A Methodological Approach to Evaluate Security Requirements Engineering Methodologies: Application to the IREHDO2 Project Context

Romain Laborde, Sravani Teja Bulusu, Ahmad Samer Wazan, Arnaud Oglaza, Abdelmalek Benzekri

Journal of Cybersecurity and Privacy

MDPI

Article

A Methodological Approach to Evaluate Security Requirements Engineering Methodologies: Application to the IREHDO2 Project Context

Romain Laborde [1,*], Sravani Teja Bulusu [1,2], Ahmad Samer Wazan [1,3], Arnaud Oglaza [1] and Abdelmalek Benzekri [1]

[1] IRIT/Université Paul Sabatier, 31062 Toulouse, France; sravny_teja@yahoo.co.in (S.T.B.); ahmad-samer.wazan@irit.fr (A.S.W.); oglaza@irit.fr (A.O.); Abdelmalek.Benzekri@irit.fr (A.B.)
[2] Sopra Steria—I2S, 31772 Colomiers, France
[3] CAT Department, College of Technological Innovation, Zayed University, Abu Dabi 4783, United Arab Emirates
* Correspondence: Romain.Laborde@irit.fr

Abstract: An effective network security requirement engineering is needed to help organizations in capturing cost-effective security solutions that protect networks against malicious attacks while meeting the business requirements. The diversity of currently available security requirement engineering methodologies leads security requirements engineers to an open question: How to choose one? We present a global evaluation methodology that we applied during the IREHDO2 project to find a requirement engineering method that could improve network security. Our evaluation methodology includes a process to determine pertinent evaluation criteria and a process to evaluate the requirement engineering methodologies. Our main contribution is to involve stakeholders (i.e., security requirements engineers) in the evaluation process by following a requirement engineering approach. We describe our experiments conducted during the project with security experts and the feedback we obtained. Although we applied it to evaluate three requirements engineering methods (KAOS, STS and SEPP) in the context of network security, our evaluation methodology can be instantiated in other contexts and other methods.

Keywords: security requirement engineering; network security; KAOS; STS; SEPP; SABSA

Citation: Laborde, R.; Bulusu, S.T.; Wazan, A.S.; Oglaza, A.; Benzekri, A. A Methodological Approach to Evaluate Security Requirements Engineering Methodologies: Application to the IREHDO2 Project Context. *J. Cybersecur. Priv.* **2021**, *1*, 422–452. https://doi.org/10.3390/jcp1030022

Academic Editors: Isabel Praça, Silvio Ranise, Luca Verderame and Habtamu Abie

Received: 11 May 2021
Accepted: 23 June 2021
Published: 13 July 2021

1. Introduction

Over the past years, the growing dependency of the business-critical applications and processes on network technologies and services has expanded the threat landscape to a large extent. Today networks constitute the main vector for attacks [1]. Thus, considering network security is crucial for ensuring business continuity in light of the growing threats.

Network security mainly concerns the security architectural needs that describe network segmentation (a.k.a., security zoning); security constraints of network devices connecting the communicating end user systems; and security constraints of the data being transferred across the communication links [2]. The ultimate aim boils down to preventing illegitimate access to the data assets that carry vital information and influence the decisions relating business critical operations [3]. An appropriate design of the network architecture provides many advantages (e.g., isolation of low trust systems, limitation of a security breach's scope, costs savings) [4].

Currently, there exists numerous network security controls (e.g., firewalls, intrusion detection and prevention systems, VPNs, etc.) to implement network security [5–8]. Each time a security control (e.g., firewall or a proxy) is added or removed to the network, it will impact the applications running over the network, the quality of service of the network and also the cost of the network with new purchase and installation [9]. For

instance, adding a DMZ (de-militarized zone) into an existing network provides a perimeter defense by isolating the internal network devices that are accessible from the public network. Nevertheless, the DMZ significantly alters the network architecture. In this regard, considering network security too late in the network development cycle (i.e., post deployment of network designs) would render additional costs and complexity in terms of incorporating any changes to the existing infrastructures. The difference in the return of security investments considered at early or at late stages of the system development cycle can range from 12 to 21 percent [10]. Therefore, it is necessary to examine network security at earlier stages, i.e., right from the planning stages of networks architecting. This perspective is known as security-by-design, which enforces the consideration of security right from the requirements analysis.

Therefore, network security requirements play a critical role since they impact the decisions related to the implementation of security controls [2]. Indeed, bad network security requirements can lead to ineffective and costly security or worth security holes in the network security design. An effective network security requirement engineering is needed to help organizations in capturing cost-effective security solutions that protect their networks against malicious attacks while meeting the business requirements.

Requirements Engineering (RE), in general, is a sub-discipline of system/software engineering, which subsumes the activities of gathering (a.k.a. eliciting), evaluating and documenting system/software requirements. Security requirements engineering (SRE) methods extend RE to the security context, by enabling the integration of risk analysis concepts. They help organizations in capturing security requirements by analyzing the business risk impact of potential threats. However, the diversity of SRE methodologies leads security requirements engineers to an open question: How to choose one? Utilization of various procedures and tools, choosing unsuitable RE instruments or utilization of inadequate techniques for eliciting requirements are among the most frequently reported issues [11]. Several comparative studies on SRE methodologies are available. Nonetheless, their evaluation is often based on a set of ad hoc criteria, hence these comparison criteria may not fit the SRE needs of every company. Thus, a systematic method must be proposed to properly capture the actual company's needs regarding an SRE methodology.

In this article, we address this issue by proposing a requirements engineering-based evaluation methodology. It helps in characterizing the appropriateness of a SRE methodology by capturing the SRE needs of the stakeholders as well as the quality characteristics of good security requirements (a.k.a. quality criteria). We developed this evaluation methodology during the IREHDO2 project where we could interview security experts involved at each step of the security process. In this project, security experts of an aircraft company wanted to improve their security process in order to increase the assurance on the final security solution enforced on their aircraft networks. More precisely, they were interested in enhancing their security requirement practices. This group of security experts included security requirement engineers, risks analysts as well as security testing experts who are involved at different levels of the security process. Our task in this project consisted in proposing the best SRE methodology which will help them in writing good security requirements. However, each security expert had a different point of view on what could constitute a good SRE methodology. As consequence, setting generic evaluation criteria appropriate to any organization is not a good idea. A better approach is to develop a process to elicit these evaluation criteria and their weight from the stakeholders. We applied this methodology to our project context. This article consolidates and extends previous contributions [12–15] to provide in a single and self-contained document explaining the whole methodology. The contributions are as follows:

1. The description of a whole methodology for evaluating SRE based on a set of generic evaluation criteria that shall be refined for each organizations.
2. The application of this methodology to the IREHDO2 project context.
3. An evaluation of three SRE methodologies based on the feedback from security experts.

The rest of the article is structured as follows. In Section 2, we provide an overview of the existing comparative studies. Next, we present our proposed evaluation methodology and its strategy in Section 3. In Section 4, we present a deep study of the characteristics of good security requirements. This work provides a basis for eliciting the characteristics of a good SRE methodology. In Section 5, we describe a process and a tool to elicit SRE methodologies evaluation criteria. Each evaluation criterion must be associated with an evaluation metric. In Section 6, we describe the implementation of our evaluation methodology in the context of IREHDO2. We present our evaluation process to evaluate three SRE methodologies (i.e., STS [16], KAOS [17] and SEPP [18]). At the end, we highlight the pros and cons of each of these three SRE methodologies elicited characteristics serving as evaluation criteria. Finally, we conclude and present our future works in Section 7.

2. Related Works

In the literature, there exists numerous works related to comparative studies that offer varying dimensions of SRE evaluation perspectives. In below, we first present an overview of the related works and later we enlist the three main issues that we identified in their evaluation perspectives. Note that, our literature review does not provide the synthesis or classification of survey studies such as in [19,20].

N. Mayer [21] provided a comparative study of the SRE methodologies based on a domain model consisting of 14 security concepts. These concepts are categorized under three groups: asset-related (e.g., business asset, system asset), risk-related (e.g., risk, impact, threat, threat agent and attack method) and risk treatment related concepts (e.g., security requirements, control).

Fabian et al. [22] proposed a conceptual framework to support the comparative study of existing SRE methodologies. When compared with the modelling framework in [21], this work highlights similar concepts related to risk analysis but also extend them with granular security analysis concepts based on the security confidentiality, integrity and availability properties. In addition, it considers the concepts related to the multi-lateral security requirements analysis that stresses on the compromise between conflicting stakeholders' security needs [23].

Munante et al. [24] extended the previous comparison frameworks [21,22] with additional evaluation concepts related to model-driven engineering (e.g., tool support, SRE modelling language, formalism). This work concludes that KAOS [25] and secure i* [26], are the most compatible for formal SRE modelling.

Souag et al. [27] proposed a comparison framework to provide a systematic mapping of the reusable concepts and patterns within the existing SRE methodologies. Accordingly, research contributions were classified into 5 categories: security patterns, taxonomies and ontologies, templates and profiles, catalogues and generic models and miscellaneous.

Uznov et al. [28] proposed a comparative analysis of security engineering methodologies using a list of 12 characteristics of security methodologies. This work provides guidelines on the selection of methodologies upon their comprehensiveness, applicability and uniqueness from the perspective of their adaptability to an industrial use.

Mellado et al. [29] provided a systematic review of comparative studies to analyze the internal/external verification and documentation support of the SRE methodologies. The evaluation is principally based on the quality characteristics of good requirements (e.g., traceability, comprehensibility, etc.) referred from the internal standard IEEE 830 [30].

N. Mead [31] provided a comparative analysis of the requirements elicitation techniques based on some criteria such as learnability, client acceptance and durability of the requirements elicitation techniques, tools support etc.

Nhlabatsi et al. [32] proposed a comparative study of SRE methodologies in order to evaluate their support capability regarding the evolution of secure software during the change management process. Accordingly, the criteria address different perspectives such as the modularization, component architectures, change propagation and change impact analysis.

Niazi et al. [33] developed a Requirements Engineering Security Maturity Model to assist software development organizations to better specify the requirements for secure software development. They conducted a questionnaire survey based on security requirements practices derived from Sommerville's requirements engineering practices [34] to end up with 79 practices classified into 7 categories. This work has the same approach involving stakeholders. However, they focus on assessing the maturity level of security requirements practices, while we aim at helping organizations in choosing a security requirement engineering methodology.

Although these related works discussed many interesting strategies and aspects, they are not sufficient to evaluate the goodness of SRE methodologies because:

- Issue A: Evaluation criteria coverage. SRE methodologies aim at covering the whole SRE process (i.e., elicitation, evaluation and documentation) [22]. From our study, we noted that the majority of the evaluation studies except [29] lack a full emphasis on the whole SRE process [35]. While some works [21,22,31] concentrate on evaluating the extent of support to security requirements elicitation at earlier stages; some others [27] focus on evaluating the extent of support to documentation in terms of reusable patterns or modelling initiatives.
- Issue B: Evaluation criteria affirmation. The evaluation criteria must result from a systematic process or standard method, such as [36], to facilitate reusability. However, in the majority of works, the proposed evaluation criteria were ad hoc and lack affirmation on why the criteria were good enough to be considered for the evaluation [28,31,32]. In this aspect only [29] can be noted as an exceptional case as the evaluation is based on the international standard [30].
- Issue C: Lack of stakeholders' involvement. This issue concerns the involvement of stakeholders' views while formulating the evaluation criteria. As highlighted in [37], the requirements process is a human endeavour, and so the requirements method or tool must be able to support the need for stakeholders to communicate their ideas and obtain feedback. In our context, stakeholders refer to the SRE methodology users (e.g., requirements engineers). However, from our observation we found that none of these comparative studies involved SRE experts while formulating the evaluation criteria for an SRE methodology. Moreover, almost all of the works acknowledge (either implicitly/explicitly) the significance of involving stakeholders' perspectives in the SRE process. For instance, [22] includes a criterion that concerns the integration of stakeholders' views to support multilateral security requirements analysis.

Altogether, we can conclude that there exists no generic evaluation methodology that can help in identifying the best suitable SRE methodology. Evaluation of an SRE methodology is an enduring problem. As long as new SRE methodologies keep arriving, this necessity of evaluation persists. Having this in mind, we decided to develop a generic evaluation methodology independent from the SRE context of use so that it can be instantiated to varying SRE contexts (e.g., network security, software security).

3. Our SRE Evaluation Methodology

We propose an SRE evaluation methodology that differentiates its strategy from previous comparative studies for two reasons. First, we involve in the process the security experts who are the SRE end-users. Secondly, identifying SRE evaluation criteria corresponds to identifying the characteristics of a good SRE methodology. This is similar to conventional requirements engineering problems, which deal with identifying the requirements of a system-to-be. In our case, the system-to-be is the good SRE methodology, noted the SRE-methodology-to-be. The evaluation criteria are indeed the requirements of the SRE-methodology-to-be, labelled as R^M.

Based on this idea, we propose an evaluation methodology built on the classical idea of requirements engineering approaches. It facilitates the elicitation of the evaluation criteria of the SRE-methodology-to-be considering the SRE context of the stakeholders. Figure 1 depicts an overview of our approach. Similar to the RE process, it subsumes three

steps: (1) identifying the problem context and eliciting the initial high-level characteristic goal. This is performed by coupling the stakeholder's working SRE context with the quality criteria of good security requirements; (2) refining the high-level characteristic goals into the final requirements of the SRE methodology-to-be (R^M); (3) the final step deals with the evaluation of the existing SRE methodologies using the elicited requirements (R^M) from step 2.

