



Méthodologie d'ingénierie des exigences de sécurité réseau

Sravani Teja Bulusu

► To cite this version:

Sravani Teja Bulusu. Méthodologie d'ingénierie des exigences de sécurité réseau. Génie logiciel [cs.SE].
Université Paul Sabatier - Toulouse III, 2019. Français. NNT : 2019TOU30084 . tel-02896486

HAL Id: tel-02896486

<https://tel.archives-ouvertes.fr/tel-02896486>

Submitted on 10 Jul 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Université Fédérale



Toulouse Midi-Pyrénées

THÈSE

En vue de l'obtention du
DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par l'Université Toulouse 3 - Paul Sabatier

Présentée et soutenue le *16 avril 2019* par

Sravani Teja BULUSU

Titre: Méthodologie d'Ingénierie des Exigences de Sécurité Réseau

JURY

Nora CUPPENS	Directrice de recherche, IMT-Atlantique	Rapporteur
Guy PUJOLLE	Professeur, LIP6, Sorbonne Université	Rapporteur
Yves ROUDIER	Professeur, I3S, Université de Nice Sophia Antipolis	Examineur
Abdelmalek BENZEKRI	Professeur, IRIT, Université Paul Sabatier	Directeur de thèse
Romain LABORDE	MCF, HDR, IRIT, Université Paul Sabatier	Co-directeur de thèse

INVITÉ

Bertrand LECONTE	Ingénieur, Airbus
François BARRERE	MCF, IRIT, Université Paul Sabatier
Ahmad Samer WAZAN	MCF, IRIT, Université Paul Sabatier

École doctorale et spécialité :

ED MITT - Ecole Doctorale Mathématiques, Informatique et Télécommunications de Toulouse

Unité de Recherche :

IRIT - Institut de Recherche en Informatique de Toulouse

Directeur(s) de Thèse :

Abdelmalek BENZEKRI et Romain LABORDE

Rapporteurs :

Nora CUPPENS et Guy PUJOLLE

“Ano bhadra krtavo yantu vishwatah”

“Que de nobles pensées nous viennent de toutes les directions”

“Let noble thoughts come to us, from all directions”

- R̥g veda

Dedicated to my beloved brother
Sri Satish Krishna Chaitanya BULUSU. . .

Résumé

La construction de réseaux sécurisés est à la fois une étape cruciale et complexe pour toute organisation. Traditionnellement, cette tâche couvre les aspects architecturaux en proposant une segmentation du réseau où des règles de sécurité différentes sont appliquées à chaque zone ; elle couvre également la sécurisation des équipements d'extrémité exploités par des utilisateurs et apporte ainsi des garanties sécuritaires pour les informations transférées sur les liaisons de communication. Le plus souvent, les aspects sécurité réseau sont pris en compte après la conception du réseau et donc tardivement. Ceci se traduit inéluctablement par une augmentation de la complexité et des coûts pour prendre en compte les modifications nécessaires qui doivent alors être apportées.

À cet égard, les exigences de sécurité revêtent une importance primordiale, car elles guident les décisions relatives à la mise en œuvre des contrôles de sécurité réseau (ex. Firewalls et proxies sécurité, VPN) répondant aux besoins de l'entreprise. En effet, de mauvaises exigences en matière de sécurité réseau peuvent conduire à une sécurité inefficace voire à des failles de sécurité dans la conception de la sécurité du réseau. Cependant, les méthodologies d'ingénierie des exigences de sécurité actuelles ne permettent pas de déduire les exigences de sécurité réseau.

Ce travail de thèse fait partie du projet de recherche DGA IREDHO2 (Intégration REseau Haut Débit Embarqué Optique 2^{ème} phase) qui concerne en particulier les réseaux avioniques du futur. Ce travail est le résultat d'une collaboration menée avec la société AIRBUS GROUP. Il a pour objectif de proposer une méthodologie d'ingénierie des exigences de sécurité pour capturer, exprimer et analyser les exigences de sécurité réseau, afin de pouvoir les dériver et gérer l'application de configurations réseau dans une approche « Top Down ». La complexité adressée vient à la fois des différences de point de vue : i) en regard de la compréhension de la problématique de la sécurité par les différentes parties prenantes, ii) de la nature des systèmes impactés et iii) de la variabilité des niveaux d'abstraction retenus dans le cycle de développement réseau.

Dans ce travail, nous avons défini une méthode qui s'appuie sur les niveaux d'abstraction proposés par la méthode SABSA (Sherwood Applied Business Security Architecture) afin de décomposer les exigences de sécurité métier en exigences de sécurité techniques. En effet, SABSA préconise l'étude de vues Métier (décisionnaire), Architecte (objectifs, risques, processus, applications, interactions), Concepteur (services de sécurité), Constructeur de réseau (mécanismes de sécurité) et Composants (produits, outils, technologies, etc.).

Les vues Métier et Architecte sont exprimées dans le formalisme STS (Social Technical Systems). Nous proposons aussi de représenter les attaques comme un système multi-agent pour l'analyse de risques des niveaux Métier et Architecte. Pour l'expression des exigences de la vue Concepteur, nous reprenons les approches de définition de zones dont nous renforçons les principes de construction/protection en s'appuyant sur les modèles formels d'intégrité. Nous proposons une méthodologie qui automatise la proposition de zones de sécurité ainsi que la dérivation des exigences de sécurité réseau à l'aide d'un ensemble fini de règles formalisées. Nous avons développé un outil qui implémente ces règles en ASP (Answer Set Programming) et facilite le calcul des modèles de zone de sécurité à coût optimal.

Enfin pour assurer la traçabilité entre les différentes vues Métier, Architecte et Concepteur, nous définissons une nouvelle notation de modélisation à partir des concepts proposés dans KAOS (Keep All Objectives Satisfied) et STS. Nous illustrons notre méthodologie à l'aide d'un scénario propre au projet IRHEDO2. Finalement, nous évaluons notre méthodologie en considérant deux scénarios supplémentaires : 1) une étude de cas de réseau d'entreprise dans le domaine du commerce électronique ; 2) un deuxième scénario issu du projet IRHEDO2.

Abstract

Building secure networks is crucial as well as challenging for any organization. Network security majorly concerns the security architectural needs that describe network segmentation (i.e., security zoning); security of network devices connecting the communicating end user systems; and security of the information being transferred across the communication links. Most often, a late consideration of security aspects (i.e., post-deployment of network design) inevitably results in an increase in costs as well as in the complexity to take into account the necessary changes that have been made to the existing infrastructures.

In this regard, network security requirements hold a paramount importance since they drive the decisions related to the implementation of security controls about business needs. Indeed, bad network security requirements can lead to ineffective and costly security or worse security holes in the network security design. Nevertheless, current security requirement engineering methodologies render no support to derive network security requirements.

This thesis work is a part of the research project DGA IREHDO2 (Intégration REseau Haut Débit embarqué Optique 2ème phase) that concerns aircrafts future generation networks. Our work is done mainly in collaboration with AIRBUS and is related to the security requirements engineering process for aircraft networks. Our objective in this project is to propose an SRE methodology for capturing and analysing network security requirements, and that facilitates the refinement into network security and monitoring configurations (TOP/DOWN approach). The complexity addressed comes at a time from the differences in point of view: i) with regard to the understanding of the issue of security by different stakeholders, ii) the nature of the systems impacted and the variability of the levels of abstraction in the network development cycle.

In this work, we defined SRE methodology based on the abstraction levels proposed by SABSA (Sherwood Applied Business Security Architecture) method in order to structure the refinement activity of business needs into network security requirements. Indeed, SABSA recommends the expression of the needs considering the Business view (decision makers), Architect's view (objectives, risks, processes, applications and interactions), Designer's view (security services), Builder's view (security mechanisms) and Tradesman's view (products, tools, technologies). We considered the first three views.

We express the *business* and *architect*'s views using STS (Social-Technical Systems) formalism. We also propose to represent attacks as multi-agent systems to facilitate the analysis of security risks at these first two views. For expressing the network security requirements captured at *Designer's view*, we propose a methodology that automates parts of the process of security zoning and network security requirements elicitation using a definite set of formalized rules derived from security design principles and formal integrity models. We developed a tool that implements these rules in ASP (Answer set programming), which facilitates calculating cost-optimal security zone models.

In the end, to ensure traceability between the three views, we defined a new modelling notation based on the concepts proposed in KAOS (Keep All Objectives Satisfied) and STS. We illustrate our methodology using a scenario specific to the IREHDO2 project. Finally, we evaluate our methodology using: 1) an e-commerce enterprise case study; 2) a new scenario specific to the IREHDO2 project.

Acknowledgements

First of all, I would like to hail all **the researchers**, otherwise known as the knowledge worshipers, who are always in pursuit of innovative ideas and solutions, which have a great potential to transform our world into a better place. PhD program is one such a journey that trains individuals for providing license to the world of research. Although, this journey would never have been possible without the support of many people. This section of this document is to acknowledge my sincere thanks to all those who have been part (directly/indirectly) of my PhD program.

Foremost, I am profoundly indebted to **Professor Abdelmalek BENZEKRI** for having granted me the opportunity to pursue my doctoral studies. It has been a privilege to work under his supervision and to learn from his outstanding experience and knowledge. As quoted by Sir Isaac Newton, *“If I have seen further, it is by standing on the shoulders of giants”*, it is no wonder that prof. Benzekri is a giant with an extraordinary vision in security field; although, I am no newton but a dwarf who is deeply honoured to receive his invaluable support and guidance. One thing I learned from him is that *“A broad imagination, organization and structured expression of ideas is vital to any research work in order to make it more valuable and reachable”*. Apart from his professional expertise, his positive state of mind and enthusiasm are truly amazing. Moreover, he is kind, joyful, and always carries himself with a great amount of energy. For instance, the way he greets “Bonjour” with his deep and loud voice itself stirs a positive vibe within the team. Finally, I consider myself very lucky for having given a chance to work with such a great professor with an inspiring personality; and to receive his support and advice that will always be helpful in my career.

Next, I would like to express my heartfelt gratitude to my co-supervisor **Associate professor Romain LABORDE** with whom I closely worked on the topics of research presented in this thesis. He has been the main pillar of my thesis work, who sowed the seeds of success of our research. I feel extremely fortunate to learn the alphabets of research under his supervision. It is him who introduced me to the topics of security requirements engineering and risk management, which I believe is the one of the best turning points of my career. He has invested a great deal of time and effort to prepare me for this dissertation. If not his availability, consistent support, and encouragement it would have been impossible for me to explore and excel these topics. I am also very indebted to his constructive comments and relentless guidance throughout this dissertation. Particularly, I must emphasize that his critiques are highly insightful and invaluable, which have only pushed me further into improving my knowledge and skills. One of his frequent critiques that got hardcoded in my mind is that *“an idea not explained in simple manner is an idea not well understood”*. He never stepped back to provide me with his valuable comments, even during his busiest times of his HDR dissertation, which shows his dedication to his job. Apart from that, I also express my gratitude for helping me with the tool implementation of one of our research results (security zone modelling), which has saved me time for other research tasks. Finally, I cannot thank him enough (or I don’t know how to) for his impeccable support and patience during my thesis writing, for it was very overwhelming and an unforgettable experience. I also sincerely acknowledge all his efforts for reviewing my papers, deliverables and of course the thesis manuscript and suggesting me with corrections and improvisations.

Further, I am extremely thankful to another supervisor of mine, **Associate Professor Ahmad Samer WAZAN**, for his excellent support, and guidance throughout my PhD program. I admire his personality, for he is someone who is calm and knows how to tackle situations with poise. He has been very patient with me, notably his moral support during the hard times of thesis writing cannot be forgotten. August 2018 is the month that I had suffered the heights of depression. If not his moral support and kind words, I was almost convinced to quit the PhD program due to the great amount of stress and anxiety that clouded my instincts. Furthermore, I would like to add my genuine appreciation and acknowledgement for his efforts to review my PhD manuscript, which he took up in the last minute, due to some unexpected circumstances.

Next, I extend my deep indebtedness to my other supervisor **Associate Professor François BARRERE**, who has been my moral support throughout my PhD program. What to say about him, he is one of the amazing persons I have ever met and has a stunning personality. He is extremely kind and always puts himself first to help others. He has always been my best interest at heart, whether it's advising me on the topics of my research; helping me in preparing documents in French when in need; helping me in understanding the French administration process and of course its beautiful culture (it is fascinating to learn about the history of French wine). I never felt like I am doing PhD in a foreign land (with different language and culture), for he has always made himself available for any professional help or guidance. I have a huge respect for him as a mentor, guide, researcher, friend, as well as a great philosopher. There was a moment when I was low-spirited for a personal reason and he said "*Teja! Be happy that we are not in middle of a war*". It is a precious lifetime quote that I would always remember.

These three years has been a roller coaster ride of emotions both from personal and professional front. The pleasure and emotion of this experience will never be diminished nor will my appreciation to the invaluable support of all my four supervisors. I cherish this experience that stands as one of the **core memories** (refer 'inside out' movie) of my life.

I hereby express my sincere thanks the jury committee members, **Professor Nora CUPPENS**, **Professor Guy PUJOLLE**, and **Professor Yves ROUDIER** for agreeing to serve on my jury committee and for helping to improve my dissertation with their thoughtful critiques.

This work is part of research project IREHDO2 funded by DGA/DGAC. The financial support is highly acknowledged. I am also grateful to all the security experts at Airbus who helped us with their useful comments. Special thanks goes to **Mr. Bertrand LECONTE**, Research engineer at Airbus, for helping us get this project and more importantly, for agreeing to honour us with his presence for my PhD dissertation.

To this chain of gratitude, I would like to add **Madam Samia BOUZEFRANE** at CNAM Paris, who is the reason that I applied to this PhD position in the first place. I highly appreciate her patience and special interest to take time for forwarding the open PhD positions to the students looking for an opportunity. In addition, I like to thank all the **anonymous reviewers** of the papers that we have submitted to various conferences, for their valuable feedback and interesting comments.

During the tenure of this work, I have collaborated with many colleagues, in the SIERA team, for whom I have my great regard. I take this opportunity to convey my thanks to Thierry Desprats, Emmanuel Lavinal, Pierrick Marie, Arnaud Oglaza, Remi Venant, Messaoud Aouadj, Frédéric Lavaud, for bearing with patience my annoying questions about French language (more frequently *comment dit-on français ...?* ☺); to toufik moad, for his help with the matters relating to bank, caf and impots during the very first days of my Phd; to Basha Kabbani for his assistance while I moved to a new residence; to all other office mates such as Daniel Marquié, Marco Winckler, Philippe Palanque, Frank Silvestre, Julien Broisin, and all other members of the SIERA team, even though our interaction has been limited to “*Bonjour ça va?*”, for I believed it is often good for morale and team spirit. Special thanks goes to Mar Pérez-Sanagustín and Anis Bey along with Arnaud and Pierrick for their valuable feedback on my dissertation practice presentation.

My gratitude also goes to **Madam Chantal MORAND** and **Madam Catherine STASIULIS** for helping with the matters relating to administrative procedures of my PhD program; to Lorène COMBES and Matthew BEGUE for their help regarding the travel procedures when I had to travel for conferences; to the university for funding my FLE French courses; to all the tutors of FLE courses for helping me to improve my French language skills (je crois que je parle un peu mieux le français ☺, sauf que la prononciation des lettres ‘R’ et ‘U’ est toujours difficile ☹). I am also grateful to my Toulouse friends Sueline, Omar, Charles, Patricia, Michelle and Fernando, as well as to my fellow Indian expats Kuntal, Santan, Sourav, Pritam, Ramakrishna for all the cherishing moments.

I would not have been here without the unending love and affection of my parents in India. It is their warm blessings and great sacrifices that keep me in this position today. I am longing to share that special thanks goes to my mother for coming out with the perfect humour during one of the depressing times of my thesis writing. Here she goes with a serious tone “*Sravani! I find myself curious to do a PhD; my research topic would be: What’s taking my daughter so long to finish writing her thesis*”, which got me involuntarily burst into laughter despite my situation at that moment.

My special thoughts are for my beloved brother (Dr. Sri Satish Krishna Chaitanya) who has always been the source of my inspiration since childhood. He has always encouraged me into reaching greater heights in my career. I profit this occasion to express my sincere gratitude to him. I dedicate every success of my career (in the past, present and future) to him, for I keep the failures with me, which I believe as occurred due to my imbecilic mistakes.

My profound gratitude goes to France, the country that laid foundation to my career in security field. I take this opportunity to thank Thales organization for funding my master program. I would also thank my professors at Eurecom Institute **Refik MOLVA**, **Yves ROUDIER**, and **Marc DACIER** for their guidance and support during my Master studies.

I conclude this section with my love and devotion to **THE MOTHER NATURE**, **MY GODESS**, for keeping me alive, safe and healthy.

Table of Contents

RESUME.....	I
ABSTRACT	III
ACKNOWLEDGEMENTS.....	IV
TABLE OF CONTENTS	VII
TABLE OF FIGURES	VIII
LIST OF TABLES	X
THESIS INTRODUCTION.....	1
RESEARCH BACKGROUND AND MOTIVATION	1
THESIS RESEARCH PROJECT IREHDO2.....	2
RESEARCH ROADMAP AND CLAIMED CONTRIBUTIONS	3
THESIS OUTLINE	5
PUBLICATIONS.....	6
CHAPTER 1 SECURITY REQUIREMENTS ENGINEERING	8
1.1 SECURITY REQUIREMENTS, A GLOBAL VIEW	8
1.2 SECURITY RISK MANAGEMENT.....	9
1.2.1 RISK ANALYSIS TECHNIQUES	10
1.2.2 RISK MANAGEMENT METHODS.....	14
1.3 SECURITY REQUIREMENTS ENGINEERING PROCESS	14
1.3.1 SRE PROCESS MODELS	16
1.3.2 SABSA (SHERWOOD APPLIED BUSINESS SECURITY ARCHITECTURE) FRAMEWORK.....	17
1.4 SECURITY REQUIREMENTS ENGINEERING APPROACHES	19
1.4.1 GOAL-ORIENTED APPROACHES	20
1.4.2 AGENT-ORIENTED APPROACHES	23
1.4.3 PROBLEM FRAMES ORIENTED APPROACHES	25
1.5 NETWORK SECURITY REQUIREMENTS.....	28
1.5.1 SECURITY ZONING REFERENCE MODELS	28
1.6 CHAPTER SUMMARY	31
CHAPTER 2 WHICH SRE METHODOLOGY FITS BEST TO THE NETWORK SRE CONTEXT?	32
2.1 COMPARATIVE STUDIES: STATE-OF-THE ART	33
2.1.1 STATE-OF-THE-ART ANALYSIS.....	34
2.2 RE-BASED SRE EVALUATION METHODOLOGY	35
2.2.1 FORMAL MODELLING NOTATION.....	36
2.3 RQ1A: WHAT ARE GOOD SECURITY REQUIREMENTS?.....	36
2.3.1 THE WEAVING METHODOLOGY	39
2.4 RQ1B: WHAT IS A GOOD SRE METHODOLOGY?.....	41
2.4.1 STEP1: PROBLEM CONTEXT AND INITIAL REQUIREMENT ANALYSIS	41
2.4.2 STEP2: REFINEMENT (R^M) REQUIREMENT ANALYSIS	43
2.4.3 ELICITED SET OF EVALUATION CRITERIA (R^M)	47
2.5 STEP3: EVALUATION OF SRE APPROACHES	49
2.5.1 EVALUATION ITERATION 1 – USE CASE SCENARIO 1	49
2.5.2 EVALUATION ITERATION 2 – USE CASE SCENARIO 2.....	57
2.5.3 EVALUATION ITERATION 2 – STS MODELLING AND FEEDBACK DISCUSSION.....	57
2.6 CHAPTER SUMMARY	59

CHAPTER 3	PROPOSED SRE METHODOLOGY – A LAYER BASED APPROACH	61
3.1	SRE METHODOLOGY CONCEPTUAL MODEL	61
3.1.1	LAYER BASED ABSTRACTION MODELLING CONCEPTS	62
3.1.2	ANTI-STIS MULTI-AGENT RISK MODELLING	66
3.1.3	OUR PROPOSED ZONE MODELLING METHODOLOGY	69
3.1.4	FORMAL APPROACH OF ZONE MODELLING METHODOLOGY	73
3.2	SRE METHODOLOGY ILLUSTRATION	77
3.2.1	BUSINESS VIEW (STIS MODELLING).....	77
3.2.2	ARCHITECT’S VIEW (STIS MODELLING).....	80
3.2.3	DESIGNER’S VIEW (ZONE MODELLING)	85
3.3	CHAPTER SUMMARY	89
CHAPTER 4	EVALUATION OF PROPOSED SRE METHODOLOGY	91
4.1	E-COMMERCE CASE STUDY DESCRIPTION	91
4.2	E-COMMERCE SCENARIO IMPLEMENTATION.....	93
4.2.1	BUSINESS VIEW	93
4.2.2	ARCHITECT’S VIEW	94
4.2.3	DESIGNER’S VIEW	99
4.3	AIRBUS SCENARIO 2 IMPLEMENTATION	102
4.4	DISCUSSION AND EVALUATION – SRE METHODOLOGY	106
4.4.1	DISCUSSION ON THE SRE METHODOLOGY EVALUATION WITH REFERENCE TO THE EVALUATION CRITERIA	106
4.4.2	DISCUSSION ON THE EVALUATION OF ZONE MODELLING METHODOLOGY	109
4.5	CHAPTER SUMMARY	110
THESIS CONCLUSION		111
THESIS BACKGROUND AND MOTIVATION		111
THESIS PROJECT AND RESEARCH QUESTIONS.....		112
SYNTHESIS OF CONTRIBUTIONS.....		113
CURRENT WORK AND FUTURE PERSPECTIVES	ERREUR ! SIGNET NON DEFINI.	
BIBLIOGRAPHY		118
APPENDIX A.	FAIR RISK TAXONOMY	127
APPENDIX B.	GRAPHICAL TOOL FOR ANALYZING THE SEMANTIC DEPENDENCIES OF THE QUALITY CHARACTERISTICS.....	128
APPENDIX C.	VERIFICATION METHODS FOR THE ELICITED EVALUATION CRITERIA	132
APPENDIX D.	SRE METHODOLOGY COMPLETE VIEW	135

Table of Figures

FIGURE 1: OUR GOAL IN IREHDO2 PROJECT	2
FIGURE 2: LAYER BASED SRE METHODOLOGY OVERVIEW.....	5
FIGURE 3: RISK MANAGEMENT GLOBAL VIEW	10
FIGURE 4: FMEA RISK CALCULATION EXAMPLE [STAMATIS 2003].....	11
FIGURE 5: CAUSE-CONSEQUENCE DIAGRAM [HOGGANVIK 2007]	11

FIGURE 6: ATTACK TREE: HACK ADMIN PASSWORD	12
FIGURE 7: ADTREE EXAMPLE: AN ATTACK ON A BANK ACCOUNT [KORDY ET AL. 2011].....	12
FIGURE 8: ATTACK GRAPH [TELMON AND CISA 2012]	13
FIGURE 9: BN FOR SYSTEM FAILURE	13
FIGURE 10: MARKOV CHAIN [HØYLAND AND RAUSAND 2009]	13
FIGURE 11: RE LIFE-CYCLE.....	15
FIGURE 12: SQUARE METHODOLOGY	16
FIGURE 13: SREP PROCESS	17
FIGURE 14: SABSA LAYERED MODEL [SHERWOOD 2005].....	17
FIGURE 15: SABSA MATRIX	19
FIGURE 16: SECURE KAOS SRE METHODOLOGY	21
FIGURE 17: ANTI-GOAL REFINEMENT SAMPLE [VAN LAMSWERDE 2004]	22
FIGURE 18: STS-ML SOCIAL MODELLING.....	24
FIGURE 19: STS THREAT EVENT NOTATION	25
FIGURE 20: PROBLEM OVERVIEW	25
FIGURE 21: SPF PATTERN FOR CONFIDENTIALITY [HATEBUR ET AL. 2007B]	27
FIGURE 22: EXAMPLE ZONE REFERENCE MODELS [SECURE ARC 2009; PROVINCE OF BRITISH COLUMBIA 2012]	29
FIGURE 23: ZONE CLASSIFICATION MODEL [GONTARCZYK ET AL. 2015].....	30
FIGURE 24: SECURITY ZONE CLASSIFIER [GONTARCZYK ET AL. 2015].....	30
FIGURE 25: OUR EVALUATION METHODOLOGY	35
FIGURE 26: QUALITY CRITERIA - CONSOLIDATED SOURCES LIST	37
FIGURE 27: COMPLETE VIEW OF CRITERIA CONCEPTUAL GRAPH	38
FIGURE 28: SEMANTIC DEPENDENCIES - CYCLIC REFERENCES	39
FIGURE 29: WEAVING THE CHARACTERISTICS OF GOOD SECURITY REQUIREMENTS.....	40
FIGURE 30: ELICITATION TOOL.....	42
FIGURE 31: SRE-METHODOLOGY-TO-BE GOALS REFINEMENT	44
FIGURE 32: EXAMPLE SCENARIO 1	50
FIGURE 33: SECURE KAOS GOAL SPECIFICATION (SAMPLE).....	51
FIGURE 34: STS SOCIAL VIEW SPECIFICATION (SAMPLE).....	53
FIGURE 35: SEPP SPF DIAGRAM (SAMPLE)	54
FIGURE 36: AIRBUS NEW SCENARIO	57
FIGURE 37: USE CASE SCENARIO 2 STS SOCIAL VIEW	58
FIGURE 38: INITIAL CONCEPTUAL MODEL OF OUR LAYER BASED SRE METHODOLOGY.....	62
FIGURE 39: FINALIZED CONCEPTUAL MODEL OF OUR LAYER BASED SRE METHODOLOGY.....	63
FIGURE 40: GENERIC GOAL REFINEMENT RATIONALE (REFER KAOS).....	64
FIGURE 41: COMPOSITE REQUIREMENT NOTATION	64
FIGURE 42: REFINEMENT LINKS OF STS AND KAOS	65
FIGURE 43: COMPOSITE REQUIREMENT REFINEMENT	66
FIGURE 44: COMPARISON BETWEEN STS AND ANTI-STS NOTATIONS	68
FIGURE 45: CW-LITE SECURITY FILTERING RULE [SHANKAR ET AL. 2006]	72
FIGURE 46: INTEGRITY VALUES OF DOMAIN AND AGENTS FOR USE CASE SCENARIO 1	72
FIGURE 47: OUR ZONE MODELLING METHODOLOGY APPROACH OVERVIEW	73
FIGURE 48: BUSINESS VIEW - SEPARATION OF CONCERNS.....	78
FIGURE 49: AIRCRAFT MAINTENANCE BUSINESS NEED LINKED TO BUSINESS DRIVERS	78
FIGURE 50: ANTI-STS FEARED THREAT AGENTS AT BUSINESS VIEW	79
FIGURE 51: ARCHITECT'S VIEW - SEPARATION OF CONCERNS	80
FIGURE 52: REFINEMENT OF BUSINESS FUNCTIONAL GOALS AND AGENTS - ARCHITECT'S VIEW.....	81
FIGURE 53: INTEGRATING SECURITY DOMAIN INFORMATION (USE CASE SCENARIO 1).....	82
FIGURE 54: SCENARIO - RISK ANALYSIS USING ANTI-STS (SAMPLE)	84
FIGURE 55: DESIGNER'S VIEW - SEPARATION OF CONCERNS	85
FIGURE 56: REFINEMENT OF NETWORK DATA FLOWS	86
FIGURE 57: ZONE MODELLING STEP1 – OUTPUT (USE CASE SCENARIO 1)	86

