide monitoring capabilities to ensure that **the access to the report of last registered anthropometric data** is available to users / other system components **24 by 7**.

6. A5- limit resources to ensure capacity and performance

    - The system shall limit the number of records presented to the user for **the last registered anthropometric data** when the request limit of **25 results** is reached.
    - The system shall notify the user that the maximum number of records was returned.

Security objective: accountability

7. AY1 - logging transactions with sensitive data

    - The system shall log every time the user **views/accesses the report of last registered anthropometric data**.
    - At a minimum, the system shall capture the following information for the log entry: **user** identification, timestamp, **access**, **report of last registered anthropometric data**, and identification of the owner of **the report of last registered anthropometric data**.

Security objective: privacy

8. PR1 - usage of personal data

    - The system shall allow the owner of **the report of last registered anthropometric data** to be notified of potential authorized uses of **the report of last registered anthropometric data**.
    - The system shall allow the owner of **the report of last registered anthropometric data** to be notified when the **report of last registered anthropometric data** is accessed by **the user**.
    - The system shall have the ability to get consent from the owner of **the report of last registered anthropometric data** before accessing **the report of last registered anthropometric data** for authorized use.
    - The system shall allow access to **the report of last registered anthropometric data** for authorized use without prior consent only under exceptional circumstances as defined by applicable privacy policy.
    - The system shall have provision for the owner of **the report of last registered anthropometric data** to withdraw consent for accessing **the report of last registered anthropometric data**.
    - The system shall have provision for de-identifying the **report of last registered anthropometric data** before access by the nutritionist, if required.

(R12) – View report of patient's biochemical data

Security objective: confidentiality

1. C1 - authorized access

    - The system shall enforce access privileges that **enable** the **patient** to **access his/her biochemical report**.

2. C3 - confidentiality during transmission

    - The system shall transmit **the patient's biochemical report** in encrypted format to and from the authorized **patient**.

Security objective: integrity

3. I1- maintaining integrity during read-type actions

    - The system shall ensure consistent understanding of **the patient's biochemical report** by the **user** when **viewing**.
    - The system shall have provision to report errors in **the patient's biochemical report** when **viewing**.

Security objective: availability

4. A2- maintaining appropriate response time

    - The system shall respond to the **user's request to access the patient's biochemical report** within **5 seconds**.
    - The system shall provide monitoring capabilities to ensure that response time falls within **5 seconds**.

5. A3- maintaining service availability
    - The system shall ensure **access to the patient's biochemical report** to the **user 24 by 7**.
    - The system shall provide monitoring capabilities to ensure that **the access to the patient's biochemical report** is available to users / other system components **24 by 7**.

6. A5- limit resources to ensure capacity and performance

    - The system shall limit the number of records presented to the user for **the patient's biochemical report** when the request limit of **25 results** is reached.
    - The system shall notify the user that the maximum number of records was returned.

Security objective: accountability

7. AY1 - logging transactions with sensitive data

    - The system shall log every time **the user accesses/views the patient's biochemical report**.
    - At a minimum, the system shall capture the following information for the log entry: **user** identification, timestamp, **access**, **the patient's biochemical report**, and identification of the owner of **the patient's biochemical report**.

Security objective: privacy

8. PR1 - usage of personal data

    - The system shall allow the owner of **the biochemical report** to be notified of potential authorized uses of **the biochemical report**.
    - The system shall allow the owner of **the biochemical report** to be notified when **the biochemical report** is **accessed** by **the user**.

- The system shall have the ability to get consent from the owner of **the biochemical report** before accessing **the biochemical report** for authorized use.
- The system shall allow access to **the biochemical report** for authorized use without prior consent only under exceptional circumstances as defined by applicable privacy policy.
- The system shall have provision for the owner of **the biochemical report** to withdraw consent for **accessing the biochemical report**.
- The system shall have provision for de-identifying **the biochemical report** before access by the **nutritionist**, if required.

(R13) – Sending the patient's nutritional registered data (goal achievements and daily nutritional progress, among others) to the nutritionist's email.

Security objective: confidentiality

1. C1 - authorized access

    - The system shall enforce access privileges that **enable** the **patient** to **send his/her registered nutritional data**.

2. C2 - confidentiality during storage

    - The system shall encrypt **the patient's nutritional registered data** and store **the patient's nutritional registered data** in encrypted format using an industry-approved encryption algorithm.
    - The system shall monitor the status and location of system components that may contain unencrypted **patient's nutritional registered** data.

3. C3 - confidentiality during transmission

    - The system shall monitor the status and location of system components that may contain unencrypted **patient's nutritional registered** data.
    - The system shall transmit **the patient's nutritional registered data** in encrypted format to and from the authorized **users**.

Security objective: integrity

4. I1- maintaining integrity during read-type actions

    - The system shall ensure consistent understanding of **the patient's nutritional registered data** by the **patient** during **display**.
    - The system shall have provision to report errors in **the patient's nutritional registered data** during **display**.

5. I2 - maintaining integrity during write-type actions

    - The system shall ensure that all mandatory information is provided for the **patient's nutritional registered data** before **sending**.
    - The system shall protect against loss of information during **sending**.
    - The system shall have provision to report errors in **the patient's nutritional registered data** during **sending**.

- The system shall have provision to report errors in **the patient's nutritional registered data** after **sending**.
- The system shall have provision to correct errors in **the patient's nutritional registered data** if errors are detected.
- The system shall ensure synchronization of **the patient's nutritional registered data** if multiple **users** can perform **sending** on **the patient's nutritional registered data** simultaneously.

6. I4 - maintaining integrity of unchangeable resources

    - The system shall not allow modifications of **the patient's nutritional registered data** by any user.

Security objective: availability

7. A3- maintaining service availability

    - The system shall ensure **the sending of the patient's nutritional registered data** to the **nutritionist 24 by 7**.
    - The system shall provide monitoring capabilities to ensure that **the sending of the patient's nutritional registered data** is available to users / other system components **24 by 7**.

Security objective: accountability

8. AY1 - logging transactions with sensitive data

    - The system shall log every time **the system** performs the **sending of the patient's nutritional registered data**.
    - At a minimum, the system shall capture the following information for the log entry: **user** identification, timestamp, **send update**, **patient's nutritional registered data**, and identification of the owner of **the nutritional registered data**.

Security objective: privacy

9. PR1 – usage of personal data

    - The system shall allow the owner of the **nutritional data report** to be notified of potential authorized uses of **the nutritional data report**.
    - The system shall allow the owner of **the nutritional data report** to be notified when the **nutritional data report** is **accessed** by **the user**.
    - The system shall have the ability to get consent from the owner of **the nutritional data report** before accessing **the nutritional data report** for authorized use.
    - The system shall allow access to **the nutritional data report** for authorized use without prior consent only under exceptional circumstances as defined by applicable privacy policy.
    - The system shall have provision for the owner of **the nutritional data report** to withdraw consent for sending **the nutritional data report**.
    - The system shall have provision for de-identifying the **nutritional data report** before **sending** by **the user**, if required.

4.4 – Requirements validation

Thanks to the oracle's validation by 3 security experts, it was possible to improve the selection of templates for each of the 14 functional requirements in NutriMetas. The initial oracle (created by one security novice) contained 73 templates. After the oracle was validated, it ended up with 93 templates. In this process, 38 new templates were added (mostly integrity, availability and accountability templates) and 18 were removed. The differences between both oracles are shown in Section 5.5.2. The process carried out with each of the three security experts is as follows:

- The researcher arranged a meeting with the security expert in which an explanation about NutriMetas, its SRS and the SD methodology was given.
- The researcher handed in his latest oracle to the security expert.
- For each functional requirement, the expert validated if all necessary security objectives and templates were being taken into account.
- For each functional requirement, the expert argued if there was a need to add, remove or replace security requirements templates.

The recommendations given by security experts in this requirements validation phase were considered in the requirements specification detailed in the previous section.

4.5 – Discussion about the requirements engineering process

In our opinion, the SD methodology is a great SRE approach. Once you have mastered it, it really helps in the identification of security requirements. It is difficult at first to apply the methodology because even if you have taken your time to learn about the security objectives and the content of the 19 templates (which is needed anyway), it is not enough preparation for starting a proper requirements elicitation process in an academic or industrial setting. Practice sessions are recommended to assimilate the concepts and how to fill in the templates before making use of this SRE approach.

As for its use in an industrial setting, we consider SD viable for eliciting the basic security needs of a software system with its 19 templates distributed among 6 security objectives. In order to meet the security demands of more complex software systems, it would be advisable to extend this template repository so that more security requirements could be elicited this way. Moreover, exploring new security objectives may be another alternative to increase the scope of this methodology even more.

Its current scope may not only help software engineers which lack security expertise, but security experts too. In our experience as security novices using SD to elicit security requirements for NutriMetas, the mere fact of being introduced to the methodology made us think about security aspects that we would not have contemplated otherwise.

It helps think about security, even without starting the process of identifying security objectives and selecting applicable templates. It goes beyond the classic CIA (confidentiality, integrity and availability) triad and for beginners, this is new and valuable knowledge. After practicing filling in various templates, you truly understand how SD works.

What we rescue from the validation with the experts is that even if you (as a security novice) have learned how to apply SD, your security requirements elicitation may not be as accurate as you would imagine. This is a technical limitation of the practitioner, not of the methodology. Therefore, it is recommended for security novices to validate their elicited security requirements with security experts first, before continuing with the following phases of the SDLC. In our case, the security experts introduced many changes to the initial oracle. It is positive to realize that most of the templates we suggested were correct, but others were not included and this is a lesson for security novices: security aspects may be easily overlooked. If the practitioner is a security expert, this methodology may serve as a checklist.

On the other hand, the use of a software requirements specification following the ISO/IEC/IEEE 29148:2011 standard gives more information to the practitioners regarding the functionality of the software system. This makes it easier for them to elicit implicit security requirements present in functional requirements. For example, other studies have used use cases to explain system functionalities and they provide less details about the context and needs of the system. We consider that having a full understanding of the functional requirements of a system leads to a better security requirements elicitation process. As a result, it is of great value for software and requirements engineers if the functionalities of the software system are explained with a greater degree of detail. An intriguing aspect about this study was to prepare and use a more complete requirements specification without knowing if that would have an influence on the results. We realized that it could be used and, in fact, participants did understand NutriMetas' context. It is a more natural and traditional specification for software engineers.

Chapter 5. Replication Process

This section explains the details needed to take into account in order to conduct a controlled experiment of this nature and the chronological order of the studies which apply the SD methodology.

The original study took place at North Carolina State University (NCSU) in 2013 (Riaz et al., 2014b). It will be named as NCSU13, for the effects of this investigation. Afterwards, the findings from three differentiated replications were published in 2016 (Riaz et al., 2017). Their main objective was to compare the applicability of the previous findings and if its conclusions can be generalized in different contexts. This family of replications include:

- UT14: first replication organized at University of Trento (UT) (Riaz et al., 2017).
- NSCU14: second replication carried out at NCSU (Riaz et al., 2017) in 2014.
- UCR15: third replication conducted at University of Costa Rica (UCR) in 2015.

In 2018, two additional replications of the UCR15 study were conducted at the University of Costa Rica, with a different setting and context. They were planned and conducted in this investigation:

- UCR18a: fourth replication carried out at University of Costa Rica (UCR) in February, 2018.
- UCR18b: fifth replication which took place at University of Costa Rica (UCR) in April, 2018.

The objective of these replications was to extend the work done in 2015 at UCR in order to evaluate this approach, compare the applicability of the previous findings and if its conclusions can be generalized in a different context. The results from UCR18a and UCR18b were analyzed with the ones obtained in UCR15, for comparison purposes. Finally, the results from UCR studies were contrasted with the original study (NCSU13) and the first two replications (NCSU14 and UT14).

Differences and similarities between UCR15, UCR18a, UCR18b and all previous replications (including the original study) will be addressed as well. Figure 6 depicts the family of replications to date:

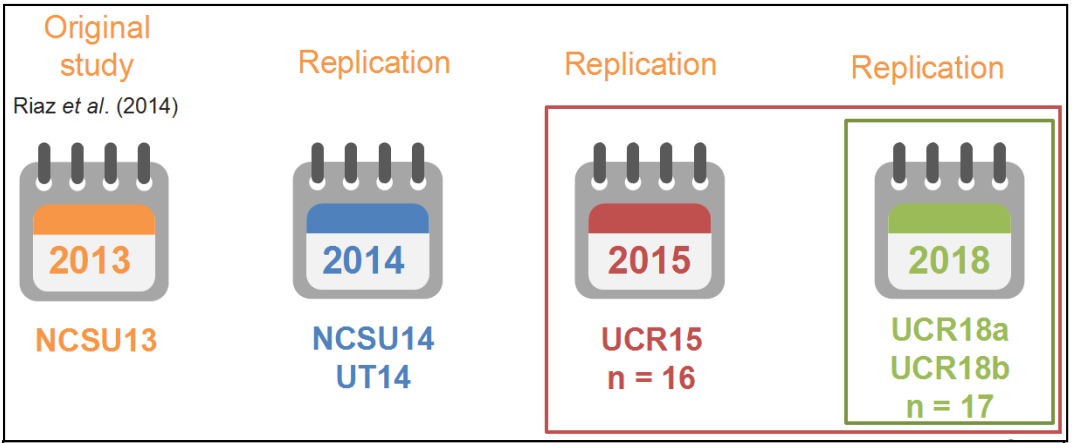


Figure 6 - Family of replications which make use of the SD methodology. UCR replications are displayed inside the red rectangle. The replications carried out in this investigation are inside the green rectangle

In all studies, participants were asked to identify security requirements based on a given domain (electronic health care or online banking). They were randomly placed by the researchers into one of the following groups: treatment or control. The first one did the exercise with the aid of suggested security requirements templates. The use of these templates was optional: it was their decision if they wanted to apply them or not. On the other hand, the control group did not have such help.

5.1 – Goals, hypothesis and metrics

"In the original experiment and all subsequent replications, we want to determine whether the use of suggested security requirements templates leads to efficient and effective elicitation of security requirements when compared to a manual approach based on personal expertise" (Riaz et al., 2017, p. 2136). The following research questions (RQ) were analyzed based on (Riaz et al., 2014b):

RQ1: What is the **quality** of security requirements elicited through the use of suggested security requirements templates?

RQ2: How **complete** are the security requirements elicited through the use of suggested security requirements templates?

RQ3: How **relevant** are the security requirements elicited through the use of suggested security requirements templates?

RQ4: How **efficient** are the security requirements elicited through the use of suggested security requirements templates?

In order to address these research questions, the following null hypothesis were tested (Riaz et al., 2014b; Riaz et al., 2017):

H01: The **quality** of elicited security requirements is unrelated to the use of suggested security requirements templates.

H02: The **quantity** of elicited security requirements is unrelated to the use of suggested security requirements templates.

H03: The **relevance** of elicited security requirements is unrelated to the use of suggested security requirements templates.

H04: The **efficiency** of elicited security requirements is unrelated to the use of suggested security requirements templates.

The metrics used for evaluating participants' responses are listed in Table 3 (Riaz et al., 2017). Data regarding quality, coverage, relevance (of security requirements) and efficiency of the process of eliciting security requirements is computed per participant. These results are used to test the hypothesis mentioned above.

In order to compute the metric for quality, a Likert-like scale of 1 to 5 (1: poor; 2: below average; 3: average; 4: above average; 5: good) is used. The quality of work done by each student was graded by using following questions as guide: (Riaz et al., 2017, p. 2137):

- "Are the requirements too general or too specific?
- Have all necessary elements of the requirements been identified (e.g., subject, resource, action, data to be logged)?
- Are there any logical inconsistencies in the requirements?
- Are different types of security objectives considered?
- For the treatment group, have the selected templates been filled-in with appropriate details?"

"For coverage and relevance, we respectively used the metrics for recall and precision computed based on an oracle of security requirements developed a priori to the conduct of the study" (Riaz et al., 2017, p. 2137). NCSU13, UT14 and NCSU14 studies analyzed the coverage and relevance in terms of the security requirements present in the oracle, as opposed to all UCR studies. UCR15, UCR18a and UCR18b measure coverage and relevance percentages based on the valid security requirements templates suggested by the participants. As a result, the way of calculating the efficiency also varies: NCSU13, UT14 and NCSU14 consider the number of security requirements in the oracle identified per minute, whereas UCR15, UCR18a and UCR18b replace security requirements with security requirements templates in that same calculation.

Table 3- Metrics used for evaluating participants' responses

Evaluation criteria	Metric type	Metrics used
Quality, *of security requirements*	Qualitative	Likert-like scale (1-5): lower score indicates lower quality.
Coverage, *of security requirements*	Quantitative	* Recall with respect to security requirements in the oracle. * Recall with respect to security requirements templates in the oracle (for UCR15, UCR18a and UCR18b).
Relevance, *of security requirements*	Quantitative	* Precision with respect to security requirements in the oracle. * Precision with respect to security requirements templates in the oracle (for UCR15, UCR18a and UCR18b).
Efficiency, *of process for eliciting security requirements*	Quantitative	* # of security requirements identified per minute. * # of security requirements templates in the oracle identified per minute (for UCR15, UCR18a and UCR18b).

5.2 – Participants

The responses of the 33 participants were evaluated. This amount represents the quantity of students in UCR15, UCR18a and UCR18b. All participants gave informed consent to use their responses for the study.

UCR15 was conducted in May 2015. In this replication, a total of 16 first and second year Master's degree students participated. They were enrolled in a Software Metrics course. Students were given 180 minutes to finish the exercise as an in-class activity, which they had to do by themselves. This task was graded with a significant percentage (25%) of their final grade. UCR15's experiment served as the basis for performing the UCR18a and UCR18b replications. Data collected in this replication is used in this work for analysis and comparison purposes.

The second replication at UCR, UCR18a, was conducted in February 2018. Similar to UCR15, 8 Master's degree students enrolled in an 8-weeks experimental software engineering intensive course participated. The task was an individual in-class activity and it was mandatory for them to complete it. They had 150 minutes (their whole lecture time) to finish this task and they knew beforehand it represented a significant percentage (20%) of their final grade.

Moreover, the third replication at UCR, UCR18b, was conducted two months later (April 2018) at University of Costa Rica. A total of 9 first and second year Master's degree students enrolled in an 18-week Design of Experiments course participated in this replication. Similar to UT14, the task was a take-home activity and they had one week to complete it. Conducting UCR18b as homework gives the opportunity to analyze if the results obtained in this study resemble those of UT14. This exercise was mandatory for them to complete as part of their course homework graded with a percentage of 10%.

For all replications, participants were told their grade would depend on the completeness and quality of the data reported in the process, the knowledge of the subject according to the base material provided and the rigorousness with which the experimental procedure was applied.

In the case of UCR15, the application domain was mobile banking and students were told to elicit security requirements from two use cases (UC1- Make payment and UC2- Retrieve account information) from a mobile payment software called Cyclos (Cyclos payment software, n.d.).

As for both UCR18a and UCR18b replications, the application domain (health care) and the software application NutriMetas (Jensen, Quesada-López, Zúñiga, Chinnock & Jenkins, 2015) were the same. Both replications' experimental procedures were supervised by one of the researchers involved in UCR15. Unlike UCR15, the participants involved in these two replications were asked to elicit security requirements from a SRS

(software requirements specification) which follows the ISO/IEC/IEEE 29148:2011 standard. The characteristics of this standard will be addressed in Section 4.3.2.

As soon as the participants finished the task, we asked them to report both their academic and work experience in Computer Science (CS), Software Engineering (SE) and Security education (Sec). No personal information about the participants was recorded. Table 4 summarizes the participants' background in UCR15, UCR18a and UCR18b for the three categories mentioned:

Table 4 – Participants' experience frequency

(*CS: Computer Science; SE: Software Engineering; Sec: Security*)

Study	Group	Number of participants	Experience (years)	Academic			Work Experience		
				CS	SE	Sec	CS	SE	Sec
UCR15	Treatment	9	> 5 years	5	4	2	2	3	0
			3-5 years	3	2	0	2	1	0
			1-2 years	0	2	1	3	2	2
			< 1 year	1	1	6	2	3	7
			No response	0	0	0	0	0	0
	Control	7	> 5 years	2	1	0	1	1	0
			3-5 years	5	4	0	5	4	0
			1-2 years	0	2	2	0	1	1
			< 1 year	0	0	5	1	1	1
			No response	0	0	0	0	0	0
UCR18a	Treatment	4	> 5 years	1	1	0	2	1	0
			3-5 years	3	1	0	1	1	0
			1-2 years	0	0	0	1	0	0
			< 1 year	0	2	4	0	2	4
			No response	0	0	0	0	0	0
	Control	4	> 5 years	4	2	0	1	1	0
			3-5 years	0	1	1	1	1	1
			1-2 years	0	1	1	2	1	0
			< 1 year	0	0	2	0	1	3
			No response	0	0	0	0	0	0
UCR18b	Treatment	5	> 5 years	2	1	0	1	1	0
			3-5 years	2	3	2	2	2	3
			1-2 years	0	0	0	1	1	0
			< 1 year	0	0	2	0	0	1
			No response	1	1	1	1	1	1
	Control	4	> 5 years	1	0	1	0	0	0
			3-5 years	0	0	0	1	0	1
			1-2 years	1	1	0	1	2	1
			< 1 year	1	2	2	1	1	1
			No response	1	1	1	1	1	1

UCR15 participants have comparable experience across all categories. Most of their work and academic experience is related to Computer Science and Software Engineering. Few work and academic experience in Security was reported.

Overall, almost all participants in UCR18a have the same reported academic and work experience. The only areas of expertise were slight differences are observed are SE

and Security academic experience: most treatment group participants reported they have less than one year of software engineering and security experience. This affirmation contrasts with the response of the participants in the control group: all of them argue they have at least one-year academic experience in SE and Sec experience ranging from "1 year" to "3-5 years" categories. Of the 8 students involved in this study, all of them reported their background information and were evenly distributed in both groups. "Similarities in background and expertise of participants indicate that differences in outcome and unrelated to a participants' experience" (Riaz et al., 2014, p.3).

On the other hand, UCR18b had a total of 9 participants (one more than UCR18a). In both control and treatment groups, one student did not give any response on their work experience or academic background, even though all participants gave consent to use their answers for this investigation, as stated before. Less work and academic experience was reported, on average, by the students in the control group.

Participants' experience is not analyzed nor compared across the five replications. "Experience is self-reported by participants and the semantics and interpretation of the word 'experience' might vary for each participant" (Riaz et al., 2017, p. 2140).

5.3 – Study environment

This section details the differences and similarities related to the study environment across all the controlled experiments.

5.3.1 – Shared aspects of study context

The author assigned the same task to all groups (regardless of whether they were treatment or control). They had to identify security requirements for a specific software application. After completing the task, participants were asked to "briefly explain the process used for identifying applicable security requirements (e.g., what information you looked at in the use case or reference material)" (Riaz et al, 2014, p.4), among other questions.

5.3.2 – Differences in study context

The most noticeable difference between the experiments was the way in which the task was presented to the students to complete it. All studies prior to UCR18a and UCR18b were conducted as an online experiment. All NCSU13, UT14, NCSU14 and UCR15 students received an URL by which they could access an online site where the task awaited them. Firstly, the system showed the students a consent form where they could explicitly allow the use of their responses for the study or not. Next, they received a unique and personal random access code. With this code, the system gave students opportunity to save the task at any given moment, leave the site and resume their work by entering it again. "The student was randomly assigned to treatment or control group and shown a screen with instructions for completing the task based on the group the student

was assigned to" (Riaz et al., 2017, p. 2140). The system recorded the total time spent by each student completing the task and if the student indeed submitted the task or not.

Grouping details prior to UCR18a were kept the same: a 2 x 2 between-subjects design was used and students were automatically assigned "to one of four groups in a round-robin fashion based on the process used for identifying requirements and the use case assigned (UC1 or UC2)" (Riaz et al., 2017, p. 2143). They received "specific sets of instructions, reference material, and task screens depending on the process (treatment vs. control) and the use case (UC1/UC2).

Figure 7 shows an example task screen as seen by treatment group students in UCR15, showing only one sentence from the use case for the purposes of this example. "We indicate the subject, resource and action elements at the end of each sentence in the use case to help with filling in the security requirements templates" (Riaz et al., 2017, p. 2144).

In the case of the control group, no security requirements templates were suggested. Without any help, participants had to think carefully before suggesting applicable security requirements templates for each sentence of the use case in question, enter them into a text area located below the use case's main flow and fill them. It was required for them to explicitly state the sentence number to which the security requirement template relates. Figure 7, for instance, displays one sentence of the use case and the security requirement template C1 is recommended for it, along with the sentence number in brackets.