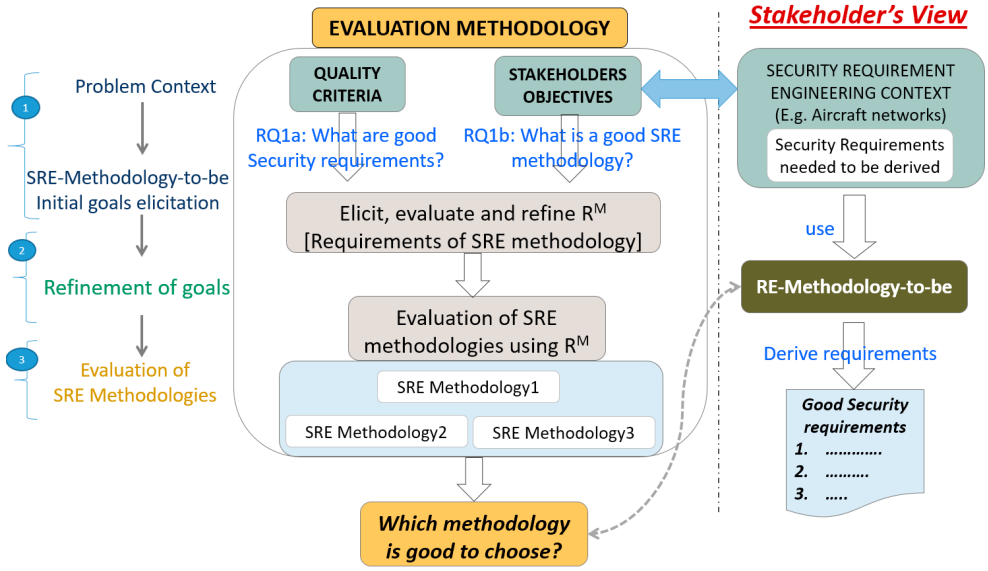

Figure 1. Our evaluation methodology.

The first step concerns the actual RE context of the system-to-be in which the requirements engineers intend to use the SRE-Methodology-to-be. The SRE methodology users represent the security experts who are considered as stakeholders in our approach. This involvement of security experts is mandatory to improve effective communication among them. Furthermore, works [17,38] consider the lack of people involvement to be one of the major requirement issues.

Ideally, the ultimate goal of the security experts is to derive good security requirements regardless of the SRE context [29,39]. The SRE methodology is a way to achieve this goal. Therefore, the elicitation of the initial characteristics features a brainstorming session driven by further refined two research questions: What are good security requirements? And what is a good security requirement engineering methodology?

The first sub-question helps to setup a common understanding of the characteristics of good quality security requirements. This study is independent from the SRE context and considers the evaluation criteria coverage issue (issue A). This understanding will eventually drive the second sub-question, which helps to capture common perspectives of stakeholders' anticipations over an ideal SRE methodology-to-be in their SRE context and handle the stakeholders' involvement issue (issue C).

Our evaluation methodology follows a pure goal-based approach for eliciting the requirements (R^M) of the SRE-Methodology-to-be. We propose to use the goal modelling notation KAOS [17] to represent the goals refinement hierarchy, since it facilitates the traceability of the refined goals. From a goal-based RE approach, the root goals represent the characteristics of good security requirements. They are refined into sub-goals that ultimately represent the anticipated characteristics of the SRE-methodology-to-be, which are eventually considered as the evaluation criteria. Thus, this goal-based refinement process helps in affirming the formulation of evaluation criteria formally, which responds to the issue B.

4. How to Qualify Good Security Requirements?

The common goal of any research contribution in the SRE community is ultimately to facilitate capturing good security requirements [29,39–41]. If the security requirements are error-prone (i.e., ambiguous, incorrect, inconsistent) then the security design will be ineffective regardless of the successful implementation of efficient security solutions [42]. For instance, let's take as an example the security requirement that states: "The data flow between device1 and device2 must be encrypted by a strong encryption algorithm". This requirement is ambiguous, because it does not explicitly define what a strong encryption algorithm means. This ambiguity may lead to false assumptions concerning the rigor of security protection, which eventually impacts the decisions related to the secure solutions.

The RE literature features numerous works describing different characteristics of good security requirements (a.k.a. quality criteria) such as unambiguous, traceable, consistent, etc. These characteristics help to verify and validate the quality aspects (i.e., the goodness) of security requirements. Nonetheless, there exists no consensus on how to characterize a good security requirement. In addition, there is no one complete and consistent set of characteristics [43].

In this context, we studied 185 definitions from 17 literature works depicted in Table 1. Different sources have listed different sets of criteria defining different characteristics of good requirements. The objective of this survey was to identify and study all the concepts discussed in each of the definitions, thereby analyzing the possibility to reach a consensus.

Table 1. Quality Criteria—consolidated sources list.

Source Identifier	Source Name	Number of Criteria Proposed in the Source
1	ISO 29148:2011 [35]	13
2	Van Lamsweerde 2009 [17]	11
3	Firesmith 2003 [39]	15
4	Wiegers 1999 [44]	10
5	Wieringa 1996 [45]	7
6	Boehm 1984 [46]	5
7	Pfleeger and Atlee 1998 [47]	8
8	IEEE 830 1998 [30]	9
9	Davis et al. 1993 [48]	20
10	Mar 1994 [43]	13
11	Sommerville and Sawyer 1997 [34]	7
12	Young 2004 [49]	15
13	Hull et al. 2010 [50]	15
14	Kar and Bailey 1996 [51]	10
15	Zielczynski 2008 [52]	13
16	Mannion and Keepence 1995 [53]	5
17	IEEE 12333:1998 [54]	9
	Total	185

Since, the list being huge and the definitions complex, we developed a semantic graph-based analysis tool using the python graph module NetworkX [55]. This allows us to formalize the definition and analyze the similarities or differences between the definitions. The nodes in the graph represent the criterion names and the concepts concerning the respective definitions. The edges correspond to the reference links expressing the semantic relation between a criterion and its associated concepts or between some criteria. For labelling the reference links (edges), we used a standard format by associating the abbreviation of the criteria with the source ID of the respective authors in Table 1 that propose the criterion. This ensures the unique interpretation of the reference links in the graph. For instance, the edge labelled M8 from node "modifiability" and node "express each requirements separately" means IEEE 830 1998 [30] (Source ID 8) refers to expressing each requirements separately in the definition of modifiability.

Figure 2 shows the complete view of the resulting semantic graph generated using Gephi tool [56]. The first result of this analysis produced a complex graph containing 212 nodes and 278 edges. The graph projects a total of 71 criterion nodes and 141 concept nodes. Every node is connected in the graph, highlighting the complexity of the semantic dependencies between the quality criteria. Such graph exhibits similarities between criteria. Indeed, when two criteria C1 and C2 refer, respectively, to a set of concepts D1 and D2 in their definitions such that D1 $\subseteq$ D2, thus we can conclude that criterion C2 is more generic than criterion C1. In addition, the semantic graph showed ambiguities in some definitions that make self-references. Figure 3 depicts an example of ambiguous definitions in which the definitions precise (PS), concise (CS) and clear (CL) from the authors result in a cyclic referencing. This opens a new problem that might require a proper research to analyze the semantic meaning of each criterion definition.

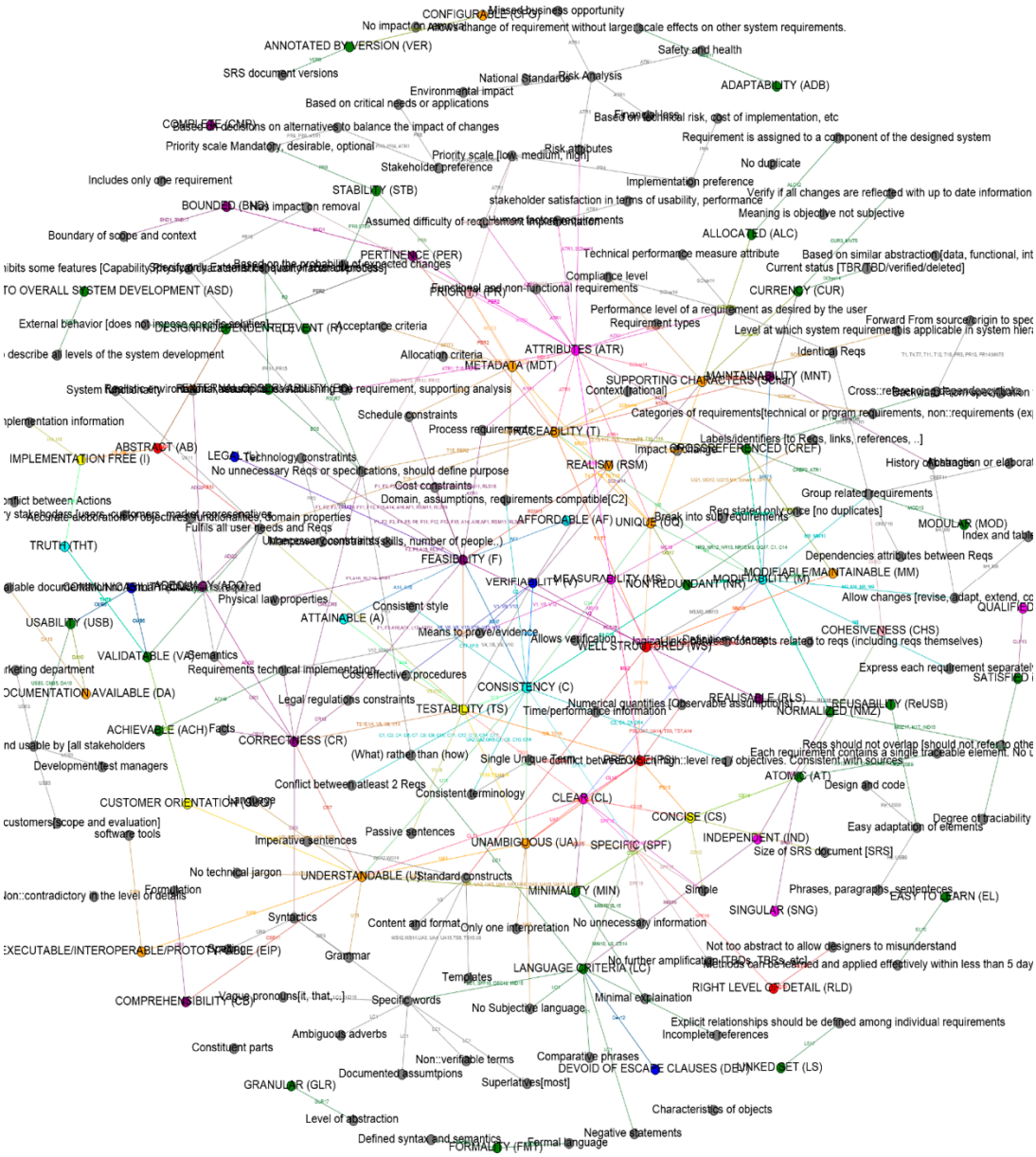

Figure 2. Complete view of the criteria conceptual graph.

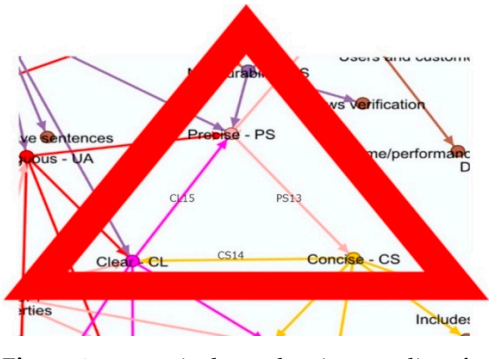

Figure 3. Semantic dependencies—cyclic references.

In our evaluation methodology context, our objective is defining an exhaustive list of the existing characteristics that can be integrated within any SRE process. Suitably, we needed something simpler that can be used as a standard basis for eliciting the characteristics of a good SRE methodology. The semantic graph helped us in identifying a total of 20 distinctive criteria definitions. However, we have observed many variations in their corresponding definitions (e.g., different authors, for similar criteria, have defined different names). This entailed into defining a weaving methodology to express these variations.

The main goal of our weaving methodology is to showcase the similarities in the criteria propositions of the unified list in a tabular format that featured a weaving strategy. Table 2 consolidates our weaving results, which extends the results in [12]. Firstly, we gave a unique identifier as a reference to each class of quality characteristics. We denote the term criterion (represented as C) that gives us a list of criteria as C1 to C20 (see Table 2). In addition, we added a one-line definition to each criterion. If the characteristic is defined in ISO 29148, we use the standard definition. Otherwise, we take references from respective authors if the characteristic description is straight forward (e.g., C8). When the characteristic descriptions seemed ambiguous (e.g., C10), we provide our own interpretation.

Secondly, we used colors (i.e., blue and orange) and typographical emphasis (i.e., bold, underlined and italic) to differentiate some special cases as follows: Criterion definitions are highlighted in blue color when defined by only a single author (e.g., C13). When different authors employ different names to describe similar quality characteristics (e.g., C3), then the respective criterion definition is emphasized in italic text. When a single author employs different names to describe similar quality characteristics (e.g., C3 by S13), then the respective criterion definition is emphasized in orange. Contrarily, when the same criterion name is used to describe different quality characteristics (same or different authors), then the characteristic name reflecting the uncommon mapping is underlined (e.g., C6 by S5 and C15 by S15).

Thirdly, we distinguish the applicability of a criterion. If mentioned *All* the Applicability column in Table 2, the respective quality characteristic requires to verify the whole set of requirements specifications document (e.g., C2). If mentioned *Each*, the respective quality characteristic targets only a single (or a specific set of) requirement(s) (e.g., C20). See the Applicability column in Table 2. Finally, we defined credibility scores to show the frequency of citations. We used qualitative scale high, medium and low. Credibility is given *high* when cited by at least 15 authors; *medium* with at least 8 authors; and *low* when at least 4 authors; *very low* when cited by less than 4 authors.

Altogether, Table 2 overviews our comprehensive review of the non-consensus issue concerning the characteristics definitions.

Table 2. Characteristics of good security requirements.