FIGURE 58: COST-BASED OPTIMIZATION RULE	86
FIGURE 59: ZONE MODELLING STEP2 – INPUT (USE CASE SCENARIO 1)	88
FIGURE 60: MEDIUM INTEGRITY LEVELS EXAMPLE (USE CASE SCENARIO 1)	88
FIGURE 61: STEP2 OUTPUT –FINAL	88
FIGURE 62: LAYERED SRE METHODOLOGY - COMPLETE VIEW	90
FIGURE 63: E-COMMERCE EXAMPLE CASE STUDY [ANSSI 2017]	92
FIGURE 64: E-COMMERCE SCENARIO SPECIFICATION - BUSINESS VIEW	94
FIGURE 65: E-COMMERCE SCENARIO - REFINING BUSINESS GOALS	95
FIGURE 66: E-COMMERCE SCENARIO - REFINING AGENT INTERACTION NEEDS	96
FIGURE 67. INTEGRITY VALUES OF DOMAINS AND AGENTS FOR THE EXAMPLE CASE STUDY	97
FIGURE 68: E-COMMERCE SCENARIO - EXPRESSING THE SECURITY DOMAIN INFORMATION	98
FIGURE 69: E-COMMERCE SCENARIO - ZONE MODELLING STEP1 (INPUT)	99
FIGURE 70: E-COMMERCE SCENARIO - ZONE MODELLING STEP1 (OUTPUT)	100
FIGURE 71: EXAMPLE INTEGRITY LEVELS FOR THE MEDIUM.....	101
FIGURE 72: E-COMMERCE SCENARIO - ZONE MODELLING (STEP2) (INPUT).....	101
FIGURE 73: E-COMMERCE SCENARIO - ZONE MODELLING (STEP2)	102
FIGURE 74: AIRBUS SCENARIO 2 - BUSINESS VIEW	103
FIGURE 75: AIRBUS SCENARIO 2 - ARCHITECT’S VIEW	104
FIGURE 76: AIRBUS SCENARIO 2 - DESIGN VIEW - ZONE MODELLING	105
FIGURE 77: OUR GOAL IN IREHDO2 PROJECT	112
FIGURE 78: RISK TAXONOMY HIERARCHY	127
FIGURE 79: GRAPH CREATION APPROACH.....	128
FIGURE 80: GRAPH CREATION APPROACH INPUT AND OUTPUT FILES.....	129
FIGURE 81: OVERVIEW OF CRITERIA GRAPH.....	130
FIGURE 0-82: CRITERIA GRAPH EXAMPLE PATTERN	131
FIGURE 83: UTILITY FUNCTIONS EXAMPLE MAPPING	135
FIGURE 84: EXAMPLE CONTROL CAPABILITY / INTEGRITY LEVEL MAPPING	135
FIGURE 85: EXAMPLE CRITICALITY / INTEGRITY MAPPING	135
FIGURE 86: EXAMPLE TRUST/INTEGRITY LEVEL MAPPING.....	135
FIGURE 87: USE CASE SCENARIO 1 - BUSINESS VIEW + ARCHITECT’S VIEW	136
FIGURE 88: USE CASE SCENARIO 1 - DESIGNERS VIEW	137
FIGURE 89: AIRBUS SCENARIO 2 - SECURITY DOMAIN INFORMATION	138

List of Tables

TABLE 1: ELICITED EVALUATION CRITERIA	47
TABLE 2: VERIFICATION METHOD FOR $R^{M6.2}$	48
TABLE 3: INDIVIDUAL EVALUATION OF SECURE KAOS	52
TABLE 4: INDIVIDUAL EVALUATION OF STS	53
TABLE 5: INDIVIDUAL EVALUATION OF SEPP.....	55
TABLE 6: SAMPLE OF THE EVALUATION RESULTS.....	55
TABLE 7: STEP 1 INPUT – SAMPLE OF PERMITTED DATA FLOWS.....	85
TABLE 8: STEP1 - IVF RULES (USE CASE SCENARIO 1)	87
TABLE 9: STEP2 – ACF RULES (USE CASE SCENARIO 1)	89
TABLE 10: STEP2 – DATA FLOW INTEGRITY REQUIREMENTS (USE CASE SCENARIO 1).....	89
TABLE 11: E-COMMERCE SCENARIO - IVF RULES (STEP1).....	100
TABLE 12: EVALUATION RESULTS OF OUR PROPOSED SRE METHODOLOGY	106
TABLE 13: PRIORITIZING IVF REQUIREMENTS AT ACCOUNTABILITY SERVER.....	108
TABLE 14: E-COMMERCE SCENARIO - PERMITTED DATA FLOWS (STEP1).....	138
TABLE 15: ACCESS CONTROL FILTER REQUIREMENTS (STEP2).....	139
TABLE 16: DATA FLOW INTEGRITY REQUIREMENTS (STEP2).....	142

THESIS INTRODUCTION

Research Background and Motivation

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 against organizations [SANS 2017]. Thus, considering network security is extremely crucial for ensuring business continuity against the odds of growing threats.

Network security majorly concerns security architectural needs that describe network segmentation (i.e., security zoning); security of network devices connecting the communicating end user systems; and security of the information being transferred across the communication links [ISO/IEC 27033 2009]. The ultimate aim boils down to one common objective, which is to prevent illegitimate access to the data assets that carry vital information and influence the decisions relating business critical operations [SANS 2015a]. An appropriate design of the architecture provides many advantages (e.g., isolation of low trust systems, limitation of a security breach's scope, costs savings) [Mariusz Stawowski 2009].

Currently, there exists numerous network security controls (e.g., firewalls, intrusion detection and prevention systems), for implementing network security. Each time a security control (e.g., firewall or a proxy) is added or removed to the network, it will have impact on the application running on the network, on the quality of service of the network, and also incur costs related to its purchase and installation [Sherwood 2005]. For example, DMZ (de-militarized zone) security constraints the network architecture. It provides a perimeter defence by isolating the internal network from public network through set of firewalls and application proxies. 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. Studies show that the difference in the return of security investments while considered at early and late stages of the system development cycle, ranges from 12 to 21 percent [Hoo et al. 2001]. Therefore it is necessary to consider network security at earlier stages i.e. right from the planning stages of architecting networks. This perspective relates to the "Security by design" principle, which enforces the consideration of security right from the requirements analysis phase.

In this regard, network security requirements play a vital role since they impact the decisions related to the implementation of security controls with regards to business needs [ISO/IEC 27033 2009]. Indeed, bad network security requirements can lead to ineffective and costly security or worse security holes in the network security design. For this, an effective network security requirement engineering is needed to help organizations to capture cost-effective security solutions that protect 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 to capture the security requirements by analysing the business risk impact of potential threats. However, majority of existing SRE methods are confined to elicit software/system security requirements [Uzunov et al. 2012], which lack support to extend to the network security context [Rehman and Gruhn 2018]. For instance, they do not help specifying security zones (e.g., DMZ) relative to a business security risk impact, which is a preliminary step for security architects in capturing other network security requirements (e.g., related to data flows) [SANS 2015a; Government of Canada, Communications Security Establishment 2007]. In this thesis work, we propose an SRE method dedicated to network security requirements engineering at early stages, which aims towards “security by design” from the network security perspective.

Thesis Research Project IREHDO2

This thesis work is a part of the research project DGA IREHDO2 (Intégration REseau Haut Débit embarqué Optique 2ème phase) that concerns aircrafts future generation networks. In this project, our work is done mainly in collaboration with AIRBUS and is related to the security requirements engineering process for aircraft networks (Figure 1). Our objective in this project is to propose an SRE methodology for capturing and analysing network security requirements, and that facilitates the refinement into network security and monitoring configurations (TOP/DOWN approach). In addition, it should be possible to determine which high-level network security requirement is impacted when any inconsistency or anomaly is found in network security and monitoring configurations (BOTTOM/UP approach). It is to note that this thesis work deals with to network security requirements. The further refinement of the network security requirements into low-level network security device configurations is out of scope of this thesis work and is a collaborated work done in parallel.

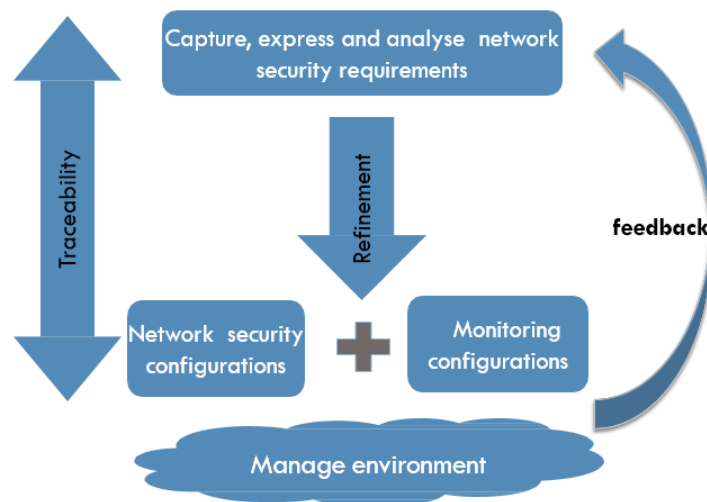


Figure 1: Our goal in IREHDO2 project

The SRE process relates to capturing, expressing and analysing network security requirements (i.e., high-level) pertaining to the business strategic as well as risk control objectives. This process is complex while compared to the capture and analysis of software/system requirements because networks principally act as an intermediary that interconnects all the software/systems within an enterprise. First, capturing network security requirements needs to consider the security problem context of all the stakeholders including the business analysts, and the security analysts of all systems connected to the network. Eventually, this will result in many interactions between numerous networked elements at varying levels of abstraction.

Next issue concerns the expression of the captured network security requirements relative to all these interactions. In particular, which language to use? Since different stakeholders are involved in the process, the language must fit their comprehensibility so that they find it at ease to describe the security problem context. Therefore, the language should consolidate different abstraction views suitable to each stakeholder. It should also express interdependencies between all the interconnected systems.

Finally, analysing network security requirements mainly deals with checking for inconsistencies and incompleteness. In particular, it is essential to determine which among the network security requirements are more important to the business. This relates to traceability issues, which require to link the network security requirements to the relative security information back to the business needs.

As a consequence, the anticipated SRE methodology for network security must facilitate involving all the stakeholders with different abstractions; to determine when to start considering network security requirements (i.e., from which abstraction level) and how they are refined from high-level business objectives.

Research roadmap and claimed contributions

The literature does not lack examples of SRE methods aiming at enhancing the quality and experience of the security requirements engineering process. The survey works [Fabian et al. 2010; Uzunov et al. 2012; Muñante et al. 2014; Souag et al. 2015] present a variety of SRE methods from the last decade. Conventionally, these SRE methods are categorized into different classes of SRE approaches such as goal-oriented, agent-oriented, problem frames and UML based modelling. Choosing or adapting an SRE approach to the network security context raises our first research question in this thesis work:

- **RQ1:** How to evaluate SRE methodologies in a network SRE context?

Each SRE approach varies in the philosophy of how to capture, express and analyse security requirements. For instance, an SRE [Van Lamsweerde et al. 2003] referring to goal modelling captures the security requirements with reference to the unwanted malicious behaviours of the system, expressed as anti-goals. Whereas an agent modelling inspired SRE [Massacci et al. 2010; Salnitri et al. 2015] captures security requirements with reference to the security constraints over the dependency interactions between the system agents. A problem frames based SRE [Hatebur et al. 2007a; Lin et al. 2003] captures security requirements with reference to the security problem environment constraints of

the system. Finally, SRE methods [Lodderstedt et al. 2002; Jürjens 2002] based on UML modelling capture security requirements with reference to the security design constraints of the system.

Given this SRE methods diversity, selecting the best suitable method to capture network security requirements seemed to be a challenging task. Although, the existing survey works [Fabian et al. 2010; Uzunov et al. 2012; Muñante et al. 2014; Souag et al. 2015] provide both an evaluation and an analysis of SRE methods to some extent, their evaluation was often based on ad-hoc criteria. Moreover, they varied in the evaluation perspectives [Karpati et al. 2011].

We propose an SRE evaluation methodology that differentiates its strategy from previous comparative studies for two reasons. First, we involve the security experts that are the SRE end-users in the whole process. 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 target system is the good SRE methodology, labelled as SRE-methodology-to-be and the evaluation criteria are indeed the requirements of the SRE-methodology-to-be.

Based on this idea, our evaluation methodology is built on the classical idea of requirements engineering approach. Our evaluation process subsumes three steps. First, we identify the problem context and elicit initial high-level characteristic goals. The latter represent the generic characteristics of good security requirements. However, the literature review on the quality characteristics revealed that there is no consensus on what a good requirement is. We propose a weaving methodology that provides a consolidated list of 20 criteria. Secondly, we refine the high-level characteristic goals into final requirements of the SRE methodology-to-be. This step specializes the generic high-level goals to a specific problem context by involving the stakeholders, i.e., the security requirement engineers. Finally, the last step deals with the evaluation of the existing SRE methodologies using the elicited requirements.

We implemented this methodology to elicit criteria for evaluating the three distinct SRE approaches in the network security context of the IREHDO2 project. We evaluated a method from each group, Secure KAOS i.e., goal-oriented, Socio Technical Security (STS) i.e., agent-oriented and Security Engineering Process using Patterns i.e., problem frames, which highlighted that they provide no support to either capture or express or analyse network security requirements. This problem derives our second research question in this thesis work:

- **RQ2:** How to define a good SRE methodology that facilitates the network security requirements elicitation and refinement i.e., from high-level business objectives into low-level network security zoning requirements.

We propose a layer based SRE methodology to verify the refinement of network security requirements from business security objectives. This approach is inspired from the SABSA [Sherwood 2005], a business risk driven enterprise security architecture development methodology framework, which defines a structured model.

The proposed conceptual model consists in three layered views of abstractions, see Figure 2. Each layered view represents the SRE context of the people (a.k.a. stakeholders) working at different

abstraction levels during a security engineering process. The *business view* deals with the elicitation of high level security objectives and constraints that are important to do business. The *architect's view* concerns the elicitation of high-level security objectives in terms of protecting the security entities involved in the fulfilment of the business objectives. We express the elicitation and refinement at these first two views using STS. In addition, to facilitate the integration of risk analysis concepts, we propose a new extension of STS called Anti-STS for modelling attacks as multi-agent systems. The *business view* and *architect's view* together correspond to the strategy and planning phases of the system security engineering process. The *Designer's view* reflects the early phases of the network security design process.

The elicitation activity at the *Designer's view* is narrowed down to network security context, where high-level network security requirements are captured. For this, we propose a methodology that automates parts of the process of security zones and network security requirements elicitation using a definite set of formalized rules derived from the Clark-Wilson lite formal security model [Shankar et al. 2006]. We implemented the rules in Answer set programming (ASP [7]) and provided a tool that facilitates calculating cost-optimal security zone models. We integrate the concepts of goal modelling (i.e. Secure KAOS) and agent modelling approaches (i.e., STS) to link these three abstraction layers.

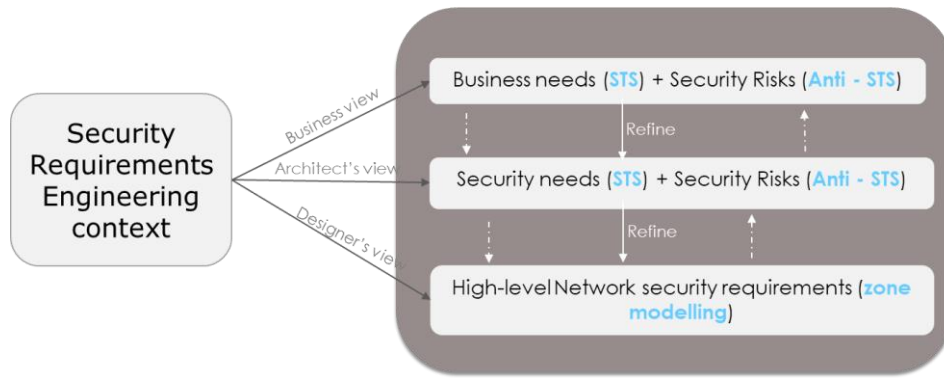


Figure 2: Layer based SRE methodology overview

The implementation of the proposed SRE approach is described using two use case scenarios from the IREHDO2 project, plus an e-commerce enterprise network case study. In this process, we show clearly how the elicited information at each abstraction layer are captured and linked so that the traceability feature is visibly justified.

Thesis outline

The thesis dissertation is structured into four chapters as follows:

Chapter 1 presents a state of the art of the underlying concepts related to security requirements engineering and security risk management. Then it provides a review of three distinct and widely known SRE methodologies i.e., STS, Secure KAOS and SEPP.

Chapter 2 describes the evaluation of the three SRE methodologies using our RE based evaluation methodology, which concerns the study of RQ1. First we describe the elicitation of evaluation criteria

with respect to the network SRE context of IREHDO2. Then we brief the evaluation analysis that includes the pros and cons of the three methodologies based on the feedback from security experts. This chapter consolidates the publications [Bulusu et al. 2016; Bulusu et al. 2017a; Bulusu et al. 2018b; Bulusu et al. 2018a].

Chapter 3 presents the integrated concepts and modelling notation of our proposed SRE methodology using the two use case scenario related to the IREHDO2 context. The *business* and the *architect's* views describes the integration of both STS and our proposed anti-agent modelling notations [Bulusu et al. 2017b] . The *design* view concerns the presentation of our proposed zone modelling methodology [Bulusu et al. 2018c].

Chapter 4 presents the evaluation and analysis on the performance of our SRE methodology using the evaluation criteria defined in chapter 2. For this, we implement an e-commerce enterprise networks SRE use case scenario [Cybedu 2017], which includes more network agents and interactions similar to a real time scenario. We could not test all the criteria. The analysis results are drawn from the initial feedback from the security experts, which proves supportability to network SRE context.

In the end, we conclude the thesis work with a discussion on the limitations, subsequent work for improving the methodology followed by our future perspectives of research.

Publications

The comprehensive list of publications resulted from this thesis work are as follows:

- Romain Laborde, Sravani Teja Bulusu, Ahmad Samer Wazan, François Barrère, Abdelmalek Benzekri. *Logic-based methodology to help security architects in eliciting high-level network security requirements*, International Conference on Emerging Security Information, Systems and Technologies (ACM SAC 2019), (to appear).
- Sravani Teja Bulusu, Romain Laborde, Ahmad Samer Wazan, François Barrère, Abdelmalek Benzekri. *A Logic-Based Network Security Zone Modelling Methodology*, International Conference on Emerging Security Information, Systems and Technologies (SECUREWARE 2018), p 67-72..
- Sravani Teja Bulusu, Romain Laborde, Ahmad Samer Wazan, François Barrère, Abdelmalek Benzekri. *A Requirements Engineering-based Approach for evaluating Security Requirements Engineering Methodologies*, International Conference on Information Technology: New Generations (ITNG 2018), p 517–525.
- Sravani Teja Bulusu, Romain Laborde, Ahmad Samer Wazan, François Barrère, Abdelmalek Benzekri. *Applying a Requirement Engineering Based Approach to evaluate the Security Requirements Engineering Methodologies*, ACM Symposium on Applied Computing – Requirement Engineering track (ACM SAC 2018), p 1316-1318.
- Sravani Teja Bulusu, Romain Laborde, Ahmad Samer Wazan, François Barrère, Abdelmalek Benzekri. *Describing Advanced Persistent Threats Using a Multi-agent System Approach*, Cyber Security in Networking Conference (CSNet 2017), p1-3.

- Sravani Teja Bulusu, Romain Laborde, Ahmad Samer Wazan, François Barrère, Abdelmalek Benzekri. *Which Security Requirements Engineering Methodology Should I Choose?: Towards a Requirements Engineering-based Evaluation Approach*, *International Conference on Availability, Reliability and Security (ARES 2017)*, p 1-6.
- Sravani Teja Bulusu, Romain Laborde, Ahmad Samer Wazan, François Barrère, Abdelmalek Benzekri. *Towards the weaving of the characteristics of good security requirements*, *International Conference on Risks and Security of Internet and Systems (CRISIS 2016)*, p 60-74

CHAPTER 1 SECURITY REQUIREMENTS ENGINEERING

Security requirements engineering (SRE) is a fundamental process for building secure systems. It deals with activities such as gathering, analysing and evaluating security requirements. They act as the baseline source for defining security specifications and configurations. This chapter is dedicated to present the concepts related to security requirements engineering.

We first provide an overview of security requirements (section 1.1). Then we present the concepts and background of security risk management (section 1.2). Next, we present the steps involved in security requirements engineering process followed by the state of the art of security requirements engineering (section 1.3). In this aspect, we focus on the underlying concepts and SRE modelling notations of three SRE methods Secure KAOS, STS and SEPP, which correspond to three distinct SRE approaches: goal-oriented, agent-oriented, problem oriented (section 1.4). We end this chapter with the discussion on network security zoning that helps in network security requirements analysis at early stages (section 1.5).

It is to note that this chapter does not include the comparative analysis of these methods. A detailed comparison and evaluation of these methods with reference to network security requirements engineering will be presented in the following chapter following the presentation of our research method for evaluation study.

1.1 Security requirements, a global view

A requirement is a familiar term that is omnipresent when building a system. It basically expresses a statement that translates or specifies a need and its associated constraints and conditions [ISO29148:2011]. In the system development context, Jackson [Jackson 2001] describes a requirement as an expected behaviour of a system component in a given problem context. Security requirements describe the anticipated functional and non-functional (i.e., quality) behaviour of systems to ensure the protection of assets from security threats [OSA 2016]. Wherein, an asset is an entity (e.g., human, software/hardware, information, system component, network device) that holds some value to business continuity [ISO31010 2009]. Security threats are the unwanted incidents that can inflict harm or a damage to business assets with unacceptable consequences [NIST 800–160 2014]. Therefore, a secured behaviour of a system is contextualized with the knowledge on what should be protected (i.e., assets) from what/whom (i.e., threats) and to what extent it requires protection i.e., degree [Fabian et al. 2010].

From a broad view, the security behaviour is determined with reference to the business security objectives that aim at preserving the CIA triad security properties i.e., confidentiality, integrity and availability. These security objectives are realized by security services list , which refers to

confidentiality, integrity, availability, authenticity, accountability, non-repudiation and reliability [ISO/IEC27000 2018]. Security control mechanisms (e.g., data encryption, firewall), otherwise called security solutions are the countermeasures that helps to prevent, detect, counteract or minimize security risks that hold potential threat to business security objectives [NIST 2013]. Wherein, a security control mechanism can address one or more security objectives. Conventionally, the security objectives and the control measures are traditionally captured through a process called elicitation [Mead and Stehney 2005]. It includes brainstorming sessions that involves stakeholders (e.g., business users, security architects, network security engineers) who claim to have interest on business system assets and associated security risks.

A risk corresponds to an unexpected future event, which hampers the objectives [ISO31010 2009]. Security risks always pertain to an asset under threat. ISO 13335 [ISO 13335 2004:2004] defines security risk as the potential that a given threat will exploit vulnerabilities (i.e., exploitable weakness) of an asset or group of assets and thereby cause harm to the organization. Security risk management team deals with the tasks of identifying and assessing security risks pertaining to business assets and proposing cost-effective security solutions (i.e., mitigations), which takes the form of security requirements. Thus, security requirement engineering and risk management practices are interdependent. They together drive the transformation of business security objectives into security requirements.

In practice, security requirements evolve iteratively during the development cycle of system that concerns with planning, designing and developing secure systems. In this process, different levels of abstraction will often be necessary to address the full range of stakeholders' objectives and security risks [ISO29148:2011] since every system element that is part of a critical business operation is eventually subjected to security protection.

1.2 Security Risk Management

Security Risk Management (SRM) literature is rather extensive and encompasses risk analysis techniques for identifying, assessing and prioritizing security risks (e.g., Attack trees [Schneier 1999], Hazop [Kletz 1999], FMEA/FMECA [Stamatis 2003]) and risk management methods for implementing risk management process related activities that involves planning, coordinating risk analysis activities (e.g., EBIOS [ANSSI 2010], CRAMM [CCTA 2001], CORAS [Lund et al. 2010]). International standards such as ISO/IEC 3100:2009, ISO/IEC 2700x and NIST provide guidelines and best practices for effective risk analysis and management activities. Figure 3 provides a summary view of the RM literature.

Risk being a future event, it is defined based on the likelihood (i.e., frequency of risk event occurrence) and impact of the risk (i.e., $\text{risk} = \text{likelihood} \times \text{impact}$) [ISO31010 2009]. A qualitative risks analysis expresses risk impact probabilities using qualitative scales (such as Low, Medium, High), which reflect subjective analysis with consensus. Quantitative risk analysis approach articulates risks in numerical terms, i.e. expected monetary loss and probability (e.g. annual loss expectancy, ALE).

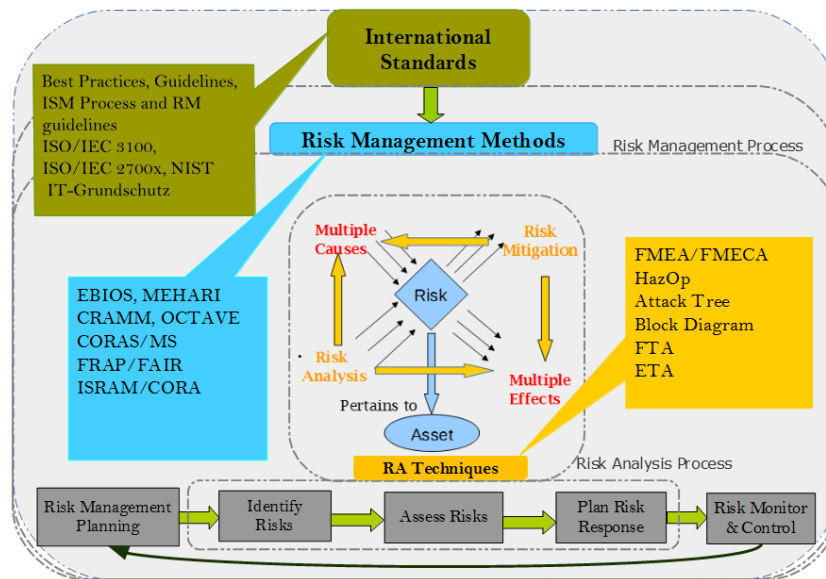


Figure 3: Risk Management global view

Furthermore, a risk analysis can also vary based on the following two strategies i.e., asset-based and threat/vulnerability based approaches [Hogganvik 2007]. Asset-based risk analysis approaches (e.g., FMEA/FMECA and Hazop) emphasize on first identifying the critical assets to business, and then assessing the potential threats that could harm or damage those assets. Threat-based risk analysis approaches (e.g., attack trees) emphasize on focus on addressing risks pertain to assets against most deadly and wide spread threats. These approaches rely on informational sources which provide detailed information on existing threats and associated list of risk events. These informational sources can be referred to threat catalogues/repositories and threat models (e.g., BSI threat catalogue [BSI 2012]; Microsoft threat model[Microsoft 2016]; OWASP top web application security risks [OWASP 2017]).

