Towards A Requirements Model of System Security Using International Standards

Kenza Meridji\textsuperscript{1}, Khaled AlMakhadmeh\textsuperscript{2}, Khalid T. Al-Sarayreh\textsuperscript{2}, Anas Abuljadayel\textsuperscript{2} and Mohammad Khalaf\textsuperscript{2}

\textsuperscript{1}Department of Software Engineering, University of Petra, Amman, 11196, Jordan
\textsuperscript{2}Department of Software Engineering, Hashemite University, Zarqa, 13115, Jordan

kmeridji@uop.edu.jo, khaled.almakadmeh@hu.edu.jo, khalidt@hu.edu.jo, anas.abuljadayel@itc.hu.edu.jo, hamoodnooman@itc.hu.edu.jo

Abstract

Currently, security requirements are defined at system levels in software engineering projects as system quality requirements. Later on system engineers must identify such system requirements into software requirements. It will be assigned to system software to conform to security at software and hardware requirements. For instance, a number of security requirements concepts and terms are described by IEEE, ISO and ECSS international standards and explained at different stages of requirements at the system, software and hardware levels.

This paper assembles and systematizes these candidate security-related descriptions into a requirements model for the specification of software functional user requirements assigned from additional system security non-functional requirements. The structure is made using the ISO-19761 international standard model of software requirements, which accept the new model to measure the functional size of software functionality and non-functionality of security systems.

Keywords: Security Requirements, Non-functional Requirements – NFR, Functional User Requirement FUR, Functional Requirements – FR, IEEE, ISO, ECSS Standards

1. Introduction

The term NFR has been used both in academia and industry but there is still no common definition for this term and the NFR domain is hard to model; the NFR are defined, interpreted, and evaluated differently by different people. The relationships between these requirements are often contradictory. Non-functional requirements have not been taken into consideration in the software engineering practice. And they are not understood enough compared to other cost factors.

Currently, software development lifecycle has several phases; one of these phases is software requirements. In this phase design engineers thoroughly identify the software functional requirements (FR) and the software non-functional requirements (NFR) at a very high level of abstraction.

Non-functional requirements have a major impact on the software system being developed and are a very important aspect to the success of software projects; such NFR’s are characterized at system levels rather than software levels.

Until now, there is no concrete way for quantifying and describing these NFR. In other way, if these requirements are not adequately addressed, potential problems may occur, i.e., if a system has inconsistencies and poor quality, which in turn will cause cost amplification and time overrun for fixing these defects and errors. Literature review
illustrates that NFR are often overlooked, weakly understood, and generically defined at the system requirements phase.

In practice, requirements are addressed at the system level [1-3], either as high-level system functional requirements (system-FR) or as high-level system non-functional requirements (system-NFR). Later these requirements can be detailed and assigned to specified-related functions, which may be implemented in hardware or software as software functional user requirements (software-FUR) [4] (see figure 1).

Specifically, System-FR are requirements which are concerned with what the system should do and what are its functionalities, which system-NFR are concerned with how the system should behave. Subsequently, at the requirements engineering phase, system-NFR can be specified and detailed as software functionality from user point of view.

Figure 1. How NFR are Mapped to Software-FUR

International standards have worked on the description of system security as a nonfunctional requirement whereas there are number of concepts provided in the ECSS, ISO 25021 and IEEE to describe various types of security requirements. Nonetheless, there is still a variation of perspectives and concepts of security in the international standards.


The Approach used in this paper is an extension to the approaches used in [9-13], as we refer to these approaches by proposing a new approach for system security non-functional requirements. This paper is focused on system security requirements to present a requirements model of system security on the basis of international standards concepts,
aligned with ISO 19761 as a base for the description and measurements of functional requirements.

For the rest of the paper, related work is presented in Section 2. A description of how software functional user requirements are viewed in the ISO-19761 is presented in Section 3. Identification of security requirements descriptions in the international standards is described in Section 4. The requirements model of system security is defined in Section 5. A case study that includes a sizing of the requirements model of system security is illustrated in Section 6. Finally, in Section 7, discussion and future work are presented.

2. Related Work

In this section, we will highlight the related works on NFR in the literature. Chung et al., [7] considered that majority of existing requirements models and requirements specification languages lacked proper treatment of quality characteristics. Chung et al., [8] mentioned that NFR have been referred to as “-ilities” or “-ities”, words ending with the string “-ility” or “-ity. A huge list of these words is presented in [8].