No	Abstract Criterion Definition	Characteristics of Good Security Requirements																		
		S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17		
		ISO29148	Lamsweerde	Firesmith	KE Wiegers	Wieringa	Boehm	Pfleeger and Atlee	IEEE830	Davis	Brain Mar	Sommerville	R R Young	Hull et al.	Karl et bailey	Peter	Mannion	IEEE 12333	Credibility	Applicability
C1	Should include all the needs of all the stakeholders	Complete	Complete	Complete	Complete	Complete	Complete	Complete	Complete	Complete	Complete	Complete	Complete	Complete	Complete	Complete	–	Complete	High	All
C2	Compatible, non-contradictory requirements	Consistent	Consistent	Consistent	Consistent	Consistent	Consistent	Consistent	Consistent	Consistent	Consistent	Consistent	Consistent	Consistent	Consistent	Consistent	–	Consistent	High	All
C3	*Accomplishable within given financial, time, legal, technological constraints*	Feasible/ Affordable	Feasible	Feasible	Feasible	Feasible	Feasible	Feasible	–	Achievable	Feasible	Realism	feasible	feasible/legal	Attainable	feasible/ realistic/ possible	Attainable/ realisable	–	High	All
C4	*Requirements must be well documented*	–	Good structuring	–	–	–	–	–	–	organized	–	–	–	–	–	–	–	configurable	low	All
C5	*Requirements should be able to refer back to its objective. Dependency or reference links between requirements should be explicity defined.*	Traceable	Traceable	Cohesiveness	Traceable	Maintainable/ traceable	Traceable	Traceable	Traceable	Traceable/ cross-referenced	Traceable	Traceable	Traceable/ Allocated	Satisfied/ Qualified/ Modular	–	–	Traceable	Traceable/ Linked set	High	All
C6	*Requirements should state what is needed but not how they are met*	Implementation Free	–	External Observ-ability	–	Truth/ Implementaion independence/ Validity	–	–	–	Design in-dependent	Minimality/ Right level of detail	–	Design In-dependent	Abstract	Implemen-tation Free	independent/ Implemen-tation Free	specific/ appropri-ate level of detail	Abstract/ Normal-ized/ granular	Medium	All
C7	*Documented requirements must be easily adaptable to new changes*	–	Modifiable	–	Modifiable	Maintainable/ Modifiable	–	–	Modifiable	Modifiable	Maintainable/ Modifiable	Adaptability	–	–	–	–	–	Modifiable	Medium	All
C8	No redundant requirements	–	–	–	–	–	–	–	–	Non-redundant	–	–	Non-redundant	Non-redundant	–	Non-redundant	–	–	low	All
C9	Every requirement is uniquely identified (i.e., number, name tag)	identification	–	Identification	–	–	–	–	–	–	–	–	Unique	Unique	Identification	–	–	Unique set	low	All
C10	*Stakeholders needs are sufficienly expressed*	–	Adequacy	Validatability	–	–	–	–	–	–	–	Validity	–	–	–	–	–	–	Medium	Each
C11	*Requirements defined are simple, using common terminology and non-technical jargon.*	–	Comprehensibility	Customer/ User Ori-entation/ Usability	–	Understandable	–	–	–	Understandable/ Concise	Simple	Comprehensibility	Concise/ simple	Clear/precise	Understandable/ minimal/ Concise	Concise/ simple/ precise	–	–	Medium	Each
C12	*Requirement statement must lead only one possible interpretation in common*	Unambiguous	Unambiguous	Lack of Ambiguity	Unambiguous	Unambiguous Communicability	–	Unambiguous	Unambiguous	Unambiguous	Unambiguous	–	Unambiguous	–	Unambiguous	Unambiguous	–	Unambiguous	High	Each
C13	Requirements defined allows evaluation - quantifiable values	–	Measurable	–	–	–	–	–	–	Precise	–	–	–	–	–	–	Measurable		low	Each
C14	*Every requirement has a purpose*	Necessary/ Bounded	Pertinence	Mandatory/ Relevance	Necessary	–	–	Relevant	–	–	Necessary	–	Necessary	–	Necessary	Necessary	–	Bounded	Medium	Each

Table 2. *Cont.*

No	Abstract Criterion Definition	Characteristics of Good Security Requirements																	
		S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	
		ISO29148	Lamsweerde	Firesmith	KE Wiegers	Wieringa	Boehm	Pfleeger and Atlee	IEEE830	Davis	Brain Mar	Sommerville	R R Young	Hull et al.	Karl et bailey	Peter	Mannion	IEEE 12333	Credibility	Applicability
C15	Requirement should accurately represent the facts and needs. Syntactically and semantically	–	–	Correctness/ Currency	Correct	–	–	Correct	Correct	Correct	Correct	–	Correct	–	–	Understandable/ Correct	–	–	Medium	Each
C16	*Non conjunctive requirements*	Singular	–	–		–	–	–	–	–		–	–	Atomic	–	Atomic	–	–	low	Each
C17	*Should define some means to prove the compliance or satisfaction of requirement with stakeholder needs, standards and constraints.*	Verifiable	–	Verifiability	Verifiable	Verifiable	Testable	Testable	Verifiable	Verifiable	Verifiable/Testable	Verifiability	Verifiable	Verifiable	Verifiable	Verifiable/Testable	–	Validatable	high	Each
C18	*Formulation of Requirement statements must follow specific criteria*	*Requirement language criteria/* Require-ments construct	–	–	–	–	–	–	–	–	Formality	–	Devoid of escape clauses/ Standard Construct	–	Standard construct	–	–	–	low	Each
C19	Requirements must be reusable by numerous stakeholders	–	–	–	–	–	–	–	–	reusable	–	–	–	–	–	–	–	–	low	Each
C20	*Individual requirements should be defined with some attributes or annotations that characterizes them (e.g., assumptions, rationale, risk realted information)*	Attributes	–	Metadata	Prioritized	–	–	–	Ranked for impor-tance/ degree of stability	Ranked for impor-tance/ Ranked for stability	–	–	–	–	Suporting characteris-tics	–	–	–	Medium	Each

Criterion C1 ensures that final set requirement specifications sufficiently express all the needs of stakeholders, respecting all the considerable aspects and scenarios. The difficulty in fulfilling this criterion corresponds to identification of all considerable aspects such as stakeholder security and risk management objectives. In a way this criterion insists on efficient requirements elicitation and risk analysis process. The common keyword used to represent this criterion is complete with credibility *high*. Applicability of this requirement implies either to an entire set of requirements or to a set of requirements.

Criterion C2 ensures that all requirements are compatible and consistent with one another. Accordingly, this criterion C2 insists on verifying if there exist any conflicts in terms of contradicting requirement statements, improper representation of viewpoints or possibility of incompatible interpretations of a statement, etc. The difficulty in fulfilling this criterion corresponds to establishment the right level of trade-off. This criterion indirectly contributes to satisfaction of requirement completeness. The common keyword used is consistent with credibility as *high*. Applicability of this criterion concerns with either an entire set of requirements or a group of requirements.

Criterion C3 ensures that all those derived requirements within the document are accomplishable within the given constrains. ISO defines some of the considerable constraints such as time, cost and process control, financial, technical, legal and regulatory. In addition, dependency constrains and domain constraints [17] can also be considered. Constraints can be viewed in two ways, one as they are imposed by stakeholders and the other based on operational context. On the whole, this criterion insists on identifying and acquiring all the possible constraints in terms of financial or technological implementations. Applicability of this criterion concerns with either an entire set of requirements or a group of requirements or an individual requirement. Credibility of this criterion is also *high*. The common keyword used is feasible, affordable and other keywords are realism and legal.

Criterion C4 ensures that all requirements within the document are well categorized and well documented in a structured manner so that it is maintainable with fewer changes. Credibility of this criterion is *low* and common keyword used is structured.

Criterion C4 ensures that all requirements within the document are prioritized and well documented in a structured manner. Credibility of this criterion is *low* and common keyword used is structured.

Criterion C5 ensures that specified within the document are traceable in both forward and backward ways. Credibility of this criterion is *high* and common keyword used is traceable. Some sources have highlighted different aspects in the same context; hence different keywords were used accordingly. The keywords are cohesiveness, allocated, satisfied/qualified.

Criterion C6 ensures requirements derived do not specify the implementation details of the solution instead it specifies what is needed. Credibility of this criterion is *medium* and the keywords used are implementation free, external observability, design independent and abstract.

Criterion C7 ensures that the document containing all set of derived requirements is modifiable and adaptable to changes. It is to note that this is similar to a Meta characteristic to criterion C4 (well structured). Credibility of this criterion is *medium* and the keywords used are modifiable, adaptability.

Criterion C8 ensures that there is no redundancy of information corresponding requirement needs. It insists during the requirements elicitation process, one must clearly be able to distinguish between redundant stakeholder needs and non-redundant stakeholder needs. Credibility of this criterion is *low* and the common keyword used is non-redundant.

Criterion 9 ensures that all the requirements in the document are uniquely identifiable. This criterion helps to achieve the traceability feature (C5). Credibility of this criterion is *low* and the common keyword used is unique.

Criterion C10 ensures that completeness feature of an individual requirement. In a way it insists on verifying if the stakeholder need is sufficiently elicited. Credibility of this criterion is *low* and the keywords used are adequacy and validity.

Criterion C11 ensures that requirements are derived using simple terminology without usage of technical jargon. Technical jargon corresponds to terminology used by different teams working in different areas of business operational environments. For example, terminology used in software development environment is difficult to be understood by individuals belonging to organizational environment. Hence, this criterion enforces that the derived requirement must be comprehensible to all the readers of the document with in the business environment. Credibility of this criterion is *medium* and the common keyword used is comprehensibility. Some sources have highlighted different aspects in the same context; hence accordingly different keywords used. They are customer or user orientation and clear.

Criterion C12 ensures that the derived requirements are precise enough and does not lead to any misinterpretations. It is to note that this criterion is different from the previous one C11 (comprehensibility). C11 insists on the aspect that there is no difficulty in the comprehension of the text (phrase or sentence), in the way it was written (focus on terminology). In addition, C12 insists on the aspect that the content of the text maintains careful precision while expressing the idea so that it does not lead to misinterpretation of the idea. In end, this criterion emphasizes on the verification that comprehension of the text is not wrong. It focuses on punctuation and meaning of terminology or vocabulary used. In a way, this criterion can be viewed as a meta-characteristic of the criterion C11. Credibility of this criterion is *high* and the common keyword used is unambiguous. Another keyword used is precise.

Criterion C13, it ensures that requirements derived can be measured with some quantifiable values. For example, consider a requirement need "a service must be available to all the customers". This need cannot be measured and while eliciting such needs, it is important to elicit measurable information. For this derived requirement for this need can say "a service must be available on an average to 'x' number of customers at 't' units of time". This way, the requirements can be measured. Credibility of this criterion is *low* and the common keyword used is measurable.

Criterion C14 ensures that the derived requirement specifies what is needed and it has not got any unnecessary information. It is to note that this criterion complements the criterion C10 (adequacy). Credibility of this criterion is *medium* and the common keywords used are necessary and mandatory. Some sources have highlighted different aspects in the same context; hence accordingly different keywords used. They are bounded, pertinence and relevance.

Criterion C15 ensures that requirement must possess accurate and up to date information. Credibility of this criterion is *medium* and the common keyword used is correct.

Criterion C16 ensures that one requirement derives one need. For example, if a requirement need says "entrance to aircraft allowed to customers with boarding pass and special emergency pass". This is not singular or atomic in nature. It is speaking allowing customers of two different types. One can split this into two as: "entrance to aircraft allowed to customers with boarding pass" and "entrance to aircraft allowed to customer's special emergency pass". This way it helps to defined more precisely what does it mean by saying special or emergency. Accordingly, we can say that this criterion C16 contributes towards C13 (measured). Credibility of this criterion is *low* and the keywords used are singular and atomic.

Criterion C17, it ensures that each of the requirements is verifiable against the constraints, standards and regulations to ensure the correctness of the requirements. This criterion somewhere again falls between C10 (adequacy) and C15 (Accurate). Credibility of this criterion is *high* and the common keyword used is verifiability.

Criterion C18, it ensures the formulation of requirement must follow some standard so that they are understandable globally. Credibility of this criterion is *low* and the common keywords used are requirement language criteria and devoid of escape clauses.

Criterion C19, it ensures that requirements must be formulated in such a way that they are reusable. This criterion emphasizes on the using some common pattern for similar

type of requirement needs. Credibility of this criterion is rare and the common keyword is usability.

Criterion C20, it ensures that every requirement should be identified with some metadata such as attributes, acceptance criteria. This way, it facilitates in their validation and evaluation. Credibility of this criterion is rare and the keyword used is Metadata.

Since the interpretation of a definition can differ between different people with varying expertise levels, the table contents can raise several questions. This implies that our weaving methodology strategy provides a way to trigger a discussion between experts. This will lead to brainstorming, which serves our purpose of initiating the elicitation process of our RE based evaluation methodology.

5. What Is a Good SRE Methodology?

Since this question depends on the SRE context, we describe the instantiation of our evaluation methodology to the network SRE context from the IREHDO2 project. The first two steps concern the elicitation of the characteristics of a good SRE methodology from the network SRE context. While the final step concerns the evaluation of the SRE methodologies with regards to the elicited characteristics.

5.1. Step1: Problem Context and Initial Requirement Analysis

The initial step of our approach consists in analysing the security problem context of the security experts. Accordingly, this step deals with interviewing the people involved in the security engineering process. We used the brainstorming technique to encourage people to exchange ideas on "the best suitable SRE methodology" that best fits their needs. However, eliciting requirements is a hard task [31]. The challenge here comes while organizing the brainstormed thoughts and ideas in a structured manner. For this, meetings/brainstorming with stakeholders must be controlled in order to be effective.