1.2.1 Risk analysis techniques

HazOp (hazard and operability) [IEC61882 2016], is an asset-based and qualitative risk analysis technique built on structured brainstorming. It helps to identify system hazards or operational risks based on set of guidewords. For example, guidewords such as EARLY, LATE, BEFORE, AFTER, could refer to deviations related to time dependent operations. The choice of suitable guidewords will strongly influence the success of the HAZOP in detecting design faults and operability problems. International standard [IEC61882 2016] describe the implementation of this technique.

FMEA/FMECA [Stamatis 2003], Failure Mode Effect/Criticality Analysis, is an asset-based RA approach. FMEA provides support to identify and assess potential failures of systems/sub-systems/individual components at early stages of development. The identified modes are classified according to their criticality, expressed using a qualitative indicator called Risk Priority Number (RPN). RPN is computed from the product of three factors Severity(S), Occurrence (O), and Detection (D) (i.e., $RPN = S * O * D$). Severity defines an extent to which a failure mode can affect the system (seriousness of adverse effects). Occurrence corresponds to the frequency rate of occurrence of failure modes (likelihood). Detection corresponds to the rate at which these failure modes are soon identified. The three factors are defined with qualitative subjective values (e.g., 10-point scale, where 1 is the least and

10 is the highest, see Figure 4). FMECA, Failure Mode Effect and Criticality Analysis, extends FMEA to quantitative criticality analysis, which specifies probability of device failure modes.

Example of a risk calculation by FMEA.

	Severity (S)	Occurrence (O)	Detection (D)	RPN=S*O*D	
Potential failure 1	2	10	5	100	
Potential failure 2	10	2	5	100	
Potential failure 3	2	5	10	100	
Potential failure 4	10	5	2	100	

Prioritize failures

Priority 1 Potential failure 4
Priority 2 Potential failure 2
Priority 3 Potential failure 1
Priority 4 Potential failure 3

Figure 4: FMEA risk calculation example [Stamatis 2003]

FTA/ETA (Fault/Event Tree Analysis) [Høyland and Rausand 2009], are threat-based risk analysis approaches used for reliability analysis. FTA helps to deduce the logical relationship of possible causes that result in unwanted events. The top node represents an unwanted incident or failure and the different events that can cause the failure, which are represented as intermediate nodes and leaf nodes. [IEC 61025 2006] provides guidelines on FTA implementation. ETA complements FTA to deduce possible consequences (adverse effects) associated with an unwanted incident. It uses binary notation (True/false) in order to represent the success and failure of an unwanted incident (event) with respect to the corresponding mitigation controls.

Cause and consequence diagram [Nielsen 1971] is a graph based model which combines the features of both Fault and Event trees to provide a complete view of cause and effect analysis, see Figure 5 (taken from [Hogganvik 2007]). On the left is the FTA, where the possible causing temporal events that could result in failure of a light were identified. On the right is ETA, where the probability consequences are shown with respect to the success and failure of risk event (i.e., light fails).

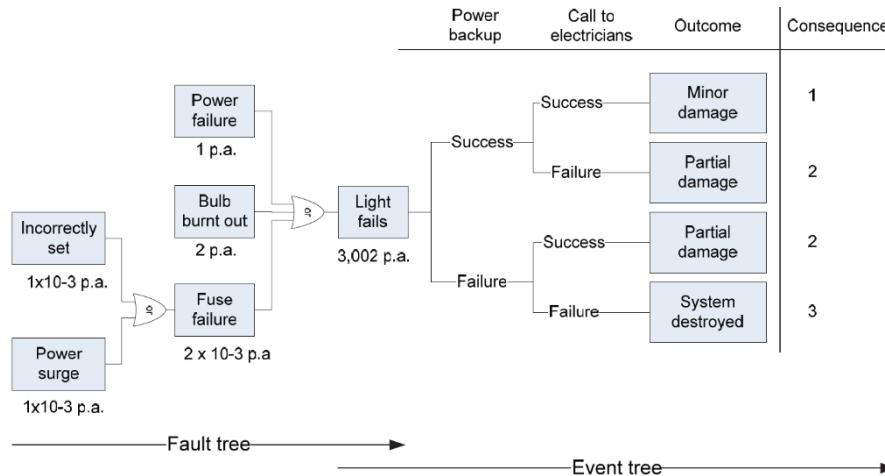


Figure 5: Cause-Consequence diagram [Hogganvik 2007]

Attack Trees [Schumacher 2003] are threat-based risk analysis approaches conceptually similar to fault trees (FTA), but are used to describe possible attack scenarios in a hierarchical decomposition using AND/OR constructs. The root node represents the attacker goal and the decomposed nodes represent the different ways to achieve the goal. Figure 6 shows an example of an attack tree to guess administrator Password. An attack tree shows a logical breakdown of the various options available to an adversary (i.e., attacker). By performing the exploits associated with one or more leaf level events

which have been carefully selected to satisfy the tree's AND/OR logic, the attacker can achieve the root level goal. Each minimal combination of leaf level events is known as an attack scenario (highlighted in dashed boxes).

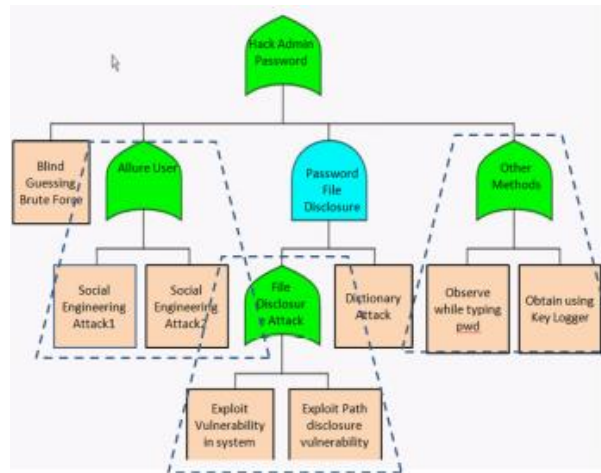


Figure 6: Attack Tree: Hack Admin password

ADTree (attack defence tree) [Kordy et al. 2011] focuses on modelling defensive strategies. Unlike cause and effect diagrams, ADTrees intend to project a combined view of possible attack scenarios and corresponding applicable mitigation controls and therefore they include attack nodes as well as defence nodes. Figure 7 depicts the example of an ADTree for an attack on a bank account.

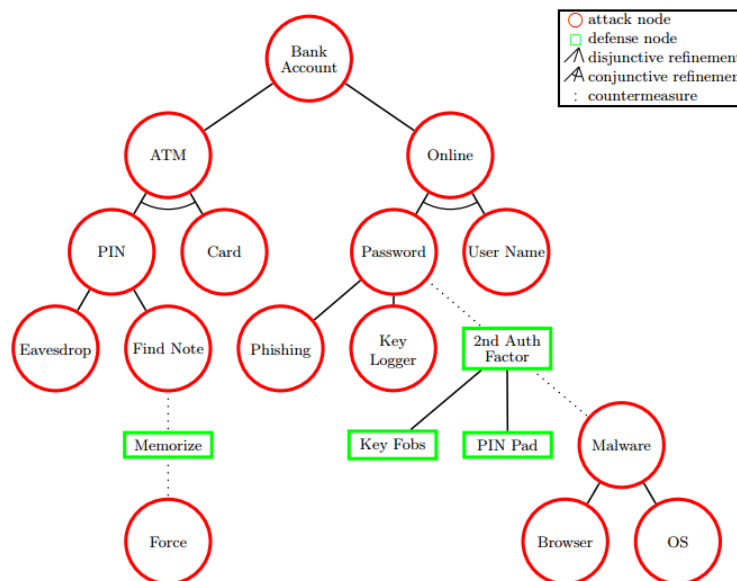


Figure 7: ADTree example: An attack on a bank account [Kordy et al. 2011]

Haruspex [Telmon and CIsa 2012] is a simulation driven asset-based risk evaluation technique. It evaluates the probability that an intelligent threat agent could successfully implement a multi-step attack against a system that results in an impact, e.g., a loss for the owner of the systems. Threat agent simulation using Haruspex is driven by attack graphs [Phillips and Swiler 1998], which helps to represent multiple attack paths. The nodes represent events and the edges represent the attack paths, see Figure 8. Haruspex methodology makes use of Monte Carlo method [Rubinstein and Kroese 2011]

attack graphs in order to define the complexity of multi-step attack as well as to determine the probability of exploiting the vulnerabilities to reach the attack goal.

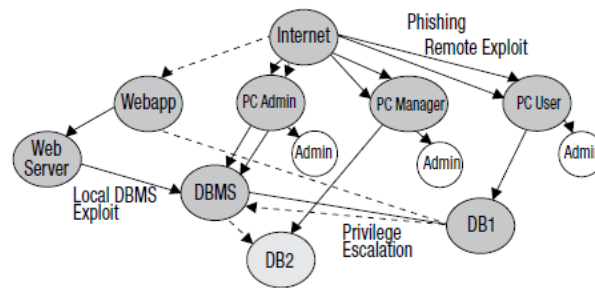


Figure 8: attack graph [Telmon and CIsa 2012]

Bayesian Network(BN) [Weber et al. 2012], is a graphical and probabilistic model based on the direct acyclic graph. Similar to the fault trees, the Bayesian networks provide the possibility of potential causes for a system failure, but based on the probability distributions. Nodes represent the variables or factors relevant to the problem and the edges represent the statistical dependencies between the nodes. Figure 9 illustrates an example of BN acyclic graph for system failure based on the conditional probability of occurrence of two conditions, i.e. buffer overflow and denial of service.

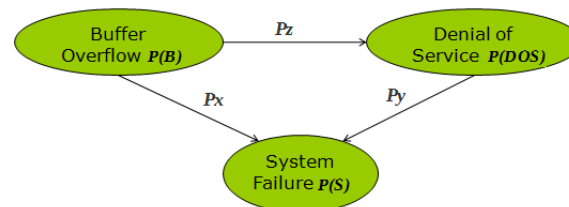


Figure 9: BN for system failure

The edges P_x defines the probability of buffer overflow causing system failure, P_y corresponds to the probability of system failure caused by denial of service and P_z is the probability of buffer overflow resulting in denial of service. Subsequently, the probable scenarios of system failure can be deduced as: (1) System failure occurred either due to Buffer Overflow or Denial of Service; (2) Buffer Overflow can result in either DOS or System Failure or both, (3) DOS can cause System Failure.

Markov Analysis [Høyland and Rausand 2009] is a stochastic mathematical analysis technique particularly applied to systems that execute probabilistic movement from one state (or condition) to another over time. For example, Markov analysis can be used to determine the probability that a service is available one day and becomes unavailable the next day due to some technical error or some vulnerability.

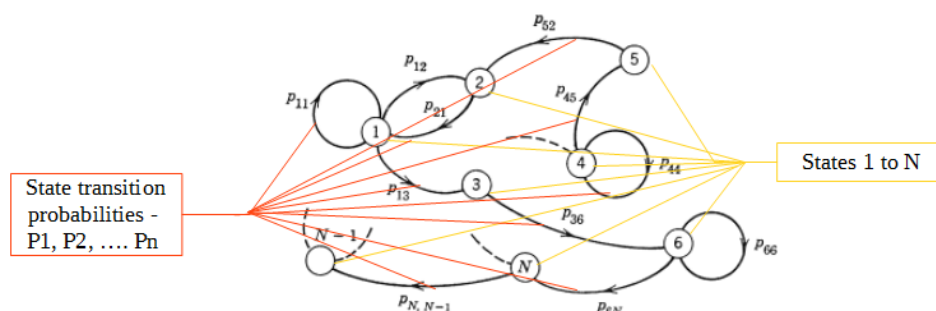


Figure 10: Markov Chain [Høyland and Rausand 2009]

Markov chains have a finite number of states, after each time step, there is the chance of state transition which is calculated as the probability of state transition (represented as P_1 , P_2 , P_n in Figure 10). IEC 61165 [IEC 61165 2006] provides guidelines on application of Markov techniques to model and analyse the reliability and maintainability of systems. However, this technique is mostly preferred to use to analyse smaller systems with strong dependencies requiring accurate evaluation since the model can get complex for large systems [Høyland and Rausand 2009].

1.2.2 Risk management methods

Several RM methods have been proposed in the past with principal aim to support identification and assessment of security threats and risks as well as to determine the risk control requirements (see Figure 3). Some example methods include:

CORAS [Lund et al. 2010; Solhaug and Stølen 2013] is a European project that defined a customized graphical modelling language to enhance effective communication and documentation of threat & risk scenarios based on ISO31000 standard;

EBIOS (Expression des Besoins Identification des Objectifs de Sécurité) [ANSSI 2010] has been developed by ANSSI in France and provides a tool based risk assessment support for governmental and commercial organizations that handle confidential or classified information;

FAIR (Factory Analysis of Information Risk) is a risk management methodology framework developed by Open Group that defines a taxonomy of risk attributes (e.g., Loss event frequency, control strength), see Appendix A for more details.

CRAMM (CCTA Risk Analysis and Management Method) [CCTA 2001] was developed by British Government agency CCTA (Central Computer and Telecommunications Agency) and provides a tool support to facilitate qualitative analysis thereby enabling effective brainstorming that involve qualified and experienced practitioners.

OCTAVE (Operationally Critical Threat, Asset, and Vulnerability Evaluation)[Marek and Paulina 2006], developed by CERT, is qualitative risk management methodology that enables operational security risk evaluation for each group of assets and threats. It is performed through self-organized brainstorming meetings that include security experts in organization.

These RM methods employ one or more risk analysis techniques i.e., either qualitative/quantitative or asset-/threat-based approaches, for estimating the risk impact. For instance, CORAS risk assessment methodology [Lund et al. 2010; Solhaug and Stølen 2013] is built on HazOp analysis [IEC61882 2016], Fault Tree Analysis (FTA) [IEC 61025 2006], (FMECA) [Stamatis 2003], as well as CRAMM [CCTA 2001].

1.3 Security requirements engineering process

A requirements engineering process subsumes the set of activities that take care of formulating requirements by carefully gathering information related to the problem context, expected behaviour and

objects in the problem world from three perspectives: WHAT, WHY, WHO [Van Lamsweerde 2009]. While, the WHAT perspective helps to study the nature of the required behaviour; the WHY perspective helps to study the rationale behind the necessity of the required behaviour; finally, the WHO perspective helps to study the nature of the entities on which (whom) the required behaviour is imposed. From a security point of view, the required behaviour concerns the protection objectives of business assets, imposed on secure systems. Figure 11 depicts an overview of conventional RE-cycle.

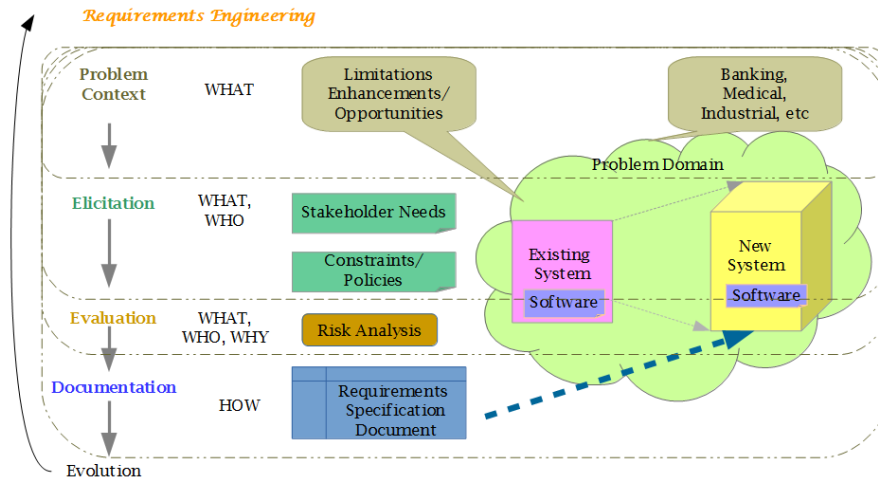


Figure 11: RE life-cycle

The main steps involved in RE life-cycle are as follows [Van Lamsweerde 2009] :

- **Problem context analysis:** A problem unstated is a problem unsolved [Ross and Schoman 1977]. Therefore, this step is dedicated to explore the ‘WHAT’ dimension of security problem. Identifying stakeholders and knowledge acquisition is one of the challenging tasks in this step [ISO29148:2011].
- **Requirements Elicitation:** This step deals with the process of capturing needs and objectives of the stakeholders. There exists various elicitation techniques such questionnaires based interviews, structured interviews, passive and active study of environment, which enable to explore the ‘WHAT’ and ‘WHY’ dimensions of their needs [ISO29148:2011].
- **Requirements Evaluation:** This step is the core of whole RE process, which conforms the evaluation requirements for their consistency, correctness and completeness. It helps to explore the ‘WHAT’ and ‘WHY’ dimensions. These aspects correspond to the quality characteristics of good requirements (e.g., complete/correct/ consistent/feasible etc.) [ISO29148:2011]. However, there exists no standard set of characteristics (we discuss this issue in chapter 2).
- **Requirements Documentation:** This step deals with structured and detailed documentation of the elicited and evaluated requirements, which describes the ‘HOW’ dimension of addressing stakeholder needs. There exists wide range of techniques and models to support this process such as natural language templates, diagrammatic notations, formal models [Van Lamsweerde 2009]. These requirement models are the essential components to document requirements for communication, inspection, negotiation, and their evolution.

1.3.1 SRE process models

SQUARE (System Quality Requirements Engineering) [Mead and Stehney 2005] defines a RE process model for eliciting categorizing, and prioritizing security requirements during the early stages of the development life cycle of information systems and applications. This methodology assumes the participation of experts from SRE as well as SRM teams.

Num.	Step	Input	Techniques	Participant	Output
1	Agree on definitions	Candidate definitions from IEEE and other standards	Structured interviews, focus group	Stakeholders, requirements team	Agreed-to definitions
2	Identify security goals	Definitions, candidate goals, business drivers, policies and procedures, examples	Facilitated work session, surveys, interviews	Stakeholders, requirements engineer	Goals
3	Develop artifacts to support security requirements definition	Potential artifacts (e.g., scenarios, misuse cases, templates, forms)	Work session	Requirements engineer	Needed artifacts: scenarios, misuse cases, models, templates, forms
4	Perform risk assessment	Misuse cases, scenarios, security goals	Risk assessment method, analysis of anticipated risk against organisational risk tolerance, including threat analysis	Requirements engineer, risk expert, stakeholders	Risk assessment results
5	Select elicitation techniques	Goals, definitions, candidate techniques, expertise of stakeholders, organisational style, culture, level of security needed, cost benefit analysis, etc.	Work session	Requirements engineer	Selected elicitation techniques
6	Elicit security requirements	Artifacts, risk assessment results, selected techniques	Joint Application Development (JAD), interviews, surveys, model-based analysis, checklists, lists of reusable requirements types, document reviews	Stakeholders facilitated by requirements engineer	Initial cut at security requirements
7	Categorize requirements as to level (system, software, etc.) and whether they are requirements or other kinds of constraints	Initial requirements, architecture	Work session using a standard set of categories	Requirements engineer, other specialists as needed	Categorized requirements
8	Prioritize requirements	Categorized requirements and risk assessment results	Prioritization methods such as Triage, Win-Win	Stakeholders facilitated by requirements engineer	Prioritized requirements
9	Requirements inspection	Prioritized requirements, candidate formal inspection technique	Inspection method such as Fagan, peer reviews	Inspection team	Initial selected requirements, documentation of decision making process and rationale

Figure 12: SQUARE methodology

The whole process of SQUARE methodology is decomposed into 9 steps summarized in Figure 12. The first four steps include activities similar to the initial planning activities of risk management, while next 5 steps correspond to requirement engineering activities. Each step is defined with input and output criteria of necessary information, major participants (i.e., stakeholders) and suggested techniques for implementing RE process.

SREP (security requirements engineering process) [Mellado et al. 2007] extends SQUARE method by integrating it with the security evaluation process given in Common Criteria (CC) [ISO/IEC 15408]. CC provides guidelines for specifying security requirements and allows developers to specify common security attributes known as Target of Evaluation (TOE), for evaluating security functionality of IT products. In the SREP process, the evaluation of security requirements is done using CC assurance requirements and the Systems Security Engineering Capability Maturity Model (SSE-CMM). Figure 13 illustrates a brief view of main activities defined as 9 steps in SREP process, similar to SQUARE and these steps are iterated at each phase system development.

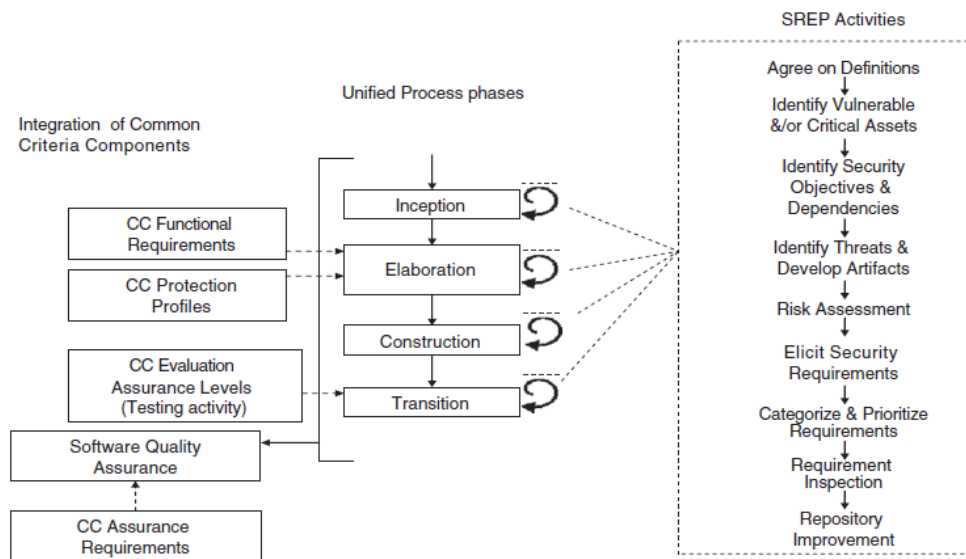


Figure 13: SREP process

1.3.2 SABSA (Sherwood Applied Business Security Architecture) Framework

SABSA (Sherwood Applied Business Security Architecture) [Sherwood 2005] is a business-driven enterprise security architecture framework intended for guiding the security architects. It defines a six-layered architecture model, with each reflecting the respective stakeholder views in the process of specifying, designing, constructing and building the business enterprise architectures. Figure 14, depicts an abstract view of SABSA layered model.

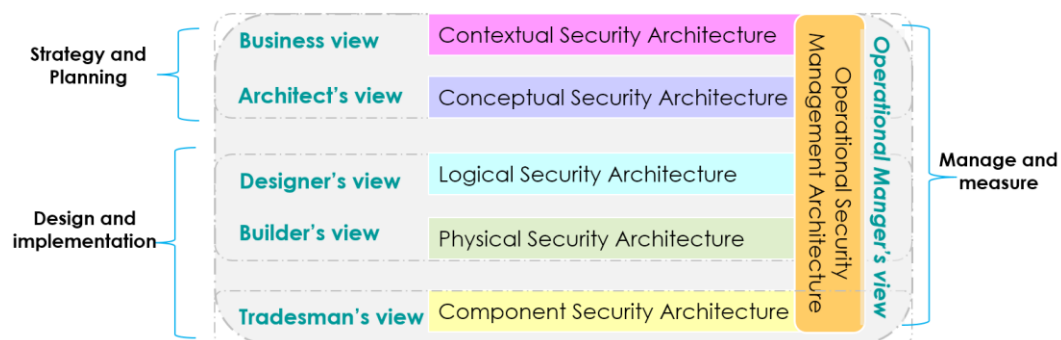


Figure 14: SABSA layered model [Sherwood 2005]

The *Business view* and *Architect's view* together correspond to the strategy and planning phases of 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 analyse 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 with the integration of protection objectives (e.g., confidentiality, integrity and availability of information) that are necessary to secure the business security needs. It is essential to integrate the defence in depth strategies to minimize security risks. This view includes the modelling of the security threats as well as the threat control objectives that drive the designing of security architecture for the system.

The *Designer's view* and *builder's view*, together, correspond to the designing phases of 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. It is essential to specify the logical elements of the secure system in order to express the business security assets. This view facilitates to model the business requirement needs in terms of logical system entities that have identity, meaning, function and structure but no physical embodiment. The *builder's view* deals with the mapping of technical security mechanisms (e.g., firewalls, hashing etc.,) to the logical security services. It is essential to define the functional and technological needs and constraints with respect to the physical system elements of the secure system. This view deals with assembling of the physical system entities that can be tangible with the technical aspects such as size, performance, capacity, and throughput.

Finally, *tradesman's view (LV5)* corresponds to the final implementation phase of system security engineering process. It deals with the selection of the tools and components for the construction and assembling of physical system elements. It is essential to consider the project specific technology needs, demands and feasibility constraints in order to accurately implement the business security system. This view facilitates to specify the business requirements needs in terms of a final set of atomic security requirements and device configurations. The latest view called *operational manager's view* corresponds to the maintenance of the system security engineering process after the deployment.

Although SABSA is not strictly a security requirement engineering approach, it provides a broad picture to understand the conventional security requirements engineering process from the perspective of business system security architecture. SABSA matrix (see Figure 15) describes a wide variety of aspects that a modeller should be able to express with the help of six interrogative questionnaires (what/why/how/who/ where/when).

These six interrogatives correspond to the different perspectives of the stakeholders at a given view. In brief, the '*What*' interrogative helps in capturing the security requirement needs of the stakeholders that correspond to the activities involved in respective phase of system engineering process. These needs are expressed in form of assets as they must be protected. The '*Why*' interrogative deals with the reasoning behind the motivation for wanting to demand security. These are expressed in terms of security risks pertaining to business needs. The interrogatives '*How*' and '*Who*' are related to capturing the process tasks goals and functions as well the respective roles and responsibilities that are needed to fulfil the business security needs. Finally, the interrogatives '*Where*' and '*When*' help to capture the

dependency constraints such as temporal and geographical location constraints confining to the business security needs.