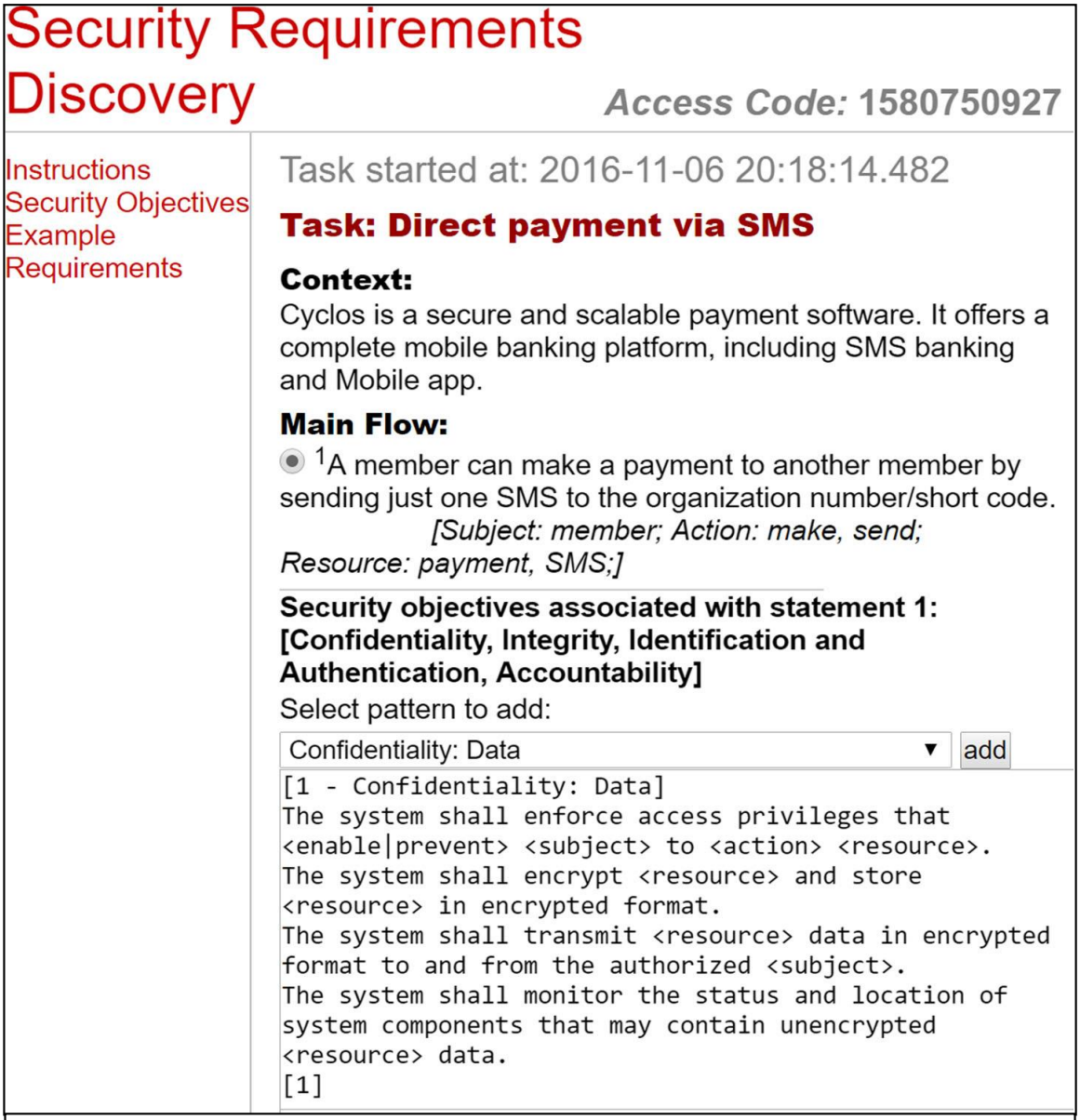


Figure 7 - Task screen for treatment group in UCR15

Source: Riaz et al., 2017, p. 2145

As for UCR18a and UCR18b, the students were told to fill in a digital form which was sent to their e-mails and send their responses back when they were finished. The digital form given to treatment group students contained a list of the 14 functional requirements (in Spanish) and the recommended templates for each requirement below. This form explicitly states the functional requirement code ranging from R0 to R13 and a short description of it. The SRS contains the requirements written in full, as well as additional information that thoroughly explains each one of them. Figure 8 shows a section of the digital form in which one of the functional requirements is stated with a recommended template below it. They must decide if they apply them or not. They are al-

lowed to other templates from Riaz et al. as well. Further details will be provided in Section 5. In the case of control students, they receive the same list, but without the recommended templates.

Authorization to experiment with these students was given days before UCR18a's course ended and due to the fact Riaz's online experiment tool had not been implemented in UCR's servers and adapted to the researchers' needs, a more straightforward approach was needed to collect their data. Opportunity to carry on experimenting with UCR18b's students came immediately after, so the same experiment methodology had to be applied.

As mentioned before, the UCR18a replication was planned as an individual activity during class time. A 4-page reference material and a 10 minute-video related to security objectives and requirements were handed to the students one week before carrying out the study. A 15-minute explanatory presentation in Spanish was given to them by one of the researchers, the day of the experiment. This presentation served as an overview of the reference material and as a means for students of clarifying doubts. They even had access to the findings from the three differentiated replications (Riaz et al., 2017). However, researchers did not know before starting the task if the participants had studied the reference material or the paper.

UCR18b was thought as a take-home activity (homework) from the beginning because the course professor only gave permission to experiment with her students this way. They had one week to complete the task. All participants in this study received the same reference material given to UCR18a, UCR15 and NCSU14. They also had access to the findings from the three differentiated replications (Riaz et al., 2017). A 15-minute explanatory presentation given to UCR18a students was given to UCR18b participants that same day, during lecture time. One of the researchers gave the students his e-mail address and phone number, in case they wanted to reach him for extra guidance before starting the task.

In both UCR18a and UCR18b, students were asked to identify security requirements for a mobile health application called NutriMetas, by analyzing its SRS. The wording used to elaborate this SRS was based on the ISO/IEC/IEEE 29148:2011 standard, which focuses on best practices regarding requirements construct and requirement language criteria, among others. Its application may help stakeholders and developers understand the requirements (of a certain software system) in a more detailed and comprehensive way. For the effects of this investigation, the application of this standard in NutriMetas' SRS may aid participants acquire a broader knowledge of what the application's functional requirements are and, possibly, lead to a better template picking and filling.

Prior studies relied on use cases to describe specific user processes for the iTrust electronic health record (EHR) system, the Cyclos mobile payment and the Virtual Lifetime Electronic Record (VLER) (Virtual Lifetime Electronic Record, n.d.). Their wording did not apply the ISO/IEC/IEEE 29148:2011 standard. It is shown in Figure 7 that little

information is provided about Cyclos' context and it was expected for participants to elicit security requirements right from the first statement in the use case ("a member can make a payment to another member by sending just one SMS to the organization number / short code"). This lack of detail may have an impact on the use of templates. Therefore, in order to mitigate this issue, the mentioned standard was used.

Additionally, like NCSU13 and UT14, UCR18a and UCR18b students were not given suggestions to fill in templates.

Finally, all participants in UCR15, UCR18a and UCR18b were native Spanish speakers. They were told they could provide their responses in English or Spanish. Templates suggested to the participants in UCR18a and UCR18b treatment groups included some extraneous suggestions as in NCSU14 and UCR15. It was their decision to make use of these suggestions or not. Also, they were not aware that some of them could be extraneous. Table 5 summarizes the differences in context factors across all studies.

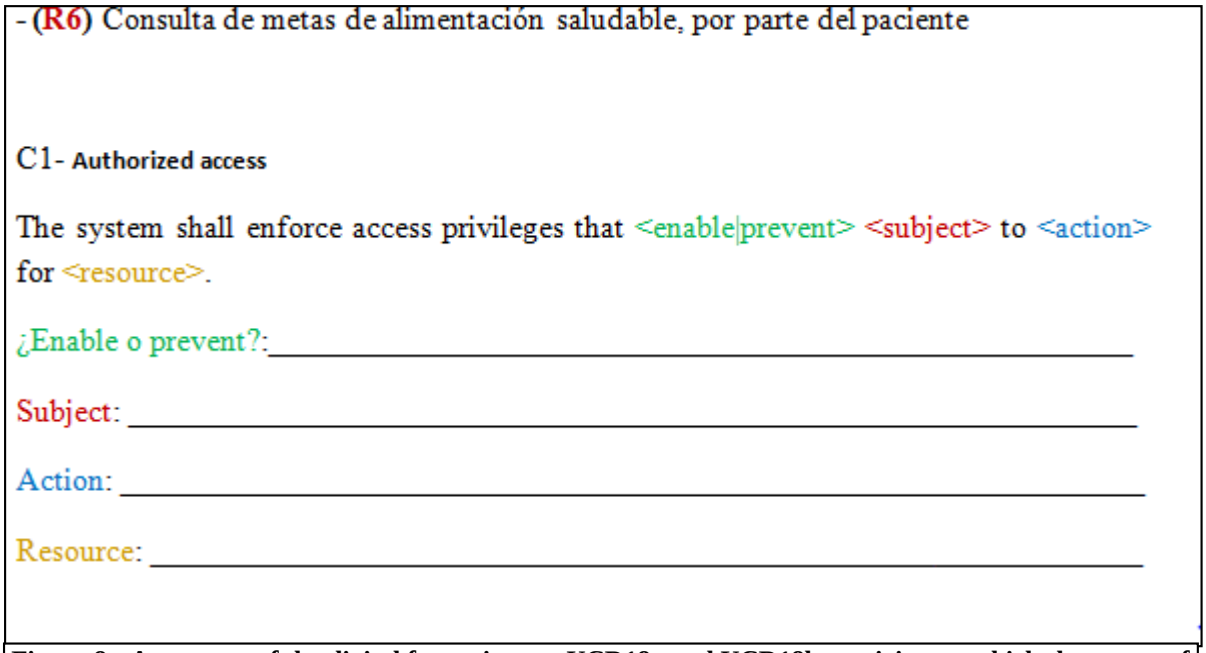


Figure 8- An extract of the digital form given to UCR18a and UCR18b participants which shows one of the 14 functional requirements in NutriMetas' SRS, and one recommended security template for it

Table 5 - Summary of context factors across different experiments

	Original study [NSCU13]	Replication-1 [UT14]	Replication-2 [NCSU14]	Replication-3 [UCR15]	Replication-4 [UCR18a] (This study)	Replication-5 [UCR18b] (This study)
Participants	50 graduate students	32 graduate students	107 graduate students	16 graduate students	8 graduate students	9 graduate students
Setting	In-class Activity, NCSU [60 min]	Take-home Activity UT, [1 week] [*Coverage & Efficiency]	In-class Activity, NCSU [60 min]	In-class Activity, UCR [180 min]	In-class Activity, UCR [150 min]	Take-home Activity UCR, [1 week] [*Coverage & Efficiency]
Security training provided	• Software Security course • 4 page reference material	• Security Engineering course • 4 page reference material	• Software Security course • 4-page reference material • 10 min video on security objectives and requirements	• 4-page reference material • 10 min video on security objectives and requirements • 15 min explanatory presentation in Spanish	• 4-page reference material • 10 min video on security objectives and requirements • 15 min explanatory presentation in Spanish	• 4-page reference material • 10 min video on security objectives and requirements • 15 min explanatory presentation in Spanish
Problem domain	Health care	Health care	Health care	Mobile banking	Health care	Health care
Source	Use case scenarios	Use case scenarios	Use case scenarios	Use case scenarios	SRS	SRS
Other changes	NA	No other changes	• Suggestion to fill in templates • Feedback on quality of responses [*Quality] • Extraneous Templates [*Relevance]	• Suggestion to fill in templates • Feedback on quality of responses [*Quality] • Extraneous [*Relevance]	• Feedback on quality of responses [*Quality] • Extraneous Templates [*Relevance] • Offline activity [*Efficiency]	• Feedback on quality of responses [*Quality] • Extraneous Templates [*Relevance] • Offline activity [*Efficiency]

5.4 – Experiment artifacts

"All participants received four pages of reference material containing a description of software objectives as well as textual clues that can indicate an implied security objective. Reference material also contained a total of 40 example security requirements grouped by security objectives" (Riaz et al., 2017, p. 2142) and a video presentation made by Maria Riaz, talking about security objectives and templates. Students were told to study the material in this particular order. This standard reference material was handed to UCR15 and UCR18a one week before the start of the experiment, so they could study it and ask researchers for advice, if necessary. A 15-minute explanatory in Spanish was given to all students before they started the task. UCR18b students attended the 15-minute explanatory presentation in Spanish first and then received the reference material, in order to complete their take-home activity (graded homework). All students from both studies had access to such material during the experiment. Researchers instructed all participants to apply as many security requirements templates as they considered necessary. This meant treatment students were allowed to use non-recommended templates to solve the security needs of any of the 14 functional requirements, if they considered it appropriate.

As it was briefly mentioned before, NutriMetas mobile health application was selected for UCR18a and UCR18b experiments (Jensen, Quesada-López, Zúñiga, Chinnock & Jenkins, 2015). It was created at UCR and the purpose of this application is to co-assist the nutritionist in the process of nutritional care by modifying patients' healthy eating habits. This Android-only application allows patient self-monitoring on a set of healthy goals previously established by their nutritionist. The application does not require patients to constantly log what they consume each day. Its approach is to have a specific behavioral tracker.

All UCR18a and UCR18b students received the NutriMetas' SRS based on the ISO/IEC/IEEE 29148:2011 standard. It contains 14 functional requirements which imply the need of security requirements. These requirements range from user (patient) activation and contact details input/modification, to goal registration (meals per day and specific beverage daily count, for instance), reminders and creation of anthropometric reports.

From NCSU14 onwards, "we did not introduce any differences in terms of the experiment material given to participants" (Riaz et al., 2017, p. 2143).

5.5 – Experiment design artifacts

Stratified random sampling was applied. This means the student population was divided into homogeneous groups before sampling. Gender, expertise in security (based on the professor's opinion for each of its students) and group size were the factors taken into account. Then, random sampling was applied within each stratum to assign each student "to one of the following processes for identifying security requirements" (Riaz et al., 2017, p. 2144):

- Treatment (T): security requirements templates are recommended to students (in their answer sheet) in order to help them in the identification of security requirements.

- Control (C): students in the control group are not suggested templates at all. Hence, a manual identification of security requirements is encouraged.

Once the researchers had assigned a group to each student, they were ready to hand them a zip file containing the experiment artifacts mentioned in Section 4.4, NutriMetas' SRS, task instructions, the answer sheet and, last but not least, the feedback questions. Two different zip files were created for UCR18a and UCR18b: one named "C", for the control group students and "T", for the treatment group. The task instructions and the answer sheet are the only differences between both files. Each group received a specific answer sheet and set of instructions, depending on whether security requirements templates were recommended or not. However, all students in UCR18a and UCR18b had to complete the same task: elicit as much security requirements as needed for NutriMetas, following Riaz's method in the experiment artifacts. The need to follow this method was emphasized.

UCR18a students received the zip files from one of the researchers by email and, when they finished the task, they send their answers and experiment feedback to their professor. The professor then sent a new zip file containing all student answers to the researchers for analysis. The students were not aware of the existence of two different zip files ("T" and "C"), nor of the treatment and control groups.

On the other hand, UCR18b students were told to download one of the two files from the course site. The researchers told each student which file to download and work with, the same day the 15-minute explanation in Spanish was given. Contrary to UCR18a students, they were aware of the existence of two separate files. Nevertheless, the researchers did not explain the reason behind this and only limited themselves to asking the participants to please follow the instructions.

As for time registration, researchers in UCR18a could efficiently register how much time each student took completing the task. All students started the experiment at the same time and as soon as they were finished, the researchers wrote down their finish

time. This was done to double-check, due to the fact participants were told to type in their start and end time in the spaces provided in the answer sheet.

For UCR18b, researchers had to rely on the elapsed time the students registered. None of them forgot to type in their start and end times.

Table 6 displays the number of participants in each group for the original experiment and its five replications. Note that UCR18 studies were based on NutriMetas' SRS:

Table 6- Number of participants in each group

Experiment	Treatment			Control			Total
	UC1	UC2	SRS	UC1	UC2	SRS	
UCR15	5	4	-	4	3	-	16
UCR18a	-	-	4	-	-	4	8
UCR18b	-	-	5	-	-	4	9
Total	5	4	9	4	3	8	33

5.5.1 – Evaluation methodology

This section covers the methodology used for evaluating the responses submitted by the participants. These responses will help the researchers evaluate the quality, coverage, relevance and efficiency of their security requirements elicitation process by computing the metrics.

5.5.2 – Oracles of security requirements templates

Two oracles were created to evaluate the coverage and relevance of the security requirements templates identified by the participants in both groups. One of them was applied in UCR18a and the second one was used to evaluate UCR18b's responses.

For each of the 14 functional requirements in NutriMetas' SRS, the author:

- Identified the security objectives related to the requirement in question.
- Selected the security requirement templates that were applicable, for each security objective present in the functional requirement.

This manual process is similar to the steps SD follows.

UCR18a's oracle was created by one of the researchers, following the SD methodology. Table 7 shows a list of the security templates related to each of the functional requirements in NutriMetas. This study did not include extraneous templates. Its SRS can be found in Appendix 1.

UCR18b's oracle is an improved version of the one used in UCR18a. UCR18a's oracle is a subset of UCR18b's oracle. It has been created by the same researcher and revised by three software security experts.

Table 8 shows UCR18b's oracle. Extraneous templates were included in this study, as done in NCSU14 and UCR, in order to analyze if treatment students question the use of recommended templates given to them or not.

Table 7 - Oracle for UCR18a

Functional requirement	Recommended templates
R0	C1, C2, C3, I2
R1	C1, C2, C3, I2
R2	C1, C2, C3, I2
R3	C1, C3, A2, A3, A4, IA2, IA3
R4	C1, C2, C3, I2
R5	C1, C2, C3, I2
R6	C1, C3, A2, A3, A4, IA2
R7	C1, C2, C3, I2
R8	I4
R9	C1, C3, A2, A3, A4, IA2, IA3
R10	C1, C3, A2, A3, A4, IA2, IA3
R11	C1, C3, A2, A3, A4, IA2, IA3
R12	C1, C3, A2, A3, A4, IA2, IA3
R13	C1, C2, C3, I2, I4, A3, PR1

Security requirement templates in this oracle: 73

Riaz's templates in this oracle: 11/19

(C1, C2, C3, I2, I4, A2, A3, A4, IA2, IA3, PR1)

Riaz's templates not used in this oracle: 8/19

(I1, I3, A1, A5, IA1, AY1, AY2, AY3)

Table 8 - Oracle for UCR18b

Functional requirement	Recommended templates
R0	C1, C2, C3, I2, IA2, AY2
R1	C1, C2, C3, I1, I2
R2	C1, C2, C3, I1, I2
R3	C1, C3, I1, A2, A3, A5, AY1, PR1
R4	C1, C2, C3, I1, I2, AY1
R5	C1, C2, C3, I1, I2, AY1
R6	C1, C3, I1, A2, A3, A5, AY1, PR1
R7	C1, C2, C3, I2, A3
R8	C3, I1, A3
R9	C1, C3, I1, A2, A3, A5, AY1, PR1
R10	C1, C3, I1, A2, A3, A5, AY1, PR1
R11	C1, C3, I1, A2, A3, A5, AY1, PR1
R12	C1, C3, I1, A2, A3, A5, AY1, PR1
R13	C1, C2, C3, I1, I2, I4, A3, AY1, PR1

Security requirement templates in this oracle: 93 (20 more than UCR18a's oracle)

Riaz's templates in this oracle: 13/19 (2 more than UCR18a's oracle)

(C1, C2, C3, I1, I2, I4, A2, A3, A5, IA2, AY1, AY2, PR1)

Riaz's templates not used in this oracle: 6/19 (2 less than UCR18a's oracle)

(I3, A1, A4, IA1, IA3, AY3)

Templates added by security experts are underlined

It is important to note the differences between both oracles. After the three software security experts gave their opinion on what templates the new oracle should include:

1. A total of 20 additional templates were included, going from 73 in UCR18a to 93 in UCR18b.
2. 13/19 of Riaz's security requirement templates were used. UCR18a's oracle made use of 11 of them.
3. 6/19 of Riaz's security templates were not used, as opposed to the 8 security templates UCR18a's oracle left unused.

5.5.3 – Mapping responses to the oracle

For the studies prior to UCR15, "the security requirements in the participants' responses were mapped to the requirements in the oracle to compute the metrics for coverage and relevance" (Riaz et al., 2017, p. 2147).

For UCR15 onwards, "we defined the oracle in terms of the templates rather than individual requirements as explained in the previous section" (Riaz et al., 2017, p. 2147). Control groups in UCR15, UCR18a and UCR18b had to explicitly mention (and fill in) which security template(s) solved the need for security for every functional requirement in NutriMetas' SRS, as their answer sheet was completely blank and no security templates were suggested to them. Treatment groups in these same studies had the opportunity to fill or ignore in the templates suggested to them, as well as making use of any other security templates in their reference material which they considered appropriate.

For each functional requirement, security requirements templates suggested by students are compared (mapped) to the ones in the oracle. For instance, if a participant suggested template C1 for functional requirement R0 and the oracle indicated C1 for that particular requirement, it counts as a true positive (TP). If the oracle contains a template for a specific functional requirement and the participant does not mention it, it is marked as a false negative (FN). On the contrary, if the student suggests a template for a requirement and it is not present in the oracle, a false positive (FP) is reported. Finally, templates out of the oracle which are not mentioned by participants count as true negatives (TN).

"The metrics for coverage and relevance provide assessment of the participants' performance in terms of how many requirements in the oracle a participant identified as well as how much of the effort was spent in identifying relevant requirements (TP) verses irrelevant ones (FP)" (Riaz et al., 2017, p. 2148). This applies for NCSU and UT studies. For UCR studies as well, but in terms of security requirements templates.

5.5.4 – Threats to validity

These are the threats to validity considered in UCR15, UCR18a and UCR18b. These threats were taken from Riaz et al. (2017), which follow Wohlin et al. (2012) sectioning, and adapted to the context of UCR replications.

5.5.4.1 - Internal validity

Selection. Stratified random sampling was applied. This means the student population was divided into homogeneous groups before sampling. Gender, expertise in security (based on the professor's opinion for each of its students) and group size were the factors taken into account. Then, random sampling was applied within each stratum to assign each student to one group (treatment or control). Even with these precautions, it is possible to create control and treatment groups in which the expertise of participants is unbalanced.

Diffusion or Imitation of Treatment. UCR18a was carried out as a class activity in exam conditions. They were not allowed to share answers nor opinions with their classmates and were not told about the existence of two groups (treatment and control). On the other hand, study UCR18b was planned as a take-home activity and students may have shared their answers. For instance, treatment group students may have collaborated with control group students. Collaboration may have even taken place within members of the same group. However, the author did not spot treatment diffusion or imitation across the groups, or similar responses among participants of the same group. Some students in the control group based their answers based on the examples available to them in the reference material, while others (two control group participants in UCR18b) preferred to specify security requirements using their own words.

Testing and Training. Treatment and control groups in both replications received the same reference material and explanatory presentation, before the start of the experiment. In addition to this, they did not receive further training relating to how and when to apply the security requirements templates. The reference material had example security requirements that may have biased students by motivating them to fill in their selected templates in a similar manner. This potential bias applies for all UCR replications. Besides, one of the researchers even offered to answer any questions by phone or email (if necessary) before UCR18a and UCR18b participants started the activity. Of both studies, only one treatment participant from UCR18b called to make sure he understood the instructions well. The author answered his questions and only referred to the concepts he explained in the presentation. He did not go beyond that explanation so that this student did not have an advantage over other participants.

Interactions with Selection. Treatment group participants received security requirement templates suggestions and they could have accepted these suggestions without questioning them (Riaz et al., 2017). In this case, they would only need to fill in the templates the document provides to them. Students behaving this way would have obtained high coverage and efficiency scores. This trend was only observed in a participant in UCR18a, who selected all templates. UCR18a did not have extraneous templates, so she achieved perfect scores in coverage and relevance. Apart from this case, no other student obtained perfect scores for both metrics. For instance, the coverage for treatment students in UCR18b ranged from 82% to 96%, with their relevance oscillating between 74% and 78%. This means treatment participants are thinking critically, even under the influence of extraneous templates.

5.5.4.2 - External validity

Representativeness of Sample Population. "The sample population should be representative of the population for which we want to draw conclusions based on the study outcomes" (Riaz et al., 2017, p. 2170). All 17 participants are enrolled in two different graduate courses of the same Master's degree in computer science, in UCR. These courses are non-security related and 88% of them stated IT to be their main career. Only two of them initially belonged to another area of expertise (one of them is a chemical engineering and the other one is an electric engineer) and moved to IT afterwards. Therefore, participants fairly represent the population of computer science graduate students. According to the feedback provided by 15 out of the 17 participants, 10 of them (about 59%) consider they have less than 1 year of academic experience in security and about 53% has worked in software security for less than a year. About 47% reported to have worked in software engineering between 1 and 5 years, and 18% argue they have worked for more than 5 years in this same area. As a result, it can be inferred that participants can be considered non-expert security practitioners, with entry-level to intermediate knowledge in software engineering.