Therefore we employed the elicitation technique proposed by SABSA [9]. The SABSA framework handles this elicitation issue by proposing a list of generic high-level business security concerns, called business attributes. These business attributes might lead to several interpretations. Interpretation of business attributes is refined for a specific problem context by a security architect who interacts with the business stakeholders. These business attributes guide the interaction during the elicitation phase. Respectively, in our case, our list of 20 quality criteria characterizing good security requirements given in Table 2 are similar to business attributes in SABSA. Based on this list, we developed an elicitation tool to trigger the discussions, see Table 3.

The first three columns consolidate Table 2. They subsume, a unique identifier, a short one-line definition and corresponding synonyms found in the literature. The last column describes the quality criteria via a set of questions, each reflecting different perspectives of the respective criterion definitions. We also provided the references from which we extracted the sample set of questions. Altogether, each of the four columns corresponds to a different way to explain each criterion in order to facilitate the understanding as well as to ignite brainstorming. This process aims at refining the generic and abstract criteria into specific and context-dependent SRE evaluation criteria that reflect the actual needs of the stakeholders involved in the SRE process.

5.2. Step2: Refinement (R^M) Requirement Analysis

The second step of our evaluation approach deals with understanding the transformation of high-level abstract characteristic goals into verifiable evaluation criteria. Respectively, the high-level abstract goals realized in step 1 are coarse-grained and should be refined into sub-goals fitting specific demands of the security experts. This refinement process is performed in collaboration with the security experts. Figure 4 provides a sample view of the criteria refinement using KAOS goal modelling notation. This goal refinement extends the study [14,15].

Table 3. Elicitation tool (sample).

No	Abstract Definitions	Criterion Names in Use	Questionnaire
C3	Accomplishable within given financial, time, legal, technical constraints	Feasible, Affordable, Legal, Achievable	► Is it possible to capture the constraints of security requirements? such as technical? Legal? Time? Financial? [35] ► Does the SRE methodology require training? is it within the project budget? [39] ► Is it possible to learn the SRE methodology notation within the project timelines? [31]
C5	Requirement should be able to refer back to its objective. Dependency or reference links between requirements should be explicitly defined.	Traceable, Cohesiveness, Allocated, Satisfied/Qualified	► Is it possible to trace the security requirements back to the business needs and vice-versa? [35] ► Is it possible to trace the group of similar requirements e.g., related to a particular abstraction level? [39]
C6	Requirements should state what is needed but not how it is met	Abstract, Design independent, External observability, Implementation free, Right level of detail, Minimality	► Is it possible to express security requirements without constraining the design solutions? [35] ► Is it possible to respect the abstraction needs of the stakeholders? [43]
C10	Stakeholders needs are sufficiently expressed	Adequacy, Validability	► Is it possible to capture the individual needs of all the stakeholders involved in the SRE process? [39] ► Is it possible to express the domain environment assumptions and domain constraints for implementing the security requirements? [17]
C11	Requirements are simple using common terminology and non-technical jargon.	Clear, Concise, Comprehensibility, Customer/User Orientation, Communicability	► Is the SRE modelling notation comprehensible to all the stakeholders involved in the SRE process? [39] ► Is the formulation of security requirements comprehensible to the stakeholders? [17] ► Does it facilitate easy communication of the ideas among the stakeholders involved at different abstraction levels of the SRE process? [45]
C20	Individual requirements should be defined with some attributes or annotations that characterizes them	Attributes, Metadata, Prioritized, Ranked for importance, Degree of stability	► Is it possible to express implementation costs of the security requirements? [35] ► Is it possible to express the risk attributes pertinent to the security requirements? [35] ► Is it possible to express the priority of the security requirements based on risk impact? [35]

The refinement uses the AND-construct and is continued until the final refined goals are realized as objectively verifiable. Thus, the leaf goal nodes become an evaluation criteria R^M. Tracing the goals refinement conforms the correctness of the final evaluation criteria R^M. Since, we used natural language to describe the criteria and goal modelling notation to express the refinement, the affirmation of evaluation criteria is self-contained in the model. The high-level goals are the quality criteria from our elicitation tool (goals in orange). The elicited interpretations are the immediate sub-goals (in green). It is to note that different colouring of goals is provided solely to facilitate the understanding of the readers and this colouring is not compliant with KAOS notation.

In below, we present our arguments relative to the refinement of six quality criteria given in Table 3. It is to note that our arguments are based on our discussions and initial feedback with the security experts involved in the brainstorming process. The goal refinement is subjected to changes with varying needs of the stakeholders.

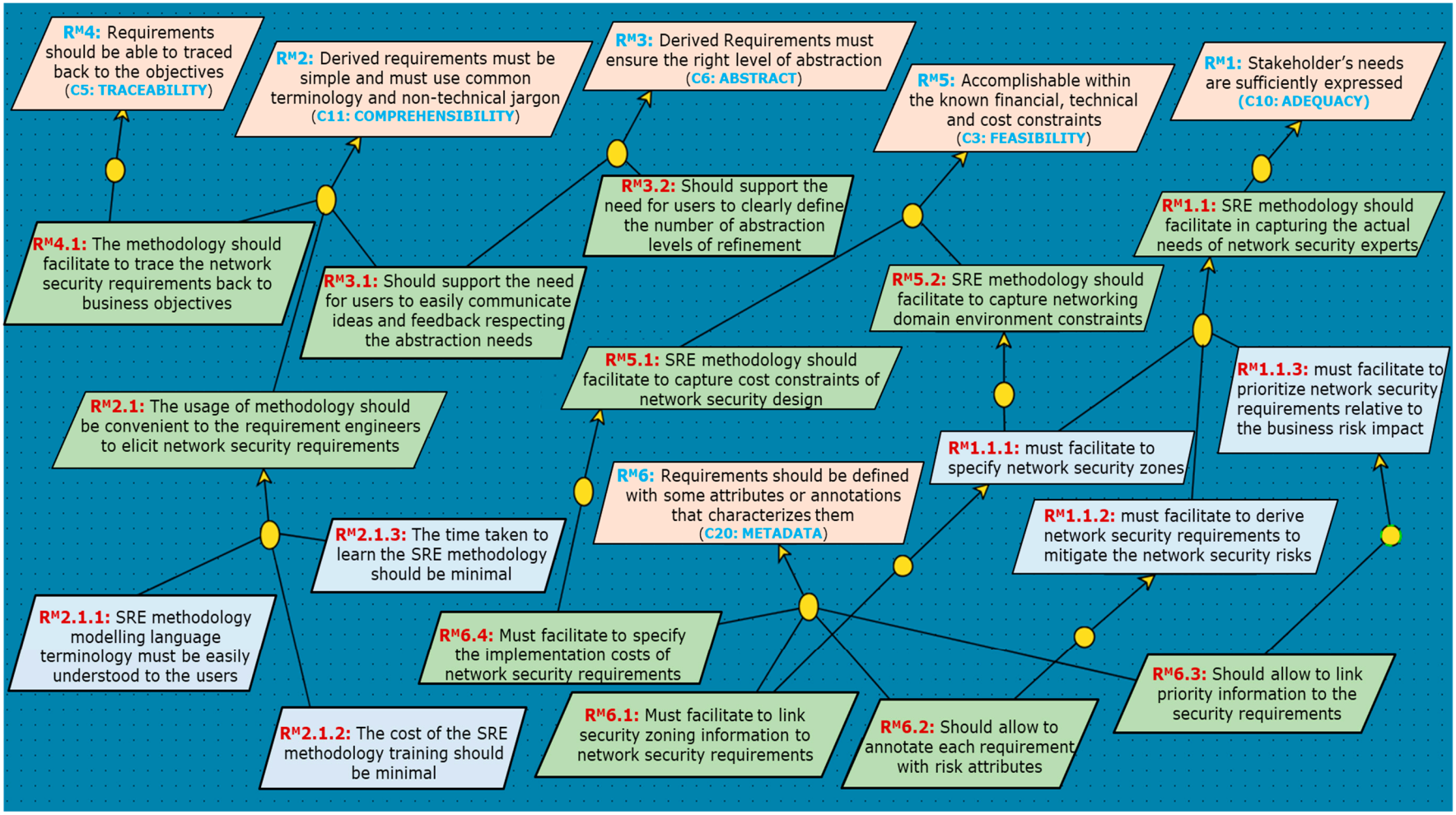

Figure 4. SRE-methodology-to-be goals refinement (Sample).

Adequacy (C10) and Metadata (C20)

The Adequacy criterion stresses that the actual needs of all the stakeholders must be captured. In our project, we refer to the security experts working in network SRE context. We refine the high-level quality goal pertaining to the adequacy quality criterion (Cf. C10 in Figure 4) into "R^M1.1: The SRE-Methodology-to-be should facilitate in capturing the actual needs of network security experts".

Since network security zoning drives the elicitation of network security requirements at early stages, we have refined this goal into three sub-goals: R^M1.1.1 and R^M1.1.2 to validate that the SRE-Methodology-to-be facilitates the specification of security zones and derived network security requirements to mitigate risks. However, in order to determine the rigor of security/validation measures required at each security zone, the security experts must also need to be able to prioritize the elicited security requirements based on the risk impact. This requisite of risk analysts is derived into R^M1.1.3, to validate if the features allow the prioritization of network security requirements.

However, to ensure the verification these information are captured [35], describes risk related information and environment constraints to be linked as requirement attributes/metadata (see C20 in Table 2). This confirms the refinement of sub-goals R^M6.1, R^M6.2, R^M6.3 from the quality criterion C20 (R^M6) in compliance with the aforementioned sub-goals (i.e., R^M1.1.1, R^M1.1.2 and R^M1.1.3).

Comprehensibility (C11)

The quality criterion comprehensibility (C11) has been refined into "R^M2.1: The methodology usage should be convenient to the requirement engineers to elicit network security requirements". This concern drives the main motivation for all of the model-based RE approaches because a security requirement not understood cannot be analyzed or implemented properly.

In this regard, the refined sub-goal R^M2.1.1 refers to verify the ease of use of the language of the SRE-Methodology-to-be itself. That means, which language is understandable to the users? A formal language? UML? Or some natural language? This aspect completely depends on the language familiarity of the stakeholders who are going to use the SRE-Methodology-to-be. If they are familiar with formal notations then using formal languages is better. If they are familiar with UML, then it is better to choose the requirement engineering methodology accordingly. Ref. [37] argues that verifying the suitability for agreement with the end-user indicates the extent to which the notation is understandable (as opposed to 'writeable') by someone without formal training. Furthermore, different stakeholders, despite their familiarity, can understand the same requirements differently, which requires negotiation schemes to calculate an agreement [57]. In addition, learning and mastering a new language has a cost in terms of both money and time [28,31]. For example, a five day training course for SABSA framework costs around 3000 € [58]. In this work, we do not consider these negotiation aspects. Instead, we confine our evaluation study to understanding the notation, learnability duration and ease of SRE modelling. This perspective is covered by the sub-goals R^M2.1.1, R^M2.1.2 and R^M2.1.3

Abstract (C6)

In practice, security requirements evolve gradually building upon the ideas/perspectives of the subject experts (e.g., business analysists, security architects and network security engineers) who work at different levels of abstraction in a network development cycle. This requires an effective communication among the subject experts and the requirement engineers. Therefore, the SRE-Methodology-to-be modelling language should support the needs of the people to easily communicate the ideas and feedback relative to the security requirements elicitation [37,45].

On the other hand, with reference to the definition of the implementation free criterion from [35], the SRE-Methodology-to-be has to ensure that the stated security requirements express only the 'WHAT' aspects of security needs and must avoid the 'HOW' aspects of

describing the security solutions. This is achieved by accommodating the separation of concerns so that the readers of the requirements specification should need to find only those parts of the requirements specification that are relevant to their area of expertise and all the other details must be hidden.

Respectively, the refinement of quality criterion abstract (C6) includes two sub-goals: $R^M3.1$ to validate that the SRE-Methodology-to-be facilitates the communication between the stakeholders and $R^M3.2$ to guarantee that the SRE-Methodology-to-be allows the specification of a number of abstraction levels for expressing the separation of concerns.

In some cases, it is possible that a sub-goal can be refined from multiple high-level goals; e.g., $R^M3.1$ sub-goal which states "the derived requirements must respect the abstraction level of respective stakeholders involved in designing and building aircraft network systems". This sub-goal was initially refined from the root goal node R^M3 related to the abstract characteristic criterion (see Figure 4). However, we discovered it can also be refined from R^M2 concerning comprehensibility characteristic criterion with a justification stating that the requirements not respecting the abstraction requirements of the stakeholders are not comprehensible. This type of refinement patterns explains the semantic dependencies between the quality characteristics. It also explicitly reflects the merging of the different security experts' points of views.

Traceability (C3) and Feasibility (C5)

Finally, goals $R^M4.1$, $R^M5.1$ and $R^M6.4$ concern the supportability of the SRE methodology with reference to traceability and feasibility characteristics (i.e., C3 and C5 in Table 2) in network SRE context. Traceability is defined as the ability to establish the link between requirements to the source business objectives [35]. These requirements express the capability to verify which high-level network security requirement is impacted when any inconsistency or anomaly is found in network security and monitoring configurations. Feasibility characteristics, on the other hand, concern the ability to verify if the security requirements are realizable within the budget schedule and technology constraints [35]. These characteristics were one of the primary objectives of IREHDO2 project.

The refinement process is performed until the final refined goals are realized as objectively verifiable. The final sub-goals (leaf nodes) are then qualified as evaluation criteria.

Table 4 provides a sample of the final list of elicited evaluation criteria in our network SRE context within the IREHDO2 project.

Table 4. Elicited evaluation criteria (Sample).