	ASSETS (What)	MOTIVATION (Why)	PROCESS (How)	PEOPLE (Who)	LOCATION (Where)	TIME (When)
CONTEXTUAL ARCHITECTURE	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 Management 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 15: SABSA Matrix

Altogether, the layers and matrix facilitates in tracing every requirement need to the components that provide a solution. It also helps in providing business justification. For instance, when some asks ‘why a particular security solution is proposed?’ the rationale will be easier by tracing back to the business requirements that drive the specific solution. However, SABSA does not provide any modelling support like SQUARE in order to integrate to an SRE process.

1.4 Security requirements engineering approaches

Security Requirement engineering (SRE) discipline features a number of SRE methods with increasing focus on integrating security risk analysis practices right from earlier stages [Dubois et al. 2010]. The underlying motivation of these methods to improve the effectiveness of SRE process thereby reducing the requirement errors.

Conventionally, the existing methods in the SRE literature are categorized into different families of SRE approaches that vary in philosophies on how to capture i.e., elicit, evaluate and document security requirements. In here, SRE approach being a word used to differentiate the underlying common SRE modelling concepts employed by the SRE methods [Fabian et al. 2010]. We mainly recognize goal-oriented, agent-oriented, problem oriented, process oriented and UML based SRE approaches [Souag et al. 2015; Fabian et al. 2010].

UML based SRE methods such as secure UML [Lodderstedt et al. 2002], and UML sec [Jürjens 2002] facilitates to express access control requirements using some standard constructs based on UML class diagrams. Similarly, abuse cases [McDermott and Fox 1999] and misuse cases [[Sindre and Opdahl 2001] extend use case diagrams to analyse malicious behaviour of users. However, these methods focus SRE modelling in later stages of SRE process (i.e., at design phases) [Dubois et al. 2010].

On the other hand, SQUARE [Mead and Stehney 2005] and SREP [Mellado et al. 2007] (mentioned in previous section 1.3) correspond to process oriented approaches that define a systematic procedure to implement SRE process. However, they provide no formal modelling support for implementing SRE process [Souag et al. 2015; Fabian et al. 2010].

As consequence, in this thesis, we principally focused on the three SRE approaches i.e., Goal-oriented [van Lamsweerde 2004; van Lamsweerde 2004], agent-oriented [Massacci et al. 2010; Salnitri et al. 2015; Paja et al. 2014] and problem frames based [Hatebur et al. 2007a; Lin et al. 2003]. These three SRE approaches propose varying formal modelling concepts to capture security requirements at early stages.

1.4.1 Goal-oriented approaches

Goals represent high-level objectives of stakeholders that correspond to the development of the future system (i.e., system-to-be). These goals play crucial role in system engineering as they capture the reasons behind WHAT and WHY aspects of system. These approaches mainly focuses on refining goals into sub-goals until each sub-goal is allocated to an agent who believed to be responsible for fulfilling those goals [Van Lamsweerde 2009]. A goal assigned to an agent is called a requirement.

Several goal-oriented SRE methods exist in the literature offering a variety of procedures for reasoning about goals. Some examples are NFR [Chung et al. 2012], KAOS [Van Lamsweerde 2009], GRL [Liu, L and Yu, E; Amyot et al. 2009] and GBRAM [Anton 1996]. KAOS SRE method is the first in line that introduce formal goal modelling concepts based on temporal logic and therefore it is widely recognized in SRE literature [Fabian et al. 2010]. In this thesis work we study Secure KAOS, a security extension to KAOS.

1.4.1.1 KAOS/Secure KAOS

KAOS (Knowledge Acquisition automated Specification or Keep All Objectives Satisfied), is a methodology developed as a joint collaboration of research work between the University of Oregon and the University of Louvain (Belgium) in 1990. Respect-IT has built a tool known as Objectiver for KAOS [Respect-IT].

KAOS methodology facilitates to mix the top-down and bottom-up approaches. A goal always captures the ‘WHAT’ perspectives of the needs. Since the ‘WHY’ and ‘HOW’ perspectives of the needs reflect the rationale behind the refinement process, the principle objective is to link goals in order to explicitly reply to the WHY and the HOW questions. This methodology is divided into four modelling activities in order to cover the whole process from the specification of goals to the specification of the system, see Figure 16.

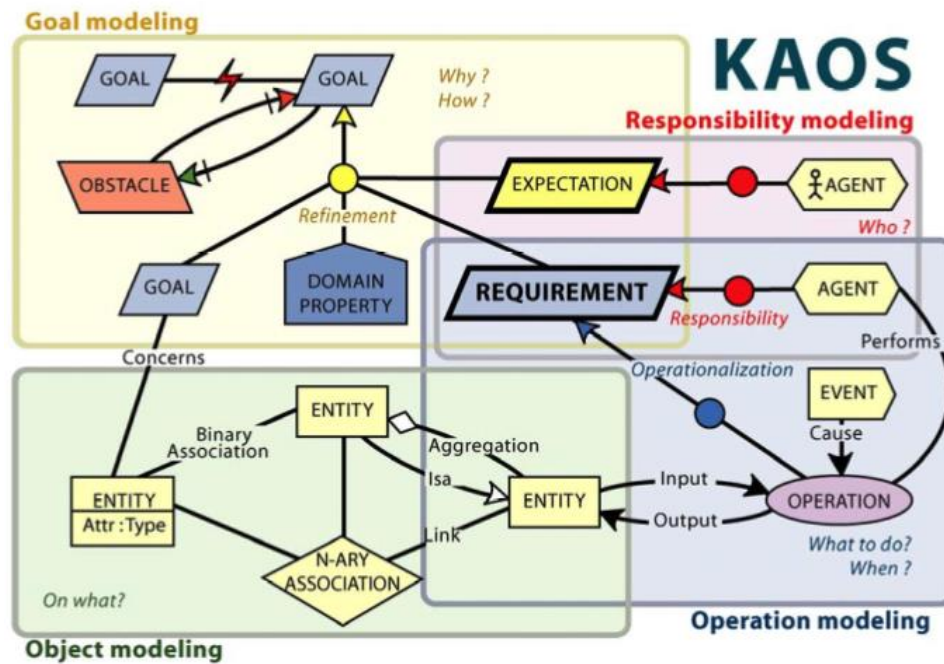


Figure 16: Secure KAOS SRE methodology

- The *goal modelling* describes the goals of the system-to-be and its environment. A goal model is a hierarchy specifying the refinement of strategic goals into requirements;
- The *responsibility modelling* defines the assignment of requirements to agents (actors of the system-to-be);
- The *object modelling* specifies the vocabulary of the domain such as entities, events, relationships etc. Object model, which is UML based, represents the correlation between the objects, entities and their association in form of a class diagram, corresponding to achieve a goal.
- Finally, the *operation modelling* defines the possible operations on the system as well as the relations with the others models. An operation model captures the behaviour of the agents to satisfy the requirements.

Altogether, this methodology highly focuses on the goal refinement thereby facilitates to verify the traceability of the security goals being refined into security requirements. Wherein, the *goal/responsibility* models correspond to the activities of early stage requirement analysis; while the *object/operation* models correspond to requirement analysis at later stages.

In Figure 16, goals (represented by blue parallelograms) are the desired properties of the system-to-be and its environment. Goal refinements are realized via the AND/OR constructs and are structured similar to acyclic graphs. An edge from goal A to goal B indicates that goal A refines goal B. Then, this edge states why goal A exists and how goal B is satisfied. A goal can be refined using an AND edge implying that the conjunction of sub-goals is necessary to satisfy the goal. It is also possible to represent alternatives using OR refinement. Then one of the sub-goal is sufficient to reach the goal. In addition, to facilitate the refinement process, KAOS defines four different types of goals based on the temporal logic (*achieve/maintain /cease/avoid*). These goal patterns have an impact on the set of possible behaviours of the system [Dardenne et al. 1993].

Sometimes, goals can be conflicting. These conflicts are defined in KAOS through a link with a flash symbol, see Figure 16. It is also possible to identify and to take into account negative events that can threaten the system-to-be using obstacles (red parallelograms). An arrow from an obstacle to a goal represents an obstruction. An arrow from a goal to an obstacle states that the goal resolves the obstacle. The goal resolving intentional obstacle is known as security goal.

Finally, the properties of the domain are expressed using purple pentagons. They explicitly state the invariants of the environment (e.g., physical laws, etc.) or the assumptions. This notion of domain property is interesting because domain property items enforce the RE analyst to explicitly specify assumptions (e.g., observing the environmental constraints). As consequence, it makes the specification of related goals more precise.

When a goal cannot refined further (i.e., atomic), it is called a requirement of the system-to-be and is assigned to an agent (to either environment or system agents). A requirement (represented by a blue parallelogram with bold edges) is about the system-to-be. When a requirement concerns an environment agent (e.g., human), it is called an expectation (yellow parallelograms in Figure 16). Distinguishing requirements and expectation seemed to be a good idea. Since requirements are assigned to system-to-be agents (actors that we will build) and expectation to environment agents (actors that we cannot control completely), it creates an impression that expectations must be dealt carefully.

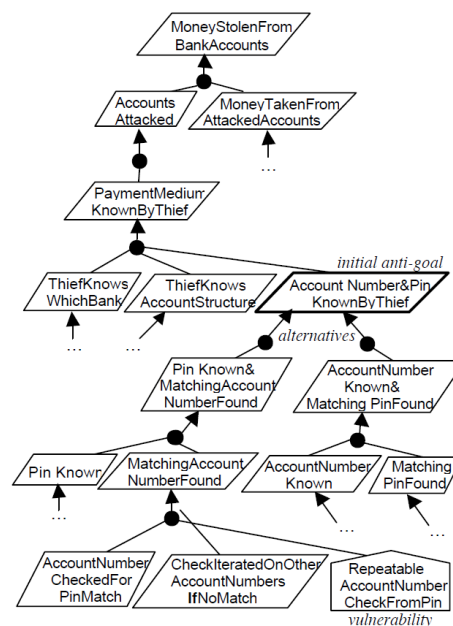


Figure 17: Anti-goal refinement sample [van Lamsweerde 2004]

Furthermore, Secure KAOS method proposes an approach to construct an anti-goal graph similar to goal hierarchy, but reflects the attacker's perspective to break system security goals i.e., intentional obstacles. Figure 17 (taken from [van Lamsweerde 2004]) illustrates an example of an anti-goal model in the context of an electronic banking service. An anti-goal refinement can be correlated to attack trees from risk analysis techniques [Schneier 1999]. Anti-goal refinement is done until leaf nodes are identified as software agents corresponding to some software vulnerabilities observable or implementable by the attacker. Security requirements are derived to avoid the adverse conditions

corresponding to these leaf nodes, similar to attack-defence trees from risk analysis techniques [Kordy et al. 2011].

1.4.2 Agent-oriented approaches

Agent-oriented approaches are centred towards Agent-oriented Software Engineering (AOSE), a software engineering paradigm corresponding to the development of complex multi-agent system [Bresciani et al. 2004]. Whereas, multi-agent system based approaches provide means to model complex systems featuring multiple agents and their goals [Rosenschein 1988].

Eric Yu's *i** framework [Yu 2011] is recognized as the first RE approach that integrates goal modelling concepts with multi-agent paradigm. The principal idea is to address the WHY dimension of SRE process from the perspective of the stakeholders involved. In this respect, it redefines agents as actors having some goals within the system or organization of interest. An *actor* can be either a *role* or an *agent*. Agents are the concrete known participants of the system (e.g., human/software/hardware entities), while roles are abstract representations of the actor concept (e.g., Student role). Roles can be played by either human agents or system agents. In addition, it presents dependency models to describe the fact that an actor depends on another actor to attain some goal [Yu 2011]. It proposes two conceptual models: 1) *Strategic dependency model*, based on the interactions between actors, and 2) *Strategic rationale model*, based on reasoning the conflictual viewpoints between these actors concerning an actor's dependencies and relationships with other actors.

Many works have extended the underlying concepts from *i** framework in order to fit to different RE contexts [Fabian et al. 2010]. Some well-known extension works include: Tropos [Bresciani et al. 2004], which extends *i** modelling concepts to software development context. Secure Tropos [Mouratidis and Giorgini 2007] extends the tropos modelling to security context. Suitably, it introduces new security concepts such as security constraints specify security related restrictions over the security dependencies between actors. Secure *i** method [Massacci et al. 2010] extends the strategic rationale dependency models of *i** framework, in order to fit to security context it introduces the notion of delegation and trust.

In this thesis we study STS (Social Technical Systems) framework [Paja et al. 2014], which extends secure Tropos and secure *i** methods featuring advantages to define fine-grained information security requirements of the system-to-be with a tool support.

1.4.2.1 Social technical systems modelling language (STS-ml)

The main idea of STS (Social Technical Systems) framework is to model not only the technical components of the system-to-be, but also its social environment where the system will be deployed. In this STS methodology framework [Paja et al. 2014], the security requirement analysis of the system-to-be (future system), is achieved through three different views: *social view*, *information view* and *authorization view*. These three views together constitute the operational view of the system-to-be.

- The purpose of the *social view* is to model the social relationships between actors for the system-to-be. Actor concept is inherited from *i**framework.

- The objective of the *information view* is to introduce primitives and relationships that permit to differentiate between the information (content) and the representation of the information (document).
- Finally, the *authorization view* drives the system designer to determine the fine-grained authorization rules over the access permissions on the documents (resources).

STS modelling language mainly supports early elicitation of security requirements by constraining the social dependency interaction relationships between the actors [Paja et al. 2014]. As consequence, identifying the actors and analysing their interactions are the core activities. Figure 18 shows the STS notation for social modelling. Goals (green rectangles with curved edges) represent actors' (pink circles) interests towards the system. Furthermore, it offers the possibility to create composite goals via the AND/OR constructs. This allows refining top-level goals into sub-goals similar to KAOS, but within the scope of an actor (Figure 18a).

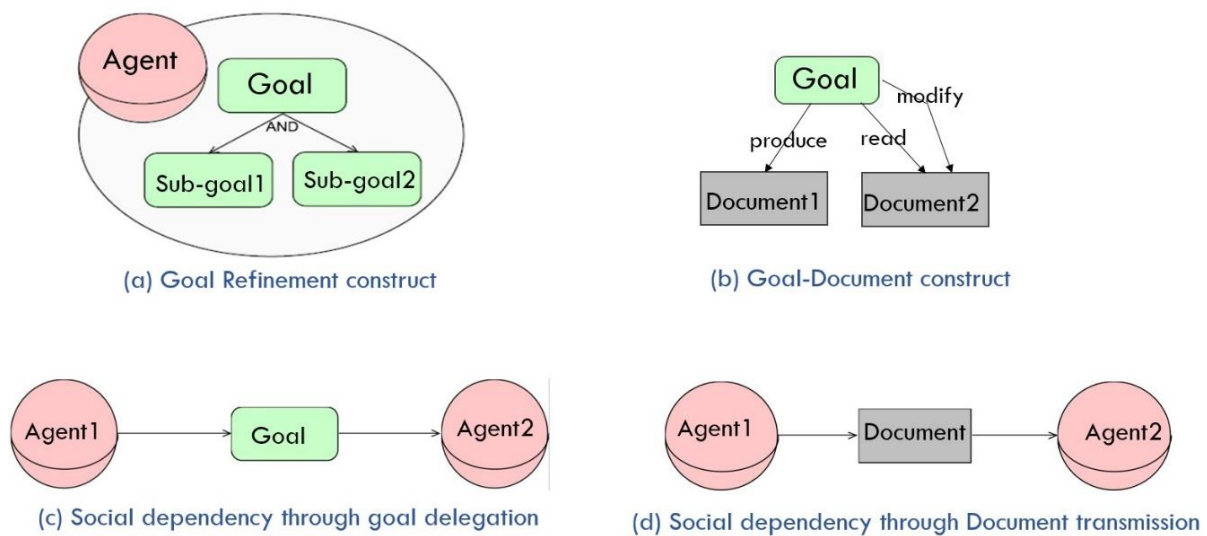


Figure 18: STS-ml social modelling

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. That means an actor depends on another actor to achieve a goal (see, Figure 18c). This aspect is similar to the *responsibility model* of the KAOS. 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 (see, Figure 18d). Different constructs have been proposed to represent the possession, the production, the modification and the read of documents [Paja et al. 2014], but within the confined scope of the agent, see (Figure 18b).

From security perspective, the security goals are implicitly expressed in the form of the security constraints over the interaction dependencies in social view. These security constraints refer to the security properties i.e., confidentiality integrity, availability, non-repudiation, accountability, reliability, as mentioned in section 1.2. Furthermore, security goals can also be observed in form of constrained permissions upon the provision of the resources in authorization view. They include read/modify/produce/transmit provisions, which will help to ensure the compliance with confidentiality

restrictions on documents as expressed in the social view. In addition, STS gives security engineers the possibility to represent threats through events that can hamper the actors' goals. The entity *Event* and the relationship *threatens* are used to represent the threats in STS (see Figure 19).

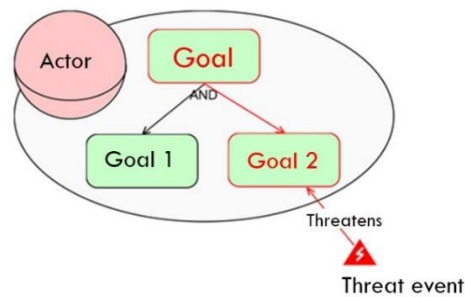


Figure 19: STS threat event notation

Finally, the STS tool can perform an implicit automated analysis hiding the complexity of the formalism. It covers three types of analysis: *Well formedness analysis*, *security analysis* and *threat analysis*. *Well formedness analysis* is used to verify the correctness and the validity of the STS models. The results are returned in terms of warnings and errors. The *security analysis* helps to detect the conflicts between security requirements. The *threat analysis* is used to show the effects over goal trees and goal/resource relationships that may have a threatening event. This threat propagation feature highlights the impact in over goal trees.

1.4.3 Problem frames oriented approaches

Problem frames (PF) are introduced by Jackson [Jackson 2001]. Unlike the STS and KAOS, problem frames approach is focused towards characterizing the constrained behaviour of system under construction as problems. Problem frames are set of patterns that classify software development problems [Fabian et al. 2010].

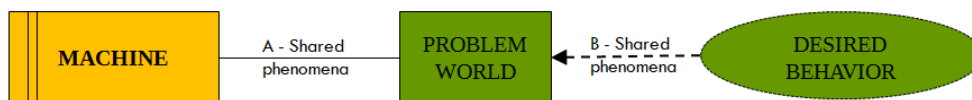


Figure 20: Problem overview

Problem frames modelling consists in three main components: problem domains, connecting interfaces, required behaviour. Figure 20 depicts a generic view of problem frames modelling related to required (or desired) behaviour.

- *Problem Domains* help to describe the problem (represented in rectangles). They correspond to system entities, people and resources that are referred as casual domains, biddable domains and lexical domains respectively. Biddable domains refer to environment agents in KAOS. Machine is a special kind of domain that refers to system-to-be (represented in rectangles with two stripes).
- *Interfaces* help to express the anticipated behaviour of the connecting problem domains. Interfaces are of two types: referenced behaviour (represented in solid straight line) and constrained behaviour (represented in dashed lines with point arrow head). In addition,

interfaces defined with shared phenomena, which represent events, operation calls, messages, and the like. The shared phenomena represent shared attributes/information that are observed by two or more problem domains during an interaction, but controlled by only one of the two connecting domains.

- Finally, the desired behaviour corresponds to the requirement (represented using dashed oval). Altogether, A *problem frame* contains a requirement that refers to certain domains via requirement references in which the requirement constrains at least one domain [Hatebur et al. 2008].

[Jackson 2001] describes 5 problem frames that represent different class of software development problems in software development context. They include required behaviour problem frame, commanded behaviour problem frame, information behaviour problem frame, simple work pieces problem frame, and transformation problem frame.

In the literature very few works have extended problem frames approaches. Abuse frames extends commanded behaviour problem frame to represent malicious behaviour through anti-requirements [Lin et al. 2003; Lin et al. 2004]. In this thesis we study SEPP [Hatebur et al. 2007b], which introduces new security problems patterns to represent security requirements.

1.4.3.1 SEPP (Security Engineering Process patterns)

SEPP [Hatebur et al. 2007b] extends the concepts and terminology of 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).

SPF provides the description of security problem context using the three main components of the PF approach (i.e., problem domains, secured behaviour, interfaces with shared phenomena). In addition, it introduces the notion of pre-conditions and post-conditions for each of the problem frames. Pre-conditions reflect the conditions that the problem environment must achieve to apply the security problem frame. The post-conditions are the formal representations of the security requirement which must be fulfilled by the machine to be built. Once the SPF is finalized, then the problem frames are instantiated as CSPF. CSPFs are the next abstraction level instantiations of the security problem frames which defines some generic security control mechanisms to solve security problems.

[Hatebur et al. 2007b] presents four SPF mapped to eight CSPF. Wherein, SPF refer to class of security problems related to authentication, confidential data transmission, and distribution of secrets and integrity preserving data transmission. However, in this thesis, we confine our study to SPF as they describe early security requirements. For instance, Figure 21 depicts the SPF diagram for confidential data transmission given [Hatebur et al. 2007b].

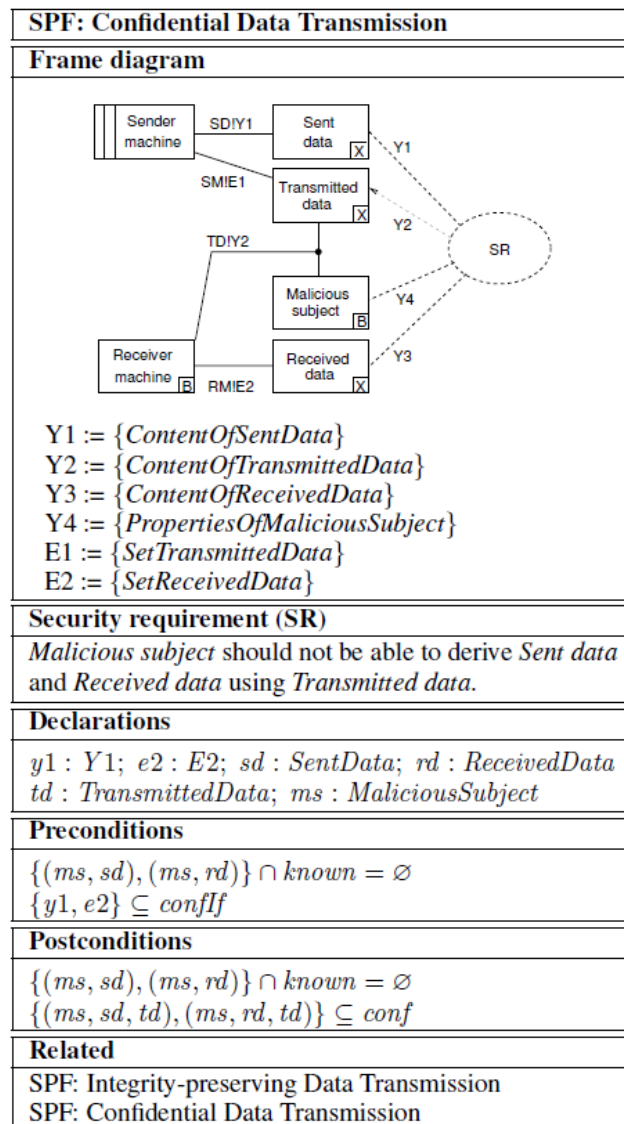


Figure 21: SPF pattern for confidentiality [Hatebur et al. 2007b]

This security requirement (SR) in this SPF constrains that transmitted data must not be derived by the malicious subject. The transmitted data is expressed as lexical domain (represented as in rectangle with 'X' in bottom corner) since it holds data being sent and received by the authentic domains (i.e., sender machine and the receiver machine). The malicious subject is expressed as biddable domain (represented as in rectangle with 'B' in bottom corner) since it refers to an attacker. However, the SPF notation does not differentiate authentic subjects to fake subjects.

The events E1 and E2 express the transmission of the data sent and received between the authentic subjects. The interfaces Y1, Y2, Y3 express the contents of the data held by the lexical domains (i.e., sent data, transmitted data and received data); while the interface Y4 express the behaviour of the malicious subject. Furthermore, the shared phenomena describe the control over the shared information defined in the declarations section of SPF. For instance, the sender machine (SM) control the events (E1) related to the transmitted contents of sent data. Frame diagram expresses this using an interface with shared phenomena denoted as SM! E1.

1.5 Network Security requirements

The study of network security requirements majorly concern with the protection of network infrastructure, access control, information flows [Government of Canada, Communications Security Establishment 2007]. From an industrial perspective, the current practice for eliciting and analysing early network security requirements is driven by security zoning, a well-known defence-in-depth strategy for network security design [Sherwood 2005; SAP 2013; ICS-CERT 2016; NIST 2014; SANS Institute 2003]. Defence-in-Depth strategy is adapted from military defence security practice, where the attacker is forced to circumvent multiple lines of defence, before being able to sneak into the secured region [Cleghorn 2013]. Likewise, the Defence-in-Depth strategy in the scope of enterprise security, aims at building multiple layered defence lines, for evading any single point of failure, in case of security breach [SANS Institute 2003].

Security zones constitute the logical grouping of network elements that are identified with similar protection requirements (e.g., data confidentiality and integrity, access control, audit, logging, etc.). Each security zone is identified with different trust levels, which exhibit the rigor of required protection. In this context, determining security zones and respective trust levels is a preliminary step for security architects in capturing other network security requirements (e.g., related to data flows), and later in selecting the right network security controls/mechanisms (such as VPN, IP Firewall, Web Application Firewall, etc.) [NIST 2014; SANS 2015a]. These security measures eventually play vital role in building security defence around the business critical assets of the organization [NIST 2014; SANS 2015a].

In the literature, several research works and best practice approaches propose various zone classification schemes and patterns [Secure Arc 2009; Province of British Columbia 2012; Government of Canada, Communications Security Establishment 2007]. Different reference works prescribe different set of zone categories as well as different guidelines and inter/intra zone communication rules.

1.5.1 Security zoning reference models

Majority of the existing works are from industrial/government sectors [Province of British Columbia 2012; Government of Canada, Communications Security Establishment 2007; Secure Arc 2009], which mainly focus on providing foundational best practice guidelines, and reference modelling patterns, for building secured networks. From a broad view, these reference models propose minimum set of zones as well as inter/intra zone interaction rules necessary to be implemented, for achieving basic logical network security design. However, there exists no restriction on either the number of zones or their category types, as they depend on the size and type of the business. Some of the commonly identified network zones include internet zone, demilitarized zone, extranet zone etc. The communication between zones are monitored and controlled by some security measures (e.g., a firewall, a gateway, etc.).