Task Representativeness. It is important for the task to truly represent the process of requirement elicitation in the industry. Each student received NutriMetas' SRS, which follows ISO/IEC/IEEE 29148:2011 standard, and was instructed to identify applicable security requirements templates. It provides information about the context of the mobile application, besides detailing its 14 functional requirements, which may help practitioners understand more about the problem domain and possibly make them consider more security requirements.

Templates Representativeness. In these replications, Riaz's security requirements templates proved to be a great resource if they are applied to identify security requirements for a mobile health application. In prior studies, the health care domain had been addressed, but not in mobile applications. Therefore, basing on the experience obtained in

these replications, this set of templates might help identify security requirements for different problem domains which have similar security objectives to the ones suggested by Riaz.

Experimental Constraints that Limit Realism. Time is a constraint that may affect the quality, coverage and efficiency of participants in the identification of security requirements. UCR15 and UCR18a participants had 180 and 150 minutes, respectively to finish the task. They may have felt pressure to finish on time. UCR18b, on the other hand, had the opportunity to work at home for a week and did not have that time constraint. In practice, eliciting security requirements for complex software systems may take longer and require the help of more than one requirements engineer. In these experiments, nevertheless, all participants worked individually and possibly the given time may not be sufficient for them to identify all templates in the oracle.

5.5.4.3 - Construct validity

Hypothesis Guessing. In general, all UCR18a and UCR18b students had access to the paper research paper in which the original experiment (NCSU13) was explained. It is unknown if they read it before carrying out the activity. The problem domain for NCSU13 did not involve NutriMetas' SRS, so the participants could not have known which templates to apply for each of the functional requirements in it. Another issue is that UCR18b participants knew about the existence of two different groups (treatment and control). Not surprisingly, they were interested in knowing why, but they received a little detailed response so as not to bias them before carrying out the take home activity.

5.5.4.4 - Conclusion validity

Reliability of Measures. Reliable conclusions must be drawn from the outcomes. For UCR18a, researchers wrote down the start time and end time of all participants. They were told to type in these times in their answer sheet as well, so we could compare our data with their self-reported times. No significant differences were observed. However, researchers could not double-check these times for UCR18b. In UCR18b, participants completed the task at home (without the online tool) and the author had to rely on the start and end times they reported. This threat may affect the efficiency scores for UCR18b, which directly affects the assessment of RQ4.

Fishing and the Error Rate. In order to minimize bias, the evaluator created and validated the oracle with three security experts before the participants took part in this experiment. As for control group students, care was taken when evaluating the quality of their responses. Some answers given by students in control groups were written in their own

words and some of them could not be easily mapped to a certain template. Nevertheless, all input was taken into consideration for the purpose of evaluation.

Violated Assumptions of Statistical Tests. In order to avoid wrong conclusions, the assumptions of the Mann-Whitney U test must not be violated. This test is a non-parametric alternative to the t-test. It does not depend on a large population in order to draw well-founded conclusions and the samples must meet these conditions:

- "The two investigated groups must be randomly drawn from the target population" (Nachar, 2008, p.15). All participants come from the graduate courses at UCR mentioned before and randomly assigned into treatment and control groups. These are two independent groups.

- "Each measurement or observation must correspond to a different participant. In statistical terms, there is independence within groups and mutual independence between groups" (Nachar, 2008, p.15). None of the participants is part of more than one group, and none of them participated in more than one study.

- "The data measurement scale is of ordinal or continuous type. The observations values are then of ordinal, relative or absolute scale type" (Nachar, 2008, p.15). Quality is measured as an ordinal variable (Likert-like scale) and coverage, relevance and efficiency are continuous variables. Coverage and relevance are measured from 0 to 100%, while efficiency is measured in templates per minute.

- The independent variable should consist of two categorical, independent groups. In our studies, the independent variable is the requirements elicitation method (with automatically-suggested templates vs. without automatically-suggested templates).

"Even if it is used on average-size samples (between 10 and 20 observations) or with data that satisfy the constraints of the t-test, the Mann-Whitney has approximately 95% of the Student's t-test statistical power" (Nachar, 2008, p.19). Therefore, it is feasible to use this non-parametric test in UCR18a and UCR18b studies (which contain fewer participants, compared to UCR15) and obtain reliable results.

Number of Participants. There is a limited amount of participants in UCR15 (16), UCR18a (8) and UCR18b (9), compared to UT14 (32), NSCU13 (50) and NCSU14 (107). Reliable conclusions may not be drawn due to these small groups. However, some of the findings of the replications in UCR are similar to the results of previous studies, in which a larger number of participants took part. For example, all treatment groups performed significantly better in the metric of coverage.

Chapter 6. Analysis of results

The results obtained in UCR15, UCR18a and UCR18b will be analyzed to conclude if the results support the hypothesis given in Section 4.1 or not. For UCR18a and UCR18b, as we have only one SRS and two treatments, the Mann-Whitney U test is used instead. This test is useful if a need of comparing the differences between two independent groups, which do not follow a normal distribution, exist. In this case, there are two independent groups: treatment and control. Also, it does not depend on a large population in order to draw well-founded conclusions (our sample is of 33 students in total, which is a small population compared to the ones in previous studies) and it strictly requires different participants in each group with no participant being in more than one group. These aspects are fulfilled by UCR15, UCR18a and UCR18b.

IBM's SPSS Statistics 1.0.0.1174 is used for the statistical analysis and R version 3.5.2 is used for plotting box plots of these results. This analysis can help us determine "the factors or the interaction between the factors leads to significantly different group means for the four metrics. "Results with $p<0.05$ are considered significant for our analysis" (Riaz et al., 2017, p. 2148).

Table 9 shows the average scores for all metrics, ranging from the original experiment to all replications done to date. UCR18a and UCR18b results will be discussed in the following subsections.

Table 9- Overall mean scores for all metrics across studies

Study	Time available for completing the task	Mean time on task	Mean quality (1-5)	Mean coverage (0-1)	Mean relevance (0-1)	Mean efficiency (req/min)
NCSU13	60 minutes in-class	~20 minutes	2.88	0.31	0.88	1.14
UT14	One week to complete at home	~47 minutes	2.70	0.36	0.77	1.07
NCSU14	60 minutes in-class	~25 minutes	2.66	0.37	0.73	1.01
UCR15	180 minutes in-class	~102 minutes	3.69	0.51	0.66	0.64*
UCR18a	150 minutes in-class	~116 minutes	4.575	0.655821918	0.675118567	0.629371243*
UCR18b	One week to complete at home	~193 minutes	3.9075	0.660178001	0.721655228	0.782612782*

* Mean efficiency (templates/min)

6.1 – UCR15 replication

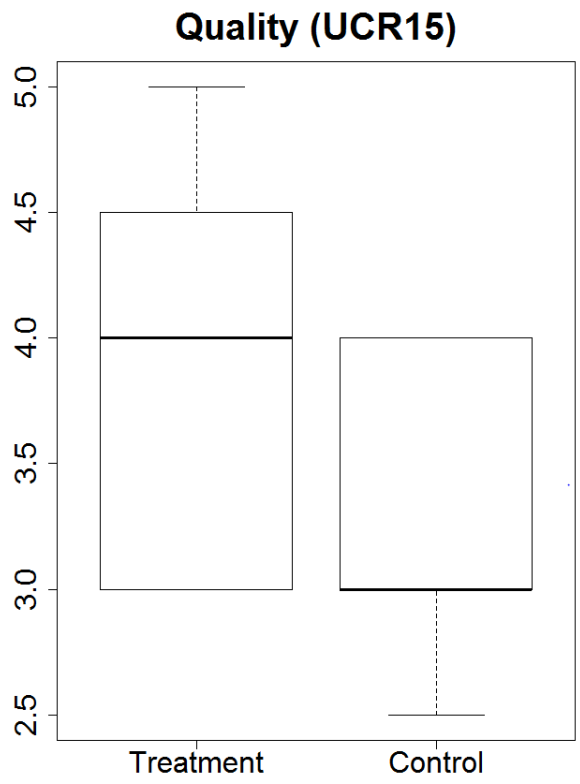
According to the data in Table 10, which displays the results of UCR15, we can conclude that the use of automatically-suggested templates proves to be a significant factor for coverage (Mann-Whitney U = 9.0 and p-value = 0.017) and efficiency (Mann-Whitney U = 7.5 and p-value = 0.020). Therefore, the null hypotheses H_{02} and H_{04} (which state that the performance of participants based on metrics for coverage and efficiency, respectively, is unrelated to the use of security requirements templates) can be rejected. On average, participants from the treatment group identified 63% of the templates in the oracle (compared to 36% in the control group) and their efficiency almost doubled the one obtained by the control group (20% vs. 11%).

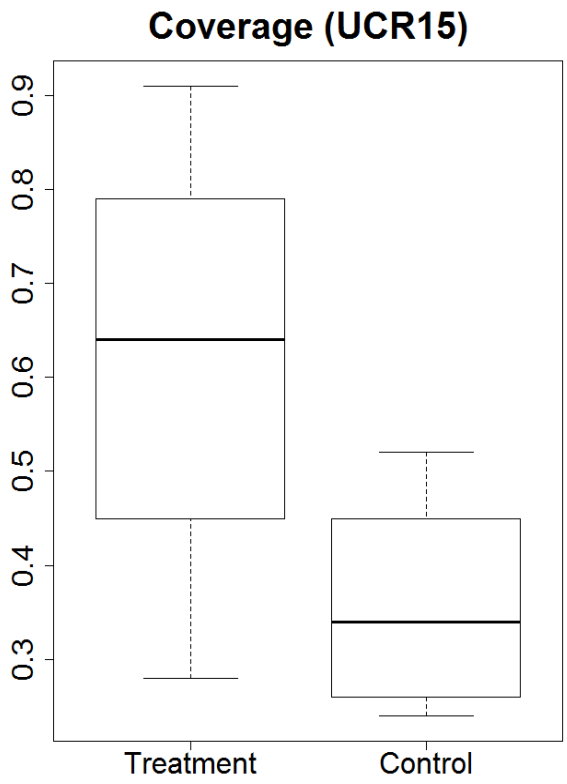
On the contrary, no significant differences were found for the metrics of quality and relevance. Therefore, the null hypotheses H_{01} and H_{03} cannot be rejected. On average, the quality of responses is slightly higher for the treatment group (3.94 vs. 3.36 for the control group). Moreover, the responses given by participants in the treatment group were on average 12% more relevant than the ones reported by the students in the control group.

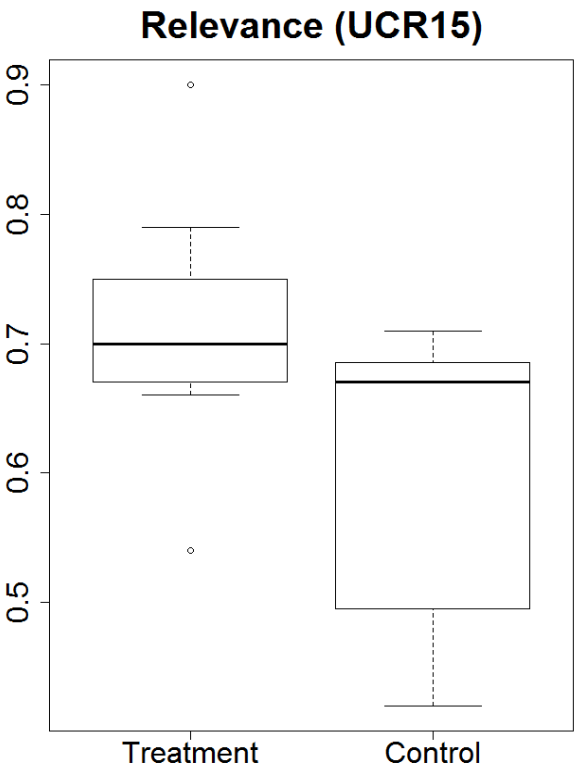
Group means were plotted in box plots for each requirement process (control vs. treatment) in Figure 9. It is evident treatment students performed better than control students across all four metrics.

Table 10- Results obtained in UCR15

Factor ↓/ Metric →		Quality (1-5)		Coverage (0-1)		Relevance (0-1)		Efficiency (templates / min)	
		Means	P-value	Means	P-value	Means	P-value	Means	P-value
UCR15	Treatment	3.94		0.63		0.71		0.20	
	Control	3.36	0.131	0.36	0.017	0.59	0.121	0.11	0.011







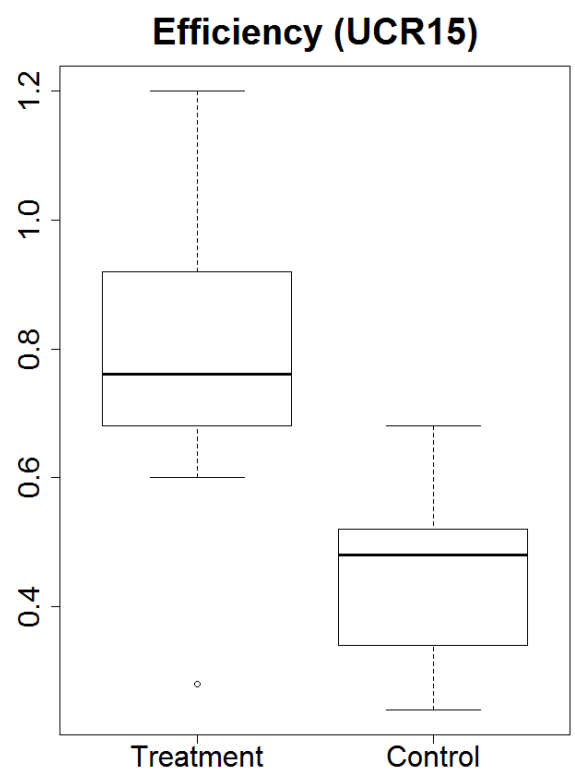


Figure 9 - Box-plots with results from UCR15 across each factor (treatment/control) and metric

6.2 – UCR18a replication

According to the data in Table 11, which displays the results of UCR18a, we concluded that the use of automatically-suggested templates is a significant factor in determining the metrics for coverage (Mann-Whitney U = 0 and p-value = 0.019) and relevance (Mann-Whitney U = 0 and p-value = 0.020). Therefore, the null hypotheses H_{02} and H_{03} (which state that the performance of participants based on metrics for coverage and relevance, respectively, is unrelated to the use of security requirements templates) can be rejected. On average, treatment group students identified 90% of the templates in the oracle and their templates were about 60% more relevant than the ones reported by the participants in the control group.

No significant differences were found for the metrics of quality and efficiency. Therefore, the null hypotheses H_{01} and H_{04} could not be rejected. On average, the quality of responses is slightly higher for the control group (4.4 vs. 4.7 for the treatment group). This may have happened because the control group did not receive template suggestions and that could have served as a motivation to consider the use of more templates (as they were not being biased by them) and put in more detail in their responses. Some treatment students, on the contrary, did not excel in these aspects probably because they followed the template suggestions without questioning them too much. 50% of this control group obtained a 3/5 for considering the use of different templates, which suggests they limited themselves to fill in only the templates they received. Their score for detail (3/5) was average as well: they did not provide detailed security requirements.

As for the efficiency, treatment students lasted on average 1.25 minutes less than the control group finishing the task (115.5 vs. 116.75, respectively). It can be inferred that, although treatment students have received help in the form of template suggestions, they took their time to do the activity. Reported templates in the treatment group range from 59 to 73 (it does not vary too much), with 73 being the total template count for this oracle. This means one student from the treatment group did not question the template suggestions at all. However, in the control group, reported templates count oscillates more: between 48 and 104. This also explains why the treatment group scored higher in coverage and relevance.

Group means were plotted in box plots for each requirement process (control vs. treatment) in Figure 10. As said before, treatment students performed better than control students across two out of four metrics: coverage and relevance.

Table 11- Results for UCR18a

Factor ↓/ Metric →		Quality (1-5)		Coverage (0-1)		Relevance (0-1)		Efficiency (templates / min)	
		Means	P-value	Means	P-value	Means	P-value	Means	P-value
UCR18a	Treatment	4.4	0.557	0.9041	0.019	0.9749	0.020	0.5996	0.564
	Control	4.75		0.4075		0.3753		0.6590	

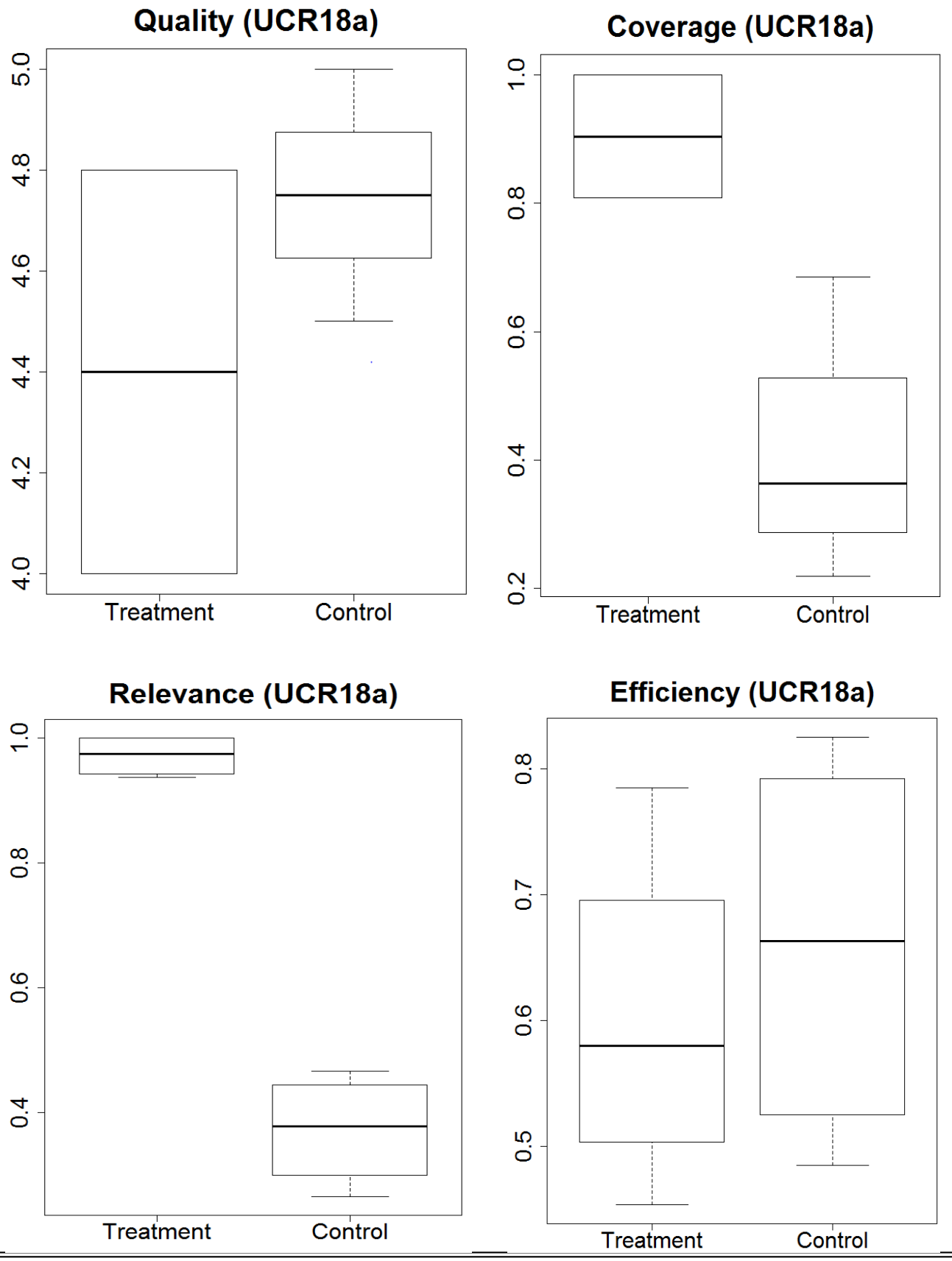


Figure 10 - Box-plots with results from UCR18a across each factor (treatment/control) and metric

6.3 – UCR18b replication

According to the data in Table 12, which displays the results of UCR18b, we can conclude that the use of automatically-suggested templates is a significant factor in determining the metrics for coverage (Mann-Whitney U = 0 and p-value = 0.014). As in NCSU13, NCSU14 and UCR15, we reject the null hypothesis H_{02}. It states that the performance of participants based on metrics for coverage is unrelated to the use of security requirements templates. Treatment group participants identified almost 91% of the templates in the oracle, on average. Students in the control group, on the other hand, identified 40% of the templates in the oracle overall.

No significant differences were found for the metrics of quality, relevance and efficiency. Thus, we fail to reject the null hypotheses H_{01}, H_{03} and H_{04}. The difference in efficiency between the treatment and control group (~65% vs. 36%) did not prove to be significant at $p < 0.05$, but it is at $p < 0.1$. This may be because one participant from the treatment group lasted 273 minutes (4 hours and 33 minutes) completing the task and reported 106 templates (efficiency of 0.39 templates/min), while another participant in this group lasted 17 minutes more (exactly 5 hours in total). This issue affects the overall efficiency of the treatment group. As a reference, the treatment student from UT14 who took the most time completing the task at home lasted 132 minutes. That is 2 hours and 12 minutes. These two UCR18b students doubled that time, mainly because they did not have time limitations at home. As for quality and relevance, treatment participants obtained higher scores than control students as well. Even though treatment group students were recommended extraneous templates, they overall identified more relevant templates as compared to the control group. However, the fact that the treatment group has obtained a mean efficiency of 76% makes it clear that students are relying heavily on the suggested templates (extraneous or not).

Group means were plotted in box plots for each requirement process (control vs. treatment) in Figure 11. Overall, treatment students performed better than control students across all four metrics.

Table 12 - Results for UCR18b

Factor ↓/ Metric →		Quality (1-5)		Coverage (0-1)		Relevance (0-1)		Efficiency (templates / min)	
		Means	P-value	Means	P-value	Means	P-value	Means	P-value
UCR18b	Treatment	4.4		0.9077		0.7607		0.6505	
	Control	3.375	0.137	0.4126	0.014	0.6825	0.219	0.3600	0.086

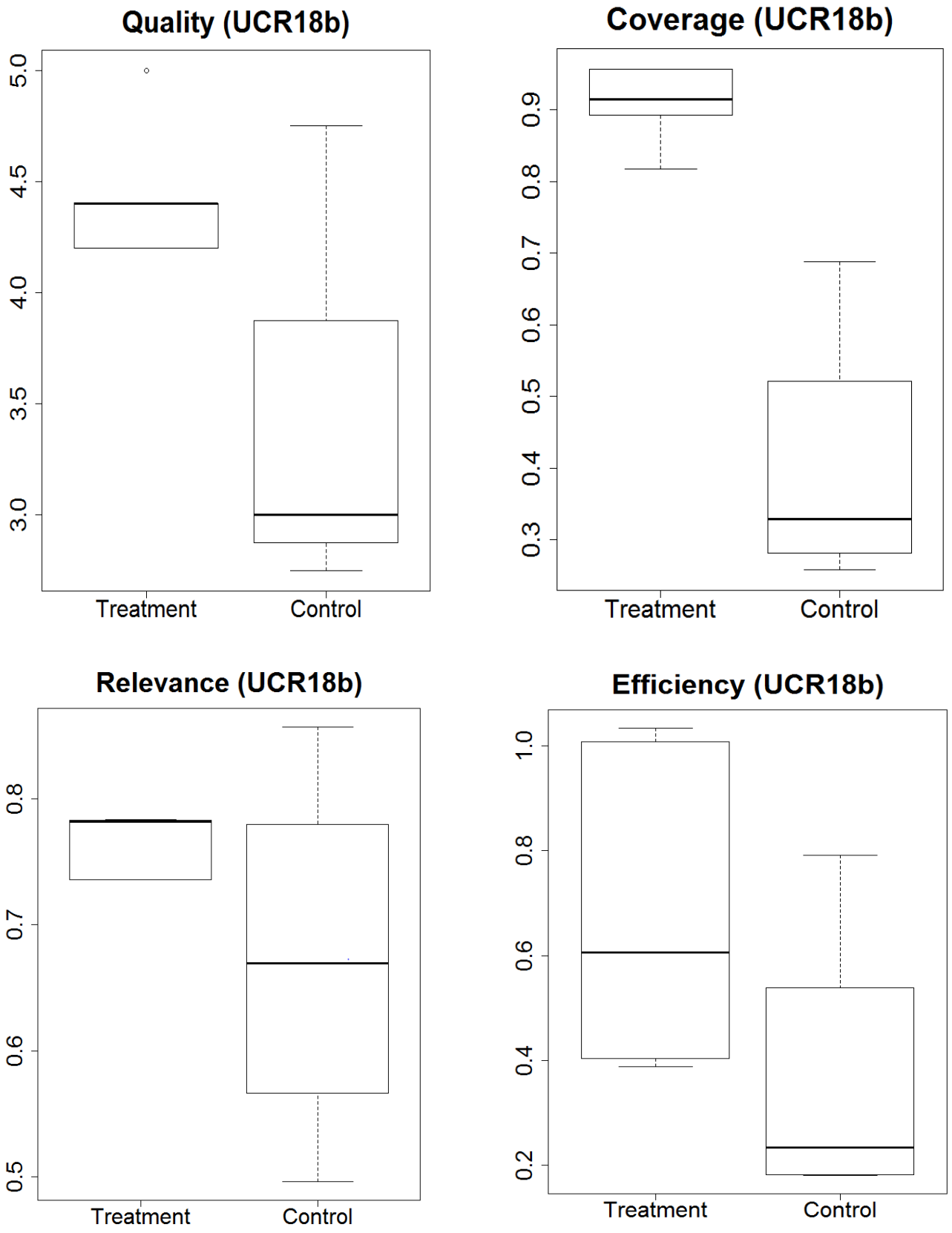


Figure 11 - Box-plots with results from UCR18b across each factor (treatment/control) and metric

6.4 – Summary of findings from individual studies

Mean metric scores from control and treatment group participants were calculated for each replication, in order to address research questions RQ1-RQ4. Table 13 summarizes the differences between the means obtained in treatment and control group for quality, coverage, relevance and efficiency (treatment mean – control mean). Additionally, the average scores for the mentioned metrics (detailed in Table 9) are plotted in Figure 12 (for UCR) and Figure 13 (for all studies).