$R^M2.1.1$: The SRE-Methodology-to-be modelling language terminology must be easily understood to the user
$R^M2.1.2$: The cost of the SRE-Methodology-to-be training must be minimal
$R^M2.1.3$: The time taken to learn the SRE-Methodology-to-be must be minimal
$R^M3.1$: The SRE-Methodology-to-be must support the need for users to easily communicate ideas and feedback respecting the abstraction needs
$R^M3.2$: The SRE-Methodology-to-be must support the need for users to clearly define the number of abstraction levels of refinement
$R^M4.1$: The SRE-Methodology-to-be must facilitate the tracing of the network security requirements back to business objectives
$R^M6.1$: The SRE-Methodology-to-be must facilitate the specification of network security zone requirements
$R^M6.2$: The SRE-Methodology-to-be must allow annotating each requirement with risk attributes
$R^M6.3$: The SRE-Methodology-to-be must allow annotating each requirement with priority information
$R^M6.4$: The SRE-Methodology-to-be must facilitate the specification of the implementation costs of network security requirements

Finally, our approach which involves the stakeholders has shown its benefits. We elicited new evaluation criteria (namely $R^M3.2$, $R^M6.1$ and $R^M6.4$) that were not proposed by previous comparison/evaluation studies.

Evaluation Methods and Metrics

An evaluation method and metric shall be associated to each evaluation criterion. The type of evaluation and respected performance metrics can differ depending on to the type of evaluation criteria. Let's consider evaluation criterion $R^M6.2$. Linking risk attributes to security requirements confirms the integration of risk analysis process to security requirement analysis. However, an evaluation metric shall define to what extent SRE methods can support this integration (Cf. Table 5).

Table 5. Verification method for $R^M6.2$.

$R^M6.2$: The SRE-Methodology-to-Be Must Allow Annotating Each Requirement with Risk Attributes	**Performance Measure**
The annotation feature is extensible. Requirements can be linked with risk attributes, i.e., asset criticality, risk event, risk likelihood, control strength)	high
At least two risk attributes, i.e., risk events and risk likelihood	medium
At least one risk attribute, i.e., risk events	low
Requirement cannot be annotated with any kind risk information	nil

The qualitative scale used for the performance measure expresses the degree of supportability, i.e., high—highly supportable, medium—partially supportable, low—less likely supportable and nil—not supportable. Wherein the colouring is used to enhance the understanding.

The type of measurements (i.e., nominal/ordinal/interval/ratio [59]) also vary with regards to the type of criteria. Nominal scaling uses some labels that don't have order to differentiate between different subjects (e.g., male/female). Ordinal scaling uses rank ordering to sort the subjects (e.g., good/neutral/poor or small/big). Finally, interval and ratio scales use some quantitative numeric values, which allows the estimation of the degree of difference between the subjects in terms of some meaningful measurement units (e.g., length, duration, angles etc.). While interval scaling describes the exact and equidistant difference between the units (e.g., number of hours or number of days) allowing statistical analysis, it does not have true zero which means nothingness. A ratio scale is the most informative as it allows ranking and differences between variables, along with the information on the value of a true zero.

For instance, the measurement scale for $R^M6.2$ reflects the ordinal scale, which verifies if each security requirement can be linked to risk attributes such as risk likelihood, control strength, threat events and risk impact with reference to FAIR taxonomy (https://www.fairinstitute.org/what-is-fair---accessed 7 August 2021). Other evaluation criteria may require different verification approach. For instance, $R^M2.1.1$ concerns comprehensibility aspects of the terminology used by SRE-methodology. The verification metric employed for this criterion can refer to the satisfaction survey of the security experts. Respectively, $R^M2.1.1$ uses ordinal scale, which sorts the understandability factor of SRE methods based on the degree of support required; $R^M2.1.3$ uses interval scale, which allows the estimation of the degree of learnability of the terminology used by the SRE-methodology-to-be in terms of number of weeks. Finally, evaluation criterion $R^M2.1.2$, related to the cost of the training -ratio scale-, requires the security experts to have knowledge on the internal budget. At the end of step 2, the verification methods and the associated metrics for each evaluation criterion is agreed (e.g., internal budget, time constraints) in accordance with the specific needs of the stakeholders (e.g., security experts) involved in the process.

Likewise, each evaluation criterion is accorded with a verification method, which will be used during the evaluation process. It is to note that these verification methods and measured metrics must be defined in compliance with the needs of the security experts who intend to use SRE methodology-to-be. This clear separation of evaluation criteria to the verification methods and respective measurement metrics permits the reusability

of the evaluation criteria as well as simplifies justifying the subjectivity of the criterion interpretation and relative evaluation perspectives.

6. Evaluation of Existing SRE Approaches

This section describes the final step of our SRE evaluation methodology. It concerns the assessment of the SRE approaches against the evaluation criteria refined in step2. Our evaluation is a qualitative assessment that is performed in collaboration with the security experts. We organized two meeting sessions (i.e., precisely 3 h brainstorming and discussions for each) with a time gap of around 2 months.

We performed the evaluation in two iterations. In the first iteration, we—the authors of this article—acted as stakeholders (i.e., SRE analysts) and tested three SRE approaches, i.e., STS, Secure KAOS and SEPP using an example scenario. First, a description of the system-to-be is presented to three different persons (playing the roles of RE analysts) from our research group whose initial knowledge fits the aforementioned methodologies the best. Then, each person has been asked to elaborate the requirements for the system-to-be using the methodology that was assigned to him/her. During this process, the persons (acting as SRE analysts) were not allowed to communicate with each other during the scenario analysis phase. Each of them had come up with a different list of security requirements for the system-to-be. The resulting SRE models have been presented during a meeting that involved four security experts. Their feedback was recorded as our initial evaluation results, which enabled us to identify and select the SRE approach found interesting to the experts at first instance.

In the second iteration, the actual security experts are directly asked to test the SRE approach selected during the first iteration. The feedback of this second iteration enabled us to finalize the evaluation results. In each iteration a different use case scenario is employed in order to ensure the credibility of the capture of the evaluation experience. In below, we provide the details of the evaluation performed in two iterations.

6.1. Evaluation Iteration 1—Use Case Scenario 1

We first present the use case scenario employed in the first iteration. The scenario concerns a task related to the aircraft maintenance process ensuring compliance to some safety regulations. The objective of this task is to monitor the on-board aircraft system by verifying specific parameters (e.g., fuel indicator) in order to ensure the readiness of the aircraft prior to taking the next trip. The Onboard aircraft system consists in two applications, i.e., aircraft control application and monitoring application (Cf. Figure 5). These two applications are connected to each other via an internal avionic bus network (represented by a red double arrow line in Figure 5).

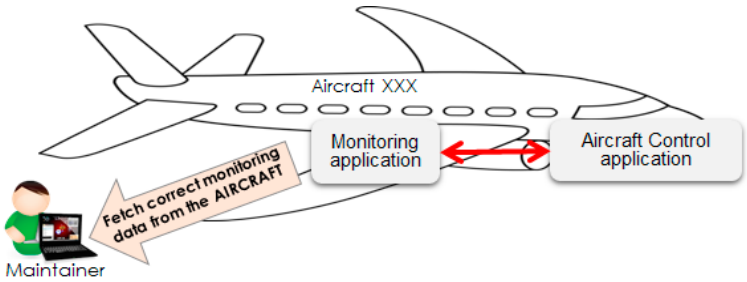

Figure 5. Example scenario 1.

The aircraft control application captures the observed parameters from various sensors and indicators of the aircraft system and transmits them to the monitoring application. Every time the aircraft has landed, the responsible person (i.e., maintainer) must connect his/her laptop to the monitoring application in order to fetch the monitored parameters. However, there is no direct access to the monitoring application. The connection is permit-

ted only using the secured network connection within the secured premises of the aircraft. The maintainer must carry the laptop to the airport ground in order to access the network. In this process, the maintainer is assumed to be trusted, while the laptop is untrusted.

The business risk impact of this scenario is expressed in terms of integrity and availability of the monitored parameters. These parameters influence the critical decision process that decides whether the aircraft is ready to fly again or has to be retained for further inspection. Non-availability of these parameters may delay the inspection process that may incur loss in terms of reducing the number of trips per day. While the lack of integrity may cause serious problems when the faulty aircraft is assumed as correctly functioning and permitted to take next trip.

Therefore, the aircraft network security design solutions must ensure a trusted transmission of parameters from the monitoring application to the aircraft control application and then to the laptop of the maintenance people. It should also ensure that the access to the aircraft control application is restricted and the laptop is allowed to connect to the monitoring application only. This implies network security requirements to closely monitor communication traffic between the laptop and the monitoring application. In this regard, some example security measures may include: a VPN connection with a strong encryption algorithm, configuring firewalls, authentication on the WIFI access point or router. The security experts wanted to confirm that these measures are sufficient to mitigate the risks. In addition, they wanted to verify the cost of the network security solutions to compare different choices; e.g., maintenance people can potentially connect to the aircraft using an Ethernet cable or a wireless connection.

The first iteration concerns the implementation of the aforementioned scenario using three SRE methods: Secure KAOS, STS and SEPP. The discussion part consolidates our observations on the SRE modelling experience of each method. We highlight the relative evaluation criteria in parenthesis to facilitate the understanding of which part of the observations are subjects to which evaluation criteria. Finally, we provide the initial feedback and evaluation results of the first iteration.

6.1.1. Scenario Implementation Using Secure KAOS

KAOS [17] (Knowledge Acquisition automated Specification or Keep All Objectives Satisfied), is a goal-oriented RE methodology developed as a joint collaboration of research work between the University of Oregon and the University of Louvain (Belgium) in 1990. The KAOS methodology mixes the top-down and bottom-up approaches. The goal modelling activity focuses on eliciting goals and refining them into sub-goals until they are atomic. Goal refinement is specified via AND/OR constructs. When a goal cannot be refined further, it is called as a requirement of the system-to-be and is assigned to an agent (to either environment or system agents) represented. When a requirement concerns an environment agent (e.g., human), it is called an expectation. Altogether, identifying and refining security goals and assigning them to agents are the core activities that are represented as goal and responsibility models. Authors of KAOS propose a formal specification language based on temporal logic. They also specify formal generic refinement patterns to validate goals refinements.

Our experimentation was conducted using the free trial version of Objectiver [60] the official tool maintained by Respect-IT. Figure 6 depicts a sample of the goal model specified in our example scenario context. It required some effort to become acquainted with the tool and its terminology through the help of guidelines and cited references (refer to R^M2.1.1 and R^M2.1.3 criteria). In particular, the security experts expressed some concerns regarding the formal temporal logic language (R^M2.1.1). They prefer the textual notation even it is more ambiguous.

KAOS drives the RE analyst to define agents later in the RE process. As a consequence, it does not help in expressing the relation between the agents and their interaction dependencies which is important for the network design. For instance, while defining network agents (e.g., an access control system) in our scenario analysis, we encountered issues when

we needed to add a new device in the network architecture. It was difficult to express the dependencies between network agents. In this regard, a conceptual diagram of the network would have been more concise and clearer. As a consequence, we had a difficulty in specifying network security requirements on network agents, particularly relative to security zoning (R^M3.1). Furthermore, KAOS provides features to express abstractions and the notation provides a good support to achieve traceability by linking goals and sub-goals (R^M4.1). However, it does not guide the security requirement engineers in asking the right questions at the right time to structure the security engineering process (R^M3.2).

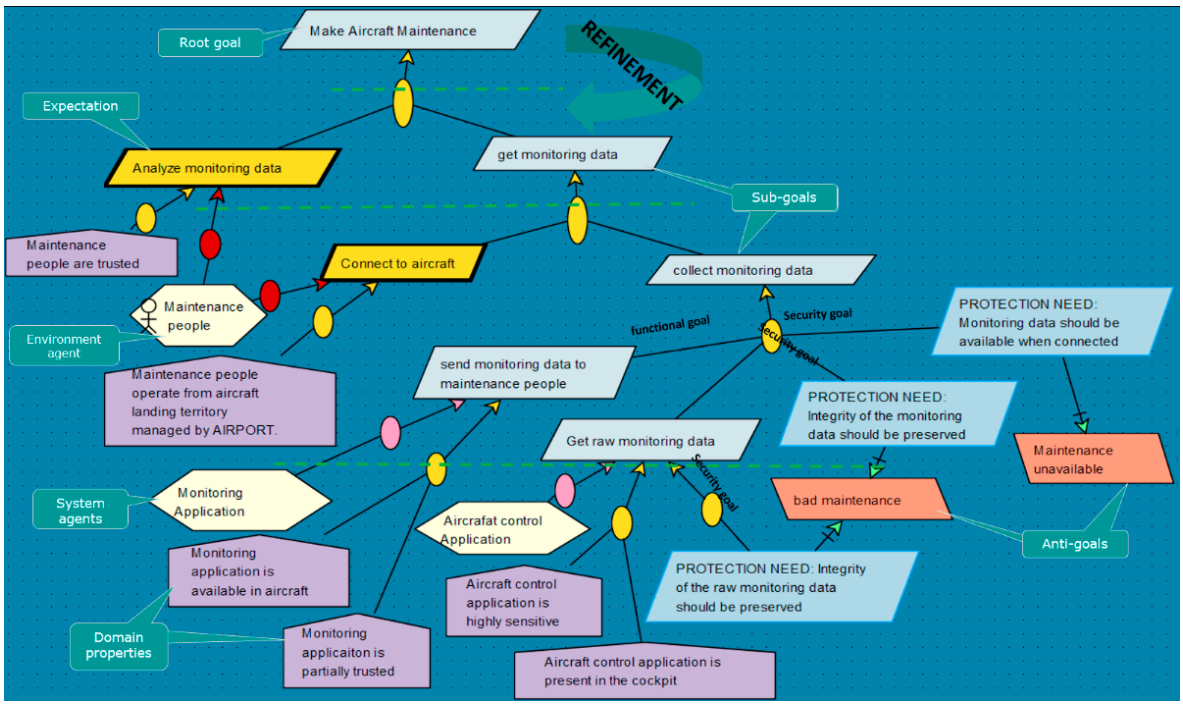

Figure 6. Secure KAOS goal specification (sample).