In brief, the Internet zone, by default, is assumed as extremely hostile and least trusted, as it is publicly accessible to everyone including the anonymous threat actors. The demilitarized zone (DMZ) is the intermediate zone that usually sits between the trusted and less trusted zone in order to reduce attacks surfaces. The extranet zone contains trust security entities that belong to an external third-part domain (e.g., external internet service provider). Respectively, the interaction rules between the security

entities, belonging to a single, or multiple zones, are determined and monitored depending on the category of zones. However, different patterns propose different rules for zone interactions.

The Figure 22 example reference patterns from British Columbia model [Province of British Columbia 2012] and Secure Arc Inc., [Secure Arc 2009] that vary in zone interaction rules. For example, the British Columbia model [Province of British Columbia 2012] describes seven zones and allows communication inside the zones and only between adjacent zones, see Figure 22(a). It defines other zones such as client security zone and high security zone that contain the set of security entities (e.g., users, desktops, servers, etc.) that are part of the enterprise. Sensitive assets are confined to highly restricted zones. In addition, the special zones such as the management zone constitutes of entities that are involved in security management activities such as monitoring, and administering the zones and their interactions.

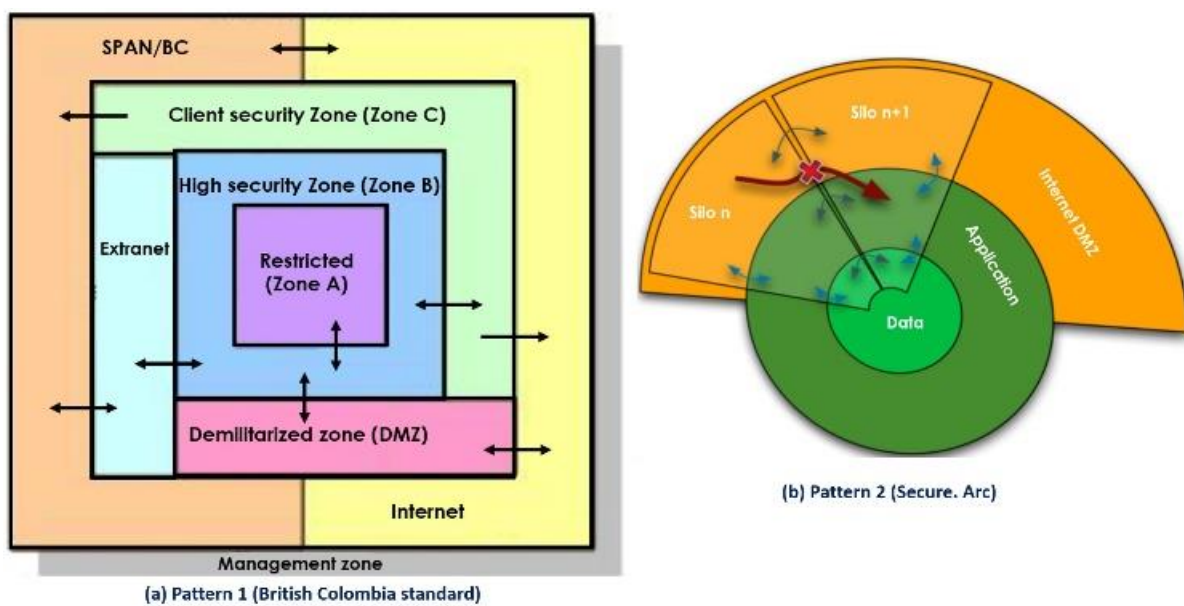


Figure 22: Example zone reference models [Secure Arc 2009; Province of British Columbia 2012]

Secure Arc [Secure Arc 2009] Defines eight zones. It also adds a parallel cross-zones segmentation concept, called silos, see Figure 22(b). Communications are allowed only between adjacent zones and within the same silo, or between adjacent silos within the same zone. That means the security entities from a silo (e.g., Silo n) are limited to communicate with the security entities that belong only to either the adjacent silo (i.e., Silo n+1) in the same zone, (i.e., Internet) or belong to the same silo (i.e., Silo n) in the adjacent zone (i.e., application zone). The aim is to limit the interaction between the zones to only dedicated traffic even though they are adjacent to each other. However, these documents are only guidelines and must be manually adapted.

On the other hand, the academic community has published only few works concerning network security zones. *Gontarczyk et al* [Gontarczyk et al. 2015] provides a standard blue-print that includes three classes of security zone (no physical measures, limited physical measures, and strong physical measures), see Figure 23. For instance, no physical security measures can be implemented in internet zone since it is publicly accessible to everyone unlike enterprise and restricted zones that are internal to enterprise.

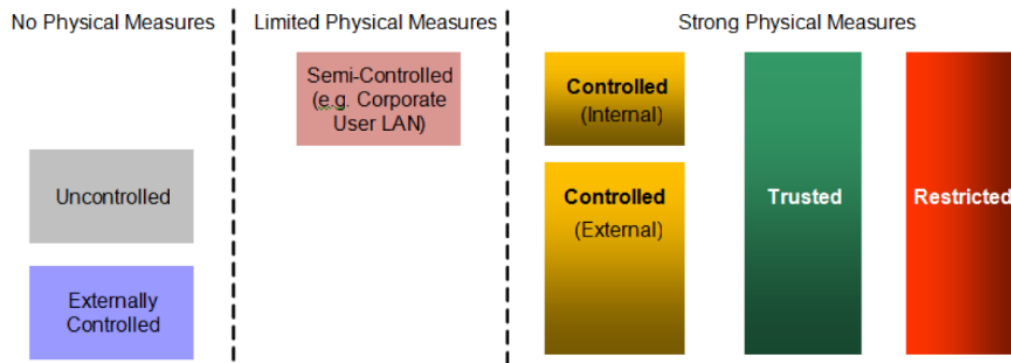


Figure 23: zone classification model [Gontarczyk et al. 2015]

In addition, the authors also provide a classifier to guide the deployment of systems/ applications, see Figure 24. However, this is a high-level guideline that must be manually adapted by the security architects. Furthermore, the classifier is ambiguous (e.g. some systems can be placed in any of the zones). They also don't define network security requirements.

	Mission Critical	Regulatory Influence	Security System	Core Infrastructure	Pilot	Business Partner	Business Support	Operational Support	Data Sensitivity	Commercial Sensitive	Self Service Web App	Proxy/Gateway	Vulnerable to Attack	Service Denial	Insider Attack	Failover Protection	Strict Service Level	Physically Secured	Mobile Technology
Zone	Business Application Classifiers												Technology & Vulnerability						
Restricted	✓	✓	✓	✓			✓	✓	✓	✓			✓			✓	✓	✓	
Trusted			✓	✓		✓	✓	✓	✓	✓			✓	✓	✓	✓	✓	✓	
Controlled			✓	✓	✓	✓	✓	✓	✓	✓			✓	✓		✓			✓
Semi Controlled			✓	✓	✓	✓					✓	✓	✓	✓		✓			✓
External Control			✓		✓	✓					✓	✓							✓
Uncontrolled			✓		✓						✓	✓							✓

Figure 24: Security zone classifier [Gontarczyk et al. 2015]

Ramasamy et al [20] proposed a bottom-up approach for discovering the security zone classification of devices in an existing enterprise network. It distinguishes between network flows actually allowed and those permitted by company policy. The actually allowed traffic correspond to the network flows related to host configurations, and network flow logs handling network statistics data. For instance, there are many network administration tools such as Nmap [6], traceroute, netstat, and SNMP-based approaches that can collect and analyse information about network configuration of devices in the infrastructure. On the other hand, the flows permitted by policy concern business interactions that include actual transmission of data. However, this work doesn't help in eliciting network security requirements.

To consolidate, a zone modelling process consists in classifying zones based on trust levels, and employing some best practice guidelines for monitoring inter/intra zone communication. However, this process takes place in an ad-hoc manner and do not integrate a rigorous approach to specify security zones and rules governing zone interactions.

1.6 Chapter Summary

Security requirements are essential components that capture business security objectives related to the protection of assets from potential threats. Network security requirements at early stages correspond to security zoning, which concerns the logical grouping of assets with regards to the degree of protection required.

Different SRE approaches inspired by the process of requirement engineering security risk analysis concepts, have proposed different methods to express and analyse requirements at early stages. The goal modelling approaches focus on the risk analysis pertaining to the achievement of system goals in the name of obstacles and anti-goals. The agent modelling approaches emphasize on the threat analysis upon securing the dependency interactions between the system agents. Finally, the problem frames oriented modelling approaches emphasize more on the problem analysis in achieving the required secured behaviour of the system.

SABSA framework provide a reference model for guiding the development of a, enterprise network security architecture with definite abstraction levels. With reference to our research objective from the project IREHDO2, it is interesting to integrate SABSA reference model to derive network security requirements from business. For this, it is important to know which of these SRE approaches suites best to facilitate this integration thereby enabling the derivation of network security requirements. In the following chapter, we present a research method concerning the evaluation of SRE methods, which helps to address this question.

CHAPTER 2 WHICH SRE METHODOLOGY FITS BEST TO THE NETWORK SRE CONTEXT?

As mentioned at the beginning of the document, this thesis work is part of the research project IREHDO2, which concerns the aircraft network security. In this project, the security experts of AIRBUS want to improve their security process to increase the assurance on the final security solution enforced on their aircraft networks. More specifically, they are interested in enhancing their SRE practices. This group of security experts is involved at different stages of the security process and comprises of security requirement engineers, risk analysts as well as the security testing experts.

Our task in the project consists in proposing the best SRE methodology, which will help them in writing good network security requirements. However, the diversity of SRE methodologies (discussed in chapter 1) leads to an open question i.e., which one of them is good to choose? In fact, each security expert from our project had a different point of view on what could constitute a good SRE methodology. As consequence, it was difficult for us to fix the evaluation criteria to analyse the suitability of the existing SRE methodologies studied in chapter 1. This issue gave rise to our first research question RQ1: How to evaluate SRE methodologies in a network SRE context?

Several comparative studies on SRE methodologies are available. Nonetheless, their evaluation is often based on a set of ad hoc criteria. These comparison criteria may not fit the SRE needs of every company. Thus, a systematic method needs to be defined to capture the company's requirements of an SRE methodology. We address this issue by proposing a requirements engineering-based evaluation methodology. It helps in characterizing the goodness of a SRE methodology by considering the SRE needs of the stakeholders as well as the quality characteristics of good security requirements (a.k.a. quality criteria). Indeed, considering the quality criteria is important to avoid requirement errors. This chapter is dedicated to present our evaluation methodology and its implementation.

We first provide an overview of the existing comparative studies (section 2.1). Next, we present our proposed evaluation methodology and its strategy (section 2.2). In this respect, we explored the refined research question RQ1a related to the study of the characteristics of good security requirements (section 2.3). The RQ1a exploration provides a basis for eliciting the characteristics of good SRE methodology (section 2.4). Finally, we describe the implementation of our evaluation methodology (section 2.5). In this regard, we describe our evaluation process to evaluate the three SRE methodologies (i.e., STS, KAOS and SEPP) presented in chapter 1. At the end, we present in brief the pros and cons of each of these three SRE methodologies using the elicited characteristics serving as the evaluation criteria.

This chapter consolidates the following contributions: [Bulusu et al. 2016; Bulusu et al. 2017a; Bulusu et al. 2018b; Bulusu et al. 2018a].

2.1 Comparative studies: State-of-the art

In the literature, there exists numerous works related to comparative studies, which offer varying dimensions of SRE evaluation perspectives. In below, we first present an overview of these 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 similar to the works [Karpati et al. 2011; Khwaja and Urban 2002].

N Mayer [Mayer 2009] 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 as 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. [Fabian et al. 2010] proposed a conceptual framework to support the comparative study of the existing SRE methodologies. When compared with the modelling framework in [Mayer 2009], this work highlights some similar concepts related to risk analysis but was extended with granular security analysis concepts based on the security properties i.e., CIA triad. In addition, it considers the concepts related to the multi-lateral security requirement analysis that stresses on the compromise of the conflicting stakeholder's security needs [Rannenberget al. 1999].

Munante et al. [Muñante et al. 2014] extends the comparison frameworks [Fabian et al. 2010; Mayer 2009] that compose additional evaluation with the concepts related to model-driven engineering (e.g., *tool support*, *SRE modelling language*, *formalism*). This work concludes that KAOS [van Lamsweerde 2004] and secure i* [Elahi and Yu 2007], are the most compatible for formal SRE modelling.

Souag et al. [Souag et al. 2015] proposed a comparison framework to provide a systematic mapping of the reusable concepts and patterns within the existing SRE methodologies. Accordingly, the research contributions were classified into 5 categories as security patterns, taxonomies and ontologies, templates and profiles, catalogues and generic models and miscellaneous.

Uznov et al. [Uzunov et al. 2012] proposed a comparative analysis of security engineering methodologies using a list of 12 criteria in the name of 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. [Mellado et al. 2010] provided a systematic review of comparative studies to analyse 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 international standard IEEE 830 [IEEE 830 1998].

N Mead [Mead 2007] provided a comparative analysis of the requirement elicitation techniques based on some criteria such as learnability, client acceptance and durability of the requirement elicitation techniques, tools support etc.

Nhlabatsi et al. [Nhlabatsi et al. 2009] proposed a comparative study of SRE methodologies in order to evaluate the extent to which they can support 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.

2.1.1 State-of-the-art analysis

Many interesting strategies and aspects were discussed in the related works. However they are not sufficient to evaluate the goodness of the SRE methodologies. In below we provide our analysis based on three issues related to requirements engineering process.

2.1.1.1 Issue A: evaluation criteria coverage

This issue concerns the evaluation criteria coverage perspective. The ultimate motive of SRE methodology is to help in capturing security requirements with regards to SRE process (i.e., elicitation, evaluation and documentation) [Fabian et al. 2010].

From our study, we noted that majority of the evaluation studies except [Mellado et al. 2010] lack a full emphasis on the whole SRE process. While some works [Fabian et al. 2010; Mayer 2009; Mead 2007] emphasize on evaluating the extent of support to security requirements elicitation at earlier stages; some others [Souag et al. 2015] focus on evaluating the extent of support to documentation in terms of reusable patterns or modelling initiatives. In this respect, *Mellado et al* [Mellado et al. 2010] is considered as an exception because, the evaluation based on the quality characteristics of requirements explicitly covers all activities of SRE process [ISO29148:2011] (as discussed in Section 1.3).

2.1.1.2 Issue B: Evaluation criteria affirmation

Evaluation studies are performed to help researchers with the analysis of the information related to the supportable characteristics of SRE methodologies. Therefore, the evaluation criteria must result from a systematic processes or standard methods like [ISO29148:2011], to facilitate their reuse.

However, in the majority of works, the proposed evaluation criteria were ad hoc and lack affirmation on why the criteria were good enough for the evaluation [Mead 2007; Nhlabatsi et al. 2009; Uzunov et al. 2012]. In this aspect only [Mellado et al. 2010] can be noted as an exceptional case as the evaluation is based on the international standard [IEEE 830 1998].

2.1.1.3 Issue C: stakeholders involvement

This issue concerns the involvement of stakeholder's views while formulating the evaluation criteria. As highlighted in [Kotonya and Sommerville 1996], the requirements process is a human endeavour, and so the requirements elicitation method or tool must support the need for stakeholders to communicate their ideas and obtain feedback. In here, stakeholders refer to the SRE methodology users (e.g., requirement 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, it is to note that almost all of the works acknowledge (either implicitly/explicitly) the significance of involving

stakeholders perspectives in the SRE process. For instance, [Fabian et al. 2010] includes a criterion that concerns the integration of a stakeholder's views to support multilateral security requirement analysis.

Altogether, from our state-of-the-art analysis, we 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).

2.2 RE-based SRE evaluation methodology

We propose an SRE evaluation methodology that differentiates its strategy from previous comparative studies for two reasons. First, we involve the security experts that are the SRE end-users in the whole process. 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 (section 1.3). In our case, the system-to-be is the good SRE methodology, labelled as 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 classic idea of a requirements engineering approach. It facilitates the elicitation of the evaluation criteria of SRE-methodology-to-be considering the SRE context of the stakeholders. Figure 25 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 characteristics goal. This is done 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.

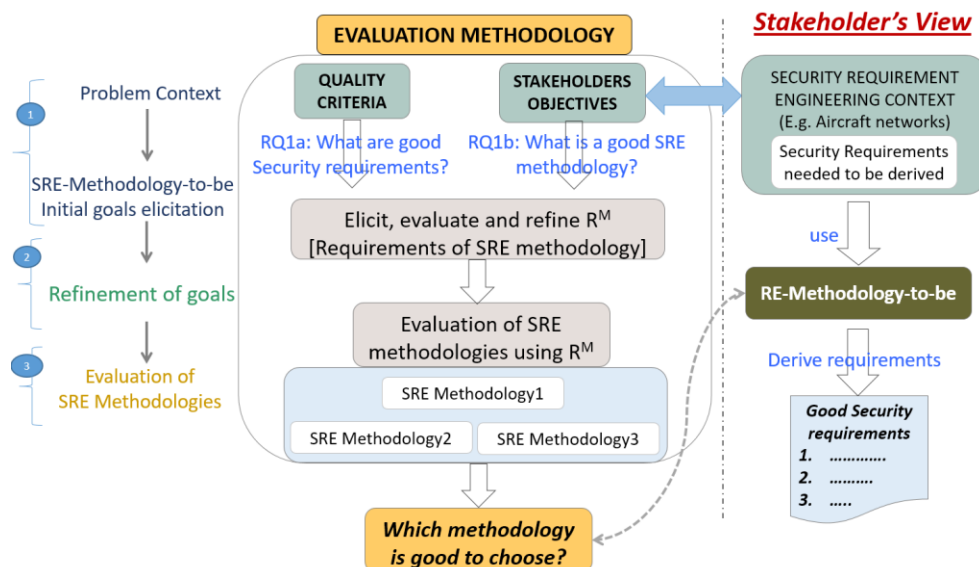


Figure 25: Our evaluation methodology

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 regarded as stakeholders in our approach. This involvement of security experts is mandatory since helps to improve effective communication among them. Besides, works [Firesmith 2007; Kotonya and Sommerville 1996; Mead and Stehney 2005; Van Lamsweerde 2009; Walia and Carver 2009] consider the lack of people involvement is one of the major requirement problems.

Since the ultimate goal of the security experts is to derive good security requirements regardless of the SRE context [Firesmith 2003; Mellado et al. 2010], the SRE methodology is a way to achieve this goal. Therefore, the second step concerns the elicitation of the initial characteristics goal, which features a brainstorming session driven by two further refined research questions, i.e., RQ1a: what are good security requirements?, and RQ1b: what is a good security requirement engineering methodology?

The study of RQ1a helps to setup a common understanding the characteristics of good quality security requirements. This study of RQ1a is independent from the SRE context and helps to watch for the **issue A** (described in section 2.1.1.1). This understanding will eventually drive the RQ1b, which helps to capture common perspectives of stakeholders' anticipations over an ideal SRE methodology-to-be in their SRE context. This study of RQ1a is independent to SRE context and helps to address the **issue C** (described in section 2.1.1.1). Indeed, this correlation between RQ1a and RQ1b has once been addressed in similar manner in the work of [Mar 1994], where three software requirements methods were compared using quality characteristics of good requirements. However, it has not defined any formal process.

2.2.1 Formal modelling notation

Our evaluation methodology follows a pure goal-based approach for eliciting requirements (R^M) of the SRE-Methodology-to-be. We propose to use the goal modelling notation KAOS [Van Lamsweerde 2009] to represent the goals refinement hierarchy, since it facilitates the traceability of the refined goals (see section 1.4.1.1). 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** (given in section 2.1.1.2).

In the following sections, we instantiate this evaluation methodology in the network SRE context, in order to elicit the characteristics of the good SRE methodology. For this, we first need to explore the research question RQ1a: "what are good security requirements".

2.3 RQ1a: What are good security requirements?

The common goal of any research contributions in the SRE community is ultimately to facilitate the capture of good security requirements [Firesmith 2003; Donald Firesmith 2007; Christian 2010; Van Lamsweerde et al. 2003; Mellado et al. 2010]. Because, 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 effective security solutions [Anderson 2010]. 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 is not explicitly defined 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 security solutions.

Sources ID	Source name	Criteria count
1	[ISO29148:2011]	13
2	[Van Lamsweerde 2009]	11
3	[Firesmith 2003]	15
4	[Wiegers 1999]	10
5	[Wieringa 1996a]	7
6	[Boehm 1984]	5
7	[Pfleeger and Atlee 1998]	8
8	[IEEE 830 1998]	9
9	[Davis et al. 1993]	20
10	[Mar 1994]	13
11	[Sommerville and Sawyer 1997]	7
12	[Young 2004]	15
13	[Hull et al. 2010]	15
14	[Kar and Bailey 1996]	10
15	[Zielczynski 2007]	13
16	[Mannion and Keepence 1995]	5
17	[IEEE 12333 1996]	9
TOTAL		185

Figure 26: Quality Criteria - consolidated sources list

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 [Mar 1994].

In this context, we have studied 185 definitions from 17 literature works depicted in Figure 26. 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 analysing the possibility to reach a consensus.

Since the list is huge, we have developed a graph-based analysis tool (see Appendix B) using python graph module Networkx [NetworkX developers 2016]. Figure 27 shows the complete view of the criteria conceptual graph generated using Gephi tool [Gephi.org].

The first result of our analysis has led us to a huge graph with 212 nodes and 278 edges. The graph projects a total of 71 criterion nodes and 141 concept nodes. Whereas, the nodes 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.

[illegible]

evident that every node is connected, highlighting the complexity of the semantic
 the quality criteria. In addition, the dependencies are found ambiguous in some
 a sample of ambiguous definitions in which the definitions precise (PS), concise
 om the authors results in a cyclic referencing, see Figure 28. This opens a new

problem that might require a proper research to analyse the semantic meaning of each of the criterion definitions.

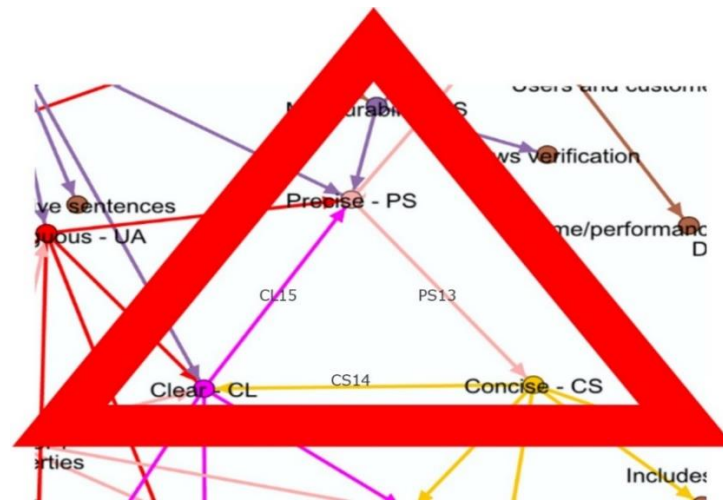


Figure 28: semantic dependencies - cyclic references

In our evaluation methodology context, our objective is to define an exhaustive list of the existing characteristics that can be integrated to any SRE process. Suitably, we needed something simpler that can be used as a standard basis for eliciting the characteristics of the good SRE methodology. Suitably, we have identified 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.

2.3.1 The weaving methodology

The main goal of our weaving methodology is to showcase the indifferences in the criteria propositions of the unified list in a tabular format that featured a weaving strategy. Table in Figure 29 consolidates our weaving results, which extends the results in [Bulusu et al. 2016]. Firstly, we give 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 1). In addition, we have given a one-line definition to each criterion. If the characteristic is defined in ISO29148, we used the standard definition. Otherwise, we have taken references from respective authors if the characteristic description is straight forward (e.g., C8). When the characteristic descriptions seemed ambiguous (e.g., C10), we gave our own interpretation.

Secondly, we used colours (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 colour, when defined by only a single author (e.g., C13). When different authors employ different names to describe similar quality characteristic (e.g., C3), then the respective criterion definition is emphasised 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 emphasised in italic text. 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).

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	Credibilt y	Applicabilt y	
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	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	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/re alistic/pos sible	Attainable / realisable	--	High	All	
C4	Requirements must be well documented	--	Good structuring	--	--	--	--	--	--	organized	--	--	--	--	--	--	--	configurabl e	low	All	
C5	Requirements should be able to refer back to its objective. Dependency or reference links between requirements should be explicitly 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 Observability	--	Truth/ Implementation independence / Validity	--	--	--	Design independent	Minimality / Right level of detail	--	Design Independent	Abstract	Implementa tion Free	Independe nt /Implemen tation Free	specific/ appropriate level of detail	Abstract/ Normalized / granular	Medium	All	
C7	Documented requirements must be easily adaptable to new changes	--	Modifiable	--	Modifiable	Maintainable / Modifiable	--	--	Modifiable	Modifiable	Maintainabi le / Modifiable	Adaptabilit y	--	--	--	--	--	Modifiable	Medium	All	
C8	No redundant requirements	--	--	--	--	--	--	--	--	Non- redundant	--	--	Non- redundant	Non- redundant Unique	Identificati on	Non- redundant	--	--	low	All	
C9	Every requirement is uniquely identified (i.e., number, name tag)	Identification	--	Identification	--	--	--	--	--	--	--	--	--	Unique	--	--	--	Unique set	low	All	
C10	Stakeholders needs are sufficiently expressed	--	Adequacy	Validity	--	--	--	--	--	--	--	Validity	--	--	--	--	--	--	Medium	Each	
C11	Requirements defined are simple, using common terminology and non-technical jargon.	--	Comprehensi bility	Customer / User Orientation / Usability	--	Understandab le	--	--	Understanda ble / Concise	Simple	Comprehe nsibility	--	Concise/ simple	Clear/ precise	Understand 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 Communicabili ty	--	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/Bou nded	Pertinence	Mandatory/ Relevance	Necessary	--	--	Relevant	--	Necessary	Necessary	--	Necessary	--	Necessary	Necessary	--	Bounded	Medium	Each	
C15	Requirement should accurately represent the facts and needs. Syntactically and semantically	--	--	Correctness / Currency	Correct	--	--	Correct	Correct	Correct	Correct	--	Correct	--	--	Understanda ble / 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 / Requirements 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 related information)	Attributes	--	Metadata	Prioritized	--	--	--	Ranked for importance / degree of stability	Ranked for importance / Ranked for stability	--	--	--	--	Supporting characteris tics	--	--	--	Medium	Each	

Figure 29: weaving the characteristics of good security requirements

In below we provide the weaving results related to colouring and typographical emphasis in form of list, to facilitate the comprehension of the table in Figure 29:

- **Characteristic definition**: Single criterion defined by only one author – [C19]
- *Characteristic definition*: Different names used for same criterion definition by different authors [C3, C5, C6, C7, C10, C11, C12, C14, C16, C17, C18, C20]
- **Characteristic Name**: Different names defined by single author maps to single criterion [C3, C5, C11, C14, C15 and C17]
- Characteristic name: Same name used by different authors to describe different criteria [S5:C6, S9:C13, S15:C15 and S17:C17]

Thirdly, we distinguish the applicability of a criterion. If mentioned *each*, it shows that the respective quality characteristic targets a single (or a specific set of) requirement(s) (e.g., C10). If mentioned *All*, it states that the respective quality characteristic targets the whole set of requirements specifications document. See *Applicability column* in the Table in Figure 29. Fourthly and finally, we defined credibility scores to show the frequency of citations. We used qualitative scale *high*, *medium* and *low*. Credibility given *high* when cited by at least 15 authors; *medium* with citation of minimum 8 authors; *low* when cited below 4 authors; very low when cited by less than four authors.