Chung et al., [15] proposed a NFR Framework that separates the concept of functionality from other quality attributes and concerns. The framework consisted of five major components: a set of goals, a set of link types, a set of generic methods, a collection of correlation rules, and a labeling procedure. Their framework was proposed for representing and using non-functional requirements during the development process.

Glinz [16] believed that NFR are related to the non-behavioral parts of a system, specifying constraints on how the system must operate, and with some restrictions placed on the development and the product being developed, other than an observable behavior, it describes a characteristic or a property that a system must exhibit.

Dewi Mairiza et al., [17] considered two different perspectives for NFR; the first one is NFRs as the requirements that describe the quality attributes that the software product must have; the second one is NFRs as the requirements that describe the properties, characteristics or constraints that a software system must exhibit.

Davis [18] considers NFR as behavioral vs. non-behavioral requirements. Functional requirements are defined as the requirements that describe what the system should do, while the rest are considered to be non-functional [19]. Cleland-Huang et al., [20] believed the usefulness of the early detection of NFRs since it enables consideration and incorporation of system level constraints into early architecture designs, and that those non-functional requirements (NFRs) help in describing very important constraints onto the development and behavior of the software system.

Paech et al., [5] proposed that the term “non-functional requirement” is used to differentiate requirements related to “how good” the software performs something in contrast to functional requirements, which focus on “what” the software does.” Kassab et al., [6] extended the taxonomy of the NFR framework by including the “hard goals NFRs” concept into the requirements engineering process in their semi-formal model.

Kotonya et al. [21] believed that NFRs are “restrictions and constraints among system services”. Gilb [22] considered requirements as constraints, qualities, functions and costs and, in which constraints, qualities and costs can be regarded as NFRs. Qualities indicates “How well the function will perform” [23].

Somerville considers the requirements for a system as the descriptions of the services provided by the system and its operational constraints [19]. Leite J.C.S.P. et. al., [24] describes NFRs as goals that might conflict among each other and must be represented as soft goals to be satisfied.

Haley et al., [25] described security requirements as adequate security requirements, meaning that, if these adequate requirements were respected, they will lead to a system security goals being satisfied, were system here means the software and, in addition,
people who use the software, and equipment around the software (computers, printers, etc.). They also claimed that three conditions must be satisfied by these adequate security requirements; the first one is “definition: one must know what security requirements are”, the second one is “assumptions. Security requirements must take into consideration an analyst’s implicit or explicit assumption that an object in the system will behave as expected.” the third one is “satisfaction: One must be able to determine whether the security requirements satisfy the security goals and whether the system satisfies the security requirements”. Also they defined security requirements as “the constraints on the functions of the system”, where these constraints implement one or more security goals.

In the SWEBOK [26], software security requirements deals with the clarification and specification of security policy and objectives of software requirements. This lays the foundation for security consideration in the software development factors. This phase include software requirement and threats. The former refers to specific functions that are required for the sake of security the latter refers to the possible ways that the security of software is threatened.

3. A Generic View of Software-FUR in ISO

The COSMIC FSM method [14] is aimed at measuring the size of software based on identifiable FUR. Once identified, those requirements are assigned to hardware and software from unifying perspective of a system integrating these two “components”.

Since COSMIC is aimed at sizing software, only the requirements assigned to the software are currently considered in its measurement procedure.

The COSMIC [14] standard is considered as a second generation of an FSM method. COSMIC method has been extensively tested and its use is increasing especially in the real time and telecommunication worlds; it is as well compatible with modern specification methods such as unified modeling languages (UML) and object oriented (OO) techniques.

COSMIC method defines the principles, rules and a process for measuring the functional size of a piece of software. ‘Functional size’ is a measure of the ‘amount of functionality’ provided by the software.

Figure 2 illustrates the data flow between software and hardware from a functional perspective in COSMIC method. One note that software is bounded by hardware on both sides, data movements can occur with four movement types (Entry, eXit, Read, Write).