Regarding integration of risk analysis concepts, the link between security requirements and risk information is explicitly expressed using the concepts of obstacles/anti-goals that are represented as relations such as obstruction/resolution. For example, in Figure 6, security goal "PROTECTION NEED: monitoring data should be available when connected" resolves anti-goal "maintenance unavailable". The resolution link is expressed using green arrowhead. These anti-goals can be further refined such as 'normal' goals resulting in the specification of attack trees. Obstacles/Anti-goals include two risk attributes likelihood and criticality (R^M6.2) while security goals are attributed with ordinal priority scale (R^M6.3). However, there is no explicit relationship defined between the priority of a security goal/requirement and the risk of an associated obstacle. In addition, it helps in observing the environmental constraints upon the goals through domain properties. (e.g., physical laws). For instance, the trust assumptions on agents are expressed using the domain properties (Cf. Figure 6).

6.1.2. Scenario Implementation Using STS

STS (Secure Socio-technical Systems) [16] implements the agent-oriented approach which mainly focuses on early elicitation of security requirements based on the social dependency interaction relationships. Similar to KAOS, the STS framework offers the possibility to create composite goals via the AND/OR constructs. However, the respective goals and sub-goals are determined in the scope of each actor. An actor can be either a role or an agent. This methodology mainly defines three views: social view, authorization view and information view. The social relationships between actors are manifested by

the relationships such as goal delegation and resource provision. This is modelled in the social view. Delegated goal implies the dependency of the interaction. An actor depends on another actor to achieve a goal. In addition, the social relationships between actors may also include the exchange of documents (informational resources) that contain necessary information for the achievement of a goal. STS captures this exchange via the relationship: document transmission (resource provision). The ownerships of the resources are modelled in the authorization view. This view also permits to explicitly define the authorization rules over the access permissions on the documents (resources). Another view called the information view allows to define the primitives and relationships that permit the differentiation between the information (content) and the representation of the information (document).

STS is supported by a tool of the same name. The tool is not open source but it is freely available to public use (R^M2.1.2). The usage of a simple terminology as well as the user-friendly tool took less effort to become used to the overall concepts and terminology (R^M2.1.1 and R^M2.1.3). The security experts provided a positive feedback concerning the comprehensibility aspects during the first intermediary meeting.

Unlike KAOS, STS does not support full-fledged traceability of the security goals being refined into security requirements (R^M4.1). From the STS point of view, the security needs are implicitly expressed either in the form of security constraints attributes (e.g., integrity/availability/non-repudiation, etc.) over the interaction dependencies in the social view; or in the form of constrained permissions (e.g., read/modify/produce/transmit) upon the provision of the resources in the authorization view. Figure 7 highlights the security constraints over delegated goals in blue dashed rectangles, which cannot be refined further into network security requirements (R^M3.1).

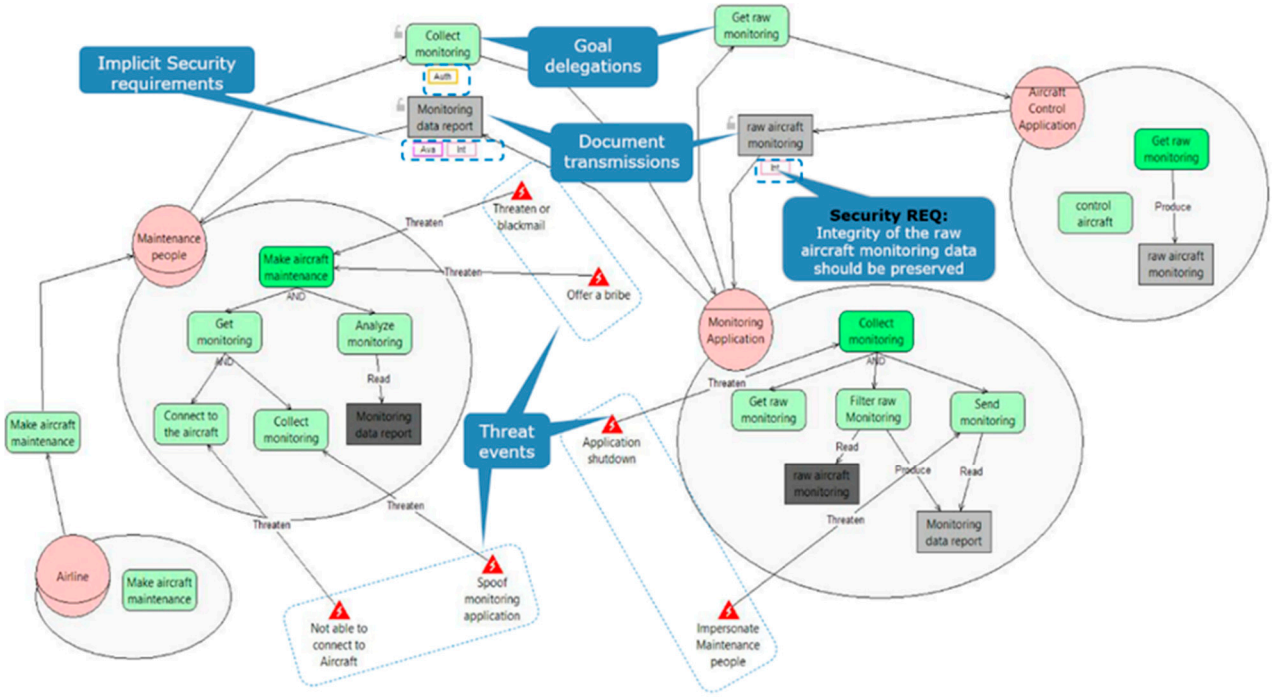

Figure 7. STS social view specification (sample).

On the other hand, STS provides a good support to express network agents and interaction dependencies. However, we had an issue in handling the number of dependencies between multiple network agents as the social view grows. Furthermore, from the STS modelling perspective, the abstractions are expressed using the three modeling views (i.e., social view, authorization view, information view). This abstraction perspective respects separation of concerns specific to authorization security problem context only. Similarly to

KAOS, STS does not guide the security requirement engineers in asking the right questions at the right time to structure the security engineering process in order to build a security architecture (R^M3.2).

In STS, the threat analysis can show the effects of a threating event over goal trees and goal/resource relationships. In addition, it defines attributes (implicit) to link countermeasures to threat events (R^M6.2). However, the threat analysis is limited to threat event propagation. Thereby it does not facilitate the expression of security needs upon the threats related to environmental risks. In addition, it does not have a facility to prioritize goals or threats such as in KAOS (R^M6.3). Figure 7 shows that the threat event "maintenance unavailable" threatens the goal "make aircraft maintenance". Likewise, the threat event "bad maintenance" threatens three goals. When the threats are propagated, the threat impact traces back to the root objective node. Respectively, the "maintenance unavailable" threatens the business objective "run business".

6.1.3. Scenario Implementation Using SEPP

SEPP (Security Engineering Process using Patterns) [18] implements the problem frames approach. In contrast with STS and KAOS, this approach views the stakeholder's requirements as constrained behavioral characteristics expected within the real physical world. A machine represents the system-to-be that helps to achieve the desired behavior. Problem world represents a part of the real world, in which the machine operates. Problem Domains represent different active entities (agents) in a given problem world upon which the machine operates in order to achieve the desired behavior. Casual domains (noted C in Figure 8) represent system agents. Lexical domains (noted X in Figure 8) represent data. Biddable domains (noted B in Figure 8) represent human agents. A problem is characterized as dependencies in terms of the shared phenomena (agents interaction attributes) between the machine and the problem domains, in a given problem context. A problem frame is therefore a problem pattern representing the common characteristics of a recognized class of problems. It separates the problems from its solution.

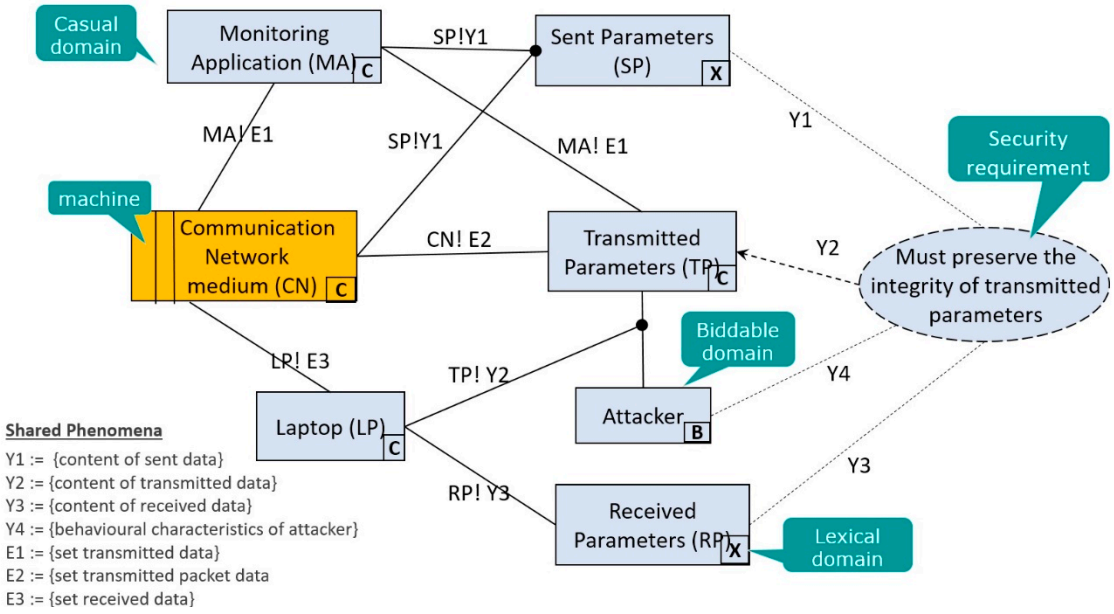

Figure 8. SEPP SPF diagram (sample).

SEPP extends the concepts and terminology of the PF to security problem analysis in a particular SRE context. It guides the RE analyst to define security problem patterns, called as security problem frames (SPF), separately from the security solution patterns, called as concertized security problem frames (CSPF). Accordingly, they have introduced four security problem frames (SPF) and eight concretized security problem frames for

authentication, confidential data transmission, distribution of secrets and integrity preserving data transmission. It is to note that, CSPF are the next abstraction level instantiations of the security problem frames which define some generic security mechanisms to solve security problems.

Figure 8 depicts a sample of our secure problem frames specification for integrity protection scheme. Security requirements are constrained on the communication network domain. The links between domains depict that there are some shared phenomena (i.e., shared attributes). The non-availability of a tool for SEPP made our experimentation hard (R^M2.1.2). It took some extra effort to become acquainted with the concepts and terminology even with the help of the cited references (R^M2.1.1 and R^M2.1.3). As a consequence, we have designed all the SPF and CSPF patterns models (for the scenario) manually which consumed more time (R^M2.1.3). In addition, there is a traceability issue (R^M4.1) as well. During our experimentation, we had other issues in knowing all the acting domains in a network environment, in particular during the early stages. A network design in hand will facilitate the problem analysis.

From the SEPP perspective, security problems (security goals) are identified using the what-if analysis technique similar to the Hazard analysis [61]. However, it does not provide explicit linking between the risk and the associated security goal (R^M6.2). This makes the risk definition implicit. The constraints on the security requirements are expressed in terms of pre-conditions attributes of SPF. These are the formalized conditions that must be satisfied by the problem environment "on prior", i.e., before applying the security problem frame. Similarly, the post-conditions attributes correspond to the formal expression of the security requirements.

Furthermore, SEPP facilitates a two-level abstraction through separation of security problems from solutions in generic perspective. However, this abstraction perspective respects the separation of concerns specific to the authorization security problem context, which does not help to explicitly describe the separation of concerns of the agents/actors specific to the network security problem context (R^M3.2). Lastly, similar to STS, it does not support attribute security requirements with either priority or implementation cost related information (R^M6.3 and R^M6.4).

6.1.4. Evaluation at Iteration 1—Results and Discussion

In Table 6, we resumed the evaluation results of the SRE methodologies to provide a consolidated view for comparison. First column displays the high-level root goals of evaluation criteria (Cf. Figure 4). Second column lists the elicited criteria that correspond to the leaf nodes in Figure 4. The remaining three columns display the respective measurement scales for each of the three SRE methods.

Each approach exhibits different features. STS is interesting when it concerns comprehensibility factors (i.e., R^M2.1.1, R^M2.1.2, R^M2.1.3), Secure KAOS is interesting in terms of traceability aspects (R^M4.1) as well as integration of risk analysis information. On the other hand, the modelling terminology and notation of SEPP seemed difficult to learn and adapt. Therefore, comparatively, STS and secure KAOS seem to be more suitable to our SRE context compared to SEPP. However, some of our requirements' R^M were not supported by any of the three methodologies (i.e., R^M3.1, R^M3.2, R^M6.1 and R^M6.4).

The criteria R^M3.1 and R^M6.1 are notably related to the network SRE context that concerns the integration of network security zoning. Without this support, it would be difficult to elicit network security requirements at early stages. Furthermore, the requirement concerning the abstraction characteristic goal cannot be respected by the three SRE methods (R^M3.2). As a consequence, expressing the security problem context of all the stakeholders including the business analysts, and the security analysts of all systems connected to the network is still an issue. Finally, none of the three methodologies allows to help in considering the cost of the security requirements (R^M6.4).

Table 6. Sample of the evaluation results.