Altogether, we confine our RQ1a study to the Table in Figure 29, which is self-contained and gives a good overview of the non-consensus issue concerning the characteristics definitions. Since the interpretation of a definition can vary between different people with varying expertise levels, the table contents can raise several questions. This implies that our weaving methodology strategy provides a way that can 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.

2.4 RQ1b: what is a good SRE methodology?

Since RQ1b is dependent on the SRE context, we describe the instantiation of our evaluation methodology to the Network SRE context IREHDO2 project. The first two steps which 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.

2.4.1 Step1: Problem context and initial requirement analysis

The initial step of our approach allows to analyse the security problem context of the security experts (mentioned in section 2.2). Accordingly, this step deals with interviewing the people involved in the security engineering process. We used the brainstorming technique to encourage the people to exchange ideas on “the best suitable SRE methodology” that best fits their needs. However, eliciting requirements is a hard task [Mead 2007]. 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 [Sherwood 2005], (see section 1.3.2). 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, the list of 20 quality criteria characterizing good security requirements given in Table1 is similar to business attributes in SABSA. Based on this list we developed an elicitation tool to trigger the discussions, see Figure 30.

No	Abstract criterion abstract definition	Criterion Names in use	QUESTIONAIRES
C3	Accomplishable within the 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? [ISO29148 2011] ► Does SRE methodology require training? is it within the project budget? [Firesmith 2003] ► Is it possible to learn the SRE modelling within project timelines? [Mead 2007]
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 the vice-versa? [ISO29148 2011] ► Is it possible to trace the group of similar requirements e.g., related to a particular abstraction level? [Firesmith 2003]
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? [ISO29148 2011] ► Is it possible to respect the abstraction needs of the stakeholders? [Mar 1994]
C10	Stakeholders needs are sufficiently expressed	Adequacy, Validatability	► Is it possible to capture the individual needs of all the stakeholders involved in the SRE process? [Firesmith 2003] ► Is it possible to express the domain environment assumptions and domain constraints for implementing the security requirements? [Van Lamsweerde et al. 2003b]
C11	Requirements defined 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? [Firesmith 2003] ► Is the formulation of security requirements is comprehensible to the stakeholders? [Van Lamsweerde et al. 2003b] ► Does is facilitate easy communication of the ideas among the stakeholders involved at different abstraction levels of the SRE process? [Wieringa 1996]
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? [ISO29148 2011] ► Is it possible to express the risk attributes pertinent to the security requirements? [ISO29148 2011] ► Is it possible to express the priority of the security requirements based on risk impact? [ISO29148 2011]

Figure 30: Elicitation tool

The first three columns consolidate the weaving results from Figure 29. They subsume, a unique identifier, a quick one-line definition and corresponding synonyms found in the literature, to the weaving results in Figure 29. Finally, the last column describes the quality criteria via a set of questions, each

reflecting different perspectives of the respective criterion definitions. We have provided the references to the related works from which we have framed 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 the brainstorming. For example,

Axel van defines the Adequacy quality criterion (C10) as follows: *“The requirements must address the actual needs of the system-to-be, explicitly expressed by the stakeholders or left implicit. The software requirements must be adequate translations of the system requirements. The domain properties must correctly describe laws in the problem world. The environment assumptions must be realistic”*.

Firesmith defines the validatability quality criterion that shares similar semantic meaning, which says: *“Individual requirements must actually fulfil the needs and desires of their primary stakeholders and should be validatable: Is it possible to ensure that each requirement is actually what the customer representatives really want and need?”*

From these above definitions, we derive the evaluation perspectives as to verify if the SRE methodology under validation facilitates to capture the actual needs of the stakeholders involved in the SRE process. In addition, the methodology also must facilitate to capture the environment assumptions as well as the domain constraints pertain to the security requirements. For example, in network environment, the domain constraints can correspond to the security zoning which will enable security experts to determine the security requirements pertinent to varying security levels of the zones. Respectively, we form the questionnaires (see the last column of C10 criterion Figure 30), which will help us to trigger the discussion with the stakeholders.

It is to note that, the table in Figure 30 lists the first set of criteria that we considered to elicit the characteristics of good SRE methodology. This selection does not mean anything particular. Instead they correspond to the initial set of quality criteria that the security experts at AIRBUS showed keen interest. We used this list for initiating our discussion in meetings/interviews during the elicitation phase. It even provoked the security experts to ask questions on their daily tasks.

2.4.2 Step2: Refinement (R^M) requirement analysis

The step2 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 into refined sub-goals fitting to specific demands of the security experts. This refinement process is also performed in collaboration with the security experts. Figure 31 provides the complete view criteria refinement using KAOS goal modelling notation. This goal refinement extends the study [Bulusu et al. 2018b; Bulusu et al. 2018a].

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 the evaluation criteria R^M . Tracing of the refinement characteristics goals 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.

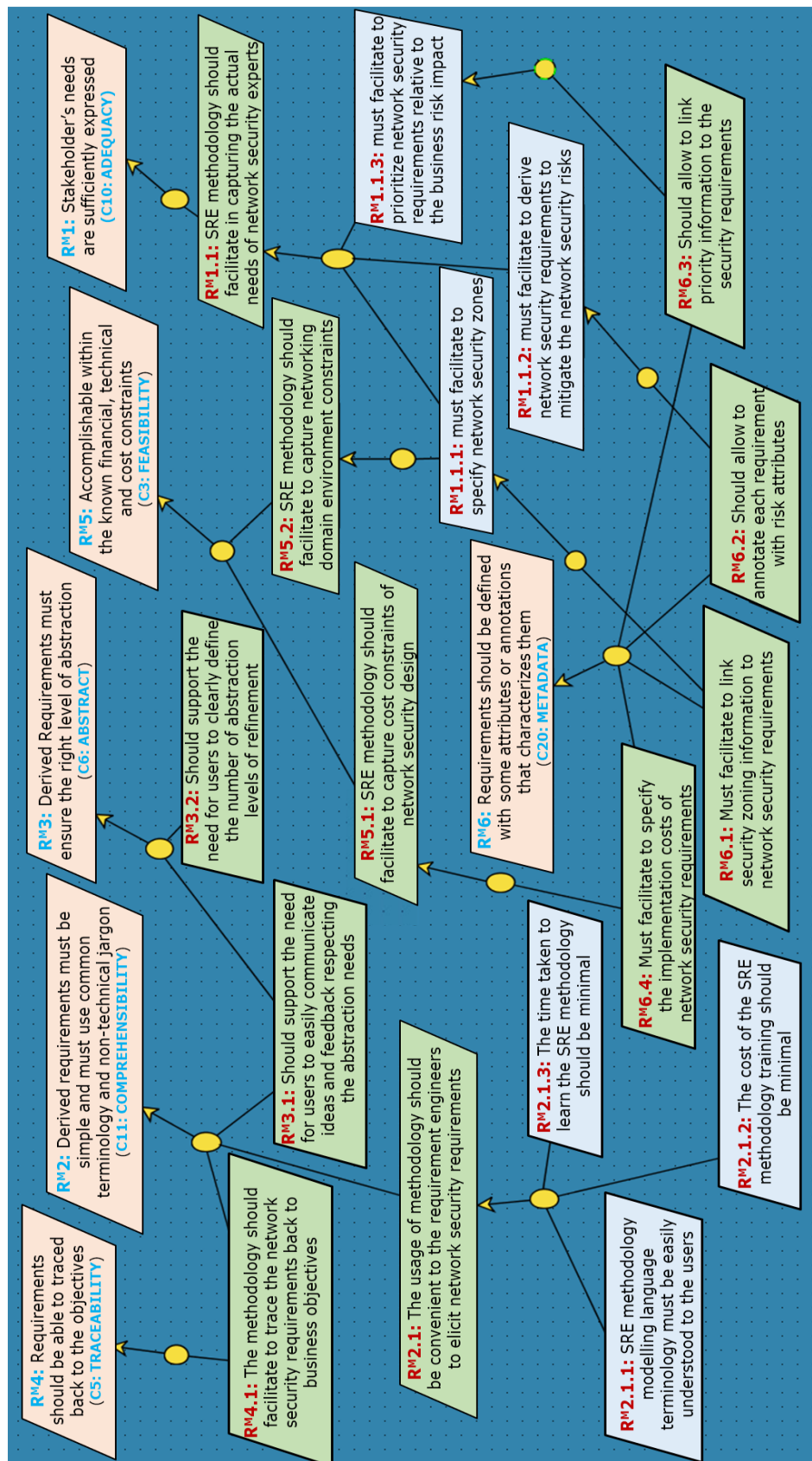


Figure 31: SRE-methodology-to-be goals refinement

In below we present our arguments relative to the refinement of 6 quality criteria given in Figure 30. It is to note that our arguments are based on our discussions and initial feedback with the security experts at AIRBUS. The goal refinement is subjected to changes with varying needs of the stakeholders.

2.4.2.1 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 the security experts at AIRBUS working in network SRE context. Respectively, we refine the high-level quality goal pertaining to the adequacy quality criterion (C10 **Figure 31**) reads “**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 as: **R^M1.1.1** and **R^M1.1.2** to validate if the SRE methodology facilitates to specify security zones and derive network security requirements to mitigate risks. However, in order to determine the rigor of security/validation measures required at each security zones, 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 facilities the prioritization of network security requirements.

However, to ensure the verification of the capture of this information, [ISO29148:2011] describes risk related information and environment constraints to be linked as requirement attributes/metadata (see C20 in Figure 29). 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**).

2.4.2.1 Comprehensibility (C11)

The refinement of quality criterion *comprehensibility* (C11) reads “**R^M2.1:** The usage of methodology should be convenient to the requirement engineers to elicit network security requirements”. This concern drives the main motivation for all the model-based RE approaches because a security requirement not understood cannot be analysed 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 RE methodology itself. That means, which language is understandable to the users? A formal language? UML? Or a natural language? This aspect completely depends on the language familiarity of the stakeholders who are going to use the methodology. 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.

[Kotonya and Sommerville 1996] 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. Besides, different stakeholders, despite their familiarity, can understand the same requirements differently, which requires negotiation schemes to calculate an agreement [Ahmad 2012]. In addition, learning and mastering a new language has a cost in terms of both money and time

[Mead 2007; Uzunov et al. 2012]. For example, a five day training course for SABSA framework costs around 3000 Euros [David Lynas]. In this thesis work, we do not consider these negotiation aspects. Instead, we confine our evaluation study to subjective analysis of degree of support to help understand the notation, learnability duration and ease of SRE modelling. This perspective is covered by the sub-goals **R^M2.1.2**, and **R^M2.1.3**.

2.4.2.2 Abstract (C6)

In practice, security requirements evolve gradually building upon the ideas/perspectives of the subject experts (e.g., business analysis, security architects, and network security engineers) who work at different levels of abstraction in a network development cycle. This requires effective communication among the subject experts and the requirement engineers. Therefore, the SRE modelling language should support the needs of the people to easily communicate the ideas and feedback relative to the security requirements elicitation [Kotonya and Sommerville 1996; Wieringa 1996].

On the other hand, with reference to the implementation free criterion definition from [ISO29148:2011], the SRE methodology has to ensure that the security requirements stated 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 if the SRE methodology facilitated the easy communication between the stakeholders and **R^M3.2** to if the SRE methodology allows to specify the definite number of abstraction levels for expressing the separation of concerns.

It is to note that, 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 31**). 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.

2.4.2.3 Traceability (C3) and Feasibility (C5)

Finally, the goals **R^M4.1**, **R^M5.1** and **R^M6.4** concern the supportability of SRE methodology with reference to traceability and feasibility characteristics (i.e., C3 and C5 in Figure 29) in network SRE context. Traceability is defined as the ability to establish the link between requirements to the source business objectives [ISO29148:2011]. So that it is possible 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 [ISO29148:2011]. As mentioned at the beginning of this document, these characteristics hold one of the primary objectives of IREHDO2 project since it enables to analyse the implementation costs of network security requirements relative to business objectives.

2.4.3 Elicited set of evaluation criteria (R^M)

As mentioned earlier, the refinement is performed until the final refined goals are realized as objectively verifiable. Only then the final sub-goals (leaf nodes) qualify as the evaluation criteria. Table 1 provides the final list of elicited evaluation criteria in our network SRE context in IREHDO2 project.

Table 1: Elicited evaluation criteria

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

Finally, our approach showed its benefits. We found new evaluation criteria (namely $R^M3.2$, $R^M6.1$ and $R^M6.4$) that were not proposed by previous comparison/evaluation studies (discussed in section 2.1).

2.4.3.1 Verification methods

The verification method expresses suggested ways to evaluate the SRE methodology against the evaluation criteria. However, the type of verification and respected performance metrics differs with respect to the type of evaluation criteria. We explain this using an example. Let's consider the evaluation criterion **R^M6.2**. Linking risk attributes to security requirements confirms the integration of risk analysis process to security requirement analysis. However, to what extent do SRE methods can support this integration features the central aspect of this verification method for **R^M6.2**, see Table 2. The performance metrics to measure the evaluation of this criterion.

The qualitative scale used for 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 types of measurements (i.e., nominal/ordinal/interval/ratio[Stevens 1946]) also vary with regards to types of criteria. The *nominal* scaling uses some labels (or names) to differentiate between different subjects. The *ordinal* scaling uses rank ordering to sort the subjects (e.g., less/more or small big). Finally, *ratio* and *interval* scales use some quantitative numeric values, which allows to estimate the degree of difference between the subjects in terms of some meaningful measurement units (e.g., length, duration, angles etc.). Ratio scales are very similar to interval scales. However, the interval scaling described the exact difference between the units (e.g., number of hours or number of days), which ratio scales allows to describe the difference in fractions (e.g., number of hours per day).

Table 2: Verification method for R^M6.2

R^M 6.2: Verification method	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

For instance, the measurement scale for **R^M6.2** reflects the nominal 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 in Appendix A. 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 example verification technique employed for this criterion can refer to the satisfaction survey of the security experts. Respectively, **R^M2.1.1** uses an ordinal scale, which sorts the understandability factor of SRE methods based on the degree of support required; **R^M2.1.3** uses an interval scale, which allows to estimate the degree of learnability of the terminology used by SRE-methodology in terms of the number of weeks. Finally, evaluation criterion **R^M2.1.2**, related to the cost of the training, 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 criteria is accorded with the verification method, which will be used during the evaluation process. In addition, the number of levels of scaling may also differ with regards to the type of verification methods. For instance, the verification of **R^M6.3** concerns verifying if the SRE methodology allows to attribute security requirements with priority information. This verification reflects binary information i.e., *yes* if it allows and *no* if it doesn't allow. Respectively, we have only two scales i.e., *high* for *yes* and *nil* for *no*.

It is to note that these verification methods and measured metrics must be defined in compliance with the needs of security experts who intend to use SRE methodology-to-be. This clear separation of evaluation criteria list to the verification methods and respective measurement metrics permits the

reusability of the evaluation criteria as well as facilitates in justifying the subjectivity of the criterion interpretation and relative evaluation perspectives. Appendix C provides the complete set of evaluation methods and measurement metrics that we have derived in our work.

2.5 Step3: Evaluation of SRE Approaches

This section describes the final step 3 of our SRE evaluation methodology. It concerns the evaluation of SRE approaches i.e., goal-oriented, agent-oriented and problem frames oriented with the help of the elicited evaluation criteria in step2. Our evaluation is a qualitative assessment that is performed in collaboration with the security experts at AIRBUS. This task has taken us to organize two meeting sessions (i.e., precisely 3 hrs brainstorming and discussions for each of the meeting) with a time gap of around 2 months.

We performed the evaluation in two iterations. In the first iteration, we authors acted as stakeholders (i.e., SRE analysts) and tested the three SRE approaches i.e., STS, Secure KAOS and SEPP using an example scenario. In this regard, 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 one of them has been asked to elaborate the system-to-be using the methodology that the person is familiar with. During this process, the persons (acting SRE analysts) were not allowed to communicate with each other during the scenario analysis phase. Each of them has come up with a different list of security requirements for the system-to-be with respect to the. The resulting SRE models have been presented during a meeting that involved the security experts. Their feedback is recorded as our initial evaluation results, which enabled us to identify and select the SRE approach the found interesting to the experts at first instance.

In the second iteration, the AIRBUS SRE experts are directly asked to test the SRE approach that found interesting to them 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 evaluation experience. In below we provide the details of the evaluation performed in two iterations.

2.5.1 Evaluation Iteration 1 – Use case Scenario 1

We first present the use case scenario used in this iteration. The scenario concerns a task related to the aircraft maintenance process ensuring the compliance with safety regulations such as Airworthiness directives. The objective of this task is to anticipate the health of 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 constitutes of two applications i.e., aircraft control application and monitoring application (see Figure 32). These two applications are connected to each other via an internal avionic bus network (represented as double arrow line in red colour in Figure 32).

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 permitted only in 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 assumed as untrusted.

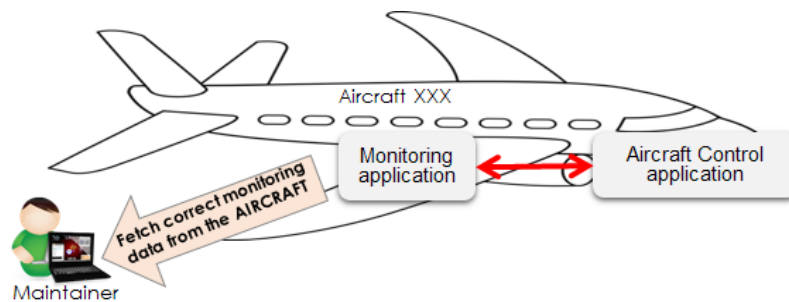


Figure 32: Example scenario 1

The business risk impact of this scenario is expressed in terms of the integrity and availability of the monitored parameters. Although these parameters do not threaten the safety of the aircraft, they may impact the critical decision related to deciding whether the aircraft is ready to fly or not. For instance, the 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 some problems when the faulty aircraft is assumed as correctly functioning and permitted to take the next trip.

Respectively, the security objectives concern the protection of the integrity and availability of the monitored parameters. The aircraft network security design solutions must ensure a trusted transmission of parameters from the monitoring application to the monitoring application and then to the laptop of 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 that network security requirements need to be specified in order to closely monitor communication traffic between laptop and the monitoring application. In this regard, some example security measures may include: VPN connection with some strong encryption algorithm, configuring firewalls, authentication on the WIFI access point or router. The security experts at AIRBUS wanted to confirm if these measures are sufficient to mitigate the risks. In addition, they wanted to verify the cost of 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 the three 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 subjected to which evaluation criteria. Finally, in section 2.5.1.4, we provide the initial feedback and evaluation results of iteration 1.

2.5.1.1 Scenario implementation using Secure KAOS

For the implementation, we used the KAOS tool known as *Objectiver* [Respect-IT]. The experimentation was conducted using the free trial version which was valid for three months (refers to the criterion $R^{M2.1.2}$). **Figure 33** depicts a sample goal model specified in our example scenario context (refer section 1.4.1.1 for the details on KAOS modelling notation). It required some effort to get acquainted with the tool and its terminology through the help of guidelines and cited references (refers to the criterion $R^{M2.1.1}$ and $R^{M2.1.3}$). In particular, the security experts expressed some concerns regarding the goal patterns based on formal temporal logic.

KAOS drives RE analyst to define *agents* later in the RE process. As consequence, it does not help in expressing the relation between the agents and their interaction dependencies. For instance, while defining network agents (e.g., firewalls) in our scenario analysis, we encountered issues when we had to add a new device in the network. In this regard, a conceptual diagram of the network would have been more concise and clear. For example, it is not clear on how to express firewall configurations that restrict access to aircraft to only maintenance people? As consequence, we had difficulty in specifying network security requirements on network agents, particularly relative to security zoning ($R^{M3.1}$). Furthermore, KAOS modelling perspective the abstractions is expressed using the four modelling activities (i.e., goal, responsibility, object /operation) (refer section 1.4.1.1). However, this abstraction perspective does not facilitate to explicitly differentiate goals/agents of the abstraction levels in network security engineering process ($R^{M3.2}$). Otherwise, KAOS modelling notation provides good support to achieve traceability ($R^{M4.1}$).

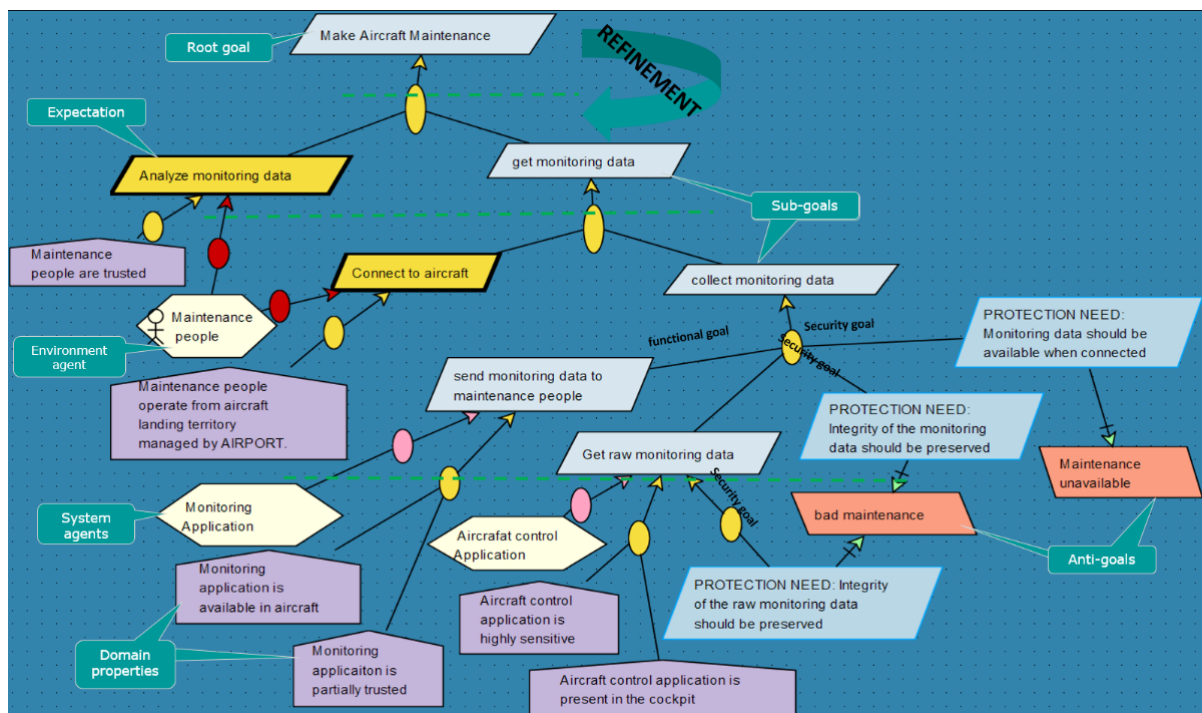


Figure 33: Secure KAOS goal specification (sample)

Coming to the integration of risk analysis concepts, a 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 33**, the security goal “PROTECTION

NEED: monitoring data should be available when connected” resolves the anti-goal “maintenance unavailable”. The resolution link is expressed using green arrowhead. These anti-goals can be further refined like ‘normal’ goals resulting in the specification of attack trees. The obstacles and 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 specifying domain properties. (e.g., physical laws). For instance, the trust assumptions on agents are expressed using the domain properties, see **Figure 33**. Table 3 consolidates the individual evaluation of Secure KAOS relative to the verification methods and measurement metrics given in Appendix C.

Table 3: Individual evaluation of Secure KAOS

Evaluation criteria (R^M)	Secure KAOS
$R^M2.1.1$: The SRE methodology modelling language terminology must be easily understood to the user	medium
$R^M2.1.2$: The cost of the SRE methodology training should be minimal	medium
$R^M2.1.3$: The time taken to learn the SRE methodology should be minimal	medium
$R^M3.1$: Should support the need for users to easily communicate ideas and feedback respecting the abstraction needs	nil
$R^M3.2$: Should support the need for users to clearly define the number of abstraction levels of refinement	nil
$R^M4.1$: The methodology should facilitate to trace the network security requirements back to business objectives	high
$R^M6.1$: Must facilitate to specify and link network security zone information	nil
$R^M6.2$: Should allow to annotate each requirement with risk attributes	medium
$R^M6.3$: Should allow to annotate each requirement with priority information	high
$R^M6.4$: Must facilitate to specify the implementation costs of network security requirements	nil

2.5.1.2 Evaluation of STS Methodology

STS is supported by a tool of the same name. In our evaluation, we explored the social modelling of STS notation (see section 1.4.2.1). Figure 34 depicts an example of secure problem frames specification. It is not an open source but the tool is freely available to public use ($R^M2.1.2$). The usage of simple terminology as well as the user friendly tool took less effort to get used to the overall concepts and terminology ($R^M2.1.1$ and $R^M2.1.3$). The security experts provided positive feedback concerning the comprehensibility aspects.

Unlike KAOS, STS does not support full-fledged traceability of the security goals being refined into security requirements ($R^M4.1$). Because, from 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 social view; or in the form of constrained permissions (e.g., *read/ modify/ produce/transmit*) upon the provision of the resources in authorization view. Figure 34

highlights the security constraints over delegated goals in blue dashed rectangle, which cannot be refined further into network security requirements ($R^{M3.1}$).