4. Identification of Security Requirements Descriptions in the International Standards

In this section, a review on security requirements perspectives, concepts and terminologies in the international standards is presented. By identifying concepts and terms in these standards, the specification of some aspects for the system security-NFR is possible. As a result of the identification of these aspects, which components to be included in the requirements model of system security can be determined. Figure 3 illustrates the mapping of system security-NFR into software-FUR for security.
4.1. Security Perspectives and Concepts as Described in IEEE

Table 1 below presents various elements of system security as defined in the IEEE standard. Defined as a non-functional requirement, from the IEEE perspective security has some factors that can provide protection to the software from anonymous access or use result in a possible modification to the system that could lead to destruction of that system.
These factors include the need to keep specific log or history through some type of backup, the need to apply certain cryptography methods to keep the information safe from accidental access, to distribute functions on different modules, to check data integrity for critical variables and to eventually restrict the communication between various areas of the program.

**Table 1. Security Perspectives, Concepts and Terminologies in IEEE-830**

<table>
<thead>
<tr>
<th>Key perspectives</th>
<th>Concepts and terminologies</th>
</tr>
</thead>
</table>
| The key perspectives of security requirements from the IEEE perspective has some factors that can provide protection to the software from anonymous access or use result in a possible modification to the system that could lead to destruction of that system. | • Cryptographically techniques;  
• Specific log or history data sets  
• Assign certain functions to different modules  
• Some areas of the program should communicate in a constrained way  
• Examine data integrity for critical variables |

4.2. Security Perspectives and Concepts as Described in ISO

The ISO standard considers security as an important functionality portion to define the software product quality. ISO standards define security as the capability of software product to protect information and data in such a way that persons or systems with unauthorized access cannot read or edit them, while people or systems that are authorized can access them. Table 2 presents the various concepts and terms defined in the ISO standards.

**Table 2. Security Perspectives, Concepts and Terminologies in ISO**

<table>
<thead>
<tr>
<th>Key perspectives</th>
<th>Concepts and terminologies</th>
</tr>
</thead>
</table>
| The key perspectives of security in ISO is expressed as an element of the software functionality that defines the software product quality | • Access Auditability  
• Access Controllability  
• Data Corruption/ Prevention  
• Data Encryption |
### Table 3. Security Perspectives, Concepts and Terminologies in ECSS

<table>
<thead>
<tr>
<th>Key perspectives</th>
<th>Concepts and terminologies</th>
</tr>
</thead>
<tbody>
<tr>
<td>The key perspectives of security</td>
<td>• Access control roles for the system, person and groups.</td>
</tr>
<tr>
<td>requirements are considered as</td>
<td>• Redundant power, redundant data and automatic restart man made should be available.</td>
</tr>
<tr>
<td>specifications in the ECSS standards.</td>
<td>• System data integrity such as integrity with firewall, antivirus, external PKI.</td>
</tr>
<tr>
<td>These specifications include some factors</td>
<td></td>
</tr>
<tr>
<td>that are related to them, as a result,</td>
<td></td>
</tr>
<tr>
<td>these factors might expose some sensitive</td>
<td></td>
</tr>
<tr>
<td>information. Also, system security</td>
<td></td>
</tr>
<tr>
<td>requirements must be defined in</td>
<td></td>
</tr>
<tr>
<td>the requirements baseline as required by</td>
<td></td>
</tr>
<tr>
<td>the ECSS.</td>
<td></td>
</tr>
</tbody>
</table>

System security in the ECSS is described as:

1. Access control roles; which are for either a person or a group of persons, and access control per system or entity;
2. Availability for redundant power, redundant data and automatic restart;
3. System data integrity such as integrity with firewall, antivirus, external PKI (encryption and decryption of data) and integrity with different types of system backup (such as automatic, time interval, durability, data versioning and runtime backups).

### 5. Definition of Requirements Model using International Standards

This section presents the mapping of concepts and terms from the international standards identified in section 4, as a result of this mapping, three types of system security requirements are identified. These security requirements types are then specified into function types of security requirements. Later in this section, a requirements model of system security is presented by using the ISO-19761 proposed model of function size measurement.

#### 5.1. IEEE, ISO and ECSS Perspectives and concepts for security are mapped

Through the collection of the various definitions, Table 4 presents the key perspectives and concepts from section 4 on software-FUR for system security-NFR.