Quality Criteria (Root Goal Nodes)	Elicited Evaluation Criteria List (R^M)	STS	Secure KAOS	SEPP
Comprehensibility (C11)	**R^M2.1.1:** The SRE methodology modelling language terminology must be easily understood to the user	high	medium	low
	R^M2.1.2: The cost of the SRE methodology training should be minimal	high	medium	low
	R^M2.1.3: The time taken to learn the SRE methodology should be minimal	high	medium	low
Abstract (C6)	**R^M3.1**: Should support the need for users to easily communicate ideas and feedback respecting the abstraction needs	nil	nil	nil
	R^M3.2: Should support the need for users to clearly define the number of abstraction levels of refinement	nil	nil	nil
Traceability (C4)	**R^M4.1**: The methodology should facilitate to trace the (network) security requirements back to business objectives	medium	high	low
Feasibility (C5) Adequacy (C10) Metadata (C20)	**R^M6.1**: Must facilitate to specify and link network security zone information	nil	nil	nil
Metadata (C20)	**R^M6.2**: Should annotate each requirement with risk attributes	low	medium	nil
Adequacy (C11) Metadata (C20)	**R^M6.3**: Should annotate each requirement with priority information	nil	high	nil
Feasibility (C5) Metadata (C20)	**R^M6.4**: Must facilitate to specify the implementation costs of network security requirements	nil	nil	nil

6.2. Evaluation Iteration 2—Use Case Scenario 2

During the first iteration of the evaluation, the authors of this article acted as stakeholders and tested the SRE models, which were then presented to the actual stakeholders (i.e., security experts) to capture their comments. The second iteration encompasses active participation of stakeholders for evaluating the STS modelling notation that holds highest rating in terms of comprehensibility aspects. The purpose of this second evaluation is capturing the direct feedback from real users' experience of SRE modelling in STS using a new use case scenario. In below, we first provide the scenario in plain-text description as it was presented to us at the initial step.

The scenario contains two business needs (Cf. Figure 9). The first business need concerns the safety measures related to the aircraft. It implies that the aircraft needs to be updated with correct load information through a portable device. The update center is a sub-component in the aircraft equipment, which reads these load parameters and performs an auto-update. The aircraft network security design must ensure the secure transmission of these parameters between the mobile devices to the on-board update center application. In addition, the direct connectivity of the mobile devices with the aircraft equipment is forbidden.

The second business need concerns the in-flight entertainment services to improve passengers' inflight condition. It describes the necessary functionalities of the aircraft equipment (e.g., passenger entertainment panel) to entertain the passengers during their flight journey. For example, one functionality is to provide the real-time flight status to the passengers. This functionality requires connectivity between the aircraft control panel, centralized on-board entertainment system component and the passengers' equipment panels. The security needs concern the availability of the aircraft's information (e.g., speed, altitude, temperature, distance, etc.) to the passengers accessing the respective entertainment system display panels. Respectively, the aircraft inflight network security design must ensure the availability of correct information to the passengers.

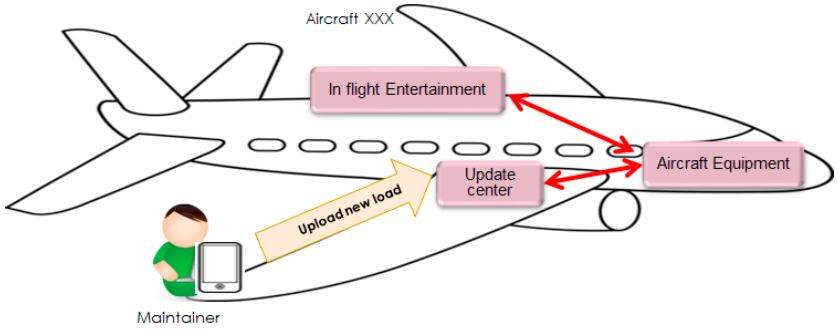

Figure 9. Second scenario.

6.3. Evaluation Iteration 2—STS Modelling and Feedback Discussion

From our initial analyses, we highlighted that STS provides no support to describe separation of concerns. In this experiment, we investigate the integration of SABSA layered approach to STS modelling in order to improve the separation of concerns.

SABSA [9] (Sherwood Applied Business Security Architecture) is a methodology framework for enterprise security architecture and service management. The main intention of this framework is to guide security architects in their task. It defines a six-layered architecture model; each layer reflecting the respective stakeholder view in the process (Figure 10). The Business view and Architect's view together correspond to the strategy and planning phases of the system engineering process. The Business view deals with the elicitation of high-level security objectives and constraints that are important to do business. It is essential to analyze the context of the business before building a secure system. This view facilitates the specification of the business security needs (e.g., reputation), operational goals and risk impacts (e.g., monetary loss) that drive the necessity of a secure system. The Architect's view concerns the integration of protection objectives (e.g., application security services, network security services, data security services, etc.) that are necessary to secure the business security needs. The Designer's view and builder's view together, correspond to the designing phases of the system security engineering process. The Designer's view deals with the specification of the security services (e.g., network entity authentication, data integrity protection, etc.) that are necessary to satisfy the control objectives. The builder's view deals with the mapping of the technical security mechanisms (e.g., firewalls, hashing, etc.) to the logical security services. Finally, tradesman view (LV5) corresponds to the final implementation phase of the system security engineering process. The latest view called operational manager's view corresponds to the maintenance management of the system security engineering process after the deployment.

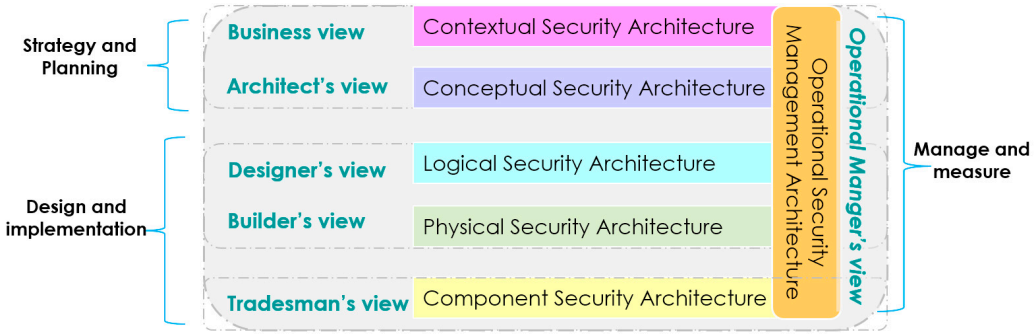

Figure 10. The SABSA layers.

SABSA is not strictly a security requirement engineering approach. This method provides a broad picture to understand the conventional security requirements engineering process starting from the perspective of the business system security architecture. The SABSA also includes a matrix that describes a wide variety of aspects that a modeler

should be able to express (Cf. Figure 11) with the help of six interrogative questionnaires (what/why/how/who/where/when). This matrix guides the security architect in asking the right questions at the right time. Thus, it structures the security engineering process.

	ASSETS (What)	MOTIVATION (Why)	PROCESS (How)	PEOPLE (Who)	LOCATION (Where)	TIME (When)
CONTEXTUAL ARCHITECURE	Business Decisions	Business Risk	Business Processes	Business Governance	Business Geography	Business Time Dependence
	Taxonomy of Business Assets, including Goals & Objectives	Opportunities & Threats Inventory	Inventory of Operational Processes	Organisational Structure & the Extended Enterprise	Inventory of Buildings, Sites, Territories, Jurisdictions, etc.	Time dependencies of business objectives
CONCEPTUAL ARCHITECTURE	Business Knowledge & Risk Strategy	Risk Management Objectives	Strategies for Process Assurance	Roles & Responsibilities	Domain Framework	Time Management Framework
	Business Attributes Profile	Enablement & Control Objectives; Policy Architecture	Process Mapping Framework; Architectural Strategies for ICT	Owners, Custodians and Users; Service Providers & Customers	Security Domain Concepts & Framework	Through-Life Risk Management Framework
LOGICAL ARCHITECTURE	Information Assets	Risk Management Policies	Process Maps & Services	Entity & Trust Framework	Domain Maps	Calendar & Timetable
	Inventory of Information Assets	Domain Policies	Information Flows; Functional Transformations; Service Oriented Architecture	Entity Schema; Trust Models; Privilege Profiles	Domain Definitions; Inter-domain associations & interactions	Start Times, Lifetimes & Deadlines
PHYSICAL ARCHITECTURE	Data Assets	Risk Management Practices	Process Mechanisms	Human Interface	ICT Infrastructure	Processing Schedule
	Data Dictionary & Data Inventory	Risk Management Rules & Procedures	Applications; Middleware; Systems; Security Mechanisms	User Interface to ICT Systems; Access Control Systems	Host Platforms, Layout & Networks	Timing & Sequencing of Processes and Sessions
COMPONENT ARCHITECTURE	ICT Components	Risk Management Tools & Standards	Process Tools & Standards	Personnel Man'ment Tools & Standards	Locator Tools & Standards	Step Timing & Sequencing Tools
	ICT Products, including Data Repositories and Processors	Risk Analysis Tools; Risk Registers; Risk Monitoring and Reporting Tools	Tools and Protocols for Process Delivery	Identities; Job Descriptions; Roles; Functions; Actions & Access Control Lists	Nodes, Addresses and other Locators	Time Schedules; Clocks, Timers & Interrupts
SERVICE MANAGEMENT ARCHITECTURE	Service Delivery Management	Operational Risk Management	Process Delivery Management	Personnel Management	Management of Environment	Time & Performance Management
	Assurance of Operational Continuity & Excellence	Risk Assessment; Risk Monitoring & Reporting; Risk Treatment	Management & Support of Systems, Applications & Services	Account Provisioning; User Support Management	Management of Buildings, Sites, Platforms & Networks	Management of Calendar and Timetable

Figure 11. The SABSA matrix (https://sabsa.org/---@2019 The SABSA Institute C.I.C).

In SABSA, network security requirement analysis (e.g., related to security zoning) starts from the designer's view (i.e., LV3). Since we demonstrated in the first iteration of evaluation that STS does not support to derive network security zoning; therefore, we confine this second experiment to the first two views of SABSA only (i.e., Business view and Architect's view). This implied that any discussion related to network security requirement analysis has been purposefully discarded. This strict confinement to the high-level abstraction improved the elicitation activity by provoking a long discussion. Indeed, this strict confinement to the SABSA abstraction has helped the team in focusing more on 'WHAT' perspectives of security needs rather than 'HOW' perspective of security needs. During this experimentation, we observed that the elicitation part of the scenario seemed clear and provoked a useful discussion.

Initially, the description of this scenario fits in just few lines. However, during the process of SRE modelling, the provoked discussion has developed the scenario details which resulted in an STS social model. In particular, new agents were introduced that were not mentioned during the textual description of the scenario. In addition, we observed that the goals of the maintainer agent were refined and improved as well. However, the second business need related to the passenger's entertainment system remained unchanged.

The overall feedback of the STS notation from the two iterations confirms that STS modelling is interesting and easy to adapt. The overall experiment lasted 2 hours, which proves that STS modelling concepts can be learnt quickly (R^M2.1.3). However, the STS modelling requires the users to know the agents and their respective initial goals before beginning the process. Our experiment shows that it is not the case because information on agents as well as their goals is evolving gradually during the elicitation process.

Although STS modelling concepts are interesting and easy to adapt, it does not help to trace the gradual growth of the social model, i.e., where to begin or how agents are

introduced in the model. In addition, considering the traceability or risk analysis aspects, STS provides less support while compared to Secure KAOS (see Table 6). Indeed, the first two layers of SABSA modelling emphasize on risk analysis and risk control objectives. Therefore, STS modelling language is an interesting choice to apply together with the SABSA layering views. However, it requires some enhancements in terms of integrating risk analysis concepts and traceability attributes.

7. Conclusions

An effective network security requirement engineering is needed to help organizations in capturing cost-effective security solutions that protect networks against malicious attacks while meeting the business requirements. However, the diversity of currently available security requirement engineering methodologies makes it difficult to select one. In this article, we presented a global evaluation methodology that we applied during the IREHDO2 project to find a requirement engineering method that could improve network security. Our evaluation methodology includes a process to determine pertinent evaluation criteria and a process to evaluate the requirement engineering methodologies. Our main contribution is to involve stakeholders (i.e., security requirements engineers) in the evaluation process by following a requirement engineering approach. We described our experiments conducted during the project with security experts and the feedback we obtained. Our evaluation of three SRE approaches (i.e., Secure KAOS, STS and SEPP) exhibited specific features of each approach. Secure KAOS facilitates traceability but cannot express the dependencies between network agents. STS is interesting regarding comprehensibility factors but cannot deal with the high number of dependencies between network security agents. Finally, the modelling terminology and notation of SEPP seemed difficult to learn and adapt.

Involving security experts in the selection process can be considered as a limitation of our evaluation methodology. However, once a SRE methodology is chosen, time and money are spent to train the users in order to make every stakeholders familiar with the chosen RE methodology and it is very unlikely that one would switch to new methodology soon. Therefore, from industrial usage perspective choosing the best suitable SRE methodology at earlier stages reduces overhead and saves time.

For future works, we would want to apply our evaluation approach to other security engineering contexts. This will help us to determine which evaluation criteria are generic and which are specific to a given security context. In the end, we intend to build a common repository to maintain the evaluations carried out in each scenario context so that there is no need to re-evaluate a SRE methodology for a similar context already considered in a previous evaluation. This can complement the requirement maturity evaluation work described in [33]. Furthermore, the verification methods and the measurement metrics can be reused on other. This knowledge will constitute a solid foundation to propose future SRE research directions towards standardizing the common perspective of good characteristics of an SRE methodology. Finally, we will focus on a global security requirements engineering process to acquire traceability from business level security objectives to network design level requirements. When structuring the SRE process using SABSA layered framework, STS modelling has been proved to be an interesting choice for expressing the business and designer's views security objectives. Based on this result and our previous work on network security zoning [62], we plan to create a new security requirements methodology that implements the SABSA framework.