The quality and efficiency of control group students in UCR18a were slightly higher than the scores obtained by their treatment group classmates. As for UCR18b, treatment group students outperformed control group students in all metrics. Based on the findings, research questions RQ1-RQ4 are addressed below:

RQ1: What is the quality of security requirements elicited through the use of automatically-suggested security requirements templates?

Of all the four metrics being evaluated in UCR15, UCR18a and UCR18b replications, quality varied the less in the treatment and control groups. We did not find any significant difference at p-value < 0.05 (nor at p-value < 0.1) between control and treatment groups in any of the replications. The use of automatically-suggested templates did not contribute to the quality of the elicited security requirements in these replications.

In UCR18a, the group with the highest average quality in its responses was the control group with 4.75/5, while the treatment group obtained 4.4/5. The difference was due to the level of detail used in the responses provided by some treatment group participants. Their security requirements were not as detailed as the ones reported by the control group. Moreover, 50% of the treatment students did not consider any other security templates, apart from the ones that were suggested to them. These aspects lowered their quality scores.

As for UCR18b, treatment students obtained a higher quality score (4.44 vs. ~3.4 achieved by the control group). In this case, 3 out of 4 participants in the control group did not follow the instructions. They did not apply the templates as expected and sometimes wrote security requirements in their own words. On the other hand, security requirements given by the treatment group were of high quality. Most of their answers needed a little more detail and consider more security templates in order to get a better score.

Moreover, treatment students in UCR15 performed slightly better on average than their classmates in the control group (3.94 vs. ~3.36). The scores obtained by the treatment group oscillated between 3 and 5 (two students obtained a 5), while the control group ranged from 2.5 to 4.

RQ2: What is the coverage of security requirements elicited through the use of automatically-suggested security requirements templates?

In both UCR18a and UCR18b replications, all treatment group participants obtained higher scores in coverage than their classmates in the control group. There is a significant difference observed between treatment and control groups at p-value < 0.05. Treatment participants in UCR18a identified 90% of the security templates in the oracle overall, while control group participants identified almost 41% of the templates in the oracle.

Two aspects from UCR18a's treatment group are noteworthy: firstly, they were not suggested any extraneous templates, which may be one of the reasons why they obtained such a high score. This means they could have been biased by this help and therefore may have not questioned the applicability of the recommended templates. In fact, one of the participants reported the 73 templates in the oracle, without adding new ones. Secondly, 50% of the students in this group applied templates which were not part of the oracle, and decided not to use some of the recommended templates as well. It can be inferred, as a consequence, that not all students who receive template suggestions will accept them blindly.

As for UCR18b, the results were very similar: students in the treatment group identified almost 91% of the templates in the oracle overall, while the control group identified 40% of the templates in the oracle. This study did include extraneous templates and even with this change, the percentage of coverage in the treatment group remained almost identical. Participants in the treatment group indeed questioned whether to apply the suggested templates or not.

UCR15's results differ significantly at p-value < 0.05 as well (63% for the treatment group vs. 36% for the control group). Apart from the three participants from the treatment group which obtained the worst coverage percentages (28%, 42% and 45%), the coverage of the remaining six participants in this group ranged from 55% to 91%. This contrasts with the percentages obtained by the participants in the control group: 4/7 students in the control group scored $\leqslant$ 34%.

Overall, it is worth remembering that most of these participants are non-experts in software security and this may explain why they missed some templates (especially the control groups, whose relevant percentages ranged from 21% to almost 69%). Apart from lack of security knowledge, time constraints may have also affected their coverage.

RQ3: How relevant are the security requirements elicited through the use of automatically-suggested security requirements templates?

In both UCR18a and UCR18b replications, the treatment groups overall obtained higher scores in relevance than their classmates in the control group. The treatment and control groups differ significantly at p-value < 0.05, for UCR18a. Responses from treatment participants in UCR18a were 97% relevant overall, while participants in the control group obtained a score of 37% in relevance. Not including extraneous templates in UCR18a may have led treatment students achieve higher relevance scores, compared to the control group.

As for UCR18b, no significant difference was found for the metric of relevance (67% for the control group and 76% for the treatment group) at p-value < 0.05. This may have happened due to a combination of various factors. Firstly, this study included the extraneous templates and it could have biased students in the treatment group. Secondly, participants in this control group performed way better than the control students in UCR18a (their mean relevance are 67% vs. 37%, respectively) and were evaluated with a more complete oracle (containing more templates). There is a curious case in UCR18b: the student who achieved the highest relevance comes from the treatment group (almost 86% of his reported templates were relevant) and obtained the lowest coverage (almost 26%). He only reported 28 templates (out of the 93 templates in UCR18b's oracle) and most of them were relevant. On average, this control group reported 64 templates, which doubles what this student reported.

In the case of UCR15, no significant difference was found for the metric of relevance either (71% for the treatment group and 59% for the control group) at p-value < 0.05.

As stated in the explanation of RQ2, lack of security knowledge and time restrictions may have had an impact on the relevance scores of all UCR15, UCR18a and UCR18b participants.

RQ4: How efficient is the process of eliciting security requirements elicited through the use of automatically-suggested security requirements templates?

No significant differences were found (at p-value <0.05) for the metric of efficiency, in UCR18a and UCR18b.

In the case of UCR18a, the control group was slightly more efficient than the treatment group (~66% vs. ~60%). Based on the results, time does not influence much in this finding: treatment students lasted on average 1.25 minutes less than the control group finishing the task (115.5 vs. 116.75, respectively). They all lasted between 93 and 130 minutes completing the task (in-class). The number of reported templates is what makes

a difference: on average, participants in the control group reported almost 11 more templates than the students in the treatment group. This is why the control group was more efficient. Probably the fact that they did not receive template suggestions made them consider more templates than the treatment group.

As for UCR18b, the treatment group more efficient than the control group by a larger margin (~69% vs. 36%, respectively). As mentioned before, this difference is not significant at $p < 0.05$, but is significant at $p < 0.1$. Contrary to UCR18a, participants in the treatment group took longer to finish the task. On average, treatment students lasted 198.6 minutes, while the mean duration of the control group was 187.75. Treatment students lasted almost 11 minutes more than their classmates in the control group. Nevertheless, what really helped this treatment group be more efficient was the quantity of templates its participants reported. The amount of templates reported by its 5 participants ranged from 106 to 121. This is higher than the quantity reported by three out of four participants (28, 44 and 55). One participant from the control group reported the most templates in UCR18b: 129. He achieved the highest coverage of his control group (almost 69%), but was the least relevant of his group and all UCR18b (about 50% of his suggested templates were relevant).

On the other hand, a significant difference was found in UCR15 at p-value <0.05 for the efficiency metric. The treatment group was ~82% more effective than the control group (0.20 vs. 0.11 from the control group). The elapsed task time did not have an influence on these results, due to the fact treatment students lasted ~101 minutes and the control students took ~103 minutes on average. It may be inferred that control group students did not understand the use cases properly, which could have lowered their efficiency score. This, combined with the fact treatment group relied on automatically-suggested templates, may explain the difference in efficiency between both groups.

As mentioned before and leaving aside RQ4, mean scores for all metrics were graphed to compare between UCR studies in Figure 11. All UCR treatment group participants demonstrated a better performance in the metrics of coverage and relevance, with more conclusive results for coverage. The difference in relevance between UCR18a groups is easily perceived. Control groups did better only in two occasions and by a little margin: in quality and efficiency, at UCR18a.

Taking all studies into account, these are still the only two times in which a control group performed better than a treatment group, as seen in Figure 12. The quality metric is the one that has benefited the least from the use of automatically-suggested templates overall. The original study (NCSU13) had almost equal quality scores for control and treatment students, while NSCU14 and UCR18b seemed to have benefited the most in this area. As for coverage and relevance, the trend observed in UCR studies is maintained: all treatment groups excelled and some of them did by a larger margin. It is particularly promising to see that extraneous templates suggestions in NCSU14, UCR15, UCR18a

and UCR18b did not affect its treatment students as much. This means they are being critical on whether or not to use a security requirement template, instead of accepting and applying the suggestions without hesitation. In terms of efficiency, treatment students in experiments prior to UCR15 benefited the most from the aid of automatically-suggested templates. This may have happened because of the way efficiency is measured in these studies (elicited security requirements/min), compared to the one used in UCR studies (reported security requirements templates/min). Time is an important factor as well. Mean time was considerably higher in UCR studies, ranging from ~102 to ~193 minutes, while mean times in NCSU and UT studies did not exceed 50 minutes.

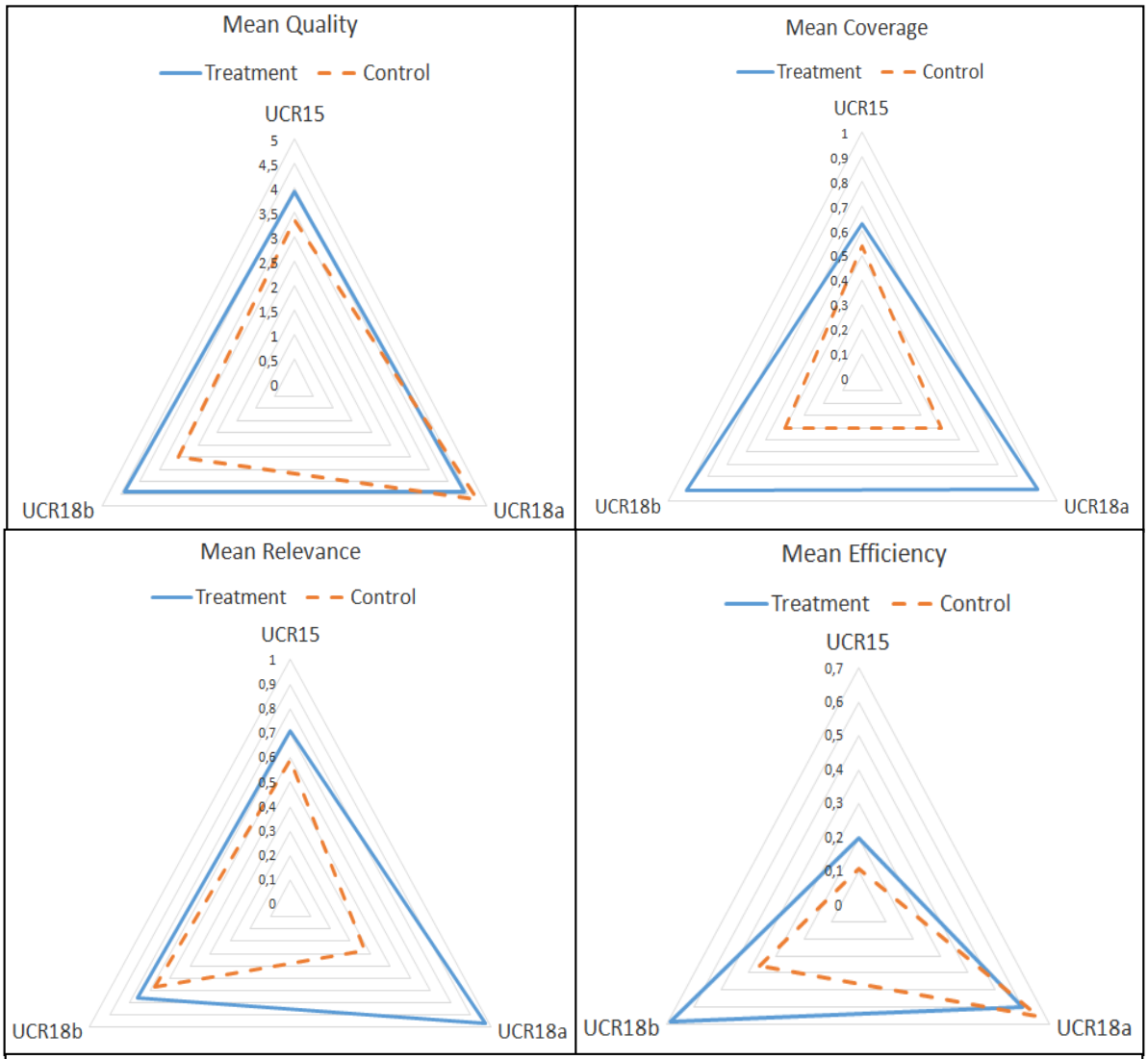


Figure 12 - Mean scores for treatment and control groups across UCR15, UCR18a and UCR18b studies

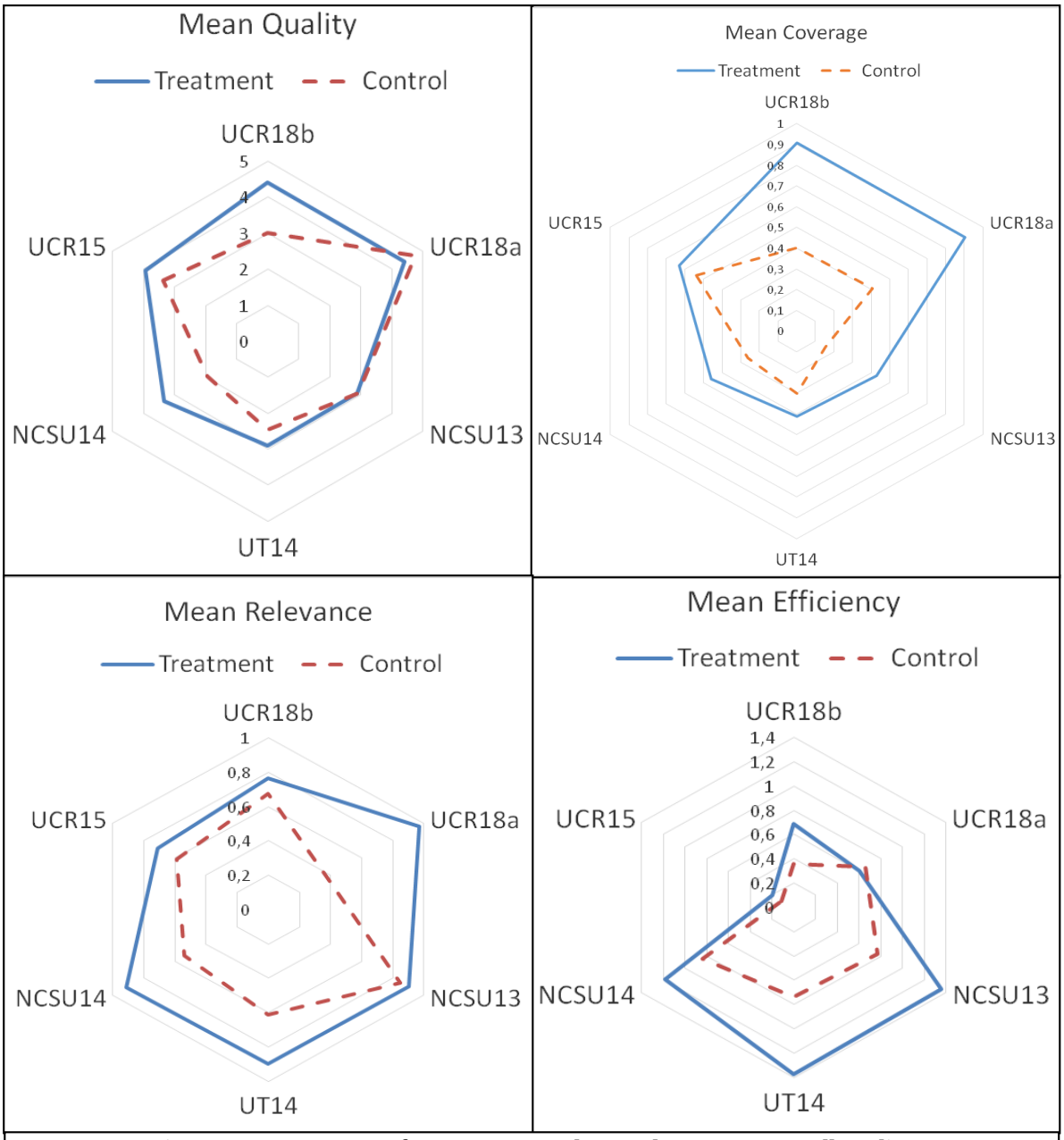


Figure 13 - Mean scores for treatment and control groups across all studies

Table 13 - Difference between treatment and control group means across studies

Study	Δ Quality (-4 to 4)	Δ Coverage (-1 to 1)	Δ Relevance (-1 to 1)	Δ Efficiency (-2 to 2)
NCSU13	-0.03	0.27 **	0.05	0.58 **
UT14	0.44	0.11 *	0.29 **	0.63
NCSU14	1.38 **	0.20 **	0.37 **	0.34 **
UCR15	0.58	0.27 **	0.12 *	0.36 **
UCR18a	-0.35	0.4966**	0.5996**	-0.0594
UCR18b	1.025	0.4951**	0.0782	0.2905*

* Significant at p-value < 0.1; **Significant at p-value < 0.05

6.5 - Results based on analysis across studies

A summary of the results from UCR15, UCR18a and UCR18b is presented below. A comparative analysis of the three studies regarding the metrics of quality, coverage, relevance and efficiency has been made. Even though there are differences between studies (oracles / use cases, in-class vs. take-home activity and the number of participants overall and per control and treatment group, among others), we can take the raw data from these three studies and apply a Mann-Whitney U test to test the four null hypotheses. As in previous analysis, results with $p < 0.05$ obtained from this combined UCR data are considered statistically significant.

Moreover, a qualitative analysis of UCR18a and UCR18b studies is presented in Section 7.2 to explore differences introduced among these two studies.

6.5.1 – Combined data analysis

The raw data obtained from UCR15, UCR18a and UCR18b was combined to perform an overall analysis of the three studies. The authors applied a Mann-Whitney U test for all treatment (18) and control (15) students in these studies. These results may be affected by the data obtained in UCR15; 16 out of the 33 participants come from this study. They make up almost 50% of the participants involved in UCR findings.

Based on the results obtained from the Mann-Whitney U test, as detailed in Table 14, it is evident that treatment participants obtained higher scores than their classmates in the control group for all metrics, with significant performance regarding coverage, relevance and efficiency. Therefore, it is possible to reject the null hypotheses H_{02}, H_{03} and H_{04}. Students assigned to the treatment group identified more requirements (~77% vs. ~39% for the control group). Their coverage was ~38% better and their responses were

~22% more relevant (~78% for the treatment group vs. ~56% for the control group participants). Besides, students in the treatment group were more efficient in the identification of security templates (~71% vs. ~48%) as well.

With a p-value = 0.114, no significant difference was found for the metric of quality. Hence, we fail to reject the null hypothesis H_{01}. The average quality score for all treatment students in the three studies was ~4.2, compared to a 3.73 of the control group students. None of the studies could individually reject this null hypothesis.

Group means and variances are provided in Table 15 for each metric across control and treatment groups of all UCR studies.

Table 14 - Results for combined UCR studies (treatment vs. control)

Factor ↓/ Metric →		Quality (1-5)		Coverage (0-1)		Relevance (0-1)		Efficiency (templates / min)	
		Means	P-value	Means	P-value	Means	P-value	Means	P-value
All UCR studies	Treatment	4.1833		0.7674		0.7838		0.7095	
	Control	3.7333	0.114	0.3867	0.000	0.5568	0.000	0.4797	0.015

Table 15 - Group means and variances for the combined data from all the studies

			Control			Treatment		
Study	Oracle / Use case	Metric	# of Obs.	Mean	Std. dev	# of Obs.	Mean	Std. dev
UCR 15	1	Quality	4	3.50	0.58	5	4.10	0.74
		Coverage	4	0.41	0.11	5	0.65	0.24
		Relevance	4	0.62	0.13	5	0.69	0.13
		Efficiency	4	0.41	0.14	5	0.70	0.32
	2	Quality	3	3.17	0.76	4	3.75	0.96
		Coverage	3	0.30	0.10	4	0.61	0.23
		Relevance	3	0.56	0.14	4	0.74	0.04
		Efficiency	3	0.49	0.18	4	0.90	0.22
UCR 18a	3	Quality	4	4.75	0.204124145	4	4.4	0.461880215
		Coverage	4	0.407534247	0.197049291	4	0.904109589	0.110724709
		Relevance	4	0.375320922	0.095136466	4	0.974916213	0.029000723
		Efficiency	4	0.659076729	0.159932377	4	0.599665758	0.138757465
UCR 18b	3	Quality	4	3.375	0.924211376	5	4.44	0.328633535
		Coverage	4	0.412646096	0.192958071	5	0.907709906	0.057732003
		Relevance	4	0.682592816	0.134597876	5	0.76071764	0.023765086
		Efficiency	4	0.360090908	0.29161825	5	0.650531256	0.272996157

6.6 – Qualitative analysis based on differences introduced among studies

UCR15, UCR18a and UCR18b studies varied in many aspects: quantity of participants, time, setting, problem domain, the way in which the requirements/uses cases were presented to participants and the support they received when performing the task. These differences are documented in Table 5. Qualitatively, it is possible to evaluate if these factors had an impact on RQ1-RQ4, by answering the following questions (Riaz et al., 2017):

1. Are participants more inclined to fill in the templates when additional support to fill the templates is provided by explicitly indicating subject, action and resource elements in the input requirements?

It can be hypothesized that giving additional support to treatment students by explicitly stating the subject, action, services and resources to them will encourage them more to fill in the templates, thus leading to a higher percentage of treatment group students willing to apply and fill in the security requirements templates. Out of the mentioned replications, only UCR15 offered this help and 100% of its treatment students filled them in. They had strong motivation because it was a graded class work. As UCR15, UCR18a was planned by the authors as an in-class activity. They had strong motivation because this activity counted heavily towards their course grade. In this case, they did not receive the extra help and all treatment students filled in the templates. It can be inferred that having limited time to finish the task, combined with the motivation, encouraged these students to really understand how to fill in the templates and rely on them, in order to obtain the highest grade possible. The same behavior was observed in UCR18b: all treatment students filled in the templates. In both UCR18a and UCR18b, motivation was a bigger factor than the additional support in determining if they were willing to fill them in or not. Having additional support did not serve as a motivation for students to fill in the templates.

Talking about the mean quality score in UCR18a, it was slightly higher in the control group (4.75 vs. 4.4). They used the templates as well and this may have happened due to the pressure these students had to complete the task in less than 150 minutes, without any help at all. Nevertheless, the average quality score of treatment group participants in UCR18b was moderately better than that obtained by its control students (4.44 vs. 3.375, respectively). This may have happened, in part, because some control group participants did not apply the templates well. Their efforts were mapped by researchers to the templates that had more resemblance to them. Therefore, the proper use of templates may be linked to higher quality responses.

2. Can participants differentiate whether a suggested security requirements template is relevant to the given use case scenario?

It can be hypothesized that treatment group participants can correctly discern if a suggested template applies or not for each of the functional requirements.

Participants from the treatment group in UCR18b were intentionally suggested both relevant and extraneous templates for each functional requirement. A total of 32 extraneous templates where suggested among the 14 functional requirements. Between 1 and 4 extraneous templates were suggested per functional requirement. According to the relevance scores shown in Table 8, the responses given by the students in the treatment

group were 76% relevant, on average. If this percentage is compared to the mean relevance obtained by treatment students in UCR18a (97%, as depicted in Table 7), we can see treatment students of UCR18b gave 21% less relevant answers than the treatment group in UCR18a. This means the majority of treatment participants from UCR18b are heavily relying on the suggested templates and may have blindly applied some of the extraneous templates as well.