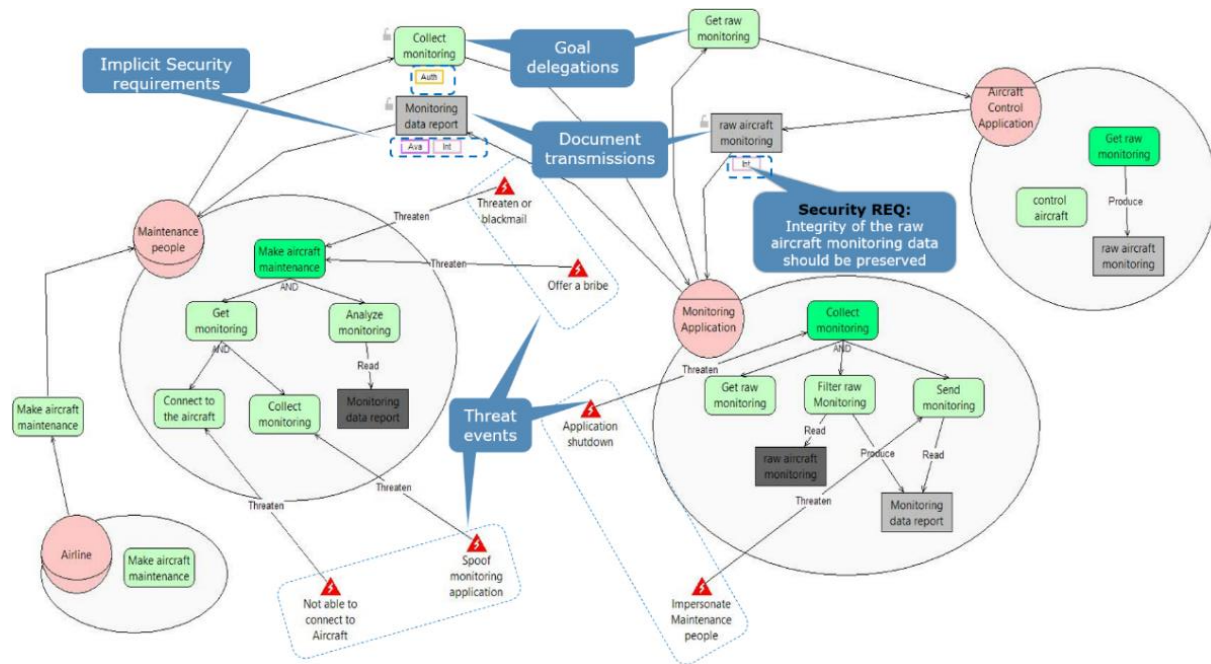


Figure 34: STS social view specification (sample)

On the other hand, STS provides good support to express network agents and interaction dependencies. However, we had an issue in handling the dependencies between the multiple network agents as the social view grow larger. Furthermore, from STS modelling perspective the abstractions is expressed using the three modelling views (i.e., social view, authorization view, information view) as described in section 1.4.2.1. This abstraction perspective respects the separation of concerns specific to authorization security problem context, which does not help to explicitly describe the separation of concerns of the agents/actors specific to network security problem context ($R^{M3.2}$).

In STS, the *threat analysis* is used to show the effects over goal trees and goal/resource relationships that may have a threatening event. Also, 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 expressing of security needs upon the threats related to environmental risks. Also, it does not have a facility to prioritize goals or threats like in KAOS ($R^{M6.3}$). Table 4 consolidates the individual evaluation of STS. Figure 34 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, on can witness how the threat impact traces back to the root objective node. Respectively, the “*maintenance unavailable*” threatens the business objective “run business”.

Table 4: Individual evaluation of STS

Evaluation criteria (R^M)	STS
$R^{M2.1.1}$: The SRE methodology modelling language terminology must be easily understood to the user	high
$R^{M2.1.2}$: The cost of the SRE methodology training should be minimal	high

R^M2.1.3: The time taken to learn the SRE methodology should be minimal	high
R^M3.1: Should support the need for users to easily communicate ideas and feedback respecting the abstraction needs	nil
R^M3.2: Should support the need for users to clearly define the number of abstraction levels of refinement	nil
R^M4.1: The methodology should facilitate to trace the network security requirements back to business objectives	medium
R^M6.1: Must facilitate to specify and link network security zone information	nil
R^M6.2: Should allow to annotate each requirement with risk attributes	low
R^M6.3: Should allow to annotate each requirement with priority information	nil
R^M6.4: Must facilitate to specify the implementation costs of network security requirements	nil

2.5.1.3 Evaluation SEPP Methodology

SEPP guides the RE analyst to define security problem patterns, called as security *problem frames* (SPF). Figure 35 depicts an example of secure problem frames specification for integrity protection scheme. The security requirements is constrained on the communication network domain. The link between domains depicts that there is some shared phenomena (i.e., shared attributes) (as described in section 1.4.3). The non-availability of a tool for SEPP made our experimentation difficult (**R^M2.1.2**). It took some extra effort to get acquainted with the concepts and terminology even with the help of the cited references (**R^M2.1.1** and **R^M2.1.3**). Therefore, 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 issues in knowing all the acting domains in a network environment, in particular during the early stages. Having a network design will facilitate the problem analysis.

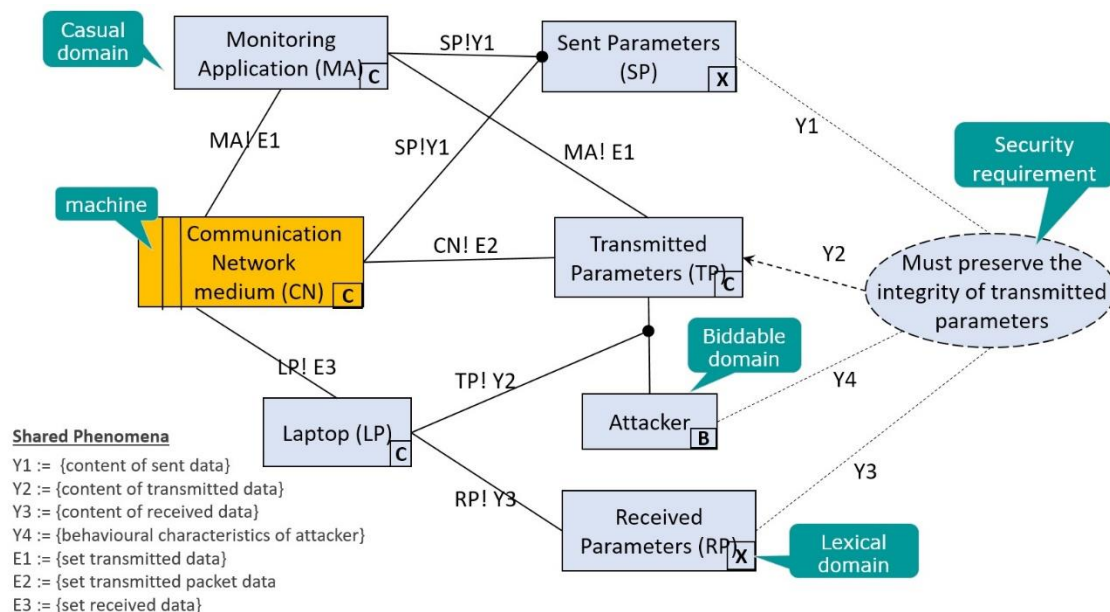


Figure 35: SEPP SPF diagram (sample)

From SEPP perspective, the security problems (security goals) are identified using the what-if analysis technique similar to the Hazard analysis [Kletz 1999]. But 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 a generic perspective. However, this abstraction perspective respects the separation of concerns specific to authorization security problem context, which does not help to explicitly describe the separation of concerns of the agents/actors specific to network security problem context ($R^M3.2$). Lastly, similar to STS, it does not support to attribute security requirements with either priority or implementation cost related information ($R^M6.3$ and $R^M6.4$). Table 5 consolidates the individual evaluation of STS.

Table 5: Individual evaluation of SEPP

Evaluation criteria (R^M)	SEPP
$R^M2.1.1$: The SRE methodology modelling language terminology must be easily understood to the user	low
$R^M2.1.2$: The cost of the SRE methodology training should be minimal	low
$R^M2.1.3$: The time taken to learn the SRE methodology should be minimal	low
$R^M3.1$: Should support the need for users to easily communicate ideas and feedback respecting the abstraction needs	nil
$R^M3.2$: Should support the need for users to clearly define the number of abstraction levels of refinement	nil
$R^M4.1$: The methodology should facilitate to trace the network security requirements back to business objectives	low
$R^M6.1$: Must facilitate to specify and link network security zone information	nil
$R^M6.2$: Should allow to annotate each requirement with risk attributes	nil
$R^M6.3$: Should allow to annotate each requirement with priority information	nil
$R^M6.4$: Must facilitate to specify the implementation costs of network security requirements	nil

2.5.1.4 Evaluation iteration 1 – consolidated results and discussion

In Table 6, we resumed the evaluation results of the SRE methodologies to provide a consolidated view for comparison (consolidates Table 3, Table 4 and Table 5). First column displays the high-level root goals of evaluation criteria (refer Figure 31). The second column lists the elicited criteria that correspond to the leaf nodes in Figure 31. The remaining three columns display the respective measurement scales for each of the three SRE methods.

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 allow to annotate each requirement with risk attributes	low	medium	nil
Adequacy (C11) Metadata (C20)	R^M6.3: Should allow to 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

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 the integration of risk analysis information. Whereas, 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, it is not known how to express the security problem context of all the stakeholders including the business analysts, and the security analysts of all systems connected to the network. Finally, the quality criterion *feasibility* has been refined into ‘The SRE-methodology-to-be helps in considering the cost of security

requirements'. However, none of the three methodologies allows to express the financial attributes (R^M6.4).

2.5.2 Evaluation iteration 2 – use case scenario 2

As mentioned earlier, the first iteration of evaluation the authors acted as stakeholders and tested the SRE models, which are then presented to the stakeholders i.e., security experts, to capture their feedback. The second iteration encompasses active participation of stakeholders (i.e., security experts from AIRBUS) for evaluating the STS modelling notation. The reason for choosing STS is evident as it holds high rating in terms of comprehensibility aspects. The purpose of this second evaluation is capture the feedback from user's real-time experience of SRE modelling in STS using a new use case scenario. In below we first provide the scenario in a plain textual description as presented to us as an initial step.

The scenario contains two business needs, see Figure 36. 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 centre is a sub-component in the aircraft equipment, which reads these load parameters and performs an auto-update. The security needs of this activity concern that the aircraft network security design must ensure the secure transmission of these parameters between the mobile devices to the on-board update centre application. Otherwise, the direct connectivity of the mobile devices with the aircraft equipment is forbidden.

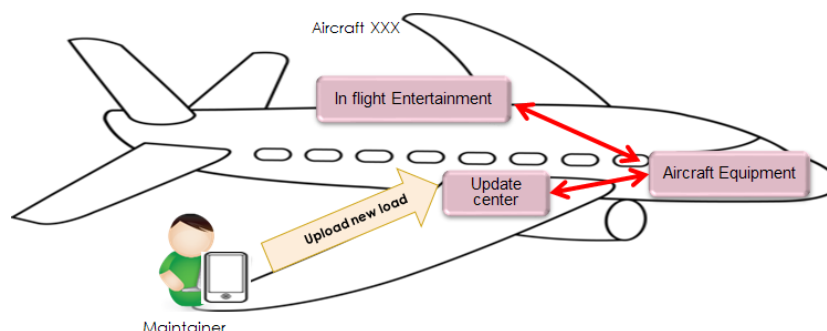


Figure 36: AIRBUS new scenario

The second business need concerns in-flight entertainment services to improve passengers' in-flight 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.

2.5.3 Evaluation iteration 2 – STS modelling and feedback discussion

From our initial analyses, we already know by now that STS provides no support to describe the separation of concerns. In this experiment, we investigate if we can apply SABSA layered approach

(given in section 1.3.2) to STS modelling (given in section 1.4.2.1) and see how it suites. In SABSA, network security requirement analysis (related to security zoning) starts from designers view (i.e., LV3). Since STS do not support to derive network security zoning therefore we confine this experiment to the first two view alone (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 has helped in improving the elicitation activity by provoking a long discussion. Indeed, this strict confinement to the 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 the provoked discussion. This has eventually developed the scenario details as well, see Figure 37.

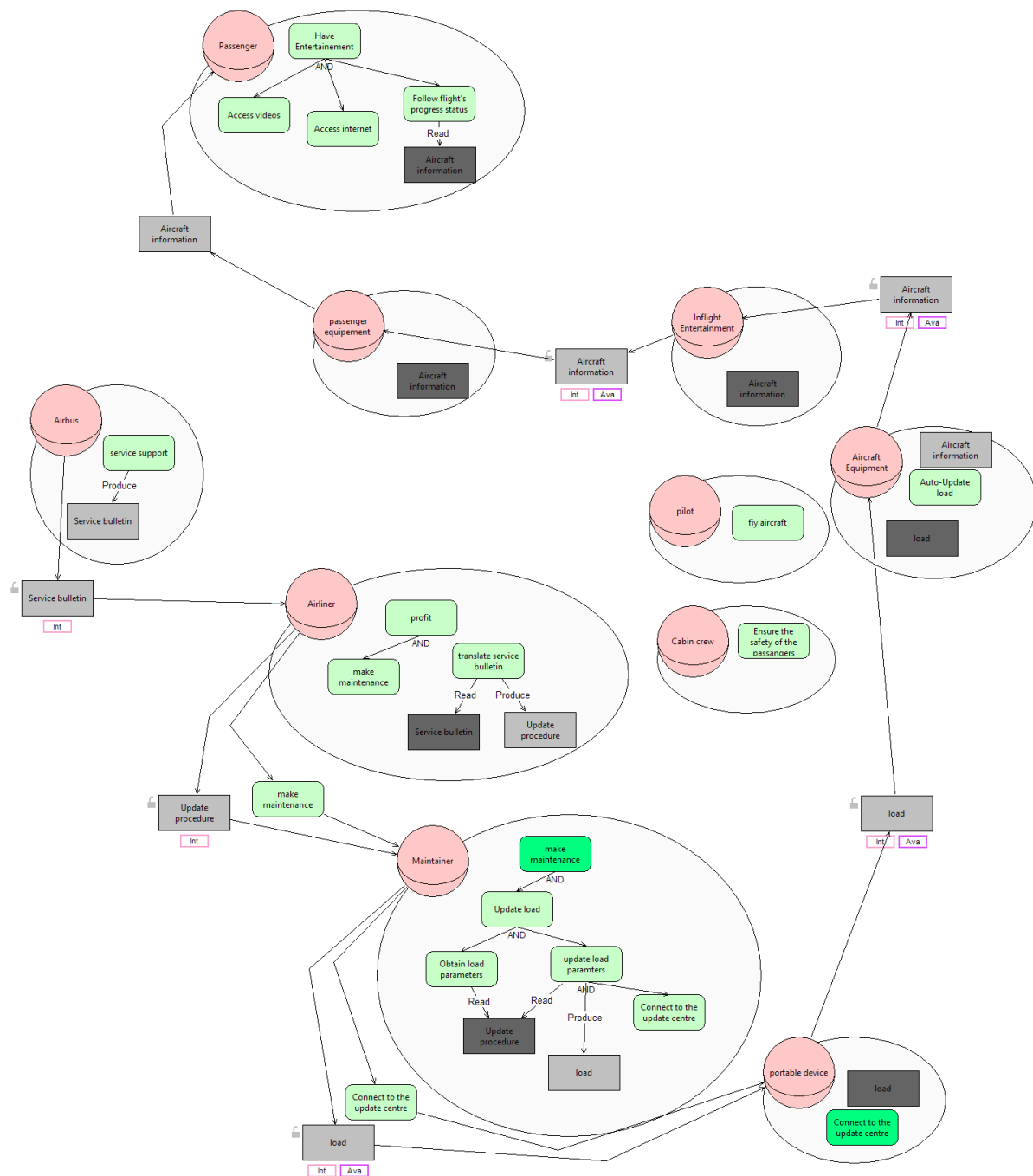


Figure 37: Use case scenario 2 STS social view

Initially, the description of this scenario was expressed in few lines. However, during the process of SRE modelling, the provoked discussion has developed the scenario details which resulted in the following social view in Figure 37. In particular, for the first business need, new agents such as AIRBUS or airline are introduced, which were not mentioned during the textual description of the scenario given in Figure 36. In addition, we observed that the goals of the maintainer agent are enhanced as well. However, the second business need related to passenger entertainment system remained unchanged. The respective textual description of the enhanced scenario after SRE modelling in STS is as follows.

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 in compliance with the aircraft service bulletin. The service bulletin comprises of necessary settings and measured parameters to ensure the safe and correct functioning of the aircraft. The aircraft manufacturing company (e.g., AIRBUS) provides the service bulletin to the airline as part of the contractual agreement. The airline translates the service bulletin to create an update procedure and assigns the task of updating the aircraft to the responsible person from the maintenance team. The person responsible for maintenance task (i.e., maintainer) generates the customized load information with reference to the update procedure and transmits them to the aircraft's update centre application through a mobile device.

To consolidate, the overall feedback of STS notation from the two iterations confirms that STS modelling is interesting and easy to adapt. The overall experiment took us 2 hours of time, which proves the fact that SRE modelling concepts can be learned in short time (**R^M2.1.3**). However, the STS modelling requires the users to know all the agents and their respective before beginning the process. Our experiment shows that in real-time it is not the case because the information on agents as well as their goals is evolved gradually during the elicitation process.

Although STS modelling concepts are interesting and easy to adapt, it does not help to describe 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 the layering but require some enhancements in terms of integrating risk analysis concepts and traceability attributes.

2.6 Chapter Summary

In this chapter, we described an evaluation methodology that facilitates the evaluation of SRE methodologies. This methodology principally explores the research questions what are good security requirements and what is a good SRE methodology. Since there exists no complete list of quality characteristics, we first provided a unified list of 20 criteria.

In our evaluation context, we considered 6 quality criteria (i.e., adequacy (C10), comprehensibility (C11), traceability (C5), abstract (C6), feasibility (C3) and metadata (C21)), which resulted in 10 evaluation criteria (i.e., R^M2.1.1, R^M2.1.2, R^M2.1.3, R^M3.1, R^M3.2, R^M4.1, R^M6.1, R^M6.2, R^M6.3 and

R^M6.4). These criteria correspond to the anticipated characteristics of good SRE methodology for network security from the perspective of security experts involved in IREHOD2 project.

The evaluation study of SRE approaches (i.e., Secure KAOS, STS and SEPP) shows that each approach exhibits different features. In particular, STS is very 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). Whereas, the modelling terminology and notation of SEPP seemed difficult to learn and adapt.

When considering SABSA layered framework for abstraction, STS modelling is an interesting choice for expressing high-level security objectives. But it requires some enhancements from risk analysis part as well as traceability aspects. Finally, we needed a methodology to define network security requirements and cost attributes relative to business risk impact.

CHAPTER 3 PROPOSED SRE METHODOLOGY – A LAYER BASED APPROACH

From our evaluation experience, we confirm that an SRE approach suitable for network security requirements analysis is missing. On the other hand, security zoning is a widely recognized practice for analysing early stage network security requirements but lacks a rigorous approach (refer section 1.5.1). These issues gave rise to our second research question RQ2: How to define a good SRE methodology that facilitates the network security requirements elicitation and refinement i.e., from high-level business objectives into low-level network security zoning requirements.

Since networks are intermediary that interconnects all the software/systems within an enterprise, we need to build a new SRE methodology, which must facilitate involving all the stakeholders with different viewpoints. It should also integrate security zoning as it concerns the security problem context of security architects, which will help in rightly distributing security controls across network security architecture relative to business risk impact.

We adopted SABSA layered framework for our SRE methodology. We consider the first three layered views of SABSA (i.e., Business view, Architect's view and designers view), which corresponding to the early phases of SRE process. We use STS modelling to express high-level security requirements at Business view and Architect's view. For risk analysis part, we propose a new extension of STS called Anti-STS for modelling attacks as multi-agent systems in a network. At Designer's view, we propose a methodology that automates parts of the process of eliciting security zones and derived network security requirements. Finally, to link the three abstract views, we propose a modelling notation that merges STS and secure KAOS modelling concepts.

This chapter is dedicated to present our three-layered SRE methodology and its illustration using the use case scenario 1 (presented in section 2.5.1). We first present the conceptual model of our proposed SRE methodology, which describes our layer based SRE modelling (section 3.1). In this respect, we describe the modelling notion of proposed SRE methodology and Anti-STS, for business and architect views. Next, we introduce the concepts as well as the formal approach of our zone modelling methodology for Designer's view. Finally, we illustrate our SRE methodology in detail using the use case scenario, which describes linking of three layers (section 3.2).

This chapter consolidates the following contributions [Bulusu et al. 2017b; Bulusu et al. 2018c; Laborde et al. 2019].

3.1 SRE methodology conceptual model

The conceptual model of our SRE methodology handles three issues: First, to apply SABSA abstraction for integrating the concepts of STS and KAOS. Second, to integrate risk analysis modelling

in order to enhance the risk analysis concepts of STS and Secure KAOS in order to suite to network environment. Third and finally, to define a formal network security zoning. We describe how our model addresses these issues in the following.

3.1.1 Layer based abstraction modelling concepts

The Figure 38 depicts the underlying conceptual model of our layer based SRE methodology. It is built based on the SABSA framework (refer section 1.3.2). Since we focus on early stages of security requirements engineering, we considered only the first three layered views. Each layered view (LV) represents the requirement-engineering context of the people working at different abstraction levels during a security engineering process. We refer to these entities as stakeholders who include business users, architects, designers, builders and the tradesmen. Starting with the security needs identified at Business view, each successive layer introduces a new level of abstraction and detail towards the design and implementation of the network security design.

Respectively, the *Business view* deals with the capture of business objectives needs and constraints that are important to do business. The *Architect's view* concerns with the capture of high-level security needs related to the protection of the security entities involved in the fulfilment of the business objectives. The elicitation activity at the *Designer's view* concerns with the capture of high-level network security requirements. Together, these views enable the separation of concerns and security needs respecting the abstraction needs of the people.

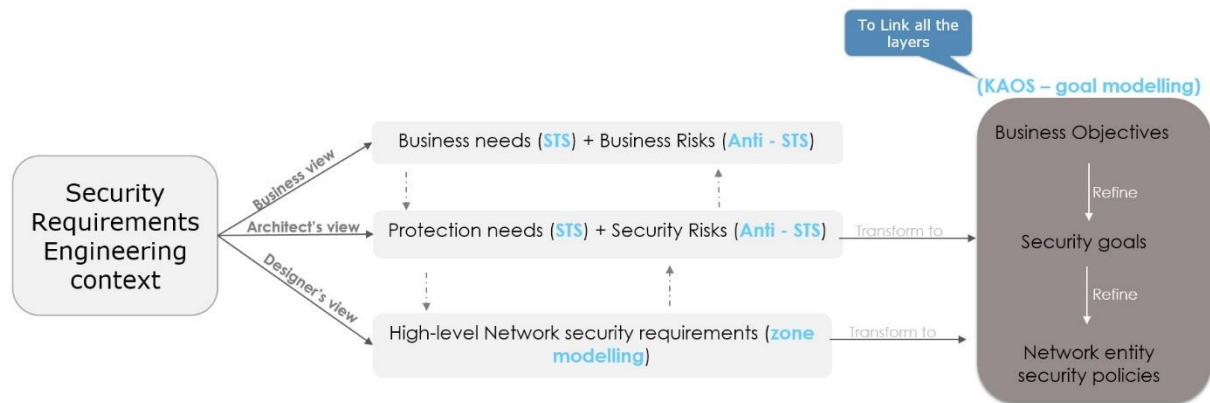


Figure 38: Initial conceptual model of our layer based SRE methodology

Because our methodology constitutes different layered views involving different stakeholders, this impelled the necessity of choosing rightful SRE methodologies that are suitable with regards the comprehensibility level of the stakeholders. So that it becomes easier to establish good communication with the stakeholders while eliciting and evaluating the requirements. Therefore, we propose to model the requirement needs and risks using different approaches at different layers. Thanks to our evaluation feedback in Table 6, which helped in choosing the SRE approaches suiting the abstraction needs.

We use STS, for expressing the needs of first two views i.e., *Business view* and *Architect's view*, since our feedback evaluation from chapter 2 shows that it is easy to adapt (see Figure 38). However, when coming to risk analysis, STS seemed less interesting, so we propose a new multi anti-agent threat model named Anti-STS that describe the social dependency between attacking agents in a network

environment. Finally, at designers view, we propose a new zone modelling methodology based on the security design principles.

Since we are using different SRE methodologies, it is necessary to link all the information and the elicited requirements goals provided in every layer. This is to ensure the traceability between the business objectives and the high-level network security requirements. For this, we propose to employ the KAOS goal modelling notation as it is good for traceability (see **Table 6**). Our idea is to transform the knowledge of each model at the end of each abstract view into KAOS goal modelling notation (see Figure 38). So that, in the end, we shall have one final specification language i.e., in KAOS.

We tested this conceptual model using the use case scenario 1 and presented the same to the security experts. From our experience, we observed an additional overhead when we had to transform the results of each view into KAOS goal modelling notation. Furthermore, our security experts expressed concerns that this conceptual model requires the users to expertise KAOS goal modelling as well. Considering this feedback, we decided to propose a new modelling notation that merges the concepts of STS and KAOS, which modifies our conceptual model as shown in Figure 39.

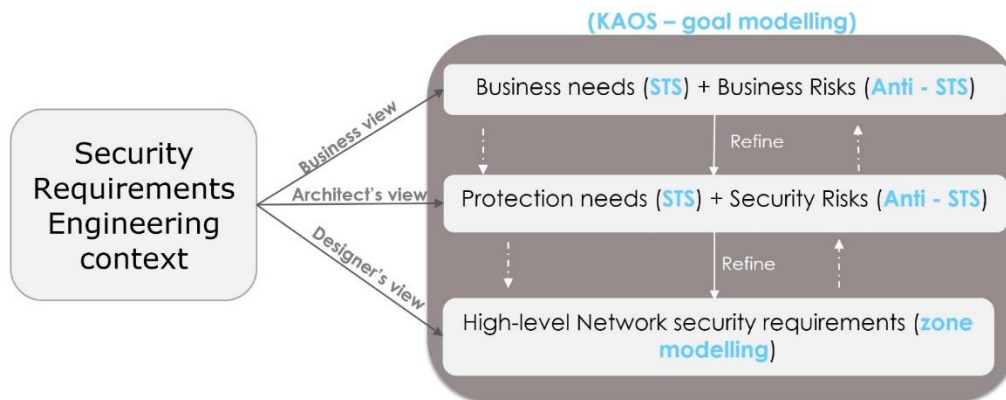


Figure 39: Finalized conceptual model of our layer based SRE methodology

3.1.1.1 Our proposed SRE modelling notation

We were careful to stay simple when coming to introducing new modelling notation as it would impact the comprehensibility aspects. Therefore, we have given priority to reuse the SRE modelling concepts of STS and Secure KAOS since SRE community is familiar with them. The purpose of merging is to accommodate abstraction layers without disturbing the modelling features of STS and Secure KAOS. This way, we achieve the traceability features as well as we can able to introduce the characteristics of the abstraction level.