#### 5.2. Types of Security Requirements

Based on the previous sections, three types of system security requirements can be derived:

- System Confidentiality.
- System Availability.
- System Integrity.
Table 4. Security Requirements in IEEE, ISO and ECSS

<table>
<thead>
<tr>
<th>System security requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Confidentiality</td>
</tr>
<tr>
<td>• Availability</td>
</tr>
<tr>
<td>• Integrity</td>
</tr>
<tr>
<td>• Access control role</td>
</tr>
<tr>
<td>• Security login</td>
</tr>
<tr>
<td>• Authentication</td>
</tr>
<tr>
<td>• Redundant power and network</td>
</tr>
<tr>
<td>• Redundant data</td>
</tr>
<tr>
<td>• Automatic restart</td>
</tr>
<tr>
<td>• Firewall</td>
</tr>
<tr>
<td>• Antivirus</td>
</tr>
<tr>
<td>• External PKI</td>
</tr>
<tr>
<td>• Backup type</td>
</tr>
<tr>
<td>• Encryption and decryption</td>
</tr>
</tbody>
</table>

5.3. Software System Security Functions to be Specified

Table 5 lists system security types identified in the previous section with specific functions assigned for each type of the three security requirements.

5.4. Identification of the System Security Function Types

As a result from the previous section, functions were assigned for each security type, function types are also identified. Three types of system security function types are identified: confidentiality function type, availability function type and integrity function type. In a more detailed manner, each of these function types then has more specific functions assigned to it. Table 6 shows these function types with their respective specific functions. Also, figures for each function type with their respective description are presented below

- Function Type 1: System Confidentiality Functions – Figure 4
  - For the access control role function, data groups are sent to and received from either security login function or authentication function.
  - For the security login function, data groups are sent to and received from either access control role function or authentication function.
  - For the authentication function, data groups are sent to and received from either security login function or access control role function.
Table 5. System Security Functions that can be assigned to Software

<table>
<thead>
<tr>
<th>I</th>
<th>Function type name</th>
<th>Security function</th>
<th>Activity (example)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Confidentiality</td>
<td>Access control role function (ACRF)</td>
<td>Per person, Per group</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Security login function (SLF)</td>
<td>User name &amp; password, Password change, Smart card, Single sign on, Automatic login</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Authentication function (AF)</td>
<td>Per person, Per group, Per entity, Per system, Smart card, Biometrics</td>
</tr>
<tr>
<td>2</td>
<td>Availability</td>
<td>Redundant power and network function (RPNF)</td>
<td>Available 24 H/7 Days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Redundant data function (RDF)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Automatic restart function (ARF)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Integrity</td>
<td>Firewall function (FF)</td>
<td>Attack detection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Antivirus function (AF)</td>
<td>Hot and cold backup</td>
</tr>
<tr>
<td></td>
<td></td>
<td>External PKI function (EPKIF)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Backup type function (BTF)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Encryption and decryption function (EDF)</td>
<td>Encryption and decryption Algorithm</td>
</tr>
</tbody>
</table>

Figure 4. System Confidentiality functions
Table 6. Function Types for Security Functions that may be Assigned to Software

<table>
<thead>
<tr>
<th>System security function types</th>
<th>System security functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. System Confidentiality (SC)</td>
<td>Access control role function (ACRF)</td>
</tr>
<tr>
<td></td>
<td>Security login function (SLF)</td>
</tr>
<tr>
<td></td>
<td>Authentication function (AF)</td>
</tr>
<tr>
<td>2. System Availability (SA)</td>
<td>Redundant power and network function (RPNF)</td>
</tr>
<tr>
<td></td>
<td>Redundant data function (RDF)</td>
</tr>
<tr>
<td></td>
<td>Automatic restart function (ARF)</td>
</tr>
<tr>
<td>3. System Integrity (SI)</td>
<td>Firewall function (FF)</td>
</tr>
<tr>
<td></td>
<td>Antivirus function (AF)</td>
</tr>
<tr>
<td></td>
<td>External PKI function (EPKIF)</td>
</tr>
<tr>
<td></td>
<td>Backup type function (BTF)</td>
</tr>
<tr>
<td></td>
<td>Encryption and decryption function (EDF)</td>
</tr>
</tbody>
</table>

- Function Type 2: System Availability Functions – Figure 5
  - For the redundant power and network function, data groups are sent to and received from either redundant data function or automatic restart function.
  - For the automatic restart function, data groups are sent to and received from either redundant power or network function or redundant data function.
  - For the redundant data function, data groups are sent to and received from either redundant power and network function or automatic restart function.