We also counted how many extraneous templates were selected by UCR18b treatment participants. They ranged from 23 to 32 templates, which is a high number. The average count of extraneous templates selected by these participants is 27.4 templates, out of 32. This means that, on average, they decided to use ~86% of the extraneous templates in ~199 minutes at home. Treatment students from UCR18a lasted on average ~116 minutes (83 minutes less), without extraneous templates being suggested to them. Therefore, we can infer from treatment students in UCR18b they really took their time analyzing which templates were applicable, instead of filling in all the templates they were recommended and finishing the task earlier. Even though all participants were instructed to fill only the templates they considered appropriate to solve the security needs of each functional requirement, some of them may have considered extraneous templates as valid suggestions as the rest of suggested templates which were indeed relevant.

3. Are there context factors, such as more time on task, which are conductive to producing better outcomes overall?

Differences in context factors, such as the possibility to give students more time to finish the task, may be the reason why variations in findings across studies may occur.

UCR15 was carried out as an in-class activity, in which its 16 participants had 180 minutes available to complete the activity. They needed ~102 minutes on average to finish it and hand in their responses. Taking both of its groups into account, their mean percentages for coverage and relevance are 51% and 66%, respectively. If these results are compared with the average mean scores of UCR18a (which was also an in-class activity), it is worth noting its participants had almost the same time to complete the task (150 minutes) and obtained 66% in coverage and 68% in relevance. Relevance did not change from one study to another (just 2%), but coverage increased 15% from UCR15 to UCR18a. In this case, a combination of both time and extraneous templates may explain these findings. UCR18a's students lasted on average 14 more minutes (~116 minutes in total) and this extra time may have helped them select more templates present in the oracle.

In the case of UCR18b, it was a take-home activity and students were told they had one week to complete it. Their time reports indicate their mean time was of ~193 minutes, with a mean coverage of 66% and 72% of their responses were relevant. Com-

paring UCR18a and UCR18b (which have NutriMetas' SRS as their common denominator), it can be inferred that the additional ~77 minutes UCR18b students lasted on average resulted in keeping roughly the same coverage (even with extraneous templates and being evaluated with a more complete oracle), increasing their relevance by 4% and their efficiency by ~15% (from ~63% to 78%). Additionally, working at home may reduce the sense of stress and pressure to finish on time that students live in a university environment, under test conditions. The only variable that decreased from UCR18a to UCR18b is quality (4.6 to 3.90). In the long run, quality may be negatively affected after working in the activity for such a long time. The wording used to fill in the templates may not be as detailed as the elapsed time advances.

A factor "that may influence the time spent on task is the expected credit or reward" (Riaz et al, 2017, p. 2162). Contrary to NCSU13 and UT14 (which gave coursework credit to students irrespective of the quality of their responses), UCR15 and UCR18a represented an important graded activity which significantly counted towards their final grade (25%). Therefore, participants had a great motivation to provide well-thought and high quality answers. UCR18b represented a lower percentage, due to the fact the activity was going to count as homework. Nevertheless, its participants proved to be motivated because they took on average the most time than the students in the original experiment and all previous replications.

Two more context factors inherent to UCR18a and UCR18b must be discussed. The first one is the use of an SRS based on based on the ISO/IEC/IEEE 29148:2011 standard. This is the first time the NutriMetas mobile application is considered for conducting a replication and, also, no other previous study has presented the requirements following this standard before. The fact that these functional requirements adhere to this standard facilitates their understanding, given that their wording, structure and level of detail remain uniform. The standard provides additional information to the reader regarding the context of the application and its purpose, a detailed description of each of the 13 functional requirements with its prerequisites, inputs, its process and outputs. Such level of detail was not given to students in UCR15, as they only received general statements of one of the two use cases from the Cyclos software (UC1- Make payment and UC2- Retrieve account information). This may be the reason why UCR18b has been the replication with the overall highest coverage, relevance and efficiency to date. It can be inferred therefore that the more participants know about the software system under study, the better they will perform. The amount of information given about the goals of NutriMetas and its functional requirements is not extensive, nor should it be. The way the information is described by the format the ISO/IEC/IEEE 29148:2011 standard follows makes reading comprehension easier. The questions the students asked the researchers were never about NutriMetas' requirements. They were just to confirm they had understood the basics of security requirements templates.

The second context factor is the use of no online tools for data collection, analysis and time tracking. Riaz et al. (2014a) developed an online tool in which students

could access the use case the system randomly assigned to them, type in their responses and send them to the researchers. Additionally, it recorded how much time each student took completing the task. This tool had to be modified to work with the NutriMetas' SRS and since there was no time to do so, added to the fact that the opportunity to work with these groups of students was sudden, we opted for a simpler solution that could be ready in less time. Hence the idea of working with text documents seemed a good alternative and it worked. Two different text documents were prepared: one of them was handed to the students in the treatment group and another one for the control group. The document received by treatment group students included the recommended templates for each of the functional requirements. The online tool worked in the same manner for each of the statements in the use cases. On the other hand, the control group received a blank document with the 14 functional requirements and its participants had to type in the template(s) they were recommending below their corresponding functional requirement and fill them in. The same applied for the online tool. Both methods work very similar in these aspects and sending these files to the researchers via email did not present any trouble. The only potential drawback of this method, which can directly affect the efficiency variable evaluated in this study, is time tracking at home. We had to rely on the time they reported and may not be as accurate as having a tool automatically taking their time.

6.7 – Breakdown of Identified Security Templates

Riaz et al. (2017) groups security requirements templates by the objectives they support. For instance, functional requirements in NutriMetas' SRS related to logging transactions with sensitive data support the security objective of accountability. Therefore, for each of the oracles used in UCR18a and UCR18b replications, we counted the number of templates in them for each security objective. To illustrate, UCR18a's oracle applied 33 confidentiality templates throughout the 14 functional requirements. We also counted how many templates in the oracle were identified on average by the participants in both treatment and control groups for each security objective. These results are plotted in Figure 14, Figure 15 and Figure 16.

In UCR18a, the treatment group obtained higher scores in all security objectives, compared to the control group. Their responses were closer to the oracle. Confidentiality and availability templates were the most used among both groups. The control group reported a high number of integrity and accountability templates, as compared to the oracle. UCR18a did not include extraneous templates, which is an important aspect to take into account.

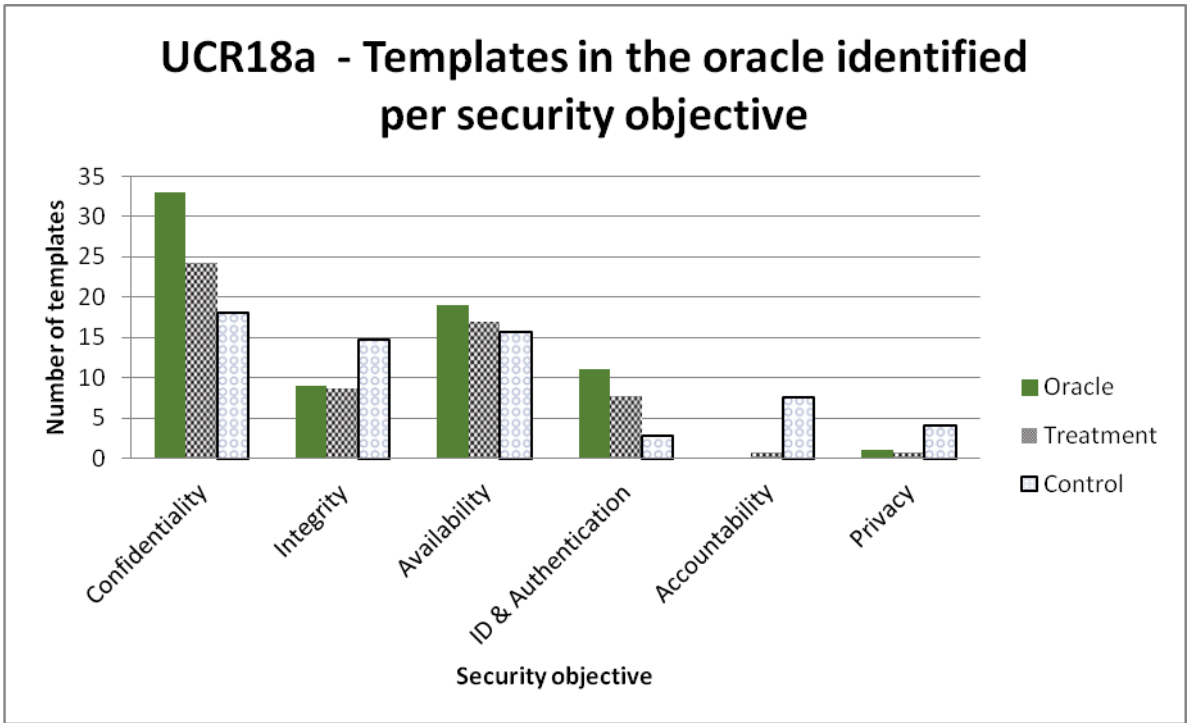


Figure 14 - Templates in UCR18a's oracle identified per security objective

As for UCR18b, the treatment group outperformed the control group in 4/6 security objectives: confidentiality, integrity, ID & authentication, accountability and privacy. Similar to UCR18a, confidentiality and availability templates were the most used among both groups. One possible reason that could explain why the answers from treatment group did not excel as much as the control group, in terms of availability and ID & authentication, is the use of extraneous templates. About 63% of the extraneous templates suggested belong to these two security objectives, which mean treatment students are relying on these suggestions. A fewer percentage of these extraneous templates are integrity templates, but they did not affect as much. Therefore, it can be inferred this is why the control group was closer to the oracle in these two security objectives.

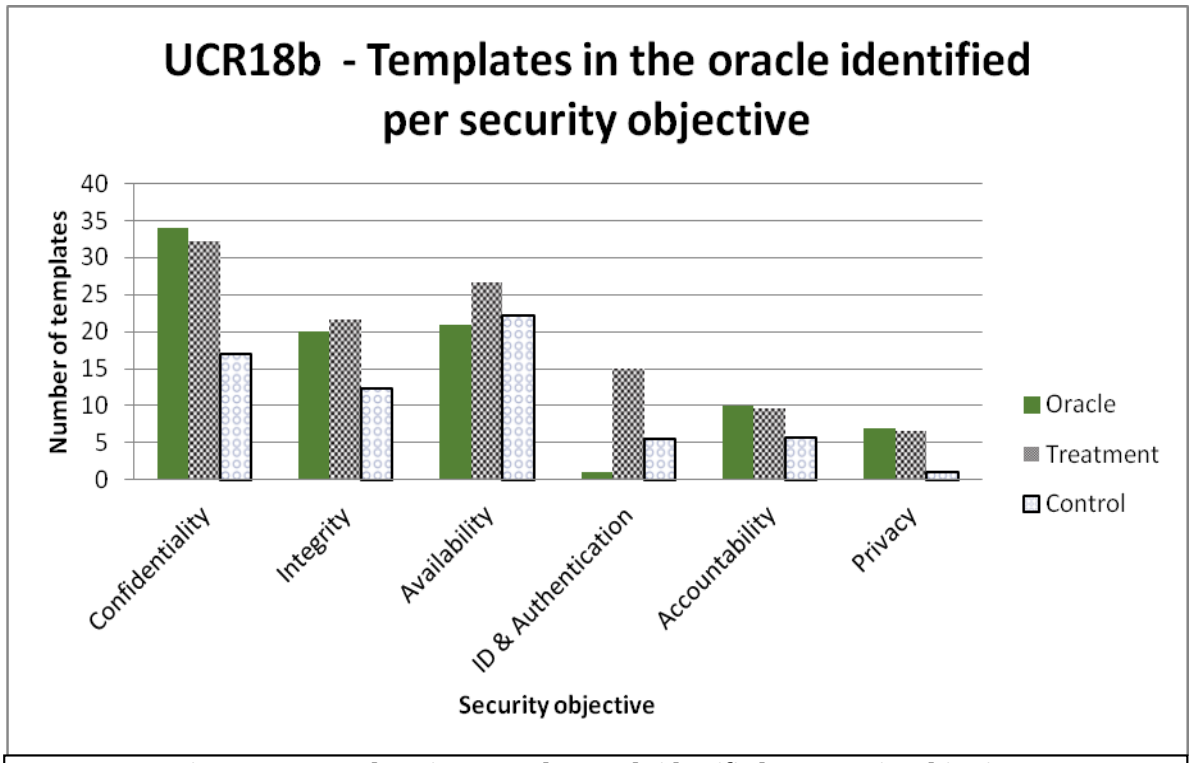


Figure 15 - Templates in UCR18b's oracle identified per security objective

In the case of UCR15, only 4 security objectives were evaluated in both mobile banking use cases. The responses from the treatment group were closer to the oracle than the ones reported by the control group in all security objectives. It is worth noting that this study involved the use of an oracle in which security requirements were detailed, not oracles as UCR18a and UCR18b. This is why the graph from UCR15 is titled "Requirements in UCR15's oracle identified per security objective", as shown in Figure 16.

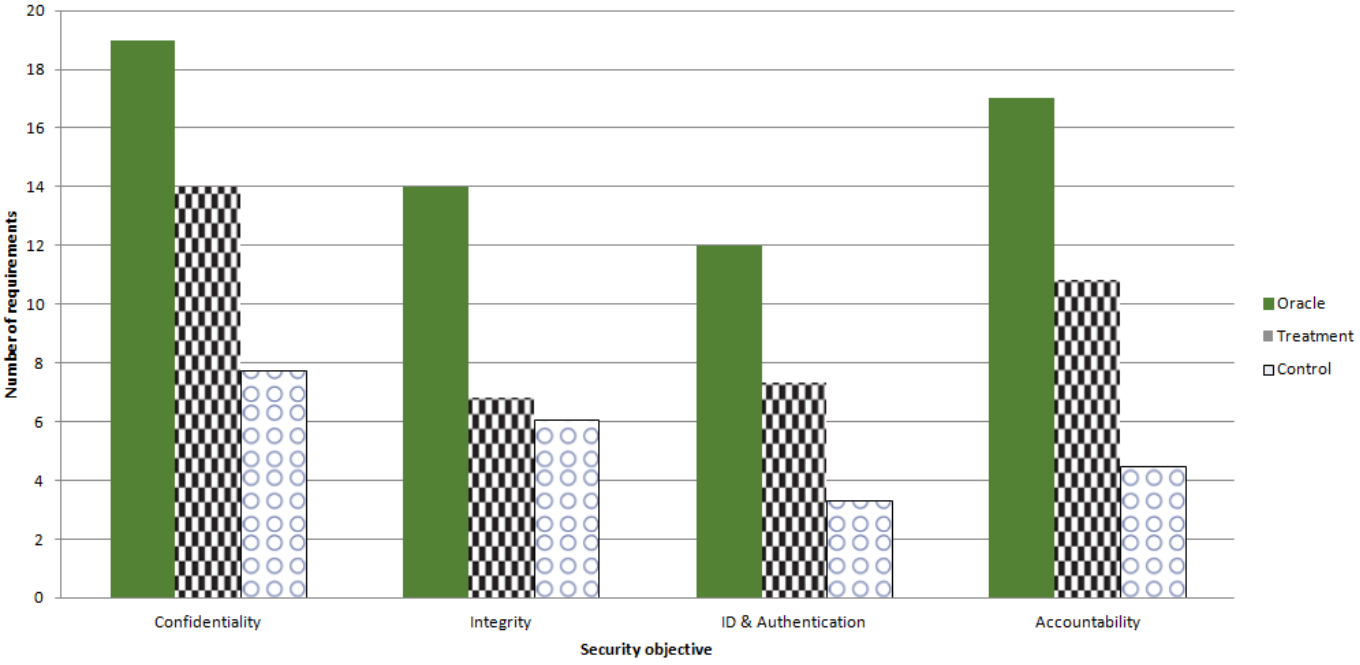


Figure 16 - Requirements in UCR15's oracle identified per security objective

6.8 – Feedback from participants

A questionnaire was given to all UCR15, UCR18a and UCR18b participants to fill after they completed the task. Feedback regarding the use of security requirements templates was asked. It was voluntary for all the studies and 31/33 participants agreed to give their opinion. Two students from UCR18b did not provide feedback.

Their opinions were translated to English and categorized (if possible) following the response categories used by Riaz et al. (2017). UCR15's responses were taken from Riaz et al. (2017) and included in the response frequency per category graph plotted in Figure 17, along with the feedback UCR18a and UCR18b participants reported:

- Provide a good starting point (19%): templates serve as an initial and basic approach; handy for the initial stages of requirements elicitation; useful for developing secure systems;
- Good coverage and applicability (19%): the six security objectives cover the main areas of concern; semantic analysis of the statements makes the process easier;
- Help in thinking about security (27%): allows non-security experts to consider more security requirements; can help security experts as well as novices; offers security requirements that only experts could recommend;
- Help in phrasing requirements (19%): improves requirement eliciting process; useful method for organizations that do not have a standardized requirements elicitation process;
- More templates and support (10%): more templates would help; reference material should include more examples regarding the use of security templates;
- Apply with caution (6%): applying the templates correctly demand the need of prior training; might make the practitioner follow the recommendations as a checklist, without critical thinking;
- Not used / answered (6%): the percentage of participants who did not give feedback on the use of security requirements templates.

In order to understand the percentages obtained, it is important to analyze Figure 17 and read the following explanation. For instance, 6 participants (3 from UCR15, 2 from UCR18a and 1 from UCR18b) mentioned (in their own words) templates "provide a good starting point". Therefore, 6/31 participants (or 19% of those 31 participants) gave feedback that relates to that category. Riaz et al. (2017) defines what to look for in participants' responses, in order to categorize them into one or more response categories. In this example, the responses of these 6 participants included aspects related to templates serving as an initial approach and being handy for the requirements elicitation stage.

Overall, students primarily argue templates help in thinking about security (27%). This may have been the most recurring response because almost 80% of the participants are software security novices. Additionally, three response categories are tied in the second place, with 19%: provide good coverage and applicability, help in phrasing requirements and provide a good starting point.

On the other hand, "more templates and support" and "apply with caution" response categories were mainly mentioned by UCR18b participants. Basing on the work and academic experience participants reported in Table 4, UCR18b holds the most experienced participants in security and this may explain why some of them gave feedback related to these response categories. Experts may tend to judge more the use of security templates and be more aware of the advantages and disadvantages of them, compared to non-expert participants which may not question the methodology as much.

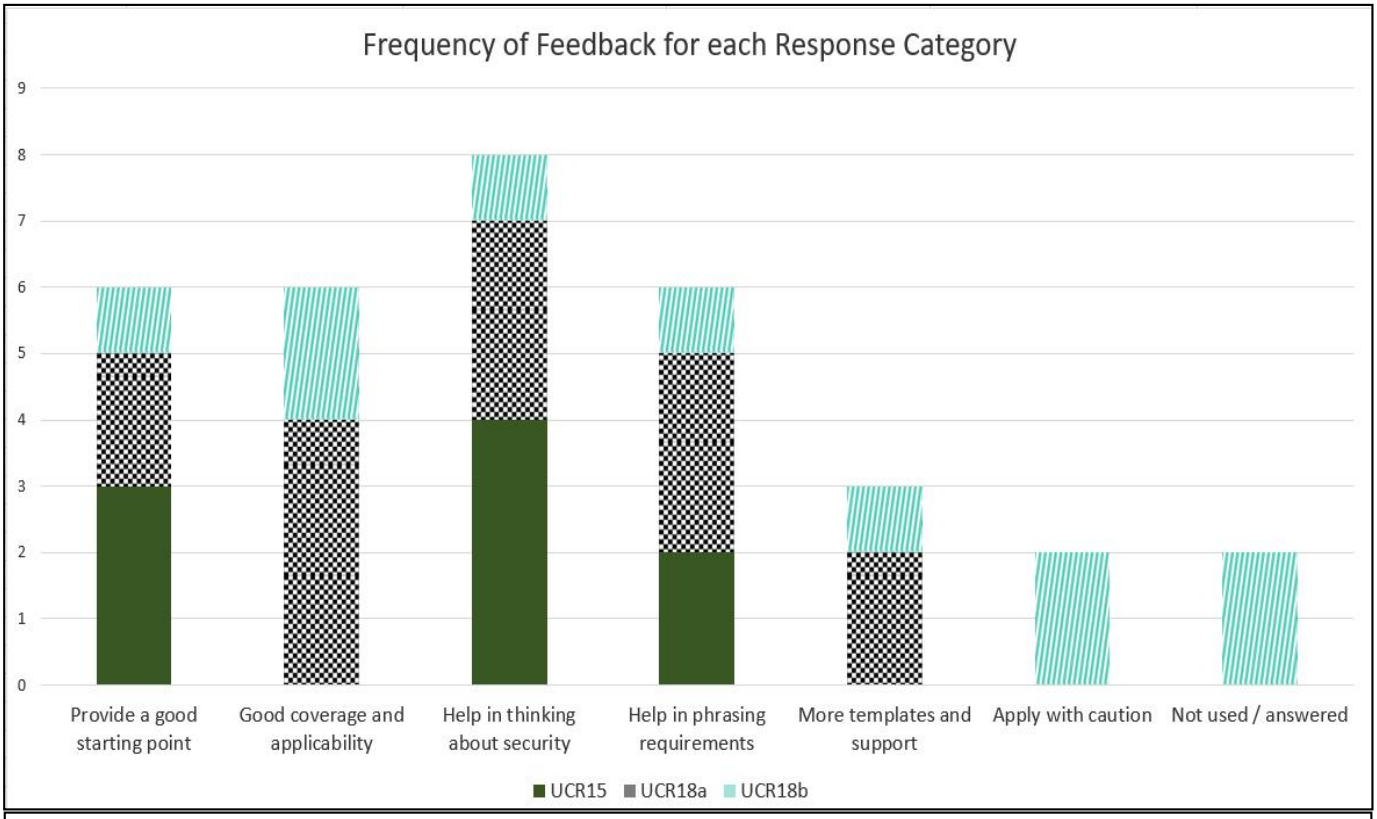


Figure 17 - Feedback related to the use of security requirements templates

6.9 – Lessons learned conducting the replications

Below are some of the lessons learned while planning, executing and reflecting on UCR18a and UCR18b controlled experiments. This analysis is based on the same five sections Riaz et al. (2017) reported.

6.9.1 – Communicating with the original experimenters

The three replications were conducted with the assistance of Maria Riaz, the original and main experimenter. Maria et al. provided the experiment material and insight on her online tools (such as SD and the one used before UCR18a to save participants' responses and track their time). Christian Quesada-López, one of the researchers who collaborated with the author, was in charge of UCR15's replication as well (the first to measure efficiency by the quantity of security requirements templates per minute, as UCR18a and UCR18b) and maintained communication with Maria Riaz over email regarding the planning of these experiments.

Based on the experience replicating in Costa Rica, maintaining close communication with the original and/or previous researchers is necessary to conduct successful replication studies. It is vital to have a deep understanding of the context factors related to the original experiment (NCSU13) and subsequent replications, in order to analyze and interpret the findings in further studies. Tacit knowledge gained by Christian Quesada (after collaborating with Maria Riaz and conducting UCR15) proved to be useful throughout all the stages involved in these replications.

Moreover, the author was part of the students which were evaluated in UCR15. Having been part of this research in the past was very helpful in understanding what a participant faces before, during and after the experiment, particularly with the online tool. Feedback and observations from the researcher were received back them and this way of work could only be achieved by learning from them. Therefore, it is important to make experiment material and artifacts available for future researchers that would like to replicate Maria Riaz's original experiment. Context factors (as mentioned before), differences between studies and observations should be communicated in order to have a better understanding of future findings.

6.9.2 – Minimizing technical setup

UCR18a and UCR18b replications were the first ones in which the online tool was not used. The idea of having "contingency plans, such as availability offline options to complete the task" (Riaz et al., 2017, p. 2167) were previously mentioned in case the tool was not available. In our case, the opportunity to work with these students was so sudden that it was necessary to develop an offline data collection option from scratch, due to the fact the online tool was still not put in place in UCR at the time being. The online tool

had to be modified in order for it to work with NutriMetas' functional requirements (it had been originally developed to work with VLER, EHR and Cyclos use cases) and there was no time, so the authors opted for an offline and manual data collection approach. Therefore, there was no need of technical setup for these replications.

The only device each participant needed to access text files in which they were told to type in their responses was a computer. Hard copies of these files were available as a contingency plan, but there was no need to use them. Students did not report problems regarding this data collection method. The only potential drawback of this method, which can directly affect the efficiency variable evaluated in this study, is time tracking at home: unlike UCR18a (where it was possible to keep a personal and more accurate record of the start and end time of each student), we had to rely on the duration time reported by each of the participants in UCR18b.

6.9.3 – Managing and reporting emerging contexts

"Replicating a study, even with all the experimental material and tools available, requires the acquisition of a considerable amount of knowledge" (Riaz et al., 2017, p. 2168). It is important as a researcher to study the original experiment, understand its findings and learn how to use its tools in an appropriate way, in order to replicate it in a good manner.