Author Contributions: Investigation, S.T.B., R.L. and A.S.W.; writing—original draft preparation, S.T.B. and R.L.; writing—review and editing, R.L., A.S.W. and A.O.; supervision, R.L. and A.B.; project administration, R.L.; funding acquisition, R.L. and A.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was mainly funded by DGA/DGAC under project IREHDO2 No. 2014930807. It was also partially supported by the Horizon 2020 research and innovation program under grant agreements No. 830929 (Project CyberSec4Europe) and No. 956562 (Project LeADS).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: We thank all the security experts from Airbus who simply discussed with us or who kindly participate to the brainstorming meetings during our experiment. Their practical feedback was essential to conduct our research work.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

CSPF	concertized security problem frames
KAOS	Keep All Objectives Satisfied
RE	Requirement Engineering
SABSA	Sherwood Applied Business Security Architecture
SEPP	Security Engineering Process using Patterns
SPF	security problem frames
SRE	Security Requirement Engineering
STS	Secure socio-Technical Systems

References

1. SANS. Securing Against the Most Common Vectors of Cyber Attacks 2017. Available online: https://www.sans.org/white-papers/37995/ (accessed on 12 July 2021).
2. ISO/IEC 27033 IT Network Security Standard. Available online: http://www.iso27001security.com/html/27033.html (accessed on 11 July 2021).
3. SANS. Infrastructure Security Architecture for Effective Security Monitoring 2015. Available online: https://www.sans.org/white-papers/36512/ (accessed on 12 July 2021).
4. Stawowski, M. Network Security Architecture. *ISSA J.* **2009**, *7*, 34–38. Available online: https://www.bluetoad.com/publication/?m=1336&i=16198&p=34&ver=html5 (accessed on 12 July 2021).
5. Laborde, R.; Kamel, M.; Barrère, F.; Benzekri, A. Implementation of a Formal Security Policy Refinement Process in WBEM Architecture. *J. Netw. Syst. Manag.* **2007**, *15*, 241–266. [CrossRef]
6. Laborde, R.; Barrère, F.; Benzekri, A. Toward authorization as a service: A study of the XACML standard. In Proceedings of the 16th Communications & Networking Symposium, Society for Computer Simulation International, San Diego, CA, USA, 7–10 April 2013; p. 9.
7. Laborde, R.; Oglaza, A.; Wazan, A.S.; Barrère, F.; Benzekri, A. A situation-driven framework for dynamic security management. *Ann. Telecommun.* **2019**, *74*, 185–196. [CrossRef]
8. Barrere, F.; Benzekri, A.; Grasset, F.; Laborde, R. A multi-domain security policy distribution architecture for dynamic IP based VPN management. In Proceedings of the Policies for Distributed Systems and Networks, Monterey, CA, USA, 5–7 June 2002.
9. Sherwood, N.A. *SABSA (Sherwood Applied Business Security Architecture)—A Business-Driven Approach*; CRC Press: Boca Raton, FL, USA, 2005.
10. Hoo, K.S.; Sudbury, A.; Jaquith, A. Tangible ROI through Secure Software Engineering. *Security Business Q.* **2001**, 1.
11. Iqbal, J.; Ahmad, R.B.; Khan, M.; Alyahya, S.; Nasir, M.H.N.; Akhunzada, A.; Shoaib, M. Requirements engineering issues causing software development outsourcing failure. *PLoS ONE* **2020**, *15*, e0229785. [CrossRef] [PubMed]
12. Bulusu, S.T.; Laborde, R.; Wazan, A.S.; Barrère, F.; Benzekri, A. Towards the weaving of the characteristics of good security requirements. In Proceedings of the International Conference on Risks and Security of Internet and Systems, Paris, France, 4–6 November 2020; Springer: Berlin/Heidelberg, Germany, 2016; pp. 60–74.
13. Bulusu, S.T.; Laborde, R.; Wazan, A.S.; Barrère, F.; Benzekri, A. Which Security Requirements Engineering Methodology Should I Choose?: Towards a Requirements Engineering-based Evaluation Approach. In Proceedings of the 12th International Conference on Availability, Reliability and Security, Reggio Calabria, Italy, 29 August–1 September 2017; ACM: New York, NY, USA, 2017; p. 29.
14. Bulusu, S.T.; Laborde, R.; Wazan, A.S.; Barrère, F.; Benzekri, A. Applying a Requirement Engineering Based Approach to Evaluate the Security Requirements Engineering Methodologies. In Proceedings of the ACM SAC RE 2018, Pau, France, 9–13 April 2018.
15. Bulusu, S.T.; Laborde, R.; Wazan, A.S.; Barrère, F.; Benzekri, A. A Requirements Engineering-Based Approach for Evaluating Security Requirements Engineering Methodologies. In *Information Technology-New Generations*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 517–525.
16. Dalpiaz, F.; Paja, E.; Giorgini, P. *Security Requirements Engineering: Designing Secure Socio-Technical Systems*; MIT Press: Cambridge, MA, USA, 2016.

17. Van Lamsweerde, A. *Requirements Engineering: From System Goals to UML Models to Software Specifications*; Wiley: Hoboken, NJ, USA, 2009.
18. Hatebur, D.; Heisel, M.; Schmidt, H. A pattern system for security requirements engineering. In Proceedings of the 2011 Sixth International Conference on the Availability, Reliability and Security (ARES), Vienna, Austria, 22–26 August 2011; IEEE: Piscataway, NJ, USA, 2007; pp. 356–365.
19. Karpati, P.; Sindre, G.; Opdahl, A.L. Characterising and analysing security requirements modelling initiatives. In Proceedings of the 2011 Sixth International Conference on Availability, Reliability and Security (ARES), Vienna, Austria, 10–13 April 2007; IEEE: Piscataway, NJ, USA, 2011; pp. 710–715.
20. Khwaja, A.A.; Urban, J.E. A synthesis of evaluation criteria for software specifications and specification techniques. *Int. J. Softw. Eng. Knowl. Eng.* **2002**, *12*, 581–599. [CrossRef]
21. Mayer, N. *Model-Based Management of Information System Security Risk*; University of Namur: Namur, Belgium, 2009.
22. Fabian, B.; Gürses, S.; Heisel, M.; Santen, T.; Schmidt, H. A comparison of security requirements engineering methods. *Requir. Eng.* **2010**, *15*, 7–40. [CrossRef]
23. Rannenberg, K.; Pfitzmann, A.; Müller, G. IT security and multilateral security. *Multilater. Secur. Commun. Technol. Infrastruct. Econ.* **1999**, *3*, 21–29.
24. Muñante, D.; Chiprianov, V.; Gallon, L.; Aniorté, P. A review of security requirements engineering methods with respect to risk analysis and model-driven engineering. In Proceedings of the International Conference on Availability, Reliability, and Security, Fribourg, Switzerland, 8–12 September 2014; Springer: Berlin/Heidelberg, Germany, 2014; pp. 79–93.
25. van Lamsweerde, A. Elaborating security requirements by construction of intentional anti-models. In Proceedings of the ICSE 2004: 26th International Conference on Software Engineering, Washington, DC, USA, 23–28 May 2004; pp. 148–157.
26. Elahi, G.; Yu, E. A goal oriented approach for modeling and analyzing security trade-offs. In Proceedings of the International Conference on Conceptual Modeling, Auckland, New Zealand, 5–9 November 2007; Springer: Berlin/Heidelberg, Germany, 2007; pp. 375–390.
27. Souag, A.; Mazo, R.; Salinesi, C.; Comyn-Wattiau, I. Reusable knowledge in security requirements engineering: A systematic mapping study. *Requir. Eng.* **2015**, *21*, 1–33. [CrossRef]
28. Uzunov, A.V.; Fernandez, E.B.; Falkner, K. Engineering Security into Distributed Systems: A Survey of Methodologies. *J. Ucs* **2012**, *18*, 2920–3006.
29. Mellado, D.; Blanco, C.; Sánchez, L.E.; Fernández-Medina, E. A systematic review of security requirements engineering. *Comput. Stand. Interfaces* **2010**, *32*, 153–165. [CrossRef]
30. IEEE 830 IEEE 830-1998—IEEE Recommended Practice for Software Requirements Specifications. Available online: https://standards.ieee.org/findstds/standard/830-1998.html (accessed on 27 May 2016).
31. Mead, N.R. *How to Compare the Security Quality Requirements Engineering (SQUARE) Method with Other Methods*. DTIC Document. 2007. Available online: https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=8257 (accessed on 12 July 2021).
32. Nhlabatsi, A.; Nuseibeh, B.; Yu, Y. Security requirements engineering for evolving software systems: A survey. In *Security-Aware Systems Applications and Software Development Methods*; IGI Global: Hershey, PA, USA, 2012; pp. 108–128.
33. Niazi, M.; Saeed, A.M.; Alshayeb, M.; Mahmood, S.; Zafar, S. A maturity model for secure requirements engineering. *Comput. Secur.* **2020**, *95*, 101852. [CrossRef]
34. Sommerville, I.; Sawyer, P. *Requirements Engineering: A Good Practice Guide*; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 1997.
35. ISO29148:2011 ISO/IEC/IEEE 29148:2011 Systems and Software Engineering—Life Cycle Processes—Requirements Engineering. Available online: https://www.iso.org/standard/45171.html (accessed on 12 July 2021).
36. ISO, I. ISO/IEC 15408-1:2009 Information technology—Security techniques—Evaluation criteria for IT security—Part 1: Introduction and general model. *Int. Organ. Stand.* **2009**. Available online: https://standards.iso.org/ittf/PubliclyAvailableStandards/c050341_ISO_IEC_15408-1_2009.zip (accessed on 12 July 2021).
37. Kotonya, G.; Sommerville, I. Requirements engineering with viewpoints. *Softw. Eng. J.* **1996**, *11*, 5–18. [CrossRef]
38. Firesmith, D. Common Requirements Problems, Their Negative Consequences, and the Industry Best Practices to Help Solve Them. *J. Object Technol.* **2007**, *6*, 17–33. [CrossRef]
39. Firesmith, D. Specifying good requirements. *J. Object Technol.* **2003**, *2*, 77–87. [CrossRef]
40. Christian, T. *Security Requirements Reusability and the SQUARE Methodology*; Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst: Fairfax County, VA, USA, 2010.
41. Van Lamsweerde, A.; Brohez, S.; De Landtsheer, R.; Janssens, D. From system goals to intruder anti-goals: Attack generation and resolution for security requirements engineering. *Proc. RHAS* **2003**, *3*, 49–56.
42. Anderson, R.J. *Security Engineering: A Guide to Building Dependable Distributed Systems*; John Wiley & Sons: Hoboken, NJ, USA, 2010; ISBN 978-1-118-00836-2.
43. Mar, B.W. Requirements for development of software requirements. In Proceedings of the INCOSE International Symposium; Wiley Online Library: Hoboken, NJ, USA, 1994; Volume 4, pp. 34–39.
44. Wiegers, K.E. Writing quality requirements. *Softw. Dev.* **1999**, *7*, 44–48.
45. Wieringa, R.J. *Requirements Engineering: Frameworks for Understanding*; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 1996.
46. Boehm, B.W. Verifying and validating software requirements and design specifications. *IEEE Softw.* **1984**, *1*, 75. [CrossRef]

47. Pfleeger, S.L.; Atlee, J.M. *Software Engineering: Theory and Practice*; Pearson Education India, 1998; Available online: https://www.pearson.com/us/higher-education/program/Pfleeger-Software-Engineering-Theory-and-Practice-4th-Edition/PGM58925.html (accessed on 11 July 2021).
48. Davis, A.; Overmyer, S.; Jordan, K.; Caruso, J.; Dandashi, F.; Dinh, A.; Kincaid, G.; Ledeboer, G.; Reynolds, P.; Sitaram, P. Identifying and measuring quality in a software requirements specification. In Proceedings of the 1st International Software Metrics Symposium, Baltimore, MD, USA, 21–22 May 1993; IEEE: Piscataway, NJ, USA, 1993; pp. 141–152.
49. Young, R.R. *The Requirements Engineering Handbook*; Artech House: Norwood, MA, USA, 2004.
50. Hull, E.; Jackson, K.; Dick, J. *Requirements Engineering*; Springer Science & Business Media: Berlin, Germany, 2010; ISBN 978-1-84996-405-0.
51. Kar, P.; Bailey, M. Requirements Management Working Group: Characteristics of Good Requirements. In Proceedings of the INCOSE International Symposium, Boston, MA, USA, 7–11 July 1996; Wiley Online Library: Hoboken, NJ, USA, 1996; Volume 6, pp. 1225–1233.
52. Zielczynski, P. *Requirements Management Using IBM Rational RequisitePro*; IBM Press/Pearson plc: Indianapolis, IN, USA, 2008.
53. Mannion, M.; Keepence, B. SMART requirements. *ACM Sigsoft Softw. Eng. Notes* **1995**, *20*, 42–47. [CrossRef]
54. IEEE 1233—Guide for Developing System Requirements Specifications. 1998. Available online: https://ieeexplore.ieee.org/document/741940 (accessed on 12 July 2021).
55. NetworkX developers NetworkX 2.1 Python Package. Available online: https://networkx.github.io/documentation/stable/# (accessed on 21 June 2016).
56. Gephi.org Gephi 0.9.2—The Open Graph Viz Platform. Available online: https://gephi.org/ (accessed on 21 June 2016).
57. Ahmad, S. *Measuring the Effectiveness of Negotiation in Software Requirements Engineering*; University of Western Australia: Crawley, WA, Australia, 2012.
58. David Lynas SABSA Foundation Courses Training—David Lynas Consulting Limited. Available online: https://www.sabsacourses.com/course-schedule/ (accessed on 7 September 2018).
59. Stevens, S.S. On the Theory of Scales of Measurement. *Science* **1946**, *103*, 677–680. [CrossRef] [PubMed]
60. Respect-IT KAOS Tool—Objectiver: HomePage. Available online: http://www.objectiver.com/index.php?id=25 (accessed on 11 July 2021).
61. Kletz, T.A. *HAZOP and HAZAN: Identifying and Assessing Process Industry Hazards*; IChemE: Boca Raton, FL, USA, 1999.
62. Laborde, R.; Bulusu, S.T.; Wazan, A.S.; Barrère, F.; Benzekri, A. Logic-based methodology to help security architects in eliciting high-level network security requirements. In Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, Limassol, Cyprus, 8–12 April 2019; pp. 1610–1619.