- Function Type 3: System Integrity Functions – Figure 6
  - For firewall function, data groups are sent to and received from the Antivirus function.
  - For antivirus function, data groups are sent to and received from either the firewall function or the external PKI function.
- For external PKI function, data groups are sent to and received from either an antivirus function or backup type function.
- For backup type function, data groups are sent to and received from external PKI function.

**Figure 6. System Integrity Functions**

### 5.5. Security Function Types: system and COSMIC modeling views

This section presents security function types in more details, that include views in both system and cosmic modelling. Each function type is discussed with its respective functions.

#### 5.5.1. System Modeling Views:

Discussed below are the three function types of security in the system modelling view, and each function type is viewed with its respective functions to specify how these functions relate to each other.

*Function type 1: System Confidentiality Functions:* The first type of security functions is the confidentiality functions, which includes three functions. A system modelling view of the data flow for these functions is described in Figure 7.

Contributions of the three functions are described as follows:

1. Access Control Role Function (ACRF): the main contribution of this function is to control who can access system resources for either a person or a group of users.

2. Security Login Function (SLF): this function provides a systematic way of identifying credentials required to obtain access to the system by checking the validity of the username and password.

3. Authentication Function (AF): in order to confirm the truth and validity of the credentials authentication function required to establish the identity of each user and giving the respective privileges for each user.

The three functions in Figure 7 use an intermediary service (IS) in order to complete and provide the user with the required functionality, and for the interaction between their services. A persistent storage is used among the three functions to share their data and system resources.
Function type 2: System Availability Functions: The second type of security functions is the availability functions, which includes three functions. A system modeling view of the data flow for these functions is described in Figure 8. The three functions are: Redundant power and network function (RPNF), redundant data function (RDF), automatic restart function (ARF).

Function type 3: System Integrity Functions: The third type of security functions is the integrity functions, which includes five functions. A system modeling view of the data flow for these functions is described in Figure 9. The five functions are: Firewall function (FF), antivirus function (AF), external PKI function (EPKIF), backup type function (BTF) and encryption and decryption function (EDF).
5.5.2. COSMIC Modeling Views: This part presents COSMIC modeling views of the three function types of security, each function type is viewed with its respective functions and the data movements between these functions.

The first function type, which is the system confidentiality function, is illustrated in COSMIC modeling view in Figure 10. An explanation of the figure is described below:

- The three functions of confidentiality, that is (ACRF, SLF, and AF), use an intermediary service to connect their services together, and to send or receive data groups.
- The functions (ACRF, SLF, and AF) use persistent storage for reading data groups of other services, and for writing their data movements on it. Persistent storage is also shared by all other security function types.
Further, Figure 11 below shows the COSMIC modeling for system availability functions:

- The three functions of Availability, that is (RPNF, RDF and ARF), use an intermediary service to connect their services together, and to send or receive data groups.
- The functions (RPNF, RDF, and ARF) use persistent storage for reading data groups of other services, and for writing their data movements on it.
Figure 11. System Availability Functions: COSMIC modeling view

Figure 12 below shows the COSMIC modeling for system integrity functions.

- The functions of integrity, that is (AF, EPKIF, BTF and EDF), use an intermediary service to connect their services together, and to send or receive data groups.

- The functions (FF, AF, EPKIF, BTF and EDF) use persistent storage for reading data groups of other services, and for writing their data movements on it.
5.6. A Requirements Model of System Security

This section presents the proposed requirements model of system security, after the identification of security requirements from international standards, and specifying security functions types, it is required to specify the relationships between these function types. This is done, using COSMIC modelling for graphical representation, in figure 13 below.

The requirements model contains three sub models that represent the three identified function types of system security. In a more specific manner, the first sub model is for the first function type of security, that is, system confidentiality function. Through the data movements that are sent or received from the functions of system confidentiality (ACRF, SLF, and AF). One can measure the functional size of the system confidentiality functions using this sub model.

The same approach applied for the first sub model can also be applied for the second and third sub models, that is, to measure the functional size for the second and third security function types; system availability and system integrity, respectively.

5.7. A Requirements Model of System Security using an SOA

Figure 14, illustrates the requirements model of system security using an SOA. The model is based on the system security requirements, functions and function types, and the use of COSMIC-SOA as explained in (COSMIC 2010).