The UCR18b replication was originally intended to be an in-class activity. Nevertheless, the professor in charge of UCR18b students preferred to arrange this experiment as a take home activity for them, due to schedule limitations. Taking into account only the studies conducted in UCR, the one that yielded the best results in efficiency (templates/min) was UCR18b (see Table 9). To delve a little more, within the replications outside of UCR, the second best in efficiency (requirements/min) was that of UT14, as shown in Table 9 as well. Similar to UT14, UCR18b's "findings can be interpreted in the context of a take home activity where participants may have more time overall to work on the task but not dedicated time for the task" (Riaz et al., 2017, p. 2168). It would be of great interest to see to what extent the nature of future take home activities influence the four variables (quality, coverage, relevance and efficiency).

Language was reported as an emerging context before (Riaz et al., 2017). For UCR18a and UCR18b replications, participants provided responses in Spanish and English. The students were told they could respond in either of the two languages, making sure they wrote down which template they were referring to each time. For researchers, this was important because this way answers could be mapped to a specific template, in case the wording of the answer was strange and it was not clear to which template the participant was referring to. These cases arose and therefore it was possible to deal with such ambiguity.

6.9.4 – Developing shared insights

"Each group of researchers had unique insights into the study they conducted" (Riaz et al., 2017, p. 2168) and it is important to "to explore potential reasons for similarities and differences observed across studies" (Riaz et al., 2017, p. 2168) as well.

As mentioned before, UCR18a and UCR18b replications were the first ones in which the online tool was not used. PDF files were handed to students for them to read NutriMetas' functional requirements and activity instructions. Text documents were created for them to type in their responses. The questionnaire at the end of the activity (in Spanish, as most of UCR students are native Spanish speakers) was also presented as a text document which they digitally edited. They saved their work and sent it via email to the researchers. This setup proved to be a great offline alternative for small groups with a single use case/SRS, but can be deficient with large groups of students because data collection and analysis may become tedious. In case resources are available and there is time to modify the online tool to fit the needs of a future replication, it is advisable to use it both with small groups, as with large.

Another change that was introduced was the way in which the requirements were detailed. NutriMetas' SRS is based on the ISO/IEC/IEEE 29148:2011 standard. The fact that these functional requirements adhere to this standard facilitates their understanding, given that their wording, structure and level of detail remain uniform. The use cases, on the contrary, may be difficult to understand and this ambiguity may lead to an inadequate identification of subjects, actions, resources and other aspects requested to the participants by these templates. For future replications, it is important to make sure that students understand the context of the software application and its functional requirements, or the process flow (if a use case is given to describe the software application under study).

Suggestions to fill in templates were not given in UCR18a and UCR18b replications. For treatment groups, the online tool suggests (inside square brackets) how to fill the empty fields of the security requirements templates as shown in Figure 6. We decided not to include this help to force the participants to be more critical when thinking about how to fill the templates. Maintaining that help could make them more dependent on the suggestions and one of the ideas behind this experiment is that they can learn to apply the method in the best way possible, without being biased by what the tool recommends. Treatment group students are already getting enough help by receiving template suggestions. In general, students of both replications were able to correctly identify the elements requested by the templates and therefore it could be inferred that this help was not necessary. Participants who did not obtain a 5/5 rating for the quality question "have all necessary elements of the requirements been identified?" (all of them were part of the control group) was because they did not follow the instructions given by the researchers, not because they did not receive suggested templates. These control group students did not rely faithfully on the wording of the templates. Hence it is vital to make sure control groups understand they cannot modify the templates at will.

6.9.5 – Working with diverse groups of participants

Participants in all UCR replications were graduate students enrolled in the same Master's degree, but enrolled in different courses. Differences in previous knowledge regarding their IT background (work, academic and software security experience) may exist and these aspects were not factored into our results. Due to language differences and even though all participants knew English, researchers had to prepare a 15-minute explanatory presentation in Spanish for all participants in order to make sure they understood Riaz's reference material and the process they had to follow during the activity. Due to the fact students in control groups responded in Spanish and all of them provided feedback in this same language, responses had to be translated by the researchers when needed.

Unlike UCR18a students, UCR18b participants inevitably did realize the existence of two different groups (control and treatment) right after the 15-minute explanatory presentation. Their professor uploaded the control and treatment zip files to the course website and students had to be assigned to control and treatment groups immediately afterwards, so that it was clear for them which files they had to work with at home. This may have created a sense of uncertainty and curiosity about it. One of the students asked why and another one inquired if some of them were going to have to work more than others. One of the researchers gave them an answer with little detail, so as not to bias them before they carried out the activity at home. They were promised that they would eventually know why and they were asked again to please follow the instructions as stated.

6.10 – SD methodology limitations

Software security encompasses many phases and activities that, together, help create secure systems. The identification of security goals or objectives, the elicitation of security requirements, the implementation of an information security governance framework (which include administrative aspects such as policies, standard, procedures and guidelines), as well as risk and threat analysis, are some of the tasks that must be taken into account. Operational and technical aspects such as database protection and security controls (backup controls, memory protection, access controls, password protection techniques) are also vital. New threats arise every day and both software systems and its staff must be prepared to respond to incidents and follow a disaster recovery plan.

The scope of this work only focuses on a methodology which covers one of the many topics related to software security: identification and generation of software security requirements. It is difficult to elicit security requirements and SD may assist in this process. By no means it intends to offer "complete" security for a software system or application. The results obtained by applying this methodology will depend on the level of security the developers and security novices/experts want (or consider necessary) for the system. For example, a software system may not specifically demand the need of a certain security requirements template and it will depend on the practitioner if it is applied or not. Maybe he/she will think about it for a few minutes, before taking a decision. Another practitioner may have no doubt about it, regardless of what he/she decided. Therefore, the practitioner has the last word and it is worthwhile mentioning that this decision

corresponds to another software security activity. SD, in this context, must be seen as an optional aid. It must not be blamed for an incorrect or insufficient security requirements elicitation, due to the fact it does not cover all security aspects.

Another point that must be clarified is that the quality of the generated security requirements (leaving the expertise with SD aside) depends on how clear the functional requirements of the software system are. Security requirements templates have blank spaces for the practitioner to fill in and, according to the methodology, their input comes from the functional requirements. If these functional requirements are ambiguous, imprecise security requirements may be generated. This situation could get worse if the practitioner has limited knowledge on the software system to which the security requirements belong.

Riaz states various limitations Security Discovery has in her doctoral dissertation (Riaz, 2016b). Even though prior replications have demonstrated the feasibility of the SD methodology in the identification of security requirements implied in functional requirements, it is important to understand its scope and applicability in different scenarios (new problem domains and the complexity of the software systems).

An industrial case study in the networks domain was conducted by Riaz and her research team, in which the SD process was carried out in order to identify security requirements for a real-world network system (Riaz, 2016b). Two requirements artifacts were provided by Cisco Systems, Inc. for this study (Riaz, 2016b):

- Product requirements document (PRD). It states the functional requirements for the system. This document served as input for the SD methodology and it contains 1218 sentences.

- Product security baselines (PSB). This is the name given to the document which outlines the security requirements for the networks system. A total of 108 security requirements were defined. This document was used to map and validate the current security requirements with the generated security requirements using SD.

For the effects of this experiment, one initial automatized mapping took place and covered 70 security requirements (65% of the security requirements in the PSB) (Riaz, 2016b). After the initial mapping, two subsequent manual mappings were conducted to revise the security requirements in the PSB that had not been completely covered by SD. The first manual mapping covered 103 (~95%) and the second one covered 105 (97%) of the 108 security requirements (Riaz, 2016b). Table 16 shows the breakdown of the coverage of security requirements in the PSB, based on the generated security requirements applying SD (Riaz, 2016b):

Table 16 - Coverage of security requirements patterns for security requirements in PSB

	Initial mapping	First manual mapping	Second manual mapping
X: Covered	70 (65%)	103 (~95%)	105 (~97%)
P: Partially Covered	11 (10%)	0 (0%)	1 (~1%)
N: Not Covered	15 (14%)	5 (~5%)	2 (~2%)
NA: Not Applicable	12 (11%)	0 (0%)	0 (0%)
Total	**108**	**108**	**108**

Although the results show that the methodology obtained a high coverage for the security requirements in the PSB document, the following limitations were reported in previous literature (Riaz, 2016b):

1. Current templates are insufficient to generate security requirements for particular topics.

   Requirements in the PSB related to training of developers, use of specific software techniques and the need to install a particular software system could not be mapped to current SD templates (Riaz, 2016b). This is understandable because Security Discoverer does not contain templates which cover these aspects. Creation of new templates related to training, management and installation may support the coverage of these kind of requirements.

2. The generated security requirements by SD may not be equivalent to the ones stated in a security requirements document for a particular software system (in case a security requirements document has been elaborated and handed to the SD practitioner), even though mapping between both occur.

   The requirements in the PSB mention particular system components, protocols and services, whereas the templates are defined in terms of generic components (Riaz, 2016b). Wording in templates cannot match the one used in the PSB because they are written for a more general context. Their vocabulary does not focus on the needs of a particular software system.

3. Insufficient detail in the functional requirement (or input sentence) may lead to an incorrect instantiation of the elements within templates.

   Detail in the input sentences may not be enough and the help of an expert (or a professional familiar with the system in question) may be required in order to properly instantiate the elements within the templates (Riaz, 2016b).

4. Different problem domains may imply the need of different security objectives and (more or new) templates, in order solve this demand.

   It has been reported that availability was the most prevalent security objective in Cisco's network software mentioned before, whereas sentences implying the need for confidentiality were the least prevalent in the PSB (Riaz, 2016b). The exact opposite was observed across all documents in the healthcare domain: they are focused mainly on confidentiality and do not go that further into availability considerations (Riaz, 2016b).

   Following this idea, the SD methodology would not be able to generate quality security requirements for a networks system (which primarily demands the need of availability), if the current template repository does not count with a robust core of availability templates. Moreover, SD only offers one template for privacy. It may or may not be enough for unexplored problem domains.

5. Security requirements templates and natural language artifacts (functional requirements documents or input sentences from which security requirements will be derived) must match in language.

   Cisco analysts argue that security templates and natural language artifacts must have a similar level of vocabulary for a better security requirements generation. Preferably, templates ought to be customized for the application domain (Riaz, 2016b) "or – more likely - the subject documents ought to be written to include more security language constructs (or implied constructs) or both" (Riaz, 2016b, p. 260)". If this match does not happen, the templates in SD may not be suitable to satisfy the security demands of complex software systems, or vice versa (a simple natural language artifact may not be compatible with highly-elaborated security requirements templates).

Chapter 7. Conclusions and future directions

Two differentiated replications of a controlled experiment (UCR18a and UCR18b) were conducted in the domain of health care to evaluate if the use of suggested security requirements templates had a significant difference on the quality, coverage, relevance and efficiency of the security requirements derived from them, as opposed to a manual approach without the help of these templates. UCR15 data was used for comparison purposes in this investigation. As in UCR15, an oracle of security requirement templates was created for each of UCR18a and UCR18b replications beforehand. Reference material was given to all participants some days before the start of the experiment, as well as a 15-minute explanatory presentation in Spanish, so they could understand Riaz et al. security concepts and clarify any doubts.

Participants involved in these three replications were graduate students enrolled in the same Master's degree, with little experience in security on average.

We introduced two major changes in UCR18a and UCR18b replications: the use of a manual approach (no online tool) and having NutriMetas' SRS follow the ISO/IEC/IEEE 29148:2011 standard. Both of them proved to be helpful in our context, particularly the manual approach as we had to work with small groups. We recommend using the online tool with larger groups, since it facilitates data collection and provides accurate time recording. There has been a discussion before on whether the use of additional resources (apart from SRS or use cases) could help students in the identification of applicable security templates (Riaz et al., 2017) and understanding of the problem domain. Based on our experience, questions asked by participants in class or via e-mail were not related to the problem domain. Having a solid understanding of the problem domain and its functional requirements is crucial to elicit more and better security requirements. The ISO/IEC/IEEE 29148:2011 standard helped the participants in this manner.

The coverage scores of treatment group participants in both UCR18a and UCR18b replications were significantly better than ones obtained by students in the control group (90% vs. 41% in UCR18a and 91% vs. 40% in UCR18b). In one of the replications (UCR18a), participants of the treatment group performed significantly better in relevance as well (97% vs. 37% in UCR18b). No case was found were the control groups performed significantly better. As for UCR15, treatment groups performed significantly in coverage (63% vs. 36 for the control group) and efficiency (20% vs. 11% for the control group). These findings hold for two different scenarios (mobile health and mobile banking) and indicate that the results may be generalizable across other scenarios that demand security considerations.

Apart from one treatment student in UCR18a who did not question the suggested templates she received, no other participant obtained perfect scores in all four variables under investigation. To some extent, their lack of security expertise and the fact that using the templates has its learning curve may have lowered their scores. Even though

templates capture human experience (expert knowledge) to solve recurring problems and may help practitioners in eliciting security requirements, the human factor will always be present. As a consequence, it is possible that participants tend to rely on the process developed by Riaz et al. (2014a) and miss essential security requirements which are not part of the templates, no matter if they are from treatment or control group. Nonetheless, 40% of the participants who gave feedback on the activity explicitly argued the templates offer good coverage and applicability and overall, their experience working with templates was positive. Most of them are willing to use this method as a starting point for the requirements elicitation process in their workplace, and not as an exhaustive list.

It is important to keep close communication with previous researchers in order to conduct successful replications. Future researchers must communicate the context factors related to their experiments and their tacit knowledge as well. Additionally, there must be an alternative plan in case of an emerging situation. These are the first replications done without the online tool and they went well because the data collection could be made in class or at home. In our case for UCR18a, we printed the reference material, the answer sheets and the questionnaires for all participants in case we did not have access to any ICT lab in UCR. Fortunately, we did not need these hard copies the day of the experiment.

For future replications, it would be of great interest to investigate new problem domains, explore new ways in which participants may feel more motivated and/or engaged before and during the task. In-class experiments give students the feeling as if they were taking an exam and it may affect their performance. Further experimentation in the way in which the use cases or functional requirements of the system in question are presented to the participants must be considered. The level of detail is important in these artifacts is important as well. Furthermore, the effect of context factors, such as the time available to complete the task, must continue under investigation in order to see if they are conductive to producing better results overall.

For practitioners, this process helps participants in the identification of a core set of security requirements, think about software security concerns and raises awareness about this topic. Security expertise can be limited even in IT professionals with many years of work and academic experience. Therefore, an underlying objective of these replications is to evaluate if software applications like SD (which may automatically identify security requirements from natural language artifacts, based on supervised machine learning) can be a viable approach to identify security requirements for a specific software system.

Part of this investigation was published in the article “Identifying implied security requirements from functional requirements” and presented at CISTI’2019 - 14th Iberian Conference on Information Systems and Technologies. This international scientific event promoted by AISTI (Associação Ibérica de Sistemas e Tecnologias de Informação)

was held in Coimbra, Portugal, between the 19th and the 22th of June, 2019. This article can be read in Appendix 2.

Appendixes

Appendix 1 – NutriMetas' functional requirements specification

Especificación de requerimientos

A continuación, se detalla la especificación de requerimientos de seguridad para el proyecto NutriMetas, el cual comprende el desarrollo de una aplicación móvil que funciona en el sistema operativo Android. Esta especificación se ha estructurado de acuerdo con las prácticas recomendadas por el estándar ISO/IEC/IEEE 29148:2011 *Systems and software engineering -- Life cycle processes -- Requirements engineering* de especificación de requerimientos.

1.1 - Propósito de la aplicación

El propósito de la aplicación es coayudar al nutricionista en el proceso de atención nutricional, mediante la modificación de hábitos alimentarios saludables de los pacientes. La aplicación permite el auto monitoreo del paciente sobre un conjunto de metas saludables establecidas previamente por su nutricionista.

La aplicación móvil provee funcionalidades para el registro diario, recordatorios y monitoreo de metas de alimentación saludable, la administración de un expediente personal con los datos nutricionales y un conjunto de reportes de seguimiento sobre los datos registrados.

1.2 - Alcance de la aplicación

La aplicación móvil administra el registro de metas de alimentación saludable y obtiene reportes de la actividad realizada y cambios obtenidos en el tiempo. La aplicación permite las siguientes funcionalidades a los usuarios:

- (R0) Activación de usuario
- (R1) Configuración de expediente: activación de usuario para consultar la información de su expediente registrado previamente por la/el nutricionista
- (R2) Configuración de metas: activación de usuario para consultar la información de las metas planteadas por su nutricionista
- (R3) Consulta de expediente: consultar toda la información de su expediente nutricional.

- (R4) Cambio de información de correo electrónico, teléfono, educación y tipo de usuario.
- (R5) Registro de datos nutricionales periódicos: registrar datos sobre las mediciones de su estado nutricional (se ingresan junto con su nutricionista)
- (R6) Consulta de metas de alimentación saludable del usuario: mostrar las metas propuestas para el usuario
- (R7) Registro de actividad diaria del cumplimiento de las metas de alimentación saludable: El usuario (paciente) desea registrar el cumplimiento de sus metas.
- (R8) Recordatorio de registro de actividad de meta: recordatorio para que el usuario registre su actividad diaria del cumplimiento de las metas
- (R9) Reporte de cumplimiento de metas
- (R10) Reporte de puntajes y medallas obtenidas por el cumplimiento de metas
- (R11) Reporte de datos nutricionales antropométricos (peso, talla, índice de masa corporal (IMC), porcentaje de grasa corporal, circunferencia de cintura y nivel de actividad física).
- (R12) Reporte de datos de bioquímica (colesterol total, LDL, HDL, triglicéridos, glicemia)

- (R13) Envío de los datos registrados al correo del nutricionista.

El prototipo de la aplicación no dará mantenimiento a las tablas de datos y configuración, esto está fuera del alcance de esta aplicación. Las tablas de datos a las cuales se les dará mantenimiento se detallan de forma explícita en este documento. Toda la información de configuración estará almacenada en las bases de datos de la aplicación previamente.

1.3 - Perspectiva del producto

El producto desarrollado es diseñado como herramienta de apoyo para el/la nutricionista y sus pacientes con el fin de dar seguimiento a las metas de alimentación saludable recomendadas. Esta es una aplicación móvil para utilizar en dispositivos móviles con el sistema operativo Android.

1.3.1 – Interfaces de usuario

Las características generales del diseño de la interfaz y los patrones de diseño son los siguientes:

- El acceso a la aplicación por dispositivo es para un usuario particular.
- La aplicación debe diseñarse para diferentes tamaños de pantallas de distintos dispositivos.
- La terminología utilizada en la aplicación debe ser entendible para los usuarios.

- Los patrones de diseño deben ser consistentes para toda la aplicación y sus distintas funcionalidades.
- Los tipos de letras deben ser consistentes para toda la aplicación.
- Los íconos deben ser consistentes para toda la aplicación.

1.4 - Funciones del producto

Las funciones del producto son las siguientes:

Control de acceso de la aplicación

- (R0) El usuario mediante su identificación y una clave puede activar su cuenta. Obtiene los datos de las metas que fueron definidas y registradas previamente por su nutricionista. Esta activación se realiza una única vez por dispositivo y con la clave proporcionada por el nutricionista. Una vez hecha la activación, el sistema reconocerá al usuario cada vez que ingrese e ingresa directamente a la sección de metas. La aplicación prototipo no sincroniza entre distintos dispositivos del mismo usuario.

- (R1) Al activar el usuario el paciente obtiene acceso a la información de su expediente previamente definido por el/la nutricionista. Este proceso carga la información en el dispositivo.

- (R2) Al activar el usuario el paciente obtiene acceso a la información de sus metas de alimentación saludable previamente definido por el/la nutricionista. Este proceso carga la información en el dispositivo.

Mantenimiento de Expediente

- (R3) Consulta de expediente: consultar toda la información de su expediente nutricional.

- (R4) El usuario puede cambiar información de correo electrónico, teléfono, educación, tipo de usuario y nutricionista.

- (R5) El usuario puede registrar junto con su nutricionista datos nutricionales tales como su peso, talla, porcentaje de grasa corporal, circunferencia de cintura, nivel de actividad física y datos de pruebas bioquímicas. Para realizar esta acción, el nutricionista debe proporcionar una clave de autorización.

- Al registrar los datos de peso y talla, el sistema calcula el IMC con base en la fórmula respectiva. Menos de 18.5: bajo peso, 18.5-24.9: normal, 25-29.9: sobrepeso, 30 o más: obesidad.
- Al registrar los datos de peso, talla, porcentaje de grasa y circunferencia de cintura, el sistema interpreta el IMC (bajo peso [Menos de 18.5], peso saludable [18.5-24.9], sobrepeso [25-29.9] u obesidad [30 o más]), el porcentaje de grasa (bajo, saludable, exceso de grasa [Mujeres: Menos de 12%: bajo, 12-31.9%: salu-

dable, 32% o más: exceso de grasa. Hombres: Menos de 4%: bajo, 4-25.9%: saludable, 26% o más: exceso de grasa]), la circunferencia de cintura (riesgo normal, riesgo incrementado [Normal: 88 cm o menos en mujeres. 102 cm o menos en hombres. Incrementado: lo contrario (mayor a 88, mayor a 102)]).

Mantenimiento de Metas

- (R6) El usuario puede consultar las metas de alimentación saludable que son parte de su plan de mejora definido en conjunto con el/la nutricionista.

- (R7) El usuario puede registrar su actividad diaria sobre el cumplimiento de cada una de las metas alimentación saludable.

- (R8) El sistema emite recordatorios para que el usuario registre los datos diarios para cada una de las metas. Este requerimiento se implementa mediante un proceso que corre como un servicio automático de notificaciones.

Reportes del sistema

- (R9) El usuario puede consultar un reporte con los datos del cumplimiento de metas.

- (R10) El usuario puede consultar un reporte de medallas obtenidas por cumplimiento de metas.

- (R11) El usuario puede consultar un reporte con sus datos antropométricos (peso, talla, IMC, porcentaje de grasa corporal, circunferencia de cintura y medidas bioquímicas).

- (R12) El usuario puede consultar un reporte con sus datos bioquímicos (colesterol total, LDL, HDL, triglicéridos y glicemia).

Envío periódico de datos al nutricionista

- (R13) Envío de los datos registrados al correo del nutricionista: cada semana, la aplicación genera una hoja de cálculo, almacena en ella los datos (registro de cumplimiento de metas, registro de uso de la aplicación, datos nutricionales y datos bioquímicos) registrados del paciente y envía el archivo al correo del nutricionista, para su revisión.

1.5 – Características de los usuarios

Los usuarios de la aplicación son usuarios regulares de teléfonos móviles con sistema operativo Android y cierta experiencia en el uso de aplicaciones móviles. Los tipos de usuarios de la aplicación son el nutricionista (que supervisa algunas funcionalidades de la aplicación) y el paciente, el cual realiza la consulta y registro diario de las metas de alimentación saludable definidas por el/la nutricionista.

1.6 – Restricciones

El sistema debe funcionar de forma eficiente para teléfonos inteligentes Android de distintas generaciones. Para el prototipo no se contempla el uso de la aplicación para otras plataformas. El sistema es una aplicación nativa para Android. El sistema debe funcionar sin conexión continua a internet.

1.7 – Suposiciones y dependencias

- El sistema funciona sobre dispositivos móviles con el sistema operativo Android.

- El sistema puede funcionar de forma independiente, sin necesidad de conexión continua a Internet o conexión con otras aplicaciones en línea.

1.8 – Requerimientos futuros

- El sistema funciona sobre dispositivos móviles con otros sistemas operativos. Por ejemplo, IOS y Windows Mobile.

- El sistema se comunica por medio de internet con una aplicación *backend* que administra el nutricionista y con la cual intercambian información en línea. Además de permitir obtener información procesada de los pacientes para el análisis de datos y apoyo a la toma de decisiones.

- El sistema permite la sincronización entre dispositivos para un usuario.

1.9 – Requerimientos específicos

Los requerimientos del producto (o funciones que el producto debe permitir realizar al usuario) son los siguientes:

Control de acceso de la aplicación

- (R0) Activación de usuario

- **Introducción**: Activación de la cuenta de usuario para usar la aplicación.
- **Prerrequisitos**: Aplicación instalada y datos del usuario definidos por el nutricionista para uso de la aplicación.
- **Entrada**: Identificación de usuario y clave de activación.
- **Proceso**: El sistema realiza la validación de datos y configura la aplicación para uso del usuario activado. El usuario se marca como activo.
- **Salida**: Se muestra un mensaje que indica que el usuario ha sido activado.

- (R1) Configuración de expediente

- **Introducción**: Activación del expediente, de acuerdo con la cuenta de usuario.
- **Prerrequisitos**: Activación de usuario.
- **Entrada**: Identificación de usuario registrado.
- **Proceso**: El sistema configura el expediente del usuario activado. Se asocian los datos del expediente al usuario.
- **Salida**: Se muestra un mensaje con sus datos.

- (R2) Configuración de metas

- **Introducción**: Activación de metas, de acuerdo con la cuenta de usuario.
- **Prerrequisitos**: Activación de usuario.
- **Entrada**: Identificación de usuario registrado.
- **Proceso**: El sistema configura las metas del usuario activado. Se asocian los datos de las metas al usuario.
- **Salida**: Se muestra un mensaje con sus metas definidas.

Mantenimiento de Expediente

- (R3) Consulta de expediente

- **Introducción**: Consultar datos de expediente.
- **Prerrequisitos**: Usuario activado.
- **Entrada**: Identificador del usuario.
- **Proceso**: El sistema recupera los datos del expediente del usuario.

- **Salida**: Se muestran los datos del expediente del usuario, nombre completo (nombre, primer apellido, segundo apellido), correo electrónico, teléfono, celular, identificación, teléfono adicional, fecha de nacimiento, edad, sexo, nivel educativo, tipo de usuario, código de nutricionista.

- (R4) Cambio de información de correo electrónico, teléfono celular, teléfono adicional, nivel educativo, tipo de usuario, nutricionista.

- **Introducción**: Actualiza datos de correo y teléfono.
- **Prerrequisitos**: Ingreso a consulta de expediente.
- **Entrada**: Identificador del usuario, correo electrónico, teléfono celular, teléfono adicional, nivel educativo, tipo de usuario, nutricionista.
- **Proceso**: El sistema realiza la actualización de datos del usuario.
- **Salida**: Se muestra un mensaje con los datos actualizados.

- (R5) Registro de datos nutricionales periódicos.

- **Introducción**: Registro de datos nutricionales.
- **Prerrequisitos**: Ingreso a consulta de expediente.
- **Entrada**: Fecha de registro, identificación de usuario, peso, talla, porcentaje de grasa corporal, circunferencia de cintura y nivel de actividad física.
- **Proceso**: El sistema realiza un registro de datos en el expediente del usuario.
- **Salida**: Se muestra un mensaje con datos registrados y calculados automáticamente.

Mantenimiento de Metas

- (R6) Consulta de metas de usuario.

- **Introducción**: Consultar datos de metas.
- **Prerrequisitos**: Usuario activado.
- **Entrada**: Identificador del usuario.
- **Proceso**: El sistema recupera los datos de las metas del usuario.
- **Salida**: Se muestran los datos y descripción de las metas planteadas.

- (R7) Registro de actividad diaria del cumplimiento de meta.

- **Introducción**: Registrar actividad de cumplimiento de metas.
- **Prerrequisitos**: Consulta de meta.
- **Entrada**: Identificador del usuario, identificador de meta, fecha de registro, cantidad registrada o cumplimiento y comentarios o anotaciones.
- **Proceso**: El sistema registra los datos de la actividad para la meta del usuario.
- **Salida**: Se muestran los datos registrados para la meta planteada.

- (R8) Recordatorio de registro de actividad de meta.

- **Introducción**: Emitir recordatorios de registro de datos de metas.
- **Prerrequisitos**: Usuario activado.
- **Entrada**: Identificador del usuario, tiempos establecidos para alertas.
- **Proceso**: El sistema recupera los datos de las metas del usuario y alerta al usuario para el registro de datos de cumplimiento de las metas.
- **Salida**: Se muestran un recordatorio de registro de datos de actividad de metas. El sistema permite ejecutar el requerimiento 6 (R6) desde este mensaje de recordatorio.

Reportes del sistema

Estos requerimientos están en revisión porque forman parte de la tercera iteración. Estos reportes implican el procedimiento de la generación de puntos y medallas. El cálculo de puntos y asignación del tipo de medalla en función de los puntos se define en una próxima iteración. A continuación, se detalla la descripción general de los requerimientos relacionados con los reportes del sistema:

- (R9) Consulta de cumplimiento de metas.

- **Introducción**: Consultar datos históricos de cumplimiento de metas.
- **Prerrequisitos**: Usuario activado.
- **Entrada**: Identificador del usuario e identificadores de metas.
- **Proceso**: El sistema recupera los datos históricos de los datos de actividad registrada para las metas de usuario.
- **Salida**: Se muestra un gráfico de tendencias para cada una de las metas establecidas.

- (R10) Consulta de medallas obtenidas

- **Introducción**: Consultar datos históricos de medallas obtenidas
- **Prerrequisitos**: Usuario activado.
- **Entrada**: Identificador del usuario.
- **Proceso**: El sistema recupera los datos históricos de medallas obtenidas para el usuario.
- **Salida**: Se muestra un pictográfico con el histórico de medallas obtenidas.

- (R11) Consulta de datos antropométricos

- **Introducción**: Consultar historia de datos antropométricos
- **Prerrequisitos**: Usuario activado.
- **Entrada**: Identificador del usuario.

- **Proceso**: El sistema recupera la historia de datos antropométricos para el usuario.
- **Salida**: Se muestra un gráfico de tendencias para cada una de las variables: peso, talla, IMC, porcentaje de grasa corporal y circunferencia de cintura.

- (R12) Consulta de datos bioquímicos

- **Introducción**: Consultar historia de datos bioquímicos
- **Prerrequisitos**: Usuario activado.
- **Entrada**: Identificador del usuario.
- **Proceso**: El sistema recupera la historia de datos bioquímicos para el usuario.
- **Salida**: Se muestra un gráfico de tendencias para cada una de las variables: colesterol total, LDL, HDL, triglicéridos y glicemia.

Envío periódico de datos al nutricionista

- (R13) Envío de los datos registrados al correo del nutricionista

- **Introducción**: Envío de los datos registrados al correo del nutricionista
- **Prerrequisitos**: Usuario activado.
- **Entrada**: Identificador del usuario.
- **Proceso**: El sistema prepara y envía semanalmente una hoja de cálculo de los datos del paciente, al correo del nutricionista.
- **Salida**: El nutricionista recibe una notificación al respecto.

1.10 – Interfaces externas

La aplicación prototipo no cuenta con interfaces con otros sistemas.

Appendix 2 – "Identifying implied security requirements from functional requirements" article

This appendix contains the preprint of the article "Identifying implied security requirements from functional requirements". It was presented at the 2019 14th Iberian Conference on Information Systems and Technologies, held in Coimbra, Portugal (19-22th June):

A. Martínez, M. Jenkins and C. Quesada-López, "Identifying implied security requirements from functional requirements," *2019 14th Iberian Conference on Information Systems and Technologies (CISTI)*, Coimbra, Portugal, 2019, pp. 1-7. doi: 10.23919/CISTI.2019.8760631

Identifying implied security requirements from functional requirements

A replication on the use of security requirements templates

Andrés Martínez
Posgrado en Computación e Informática
Universidad de Costa Rica
andres.martinezmesen@ucr.ac.cr

Marcelo Jenkins, Christian Quesada-López
Escuela de Ciencias de la Computación e Informática
Universidad de Costa Rica
{marcelo.jenkins, cristian.quesadalopez}@ucr.ac.cr

***Abstract* — The elicitation of software security requirements in early stages of software development life cycle is an essential task. Using security requirements templates could help practitioners to identify implied software security requirements from functional requirements in the context of a software system. In this paper, we replicated a previous study that analyzed the effectiveness of security requirements templates to support the identification of security requirements. Our objective was to evaluate this approach and compare the applicability of the previous findings. We conducted the first replication of the controlled experiment in 2015, and subsequently conducted two differentiated replications in 2018. We evaluated the responses of 33 participants in terms of quality, coverage, relevance and efficiency and discussed insights regarding the impact of context factors. Participants were divided into treatment (security requirements templates) and control groups (no templates). Our findings support some previous results: treatment group performed significantly better than the control group in terms of the coverage of the identified security requirements. Besides, the requirements elicitation process performed significantly better in relevance and efficiency metrics in two of the three replications. Security requirements templates supported participants to identify a core set of the security requirements and participants were favorable towards the use of templates in identifying security requirements.**

Keywords - security requirements; requirements engineering; software engineering; replication; empirical study.

I. INTRODUCTION

Software security is an essential non-functional requirement for software systems [1]. Secure software development lifecycle (SSDL) is an overall security software methodology and practical approach during software development that ensures different security objectives are considered in the development of a system. Identifying security problems before coding is essential for the success of a software development project. Defining security and privacy requirements is a crucial part of the application development process. This planning could allow development teams to identify key security objectives and take actions accordingly from early stages of development [1].

Security requirements engineering (SRE) can provide a foundation for developing secure systems [2]. The elicitation of software security requirements in early stages of software development life cycle is a vital task [3]. Although there are methods and processes for SRE [4], oftentimes the development teams do not focus on security during the early stages of software development [5]. Furthermore, security expertise in development teams may be limited and eliciting security requirements could be unspecified or inadequately specified. [6]. Security requirements can be captured in reusable templates and instantiated in the context of a software system [3]. Security requirements templates could support the elicitation process by suggesting security requirement implied from existing functional requirements [3]. Moreover, by providing guidance on applicable security requirements, security non-experts can consider security concerns in SRE tasks [2].

Riaz et al. [2, 7] developed a semi-automated process for identifying candidate security requirements implied by existing functional requirements. The process suggests templates by automatically parsing requirements and identifying security requirements. These security requirements patterns are reusable across multiple software systems and domains. Using security requirements templates (based on similar security objectives such as confidentiality, integrity, identification & authentication, availability, accountability and privacy) could help practitioners identify implied software security requirements from functional requirements in software systems [8]. These implied security requirements could be identified and specified to create a set of security requirements for the software system [2].

In this paper, we replicated a previous controlled experiment [3] that analyzed the effectiveness of security requirements templates to support the identification of security requirements from functional requirements in the context of a mobile health application [9, 10]. The suggested templates proposed in [2] were manually instantiated to generate specific security requirements. Our objective was to evaluate the security requirements templates approach and compare the applicability of previous findings presented in [3]. We conducted the first replication in 2015 [8], and subsequently conducted two differentiated replications in 2018. In total, we evaluated the responses of the 33 participants in terms of quality, coverage, relevance and efficiency and discussed insights regarding the impact of context factors. We analyzed the following research questions based on the original experiment and its subsequent replications [3, 8]:

RQ1: What is the coverage of security requirements elicited using suggested security requirements templates?

RQ2: How relevant are the security requirements elicited using suggested security requirements templates?

RQ3: What is the quality of security requirements elicited using suggested security requirements templates?

RQ4: How efficient is the process of eliciting security requirements using suggested security requirements templates?

The rest of the paper is structured as follows: Section 2 describes background and related work. Section 3 details the replication process and Section 4 presents the analysis of results. Finally, Section 5 outlines our conclusions.

II. BACKGROUND AND RELATED WORK

Software security requirements can be defined as "constraints on the functions of the system, where these constraints operationalize one or more security goals" [11] and they must describe the desired security behavior of a system [7]. SRE is the process of eliciting, specifying and analyzing the security requirements for a software system. It is concerned with the prevention of harm in the real world and considering security requirements as functional requirements [12]. Several approaches exist to support SRE by providing guidelines for the identification of security objectives, resources, assets, and associated security threats for a software system [3]. Some of them involve the use of conceptual frameworks [13], and requirement models [4]. However, these approaches still demand security experience, understanding about security objectives pursued by the system and the definition of security requirements for these objectives [3]. These methodologies rarely offer detailed advice on how to contextualize security requirements for a system [6].

Security objectives are the desired security goals or properties of a system [14]. According to [2], there are six high-level security-specific goals defining the outcomes a system must ensure or prevent: (C) Confidentiality, the degree to which the data is disclosed only as intended. (I) Integrity, the degree to which a system or component guards against improper modification or destruction of computer programs or data. (A) Availability, the degree to which a system or component is operational and accessible when required for use. (ID) Identification & Authentication, the need to establish that a claimed identity is valid for a user, process or device. (AY) Accountability, the degree to which actions affecting software assets can be traced to the actor responsible for the action. (PR) Privacy, the degree to which an actor can understand and control how their information is used. Security objectives of software systems are not only technical aspects (from the perspective of systems development), but also operational and administrative aspects [7].

Based on a list of security objectives, a list of applicable security requirements templates can be suggested, selected and instantiated by a requirements engineer in the context of a software system. For each functional requirement, you can determine if it can be associated with one or more security objectives [3].

III. REPLICATION PROCESS

In the following sections, we provide details about the methodology for conducting the three replications. The original study was conducted at North Carolina State University (NCSU13) [3]. After this experiment, a family of replications was conducted at University of Trento (UT14), North Carolina State University (NCSU14) and University of Costa Rica (UCR15) [8]. This paper reports the results of three controlled experiments conducted at University of Costa Rica (UCR15 and UCR18a, UCR18b), involving 33 graduate students attending master degree courses, to evaluate the last two steps of the Security Discovery (SD) process [3] using security requirements templates.

This study is intended to compare and extend previous results, highlighting the similarities and differences in the use of security requirements templates in order to evaluate if the initial findings could be generalized in a different setting. In our replication, we reused the goal, research questions, hypothesis, and analysis procedure reported in [8] as one of the problem domains was healthcare as well. We reported this paper based on the set of guidelines for reporting replications proposed in [15]. We detail the design of the study based on [8] and report the results of individual studies and the analysis across three studies conducted at University of Costa Rica.

A. Goals, hypotheses and metrics

The goal of the replications is to analyze the effectiveness on the use of security requirements templates to systematically support the process of identification and specification of implied security requirements from functional requirements in the context of a mobile health application. Based on the research questions, we test the following null hypotheses:

H_{01}: The coverage of elicited security requirements is unrelated to the use of suggested security requirements templates.

H_{02}: The relevance of elicited security requirements is unrelated to the use of suggested security requirements templates.

H_{03}: The quality of elicited security requirements is unrelated to the use of suggested security requirements templates.

H_{04}: The efficiency of elicited security requirements is unrelated to the use of suggested security requirements templates.

Results regarding coverage, relevance, quality (of security requirement) and efficiency (of the process of eliciting security requirements) were used to test the hypotheses. The following metrics were calculated:

- Coverage (Quantitative): Recall with respect to security requirements templates in the oracle of security requirements developed a priori to the conduct of the study.
- Relevance (Quantitative): Precision with respect to security requirements templates in the oracle of security requirements developed a priori to the conduct of the study.
- Quality (Qualitative): Quality scores were evaluated by two researchers based on elements of the templates been identified, level of detail in requirements and templates specification, any logical inconsistencies, different types of security objectives considered, and. We used a Likert-like scale (1: poor; 2: below average; 3: average; 4: above average; 5: good) [8].
- Efficiency (Quantitative): # of security requirements templates in the oracle identified per minute.

B. Participants

In total, we evaluated the responses of the 33 participants. All participants gave informed consent to use their responses for the study. UCR15, was conducted in May 2015. The participants were 16 first and second year Master's degree students enrolled in a course on Software Metrics. The exercise was graded with a significant percentage (25%) as an individual in-class activity. UCR18a, was conducted in February 2018 with second year Master's degree students enrolled in a course on Empirical Software Engineering. The task was mandatory and an individual in-class activity for all the students. The exercise was graded with a significant percentage (20%). Finally, UCR18b was conducted in April 2018 a take-home activity with first and second year Master's degree students enrolled in a Design of Experiments course. They had one week to complete it. This exercise was mandatory for all students to complete as part of their course homework graded with a percentage of 10%.

For all replications, participants were told their grade would depend on the completeness of the data reported in the process, the knowledge of the subject according to the base material provided and the rigorousness with which the experimental procedure was applied. At the end of each experiment, we asked participants to report both their academic and work experience in three categories: Computer Science (CS), Software Engineering (SE) and Security related education (SC). No personal information of the participants was recorded. Table 1 presents the demographic summary for the three replications. Demographics for previous studies are detailed in [8]. Overall, participants between replications have comparable experience across various experience categories. Most of the participants have little experience in security. Seven participants reported more than 3 years of experience in security (21%).

TABLE I. PARTICIPANTS EXPERIENCE FREQUENCY

Study	Group	No. of subjects	Experience (years)	Academic			Work		
				CS	SE	SC	CS	SE	SC
UCR15	Treatment	9	> 5 years	5	4	2	2	3	0
			3-5 years	3	2	0	2	1	0
			1-2 years	0	2	1	3	2	2
			< 1 year	1	1	6	2	3	7
	Control	7	> 5 years	2	1	0	1	1	0
			3-5 years	5	4	0	5	4	0
			1-2 years	0	2	2	0	1	1
			< 1 year	0	0	5	1	1	1
UCR18a	Treatment	4	> 5 years	1	1	0	2	1	0
			3-5 years	3	1	0	1	1	0
			1-2 years	0	0	0	1	0	0
			< 1 year	0	2	4	0	2	4
			NR	0	0	0	0	0	0
	Control	4	> 5 years	4	2	0	1	1	0
			3-5 years	0	1	1	1	1	1
			1-2 years	0	1	1	2	1	0
			< 1 year	0	0	2	0	1	3
UCR18b	Treatment	5	> 5 years	2	1	0	1	1	0
			3-5 years	2	3	2	2	2	3
			1-2 years	0	0	0	1	1	0
			< 1 year	0	0	2	0	0	1
			NR	1	1	1	1	1	1
	Control	4	> 5 years	1	0	1	0	0	0
			3-5 years	0	0	0	1	0	1
			1-2 years	1	1	0	1	2	1
			< 1 year	1	2	2	1	1	1
			NR	1	1	1	1	1	1

C. Experiment environment

This section details the differences and similarities related to the replication environment across all the experiments. The original experiment [3] and subsequent replications [8] including UCR15 were conducted as an online activity. An online site was provided to participants to complete the task. The system randomly assigned each participant to control or treatment group. The system automatically recorded total time spent completing the task. UCR18a and UCR18b replications were conducted using printed and digital materials. Both, digital and printed documents were provided to participants to complete the task. Participants were told to fill in a digital form that was sent by e-mail. Answers were received by e-mail when they were finished. Researchers randomly assigned each participant to control or treatment group. Total time spent completing the task was manually recorded by participants. After completing the task, participants were asked to (a) briefly explain the process used for identifying implied security requirements and (b) provide feedback regarding the use of security requirements templates.

All participants in UCR15, UCR18a and UCR18b replications were native Spanish speakers. They were allowed to provide their responses in English or Spanish. Templates suggested to the participants included some extraneous suggestions as in a previous study [3]. It was their decision to make use of these suggestions or not. In addition, they were not aware that some of them could be extraneous. Table 2 summarizes the differences in context factors across three replications. Context factors for previous studies were detailed in [8].

TABLE II. CONTEXT FACTORS ACROSS REPLICATIONS

	UCR15	**UCR18a**	**UCR18b**
Participants	16 graduate students	8 graduate students	9 graduate students
Setting	In-class Activity [180 min]	In-class Activity [150 min]	Take-home Activity [1 week]
Security training provided	4 page reference material, 10 min video on security objectives and requirements and a 15 min explanatory presentation in Spanish	4 page reference material, 10 min video on security objectives and requirements and a 15 min explanatory presentation in Spanish	4 page reference material, 10 min video on security objectives and requirements and a 15 min explanatory presentation in Spanish
Domain	Mobile banking	Mobile healthcare	Mobile healthcare
Type	On-line activity	Off-line activity	Off-line activity
Source	Use cases scenarios	SRS	SRS
Other changes	Suggestions to fill in templates. Feedback on quality of responses. Extraneous Templates	Feedback on quality of responses. Extraneous Templates	Feedback on quality of responses. Extraneous Templates

D. Experiment artifacts

The experiment material consists of the reference material given to the participants, as well as the use case scenarios and the software requirement specification (SRS). All participants received four pages of reference material containing a description of security objectives and textual clues that can indicate an implied security objective. Forty examples of security requirements grouped by security objectives and an explanation in video was provided as well. A 15-minute explanatory presentation in Spanish was given to explain the task. The material was provided at least two days prior to the start of the task, so they could study it and ask researchers for advice, if necessary. Participants were told to study the material in this particular order. During the experiment, both groups had access to the reference material and the treatment group received the suggested reusable security requirements templates grouped by security objectives. For participants in UCR15, additional details for filling in templates such as the subject, action and resource elements. Researchers instructed all participants to find as many security requirements as they could. This meant treatment students were allowed to use any template, if they considered it appropriate.

UCR15 participants received the use case scenarios, while UCR18a and UCR18b received the SRS describing the software system under analysis. For the use case scenarios, each of the use cases focus on a single unit of functionality, so that participants can easily understand the scope of the requirements. Understanding the use case requires no knowledge of the domain. For UCR15, we selected two use cases from the Cyclos mobile payment software [16]. Cyclos is a mobile banking platform including SMS banking and a mobile application. The first use case, make payment, is related to the functionality of making online payment to another customer. The second use case, retrieve account information, is related to the functionality for querying account information. Both use cases, via SMS.

For the SRS, functionality was described based on the ISO/IEC/IEEE 29148:2011 standard. For UCR18a and UCR18b, to assess generalizability of findings across domains and types of requirement representation, we selected the entire set of requirements from NutriMetas [9, 10]. This mobile health application was created at UCR and its purpose is to co-assist the nutritionist in the process of nutritional care by modifying patients' eating habits. This Android-only application allows patient self-monitoring on a set of healthy goals previously established by the nutritionist. Patients are not required to provide input on what they consume daily. The approach is to have a specific behavioral tracker with a simple and easy to use design. The SRS contains 14 functional requirements that imply the need of security requirements [9, 10]. These requirements range from user (patient) activation to goal registration (e.g. meals per day and specific beverage daily count), reminders and reports.

Additionally, a set of oracles of security requirements were prepared and validated in advance in order to evaluate the coverage and relevance of the security requirements identified by the participants. For UCR15, oracles reported in [8] were used. For the first use case, 110 security requirements were identified and 35 security requirements were identified for the second use case. The oracles for UCR18a and UCR18b were created based on the 14 functional requirements in NutriMetas' SRS. For each requirement, researchers identified implied security objectives associated with the statement. After that, the security requirement templates were selected and manually instantiated in the context of the mobile application. Finally, three security experts validated the oracles. A total of 93 security requirements templates were identified. We provide a summary of templates associated with each requirement in Table 3. The summary of templates associated with each of the use cases is detailed in [8].

E. Experimental process and evaluation

For each replication, participants were randomly divided into treatment (security requirements templates) and control groups (no templates). Participants were given the same task of identifying security requirements. The treatment group worked with suggested security requirements templates whereas the control group did not. Participants did not know about the existence of different groups or use cases.

TABLE III. SECURITY TEMPLATES BY REQUIREMENT

Functional requirement	**Suggested security templates**
R0 – User activation	C1 - authorized access, C2 - confidentiality during storage, C3 - confidentiality during transmission, I2 - write-type actions, IA2 - unique accounts, AY2 - logging authentication events.
R1 – Patient's medical record configuration	C1, C2, C3, I1 - read-type actions, I2.
R2 – Goal settings	C1, C2, C3, I1, I2
R3 – View patient's medical record	C1, C3, I1, A2, A3 - service availability, A5 - capability and performance, AY1 - logging transactions with sensitive data, PR1 - usage of personal information.
R4 – Update patient's personal details	C1, C2, C3, I1, I2, AY1
R5 – Insertion of periodic nutritional data	C1, C2, C3, I1, I2, AY1
R6 – View personal goals	C1, C3, I1, A2, A3, A5, AY1,
R7 – Insertion of goals	C1, C2, C3, I2, A3
R8 – Record personal daily goals	C3, I1, A3
R9 – View achieved goals	C1, C3, I1, A2, A3, A5, AY1,
R10 – View achieved medals	C1, C3, I1, A2, A3, A5, AY1,
R11 – View anthropometric data	C1, C3, I1, A2, A3, A5, AY1,
R12 - View biochemical data	C1, C3, I1, A2, A3, A5, AY1,
R13 – Send personal registered nutritional data to the nutritionist	C1, C2, C3, I1, I2, I4 - unchangeable resources, A3, AY1, PR1

Table 4 presents the number of participants for replication. Templates suggested by participants (per functional requirement) are compared (mapped) to the ones in the oracle. For instance, if a participant suggested template C1 for functional requirement R0 and the oracle indicated C1 for that particular requirement, it is considered a true positive (TP). If the oracle contains a template for a specific functional requirement and the participant does not mention it, it is marked as a false negative

(FN). On the contrary, if the student suggests a template for a requirement and it is not present in the oracle, a false positive (FP) is reported. Templates which are not mentioned by participants or the oracle count as true negatives (TN). Coverage and relevance provide assessment of the performance in terms of how many security requirement templates in the oracle a participant identified, as well as the effort spent in identifying relevant security requirement templates (TP) versus irrelevant ones (FP). Lastly, post-study survey responses by treatment and control groups are analyzed to evaluate suggestions for improvement and general reflections on the process.