The model of COSMIC-SOA separates functions into distinct services. These services communicate with each other by exchanging data in a well-defined, common format. The
COSMIC requirements model in figure 13 is considered as a high-level requirements model; this model instead is more detailed and can be used for the specification and measurement of functionality when assigned to software-FUR.

Figure 13. COSMIC Requirements Model of System Security using International Standards
Figure 14. COSMIC-SOA Requirements Model of System Security
6. Case Study: A Sizing of the Requirements Model of System Security

Case Study: automatic teller machine (ATM): In this section, a case study is presented on the proposed requirements model of system security by applying it to an automatic teller machine (ATM) system. The ATM system allows the user to do a number of cash transactions: withdraw, deposit, view account balance. The automatic teller machine is connected with a bank computer through a link between them, and it has software that communicates with the software on the bank computer to perform various cash transactions and to access the accounts database for information on accounts. The ATM consists of a screen for displaying information to the user, a numbered keypad to enter information to the ATM and a card slot reader for reading the ATM card. The ATM requirements block diagram is presented below in Figure 15, where the ATM software boundary is surrounded by a dashed box.

![Figure 15: ATM requirements Block Diagram](image-url)
The user of the ATM inserts his card into the card reader slot, and the ATM interacts with the user by asking to enter the PIN of the card, after the information is verified, the ATM will communicate with the bank computer in order to access the accounts database to get the balance information. When the user inputs the desired transaction, the ATM performs it by connecting with the bank computer to do the specified transaction and store it in the accounts database. The transaction ends with the cash dispensed to the user and a receipt of balance information is printed.

Stakeholders of a project are people who are interested in a project; in this case is the ATM. The proposed requirements model of system security can be used to derive some security requirements related to the ATM software, then these can be assigned to the ATM software as security-FUR.

Since the proposed requirements model specifies a number of security functions that are identified through international standards, it can be used in the requirements engineering phase in order to make connection between the research and practice, by doing the steps below:

- Step 1: Determine the security functional requirements at a high level of the ATM using the proposed requirements model of system security as the reference.
- Step 2: Assign system security-FUR to software functions that can be applied to the ATM to add it to this case study.
- Step 3: The software-FUR for system security assigned to software for this case study is measured.

The above steps will be presented in a more specific way:

**Step 1: Specification and addition of security requirements at a high level.** In the ATM case study, stakeholders demanded four security requirements (R1, R2, R3 and R4) at the system level. In the proposed requirements model, security requirements are allocated to 11 security functions (F1 to F11):

- R1: The ATM should check if the inserted card is a valid cash card.
- R2: The ATM verifies if the entered PIN and the card bank code are correct by connecting with the bank computer.
- R3: The ATM should communicate with various bank computers of different banks through network links.
- R4: The ATM should decrypt the information on the bank card to read it.

The security requirements described earlier in this section correspond to a number of the security functions specified in the proposed requirements model of system security, mentioned below:

- R1: Authentication function
- R2: Security login function
- R3: Redundant power and network function
- R4: Encryption and decryption function

**Step 2: Assign these system security requirements to software functions, in order to add them to the ATM.** For the objectives of the ATM case study, the system security requirements (R1, R2, R3 and R4) specified above are assigned to the ATM components as described below:

- R1 (Authentication function): assigned to the ATM software, i.e. the ATM should have the ability to check if the inserted card is a valid cash card.
- R2 (Security login function): assigned to the ATM software and the card reader hardware’s new software functions to be added, i.e. the ATM should check
and verify that the entered PIN and the card bank code are correct by connecting to the accounts database.

- **R3 (Redundant power and network function):** assigned to the ATM software, the ATM should have the ability to connect with various bank computers of different banks through a network links.

- **R4 (Encryption and decryption function):** assigned to the ATM software, the ATM can decrypt the information on the bank card in order to read the required information.

The figure below presents the proposed requirements model of system security for the updated ATM with the allocations above. The software functions assigned to the existing software (ATM) are functions (R1, R2, R3 and R4) in the proposed model, while (R2) is the function assigned to the hardware in the updated card reader component.

**Step 3: Measuring the software-FUR for system security requirements.** The software developer is the one who has interest in measuring the added system security requirements in the form of new software functions to develop them. Measurement can be done using the COSMIC method (ISO-19761), and is applied on the added functional user requirements of system security assigned to software for the ATM. The measurement scope includes just those functions assigned to software not those assigned to hardware.