IV. ANALYSIS OF RESULTS

We present and discuss the results from each replication. Results are presented in terms of quality, coverage, relevance and efficiency and findings are discussed according to context factors. Table 5 provides mean scores for all metrics across studies for a high-level overview of the results. In Table 6, we provide a summary of differences in mean scores between treatment and control groups for each of the metrics (treatment mean – control mean). We also indicate whether the difference is significant.

TABLE IV. NUMBER OF PARTICIPANTS IN EACH GROUP

Group	UCR15	UCR18a	UCR18b	Total
Treatment	9	4	5	18
Control	7	4	4	15
Total	16	8	9	33

TABLE V. OVERALL MEAN SCORES FOR ALL METRICS ACROSS STUDIES

Study	Time	Time on	Quality	Coverage	Relevance	Efficiency
NCSU13 [3]	60 min	~20 min	2.88	0.31	0.88	1.14
UT14 [8]	One week	~47 min	2.70	0.36	0.77	1.07
NCSU14 [8]	60 min	~25 min	2.66	0.37	0.73	1.01
UCR15 [8]	180 min	~102 min	3.69	0.51	0.66	0.64
UCR18a	150 min	~116 min	4.57	0.66	0.67	0.63
UCR18b	One week	~193 min	3.91	0.66	0.72	0.78

A. Results based on individual experiments

1) First replication

Based on the results of UCR15, we found the requirement eliciting process (treatment vs. control) to be a significant factor in determining the metrics for coverage (Mann-Whitney U = 9.0 and p-value = 0.017) and efficiency (Mann-Whitney U = 7.5 and p-value = 0.020). Therefore, we can reject the null hypotheses H_{01} and H_{04} that state that the performance of participants based on metrics for coverage and efficiency respectively, is unrelated to the use of security requirements templates. On average, treatment group students identified 63% of the templates in the oracle (compared to 36% in the control group) and their efficiency almost doubled the one obtained by the control group (20% vs. 11%). No significant differences were found for the metrics of quality and relevance. Thus, we fail to reject the null hypotheses H_{02} and H_{03}. On average, the quality of responses is slightly higher for the treatment group (3.94 vs. 3.36 for the control group). Moreover, the responses given by participants in the treatment group were on average 12% more relevant than the ones reported by the students in the control group.

2) Second replication

Based on the results of UCR18a, we found the requirement eliciting process (treatment vs. control) to be a significant factor in determining the metrics for coverage (p-value = 0.019) and relevance (p-value = 0.020). On average, treatment group students identified 90% of the templates in the oracle and their templates were ~60% more relevant than the ones reported by the participants in the control group. No significant differences were found for the metrics of quality and efficiency. Thus, we fail to reject the null hypotheses H_{03} and H_{04}. On average, the quality of responses is slightly higher for the control group (4.4 vs. 4.7 for the treatment group). This may have happened because the control group did not receive template suggestions and that could have motivated it to consider using of more templates (as they were not being biased by them) and provide more detailed answers. Some treatment students, on the contrary, did not excel in these aspects probably because they followed the template suggestions without questioning them too much. 50% of this control group obtained a 3/5 for considering the use of different templates, which suggests they limited themselves to fill in only the suggested templates. Their score for detail (3/5) was average as well: they did not provide detailed security requirements. As for the efficiency, treatment students lasted on average 1.25 minutes less than the control group finishing the task (115.5 vs. 116.75, respectively). It can be inferred that, although treatment students have received help in the form of template suggestions, they took their time doing the activity. Reported templates in the treatment group range from 59 to 73, with 73 being the total template count for this oracle. This means one student from the treatment group did not question the template suggestions at all. However, in the control group, reported templates count oscillates more: between 48 and 104. This also explains why the treatment group scored higher in coverage and relevance.

3) Third replication

Based on the results of UCR18b, we found the requirement eliciting process (treatment vs. control) to be a significant factor in determining the metrics for coverage (p-value = 0.014) and efficiency (p-value = 0.086). Treatment group participants identified almost 91% of the templates in the oracle, on average. Students in the control group, on the other hand, identified 40% of the templates in the oracle overall. No significant differences were found for the metrics of quality and relevance. Thus, we fail to reject the null hypotheses H_{02} and H_{03}. The difference in efficiency of security requirements templates between treatment and control group (~65% vs. 36%) is not significant at $p < 0.05$, but is significant at $p < 0.1$. This may be because one participant from the treatment group lasted 273 minutes completing the task and reported 106 templates (efficiency of 0.39 templates/min), while another participant in this group lasted 17 minutes more. This issue affects the overall efficiency of the treatment group. As a reference, the treatment student from UT14 who took the most time completing the task at home lasted 132 minutes. These two UCR18b students doubled that time, mainly because they did not have time limitations at home. As for quality and

relevance, treatment participants obtained higher scores than control students as well. Even though participants in the treatment group were recommended extraneous templates, they overall identified more relevant templates as compared to the control group. However, the fact that the treatment group has obtained a mean efficiency of 76% confirms that students are relying heavily on the suggested templates (extraneous or not).

TABLE VI. DIFFERENCE BETWEEN TREATMENT AND CONTROL GROUP MEANS ACROSS ALL REPLICATIONS

Study	Δ Coverage	Δ Relevance	Δ Quality	Δ Efficiency
NCSU13	0.27 **	0.05	-0.03	0.58 **
UT14	0.11 *	0.29 **	0.44	0.63
NCSU14	0.20 **	0.37 **	1.38 **	0.34 **
UCR15	0.27 **	0.12 *	0.58	0.36 **
UCR18a	0.50 **	0.60 **	-0.35	-0.06
UCR18b	0.50 **	0.08	1.02	0.29 **

* Significant at p-value < 0.1, **Significant at p-value < 0.05

B. Results based on analysis across studies

The combined data obtained from UCR15, UCR18a and UCR18b was used to perform an aggregate analysis of the metrics. A Mann-Whitney U test was applied for all participants in treatment group (18) and control group (15). Based on the results obtained from the Mann-Whitney U test, the participants in the treatment group performed better than their classmates in the control group for all metrics, with significant performance regarding coverage, relevance and efficiency. Therefore, it is possible to reject the null hypotheses H_{01}, H_{02} and H_{04}. Participants in the treatment group identified more requirements (~77% vs. ~39% for the control group). Their coverage was ~38% better than the control group and their responses were ~22% more relevant (~78% for the treatment group vs. ~56% for the control group). Participants in the treatment group were also more efficient in the identification of security templates (~71% vs. ~48%). No significant difference was found for the metric of quality (p-value = 0.114). The average quality score for all treatment students in the three studies was ~4.2, compared to a 3.73 of the participants in the control group

C. Breakdown of identified security templates

For UCR18a and UCR18b, we counted the number of templates in NutriMetas' oracle per security objective and compared it with the average count obtained by their treatment and control groups. In UCR18a, the treatment group performed better than the control group in all security objectives. Their responses were closer to the oracle. Confidentiality and availability templates were the most used between both groups. The control group reported a high number of integrity and accountability templates, as compared to the oracle. In UCR18b, the treatment group performed better than the control group in 5/6 security objectives: confidentiality, integrity, ID & authentication, accountability and privacy. Similar to UCR18a, confidentiality and availability templates were the most used between both groups. One possible reason that could explain why the treatment group was not as close as the control group to the oracle, in terms of availability and ID & authentication, is the use of extraneous templates. About 63% of the extraneous templates suggested belong to these two security objectives, which means treatment students are relying on these suggestions.

D. Discussion

We analyzed different aspects of the replications answering the following two questions: Can participants differentiate whether a suggested security requirements template is relevant to the given use case scenario? Are there context factors, such as more time on task, which are conductive to producing better outcomes overall? Participants in treatment group relied heavily on the suggested security templates. Participants seemed to have blindly applied some of the extraneous templates as well. The average count of extraneous templates selected by these participants is 27.4 templates, out of 32. Therefore, we can infer from treatment students in take home activity they took their time analyzing which templates were applicable, instead of filling in all the templates they were recommended and finishing the task earlier. Furthermore, participants with more experience in security were more willing to suggest more security requirements and apply the security requirements templates more carefully.

Two main context factors inherent to UCR18a and UCR18b must be discussed. The first one is the use of SRS based on the ISO standard. The fact that these functional requirements adhere to this standard facilitates their understanding, given that their wording, structure and level of detail remain uniform. The standard provides additional information to the reader regarding the context of the application and its purpose, a detailed description of each of the 14 functional requirements with its prerequisites, inputs, its process and outputs. Replications using this SRS had the overall highest coverage. It could be hypothesized that if participants know details about the software system under study, they would perform better in this metric. Besides, the use of this requirements document seem not to affect performance in the other metrics. This may confirm that this approach could be used with this type of requirement representation and instantiated to the health and mobile domains. Finally, the idea of working with digital documents seemed a good alternative and it worked. Both methods work very similar in these aspects and sending these files to the researchers via email did not present any trouble. The only potential drawback of this method, which can affect the efficiency variable in this study, is time tracking at home.

We compiled the feedback provided from participants regarding the use of security requirements templates. Participants' opinions were categorized following the categories used in [8]. Participants stated that this process provides a good starting point (20%) useful for developing secure systems, it has good coverage and applicability (40%) based on the six security objectives, it helps in thinking about security (27%) mainly for non-security experts, and it helps in phrasing requirements (27%) improving the requirement eliciting process. They claimed that more templates and support (20%) would help, and finally, it had to be applied with caution (13%) to not choose templates without critical thinking.

V. CONCLUSIONS

We replicated a previous study that analyzed the effectiveness of security requirements templates. Our results support some previous results: treatment group performed significantly better than the control group in terms of the coverage of the identified security requirements. Besides, the requirements elicitation process performed significantly better in relevance and efficiency metrics in two of the three replications. Security requirements templates supported participants in the identification of a core set of the security requirements and participants were favorable towards the use of templates in identifying security requirements. This study showed the feasibility of the approach to be instantiated in the context of a mobile health application to generate specific security requirements templates. For practitioners, this process may help them identifying a core set of security requirements; think about software security concerns and raises awareness about this topic.

For future replications, it would be interesting to investigate new problem domains, and explore new ways in which participants may feel more engaged before and during the elicitation process. Furthermore, the effect of context factors must continue under investigation in order to evaluate if they are conductive to producing better results.

ACKNOWLEDGMENT

This study was part of the first author's final applied research work, advised by members of the Empirical Software Engineering Group (ESEG) from the University of Costa Rica. The authors would like to thank Gabriela Barrantes, Ricardo Villalón and Gustavo Esquivel for validating the security requirement templates used in the replications. We also would like to thank practitioners of the study for their participation.

REFERENCES

[1] Karim, N. S. A., Albuolayan, A., Saba, T., & Rehman, A. (2016). The practice of secure software development in SDLC: an investigation through existing model and a case study. Security and Communication Networks, 9(18), 5333-5345.

[2] Riaz, M. (2016). Inferring Security Requirements from Natural Language Requirements Artifacts (doctoral dissertation). North Carolina State University, North Carolina, USA.

[3] Riaz, M., Slankas, J., King, J., & Williams, L. (2014). Using templates to elicit implied security requirements from functional requirements-a controlled experiment. In Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (p. 22). ACM.

[4] Fabian, B., Gürses, S., Heisel, M., Santen, T., & Schmidt, H. (2010). A comparison of security requirements engineering methods. Requirements engineering, 15(1), 7-40.

[5] Mellado, D., Blanco, C., Sánchez, L. E., & Fernández-Medina, E. (2010). A systematic review of security requirements engineering. Computer Standards & Interfaces, 32(4), 153-165.

[6] Riaz, M., & Williams, L. (2012). Security requirements patterns: understanding the science behind the art of pattern writing. In 2012 IEEE Second International Workshop on Requirements Patterns (RePa 2012) (pp. 29-34). IEEE.

[7] Riaz, M., King, J., Slankas, J., & Williams, L. (2014). Hidden in plain sight: Automatically identifying security requirements from natural language artifacts. In Requirements Engineering Conference (RE), 2014 IEEE 22nd International (pp. 183-192). IEEE.

[8] Riaz, M., King, J., Slankas, J., Williams, L., Massacci, F., Quesada-López, C., & Jenkins, M. (2017). Identifying the implied: Findings from three differentiated replications on the use of security requirements templates. Empirical Software Engineering, 22(4), 2127-2178.

[9] Jensen, M., Quesada-López, C., Zúñiga, G., Chinnock, A., & Jenkins, M. (2015). Design, Development and Evaluation of a Mobile Application for Goal Setting and Self-Monitoring of Dietary Behaviors. Journal of the Academy of Nutrition and Dietetics, 115(9), A67.

[10] Quesada-López, C., Jensen, M. L., Zúñiga, G., Chinnock, A., & Jenkins, M. (2016). Design, development and validation of a mobile application for goal setting and self-monitoring of dietary behaviors. In Central American and Panama Convention (CONCAPAN XXXVI), 2016 IEEE 36th (pp. 1-6). IEEE.

[11] Haley, C. B., Moffett, J. D., Laney, R., & Nuseibeh, B. (2006). A framework for security requirements engineering. In Proceedings of the 2006 international workshop on Software engineering for secure systems (pp. 35-42). ACM.

[12] Salini, P., & Kanmani, S. (2012). Elicitation of security requirements for e-health system by applying Model Oriented Security Requirements Engineering (MOSRE) framework. In Proceedings of the Second International Conference on Computational Science, Engineering and Information Technology (pp. 126-131). ACM.

[13] Mead, N. R., & Stehney, T. (2005). Security quality requirements engineering (SQUARE) methodology (Vol. 30, No. 4, pp. 1-7). ACM.

[14] Schumacher, M., Fernandez-Buglioni, E., Hybertson, D., Buschmann, F., & Sommerlad, P. (2013). Security Patterns: Integrating security and systems engineering. John Wiley & Sons.

[15] Carver, J. C. (2010). Towards reporting guidelines for experimental replications: A proposal. In 1st international workshop on replication in empirical software engineering.

[16] Mobile banking with Cyclos. (n.d.). Retrieved from https://www.cyclos.org/mobilebanking/

References

Al-Hamdani, W., & Dixie, W. (2009). Information Security Policy in Small Education Organization. *Information Security Curriculum Development Conference (InfoSecCD)*, Kennesaw, Georgia, USA, September 25-26, pp.72-78. doi: 10.1145/1940976.1940991

Bani-Salameh, H., & Al jawabreh, N. (2015). Towards a Comprehensive Survey of the Requirements Elicitation Process Improvements. *Proceedings of the International Conference on Intelligent Information Processing, Security and Advanced Communication (IPAC '15)*. Batna, Algeria, 23-25 November. doi: 10.1145/2816839.2816872

Barcelos, L. & Penteado, R. (2017). Elaboration of software requirements documents by means of patterns instantiation. *Journal of Software Engineering Research and Development*. 5(3), doi: 10.1186/s40411-017-0038-9

Beckers, K., Côte, I. & Goeke, L. (2014). A Catalog of Security Requirements Patterns for the Domain of Cloud Computing Systems. *Proceedings of the 29th Annual ACM Symposium on Applied Computing (SAC '14)*, pp. 337-342. doi: 10.1145/2554850.2554921

Bennaceur, A., Bandara, A., Jackson, M., Liu, W., Montrieux, L., Tun, T., Yu, Y., & Nuseibeh, B. (2014). Requirements-driven mediation for collaborative security. *Proceedings of the 9th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, Hyderabad, India, pp. 37-42. doi: 10.1145/2593929.2593938

Bourque, P., & Fairley, R.E., eds. (2014). Guide to the Software Engineering Body of Knowledge, Version 3.0. IEEE Computer Society.

Braber, F., Hogganvik, I., Lund, M., Stølen, K & Vraalsen, F. (2007). Model-based security analysis in seven steps – a guided tour to the CORAS method. *BT Technology Journal*, *25*(1), Massachusetts, USA, pp. 101-117. doi: 10.1007/s10550-007-0013-9

Bulusu, S., Laborde, R., Wazan, A., Barrère, F., & Benzekri, A. (2017). Which security requirements engineering methodology should I choose? Towards a requirements engineering-based evaluation approach. *Proceedings of the 12th International Conference on Availability, Reliability and Security*. Reggio Calabria, Italy. doi: 10.1145/3098954.3098996

Callele, D., & Makaroff, D. (2006). Teaching Requirements Engineering to an Unsuspecting Audience. *Proceedings of the 37th SIGCSE technical symposium on computer science education*. *38*(1), New York, USA, pp. 433-437. doi: 10.1145/1121341.1121475

Cerpa, N., & Verner, J. (2009). Why did your project fail? *Communications of the ACM – Finding the Fun in Computer Science Education*, *52*(12), pp. 130-134. doi: 10.1145/1610252.1610286

Chemuturi, M. (2013). *Requirements Engineering and Management for Software Development Projects*. New York, Springer Science.

Cyclos payment software. (n.d.). Retrieved from https://www.cyclos.org/

Cysneiros, L. and Sampaio do Prado, J. (2002). Non-Functional Requirements: From Elicitation to Modelling Languages. *Proceedings of the 24th International Conference on Software Engineering (ICSE)*, Orlando, Florida, USA, pp. 699-700.

El-Hadary, H., & El-Kassas, S. (2014). Capturing security requirements for software systems. *Journal of Advanced Research*, *5*(4), Cairo, Egypt, pp. 463-472. doi: 10.1016/j.jare.2014.03.001

Fabian, B., Gürses, S., Heisel, M., Santen, T. & Schmidt, H. (2010). A comparison of security requirements engineering methods. *Requirements engineering (RE'09)*, 15(1), pp. 7-40. doi: 10.1007/s00766-009-0092-x

Firesmith, D. (2003a). Analyzing and Specifying Reusable Security Requirements. *Proc. Solid Freeform Fabrication Symposium*, pp. 507-514.

Firesmith, D. (2003b). "Engineering security requirements," *J. Object Technology*, vol. 2, p. 16.

Firesmith, D. (2004). Specifying Reusable Security Requirements. *Journal of Object Technology*, 3(1), pp. 61-75. doi:10.5381/jot.2004.3.1.c6.

Haley, C., Moffett, J., Laney, R., & Nuseibeh, B. (2006). A Framework for Security Requirements Engineering. *Proceedings of the 2006 International Workshop on Software Engineering for Secure Systems (SESS)*, May 20–21, Shanghai, China, pp. 35-42. doi: 10.1145/1137627.1137634

Haley, C., Moffett, J., Laney, R., & Nuseibeh, B. (2008). Security requirements engineering: A framework for representation and analysis. *IEEE Transactions on Software Engineering*, *34*(1), pp.133-153. doi: 10.1109/TSE.2007.70754

Hamid, B. & Weber, D. (2018). Engineering secure systems: Models, patterns and empirical validation. *Computers & Security*, 77, pp.315-348. doi: 10.1016/j.cose.2018.03.016

Institute of Electrical and Electronics Engineers. (1990). IEEE Standard Glossary of Software Engineering Terminology. Retrieved from http://standards.ieee.org/findstds/standard/610.12-1990.html

Jensen, M., Quesada-López, C., Zúñiga, G., Chinnock, A., & Jenkins, M. (2015). Design, Development and Evaluation of a Mobile Application for Goal Setting and Self-Monitoring of Dietary Behaviors. *Journal of the Academy of Nutrition and Dietetics*, 115(9), A67. doi: https://doi.org/10.1016/j.jand.2015.06.240

Lai, H., Peng, R., & Ni, Y. (2014). Evaluating the BPCRAR Method: A Collaborative Method for Business Process Oriented Requirements Acquisition and Refining. *Communications in Computer and Information Science (CCIS), 432*, pp. 90-104.

Mead, N., & Stehney, N. (2005). Security quality requirements engineering (SQUARE) methodology. *Proceedings of the 2005 workshop on Software engineering for secure systems—building trustworthy applications (SESS '05), 30*(4), pp. 1-7. doi: 10.1145/1083200.1083214

Mead, N., Houg, E. & Stehney II, T. (2005). Security quality requirements engineering (SQUARE) methodology. *Technical Report CMU/SEI-2005-TR-009 Software Engineering Institute*, Carnegie Mellon University.

Mellado, D., Blanco, C., Sánchez, L., & Fernández-Medina, E. (2010). A systematic review of security requirements engineering. *Computer Standards & Interfaces 32*(4), pp. 153-165. doi:10.1016/j.csi.2010.01.006

Nachar, N. (2008). The Mann-Whitney U: A Test for Assessing Whether Two Independent Samples Come from the Same Distribution. *Tutorials in Quantitative Methods for Psychology*, 4(1), pp. 13-20. doi: 10.20982/tqmp.04.1.p013.

Nguyen, T., Vo, B., Lumpe, M., & Grundy, J. (2012). REInDetector: A Framework for Knowledge-Based Requirements Engineering. *Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pp. 386-389. doi: 10.1145/2351676.2351754

Riaz, M. & Williams, L. (2012). Security Requirements Patterns: Understanding the Science Behind the Art of Pattern Writing. *Proceedings of the Second International Workshop on Requirements Patterns (RePa)*, Chicago, Illinois, USA, pp. 29-34. doi: 10.1109/RePa.2012.6359977

Riaz, M., Breaux, T., & Williams, L. (2015). How have we evaluated software pattern application? A systematic mapping study of research design practices. *Information and Software Technology*, 65, pp. 14-38. doi: 10.1016/j.infsof.2015.04.002

Riaz, M., King, J., Slankas, J., and Williams, L. (2014a). Hidden in Plain Sight: Automatically Identifying Security Requirements from Natural Language Artifacts. *Proceedings of the 22nd International Conference on Requirements Engineering (RE)*, Karlskrona, Sweden, Aug 25-29, pp. 183-192. doi: 10.1109/RE.2014.6912260

Riaz, M., King, J., Slankas, J., Williams, L., Massacci, F., Quesada-López, C., & Jenkins, M. (2017). Identifying the implied: Findings from three differentiated replications on the use of security requirements templates. *Empirical Software Engineering (ESE), 22*(4) pp. 2127-2178. doi: 10.1007/s10664-016-9481-1

Riaz, M., Slankas, J., King, J., & Williams, L. (2014b). Using templates to elicit implied security requirements from functional requirements - A controlled experiment. *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pp. 1–10. doi: 10.1145/2652524.2652532

Riaz, M., Stallings, J., Singh, M., Slankas, J. & Williams, L. (2016a). DIGS – A framework for discovering goals for security requirements engineering. *Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, Ciudad Real, Spain. doi: 10.1145/2961111.2962599

Riaz, M. (2016b). *Inferring Security Requirements from Natural Language Artifacts* (Doctoral dissertation, North Carolina State University, North Carolina, USA). Retrieved from https://repository.lib.ncsu.edu/bitstream/handle/1840.16/11378/etd.pdf?sequence=2&isAllowed=y

Romero-Mariona, J. (2009). Secure and Usable Requirements Engineering. *24th IEEE/ACM International Conference on Automated Software Engineering (ASE '09)*, Auckland, New Zealand, November 16-20, pp. 703-706. doi: 10.1109/ASE.2009.81

Salini, P., & Kanmani, S. (2012). Elicitation of Security Requirements for E-Health System by applying Model Oriented Security Requirements Engineering (MOSRE) Framework. *Proceedings of the Second International Conference on Computational Science, Engineering and Information Technology (CCSEIT)*, Coimbatore UNK, India, pp. 126-131. doi: 10.1145/2393216.2393238

Schmidt, D., Fayad, M., & Johnson, R. (1996). Software Patterns. *Communications of the ACM, 39*(10), pp. 37-39. doi: 10.1145/236156.236164

Schumacher, M. & Roedig, U. (2001). Security engineering with patterns. *Proceedings of the 8th Conference on Pattern Languages of Programs (PLoP)*.

Schumacher, M., Fernandez-Buglioni, E., Hyberston, D., Buschmann, F., & Sommerlad, P. (2006). *Security Patterns: Integrating Security and Systems Engineering*. West Sussex, England: John Wiley & Sons, Ltd.

Tantithamthavorn C., Ihara A., Hata H., & Matsumoto K. (2014) Impact Analysis of Granularity Levels on Feature Location Technique. In: Zowghi D., Jin Z. (eds) Requirements Engineering. *Communications in Computer and Information Science (CCIS)*, 432, pp. 135-149. Springer, Berlin, Heidelberg

Tu YC., Tempero E., & Thomborson C. (2014) Evaluating Presentation of Requirements Documents: Results of an Experiment. In: Zowghi D., Jin Z. (eds) Requirements Engineering. Communications in Computer and Information Science, 432, pp. 120-134. Springer, Berlin, Heidelberg

Virtual Lifetime Electronic Record. (n.d.). Retrieved from https://www.health.mil/Military-Health-Topics/Technology/VLER-HIE

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M., Regnell, B., & Wesslén, A. (2012). Experimentation in Software Engineering. Springer, Berlin, Heidelberg

Yahya S., Kamalrudin M., Sidek S., & Grundy J. (2014) Capturing Security Requirements Using Essential Use Cases (EUCs). In: Zowghi D., Jin Z. (eds) Requirements Engineering. Communications in Computer and Information Science, 432, pp. 16-30 Springer, Berlin, Heidelberg