The information required to measure the functional size is available since specifying these security requirements with the use of the structure of the proposed requirements model of system security has made them coordinated with the COSMIC model of FUR. From the above, it can be assumed that: each specified security function has a single data group. The overall functional size as given in the ISO-19761 for the newly added security functions is gathered through the addition of all the data movements for each distinct security function.

Measurement results of the newly added security functions to the ATM are presented in table 7. The total functional size of the added software functions that is required to meet the system security requirements is 26 CFP.

**Table 7. Total Functional Size**

<table>
<thead>
<tr>
<th>Requirements model for security functions</th>
<th>FUR</th>
<th>E</th>
<th>X</th>
<th>R</th>
<th>W</th>
<th>Size in CFP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication function</td>
<td>R1</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Security login function</td>
<td>R2</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Security login function</td>
<td>R2</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Redundant power and network function</td>
<td>R3</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Encryption and decryption function</td>
<td>R4</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>One Intermediary service between R1, R2</td>
<td></td>
<td>4</td>
<td>4</td>
<td>-</td>
<td>-</td>
<td>8</td>
</tr>
<tr>
<td>and R4</td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>One Intermediary service between R3, R4</td>
<td></td>
<td>4</td>
<td>4</td>
<td>-</td>
<td>-</td>
<td>8</td>
</tr>
</tbody>
</table>

**Functional Size** 8 8 5 5 26 CFP
Figure 16. Requirements Model of System Security for the Updated ATM

7. Discussion and Future work

This research paper presented a proposed requirements model of system security, through the use of international standards in identifying different types and concepts of security. This research paper extends earlier work by Al-Sarayreh et al [9-13] in presenting such requirements model for system security.
This model was achieved by the process of identifying international standards views and concepts on system security. And then mapping these views and concepts to specify the three main function types of system security: system confidentiality, system availability and system integrity.

Each of these function types has its respective functions which accomplish its main functionality by communicating and interacting together through an intermediary service and with a shared persistent storage to store its data and system resources.

A small case study was introduced for the proposed model, applied on the ATM machine system; some of the security functions were applied through the proposed requirements model of system security on the ATM machine to improve its security. The case study showed that the model is applicable and has efficient results.

Future work includes applying this model on larger and more complex systems to assure its efficiency and validity for variety of systems and to make sure it is useful across different communities.

References

Kenza Meridji is an Assistant Professor in Software Engineering Department at University of Petra, Jordan. She completed her Ph.D. in Software Engineering from University of Québec (Canada) in 2009. She received his Master in Software Engineering from Concordia University 2005. Her research interest includes software engineering fundamentals, software principles and software requirements.

Khaled Almakadmeh is an Assistant Professor in the Software Engineering Department at Hashemite University, Jordan. He completed his Ph.D. in Software Engineering at University of Quebec (2013). He received his Master in Information Systems Security from Concordia University (2010) and his B.S. in computer science from Jordan University of Science & Technology (2008). His research interest includes software requirements, software metrics and software quality.
Khalid T. Al-Sarayreh is assistant professor of Software Engineering at Hashemite University in Jordan. He has a PhD degree in Software Engineering from École de Technologie Supérieure (ÉTS)—University of Québec (Canada). He also has a Doctoral degree in Computer Information Systems, an MSc in Computer Engineering (Embedded Systems) and a BS degree in Computer Science from Jordanian universities. From 2002 to 2005, he was at the KADDB (King Abdullah II Design and Development Bureau). From 2005 to 2006, he was an Assistant Professor at the Jordan University, and from 2006 to 2008, he was an Assistant Professor at the Faculty of Information Technology at the Applied Sciences University (Jordan). His research interests include: Software Real-Time Embedded Systems, Computer Networks, Non-functional Requirements for Embedded Systems and Applied Artificial Intelligence.

Anas Abuljadayel is a Master student in the Software Engineering Department at The Hashemite University, Jordan. He completed his B.S. in Software Engineering from The Hashemite University, Jordan in 2012. His research interest is Software Engineering with focus on Software Testing, Software Maintenance, Software Requirements and Software Quality.

Mohammad Khalaf is a Master student in the Software Engineering Department at The Hashemite University, Jordan. He completed his B.S. in Software Engineering from The Hashemite University, Jordan in 2013. His research interest is Software Engineering with focus on Software Maintenance and Cloud Computing, Software Requirements and Software Quality.