In practice, the business users have an idea and the same idea is taken forward to the people working at different levels of the security engineering process. In this process, the information needed to realize the idea into proper requirements that are revealed in an incremental manner at each abstraction view. Goal refinement on Secure KAOS reflects this perspective, which allows the refinement of business objectives into security requirements. While a goal always captures the ‘WHAT’ perspectives of the needs, the ‘WHY’ and ‘HOW’ perspectives of the needs reflects the rationale subjected to the refinement process (see Figure 40). Also, in order to justify the refinement of the goals from high-level objectives we need to clearly specify the rationale behind the goal refinement.

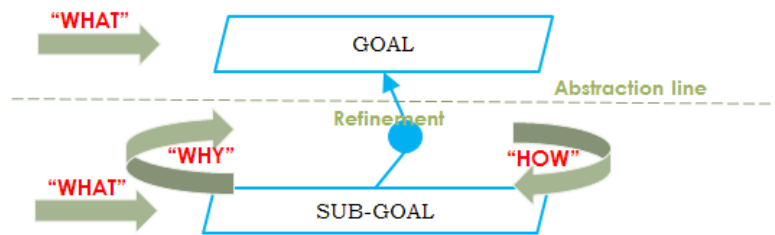


Figure 40: generic goal refinement rationale (refer KAOS)

This goal refinement is similar to the underlying perspective of abstraction layering of our conceptual model. Wherein, in our model, each layer comprises of STS modelling concepts i.e., social actors, goals and document resources. As the refinement proceeds to the next levels, the precision over the goal refinement enhances and the social actors at high abstraction level are evolved into the system entities. To express this type of refinement, we introduce a new notion called *composite requirement*, which takes the form of goal used in KAOS model, see Figure 41.

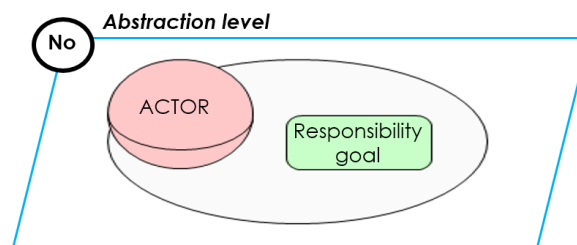


Figure 41: Composite requirement notation

The composite requirement is expressed using a parallelogram with blue coloured frame that encompasses STS modelling, see Figure 41. Each composite requirement includes social actors concerned with the business need. This implies that the composite requirement helps to confine the grouping the social actors with respect to an abstraction level. This composite requirements corresponds to a goal in KAOS, however, the goal information is expressed using STS social modelling notation. Every composite requirement frame must contain at least one actor with a responsibility goal which is visually expressed as shown in Figure 41. This is where we successfully integrate the abstraction levels by retaining the SRE modelling features of STS and KAOS modelling notations.

In addition, the composite requirement comes with two mandatory attributes. First, attribute concerns numbering to achieve **uniqueness of the level**. The second attribute concerns the meta-data defining the layered view to express the **abstraction view** the composite requirement belongs to it. It is to note that unlike to SABSA, in our notation, an abstraction view may include more than one refinement levels. For instance, we may require two refinement levels to express the Business view. This improves the flexibility of refinement, which is not possible in SABSA.

To consolidate, we formally express these integration rules as follows. Let S be the network system-to-be of our research project context is formally represented as: $S = \langle \text{COMPREQ}, \text{AGENT}, \text{GOAL}, \text{SCOPE}_G^A \rangle$, Where,

- GOAL is the set of responsibility goals of an agent
- AGENT is the set of agents
- COMPREQ is the set of composite requirements

- $\text{SCOPE}_G^A \subseteq \text{GOAL} \times \text{AGENT}$ is a relation that states a goal is within the scope of an agent.
- $\text{NUM}: \text{COMPREQ} \rightarrow \mathbb{N}$ returns the identification number of the refinement level of the composite requirement
- $\text{LV}: \text{COMPREQ} \rightarrow \{\text{Business view, Architect's view, Designer's view}\}$ returns the layered abstraction view of the composite requirement
- $\text{REFINE_LEVELS} \subseteq \text{LV} \times \text{NUM}$ is a relation that states the an abstraction view has a refinement level

Respectively, the newly introduced formal rules related to our composite requirements are as follows:

Composite Requirement – Rule1: The scope of every composite requirement must contain at least one agent. Wherein, the scope of every agent must contain at least one goal [Paja et al. 2014].

$$\forall cr \in \text{COMPREQ}, \text{card}(\{a \mid a \in \text{AGENT}, (a, cr) \in \text{SCOPE}_{cr}^A\}) \geq 1$$

$$\forall a \in \text{AGENT}, \text{card}(\{g \mid g \in \text{GOAL}, (g, a) \in \text{SCOPE}_a^G\}) \geq 1 \text{ [Paja et al. 2014]}$$

Composite Requirement – Rule2: Every composite requirement must be attributed with the level of abstraction view i.e., Business view or Architect's view or Designer's view. Wherein, an abstraction view may contain more than one refinement levels.

$$\forall cr \in \text{COMPREQ}, \exists \text{LV}(cr) \mid \text{LV}(cr) \in \{\text{business view, architect's view, designer's view}\}$$

Composite Requirement – Rule3: An abstraction view can have more than one refinement levels.

$$\forall cr \in \text{COMPREQ}, \text{LV}(cr), \text{card}(\{n \mid n = \text{NUM}(cr), (n, \text{LV}(cr)) \in \text{REFINE_LEVELS}\}) \geq 1$$

In addition, our notation retains the delegation features of STS modelling. Furthermore, in KAOS, goals assigned to environment agents (i.e., human agents) are described as expectations. We add this concepts to STS modelling. If a goal is delegated to an environment agent, then the delegated responsibility goal is expressed as an **expectation**. An actor is considered as an environment agent in two cases: either the actor is not managed by the organization or the actor is identified as human agent. An expectation allows us to bring the notion of trust over such entrusted actors towards the fulfilment expected goals or delegated expectations. Furthermore, the refinement composite requirement as well as the responsibility goals can be refined using **AND/OR** constructs, see Figure 42.



Figure 42: Refinement links of STS and KAOS

However, the characteristic features of refinements differ with respect to the type of refinement. If the refinement of the responsibility goal is within the scope of the layered abstraction view, then goal refinement is specified within the same composite requirement frame; otherwise the goal refinement is expressed in new composite requirement frame. In other words, a responsibility goal of an agent can be refined within the same composite requirement or in a new composite requirements depending on the

scope of the refinement relative to layered abstraction view. For instance, in Figure 43, the refinement of the responsibility goal of the actor at highest abstraction level is expressed in the lower abstraction level. Respectively, the composite refinement links comes with a *rationale* attribute to allow the description of the reasoning behind the refinement. This facilitates the goals' tracing both forward and backward. We formally express this rule as follows:

- $\text{REFINE_LINK} \subseteq \text{COMPREQ} \times \text{COMPREQ}$ states the refinement relation between two composite requirements
- **Rationale:** REFINE_LINK returns the rationale behind the refinement of a composite requirements

Composite Requirement – Rule4: Every refinement link of a composite requirement must be explicitly defined with a rationale statement

$$\forall cr1, cr2 \in \text{COMPREQ}, \forall (cr1, cr2) \in \text{REFINE_LINK}, \exists \text{Rationale}(\text{REFINE_LINK})$$

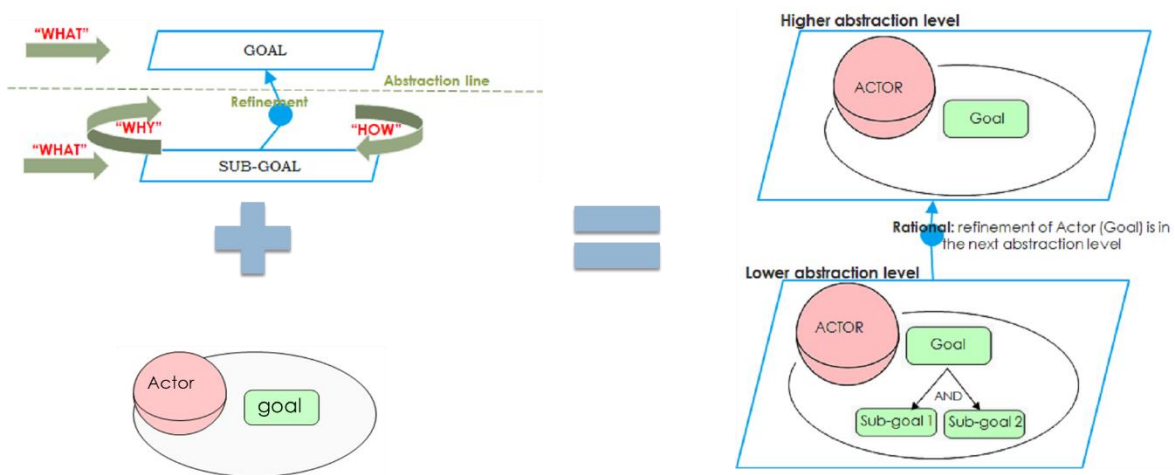


Figure 43: composite requirement refinement

However, it is to note that, this rationale of refinement is not supported by KAOS or STS modelling. This can be observed in Figure 43. On the left side we have KAOS goal refinement plus STS actor representation. In our conceptual model the merging of these two notations can be expressed using two composite requirements as shown on the right side of the Figure 43. The additional attributes of goals and agents relative to abstraction views will be discussed during the scenario illustration in section 3.2.

3.1.2 Anti-STS Multi-agent risk modelling

From our evaluation study, we have observed that STS notation does not provide adequate support to drive the security risk analysis (refer section 2.5.1.2). Although the threat propagation is an interesting functionality for tracing back the threat impact to the business objective, it is not sufficient to assess the level of threat. As consequence, it is difficult to detect what the potential threat event can have on business objectives, and what rigor of security protection need to be advocated [Matulevicius et al 2012]. For example, how severe is the threat? What are the possible threat scenarios? How often does it occur? Who could potentially manifest such threats? Answering these questions would drive the risk analysts to calculate the probability and frequency of occurrence of the threat event. To some extent, Secure

KAOS enables to express attacker goals (i.e., anti-goals) hierarchy similar to the attack trees (see Figure 17), which describe attack scenarios. In addition, it also allows to express the risk attributes such as likelihood and criticality of anti-goals (refer section 2.5.1.1). However, in secure KAOS, there is no clear link between the likelihood of the risks and the anticipated threat capability or to the rigor of protection capability required to mitigate the risks.

Besides, Information System Security Risk Management (ISSRM) domain model [Mayer 2009] advocates that an SRE methodology must provide support to a minimum of three risk management concepts i.e., assets, risks and risk treatments. Adding to this, [Matulevicius et al 2012] argues that SRE methodologies must provide a concrete modelling constructs to represent attack agents and anti-goals to complement risk management concepts as prescribed in the (ISSRM) domain model [Mayer 2009]. Indeed, [Matulevicius et al 2012] extends secure Tropos to link the threat goals of threat agents with the system goals using the relationships: *exploits* and *attacks*. However, they imagine the attacking agents as a single entity. As consequence, they cannot help to represent complex attacks like APTs (i.e., advanced persistent threats) in which multiple agents act collaboratively to infiltrate enterprise networks.

Apart from the SRE modelling support to risk analysis, there exists enormous risk analysis techniques built on the modelling perspectives of *attack trees* [Schneier 1999] and *attack graphs* [Phillips and Swiler 1998] (refer section 1.2.1). In this regard, Kordy et al [Kordy et al. 2014] presents a detailed literature review of risk analysis techniques and compare them with respect to the type of their formalisms; their modelling capabilities and usability. From our understanding, we see that these attack modelling techniques confined their modelling perspectives to analyse the attack behaviours having conventional attacks in mind. As a consequence, they do not provide support to analyse the coordinated behavioural aspects of the multiple attacking agents. This situation again advocates the need for a new modelling constructs to fulfil this gap.

To complement this situation, in this thesis, we propose a threat profile based risk modelling technique built on the agent-oriented modelling (similar to STS). A threat profile, in general, includes the information about critical assets, threat actors, and threat scenarios [SANS 2015b]. We argue that attackers are not lonely agents and they too have to rely on some malicious users or malicious programs in order to launch an attack. For instance advanced persistence threats (APT) include coordinated interactions between multiple attacking agents that execute multi-step attack methods to their ultimate motives (e.g., stealing data, stealing money, espionage, etc.) [Chen et al. 2014]. Analyzing the social and technical dependencies between the threat actors (i.e., individual or group, system element or software program, etc.) will help in anticipating the factors of likelihood and threat capabilities.

Suitably, we decided to represent the attack models in form of multi-agent systems comprising of several attacking agents. We considered the following list of anticipated requirements with reference to the characteristic features of composite attacks (e.g., APT). It is to note that the list is not exhaustive and describes the minimum requirements needed to model APT:

- **Req1**– Must use unambiguous modelling language.
- **Req2**– Must facilitate to express all the attacking agents
- **Req3** – Must facilitate to express all the threat goals
- **Req4** – Must facilitate to express all the attack methods

- **Req5** – Must facilitate to trace the attack path
- **Req6** – Must facilitate to express the social and technical dependencies among the attacking agents (i.e., human/system).
- **Req7** – Must facilitate to express the protection goals
- **Req8** – Must facilitate to express the used attack patterns.

Defining yet another modelling language is not a viable option to us as it can impact comprehensibility aspects. Therefore, we propose to integrate the social modelling concepts from STS modelling and the Anti-goal modelling concepts from Secure KAOS [Van Lamsweerde et al. 2003] (**Req1**). We name this model as Anti-STS. Figure 44 depicts the comparison between STS and Anti-STS.

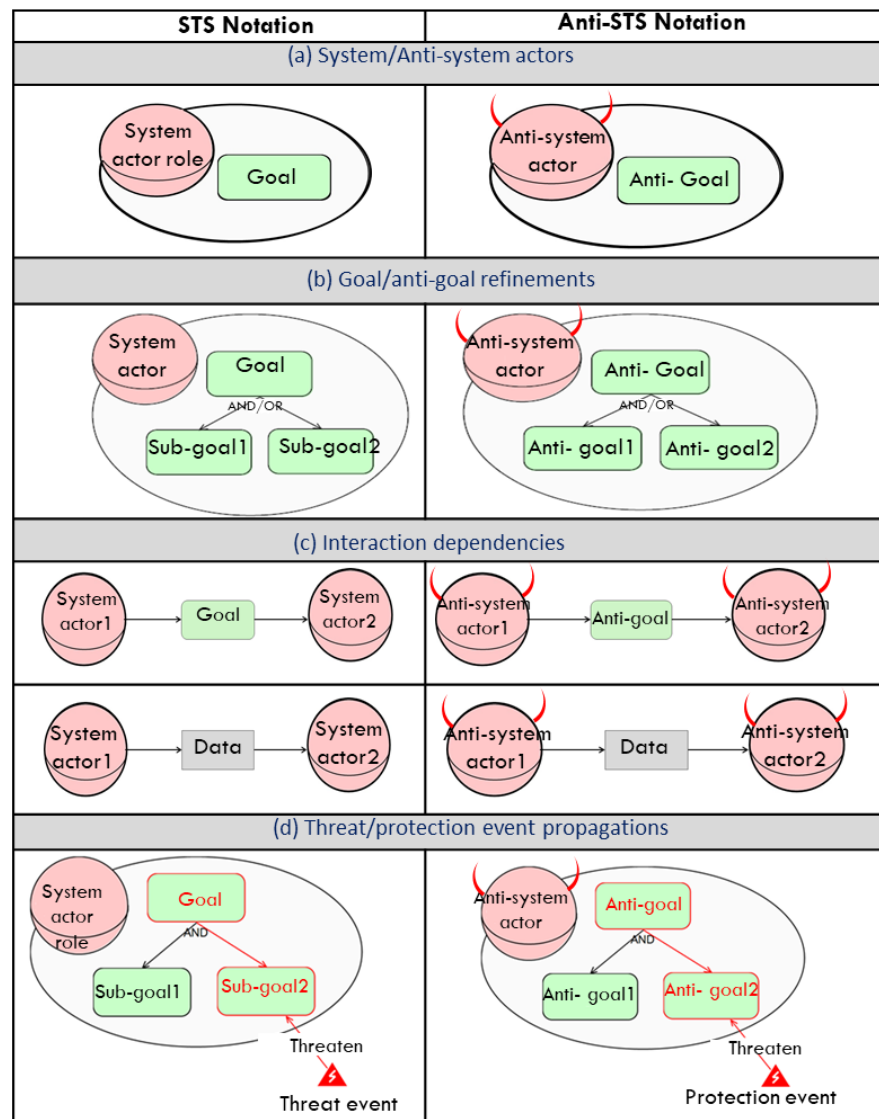


Figure 44: Comparison between STS and Anti-STS Notations

We represent the attacking agents with horns [] in order to differentiate the malicious actors with good ones. Anti-STS, represents an anti-system of attackers who collaboratively work to manifest a threat (**Req2**). The malicious objectives are represented as the anti-goals within the scope of each threat actor, see (Figure 44a). Threat scenarios are driven by some malicious motive (e.g., spoil reputation,

gain money) and follow a procedural approach. The whole composite attack scenario is divided as responsibility goals of the respective threat actors and expressed as their root Anti-goal nodes (**Req3**). Subsequently, the steps involved in the attack methods are expressed as hierarchical threat actor/agent anti-goals trees using AND/OR constructs (**Req4**), (Figure 44b). Unlike Attack trees, a threat actor/agent can have more than one anti-goal trees. That means, the deliberate threat/Attack goals are defined as the root nodes. The tree-structured refinements reflects how the threat goals can be fulfilled.

The social/technical interaction dependencies are expressed similar to STS notation goal delegations and data exchange, except that here the intentions are towards manifesting a threat scenario (Figure 44c). These social and technical dependency interactions allow to comprehend the strategic plan as well as to trace the attack path between the disjointed anti-goal models (**Req5** and **Req6**). This allows to simplify the sophistication of composite attacks (like APTs) by breaking the complex strategic attack/threat scenario into multiple anti-goal models (**Req1**). Finally, we use threat propagation functionality in STS to represent the propagation of protection events (i.e., protection needs/objectives) such as defensive strategy goals to mitigate particular malicious goal (Figure 44d). Accordingly, the protection event propagation allows not only to link the protection goals to the anti-goals, but also it facilitates to view the protection impact on the whole Anti-goal tree model (**Req7**).

We adopt the following steps to gradually develop the expression of an APT in our Anti-STs Model: (Step 1) identify threat actors/agents, (Step 2) identify the respective root anti-goals, (Step 3) refine the anti-goals based on the threat intelligence information available in reference documents, (Step 4) identify the possible interaction dependencies, (Step 5) identify control measures are given and express them as protection events. While first two steps are confined to Business view and the other two steps are confined to Architect's view.

Since we are already using STS modelling notation for SRE at business and Architect's view, Anti-STs would be practical to accommodate to the integration of existing attack modelling perspectives described in [Kordy et al. 2014]. Since we are already using STS modelling notation for SRE at business and architects view, Anti-STs would be practical to accommodate to the integration of existing attack modelling perspectives described in [Kordy et al. 2014]. We will present in more details the application of our Anti-STs modelling in section 3.2. This will show how this technique has been integrated in our methodology.

3.1.3 Our proposed Zone modelling methodology

In the first two views of our SRE conceptual model we propose to use STS concepts. In the designers view (LV3), we start to consider the network security needs. Our evaluation study shows that, since STS is an agent-oriented modelling approach, it gets complicated when starting to analyse the interaction dependencies between the network devices. In addition, there is no other SRE approach that facilitates the elicitation and analysis of network security requirement needs. Therefore we propose a new modelling technique, which is based on the zone segmentation principles in order to introduce the notion of network security analysis.

3.1.3.1 Analysing the risk of the enterprise network

The current practice for eliciting and analysing early network security requirements is driven by network zoning [8]. Network zoning (a.k.a. network segmentation) is a key defence-in-depth strategy for enterprises that segregates and protects key company assets and limits lateral movements of attackers across corporate network in case of intrusion. Grouping assets in terms of segmented zones with varying trust levels enables organizations to distribute security controls across the network relative to the criticality as well the risk exposure of the systems within zones. Therefore, to facilitate the integration of security zoning concepts, in our zone modelling methodology we mainly consider three elements: domains, zones and agents.

A *security domain* represents the organizational authority, which controls and manages the entities (i.e., servers, software, data, users, etc.) that belong to it. We call these entities as *agents* who are assigned with different set of responsibility goals to achieve business needs. Furthermore, a security domain can be refined into sub-domains highlighting different policies or procedures within the same organization. As mentioned earlier, in our SRE methodology, we categorize agents into two groups. *System agents* refer to entities under direct control such as software/hardware systems that are developed and/or maintained by the enterprise. *Environment agents* are not under direct control and refer to humans, or to some purchased third party software/hardware. For example, in the given use case scenario 1, we assume two domains: the airport ground domain and the aircraft domains. The aircraft domain consists in the system agents: aircraft control application and aircraft monitoring application. The airport ground domain consists in the environment agent i.e., maintenance people. Finally, *security zones* constitute logical grouping of agents with common protection requirements. As consequence, our methodology mainly aims at grouping *agents* within *security zones* managed in *security domains* and eliciting the related network security requirements.

We consider that some external risk analysis has assessed security domains and agents to provide domain control capability, trust of environment agents and criticality of system agents. *Security domains control capability* describes the maturity of an enterprise to deploy security controls and/or its capability to control a given environment. In other words, a well-controlled domain means that the security management activity within the domain is mature. For example, in the given use case scenario 1, the aircraft domain is mature since it is equipped with well-trained employees (i.e., cabin crew and pilots). Thus, we consider that this domain is well controlled by the airline company. On the other hand, the aircraft ground domain may consists in people belonging to airport management as well as competitive airlines. Therefore, we consider this domain is less controlled by the airline of interest.

Environment agents are given a *trust level*, which specifies the degree of the trustworthiness over the expected behaviour of environment agents in a given context. Because it will eventually impact the responsibility goals entrusted to them. For instance, in our example scenario1, the maintenance people are considered as partially trusted because their goal is to collect aircraft parameters by connecting their laptop to the monitoring application. If the laptop is vulnerable to viruses, this may comprise the goal of the maintenance people.

Finally, system agents are evaluated based on their *criticality* levels. *Criticality* level determines the sensitivity to threats and their risk impact on the overall business. For instance, in our example scenario1,

aircraft control system controls the dynamics of the aircraft (such as engine speed, flight's direction, inflight temperature and pressure). Its failure is intolerable since it threatens the safety of the passengers along with huge monetary loss and reputation. Thus, aircraft control system is considered as highly critical. On the other hand, the monitoring application's role is critical only to the aircraft maintenance process. In case of disruption of the responsibility goal "collect monitoring data" it may also incur monetary loss, which can be tolerable to some extent but would damage the reputation in long run. Therefore, considering the risk tolerance factors, monitoring application is considered partially critical.

Since Our Anti-STS model facilitates to trace the attack goals to deliberate disruption of business needs, it will help in integrating the risk analysis techniques such as FMECA (failure mode criticality analysis) in order to analyse the risk impact of the system agents on the business needs in case of failures.

3.1.3.2 Integration of three core security design principles

Through security zoning, in the end we expect to achieve secure network design. Therefore, we build our security zoning strategy compliant with the three core security design principles i.e., complete mediation, least privileges and formal integrity security models for information flows.

First, the complete mediation principle enforces that every access to every an entity must be checked for authority [Saltzer and Schroeder 1975a]. By default, this complete mediation rule is checked for client-server models. Applying this principle to the context of security zoning, that means that every access to the zones (otherwise data flows between the zones) are controlled with some security mechanisms. We also identify and ensure that server and client do not reside in same zone as it will be conflict. Second, the principle of least privileges [Saltzer and Schroeder 1975a] requires to limit users to access only what is necessary for their legitimate purpose. We translate this principle in the context of network security zoning as a user can access a zone only if he is granting access to all the services within the zone. Here we map the services level privileges to the network level privileges. This limit the propagation of an attack on a service to only the zone. Finally, formal models of integrity fosters to avoid critical systems to consume untrusted/fake information. We consider these models to introduce the security verification on data flow by validating the integrity of the data flows between zones.

In practice, there exists several models for integrity such as Biba [Biba 1977], clark-wilson [Clark and Wilson 1987], which propose abstract solutions to preserve the integrity of information flows. These models are widely used in current operating systems for improving the integrity protection of the information flows in inter-process communications (e.g., Microsoft Windows Integrity Mechanism [Microsoft]). We adapt the model of Clark-Wilson lite [Shankar et al. 2006] (lighter version of clark-wilson model), for verifying the integrity property of traffic flows traversing multiple zones.

In CW-lite model, all information flowing from low integrity subjects to high integrity subjects must be filtered. The filter is placed at the receiving subject's side. Figure 45 shows the formal notation of the CW-lite integrity principle. This predicate should be read as follows: "if a subject s receives an information flow from a subject s_i at interface I , then either there is an integrity validation filter at interface I or the integrity level of s_i is greater or equal to the integrity of subject s ".

$$flow(s_i, s, I) \wedge \neg filter(s, I) \rightarrow (int(s_i) \geq int(s))$$

Figure 45: CW-lite security filtering rule [Shankar et al. 2006]

Here, the integrity filters correspond to integrity validation procedures that sanitize information or block it. For instance, in network security analysis context, an integrity validation filter can be a web application firewall that checks SQL statements or URL formats. Integrity models complete the principle of complete mediation by checking the content of flows.

3.1.3.3 Unification of risk analysis information through integrity levels

The three factors i.e., control capability of domains, criticality and trust levels of agents enables the integration of risk analysis process in network requirement analysis context. And to facilitate this integration, we unify capability, trust and criticality within the concept of integrity since integrity is a pivot concept of risk and trust. For example, scientific articles published in the security conferences are more trustful than those published in teenager blogs. Here, the integrity of the published content is related to trust when considering the external or unmanaged systems. Likewise, integrity is also related to risk. For example a critical system must be consistent, « honest », which means it requires high level of integrity. In addition, systems take decisions based on information (e.g. a program executes an algorithm based on its inputs). If input information is wrong, then decisions can be wrong too. Consequently, we will only permit critical system to consider information with high level of integrity (i.e. high level of assurance).

In our methodology, this unification of risk and trust within integrity allows us also to integrate formal integrity security models with security zone modelling design principles for addressing the risks pertaining to traffic flows and information assurance. Respectively, the integrity of an agent reflects the assurance of an expected behaviour. This fits with the concept of trust related to environment agents. Since they are not directly under the control of the enterprise, we assess the confidence that enterprise can have on respective entity (in best case) in terms of its expected behaviour. As consequence, the trust assessment can be transformed into a maximum integrity value representing an assumption.

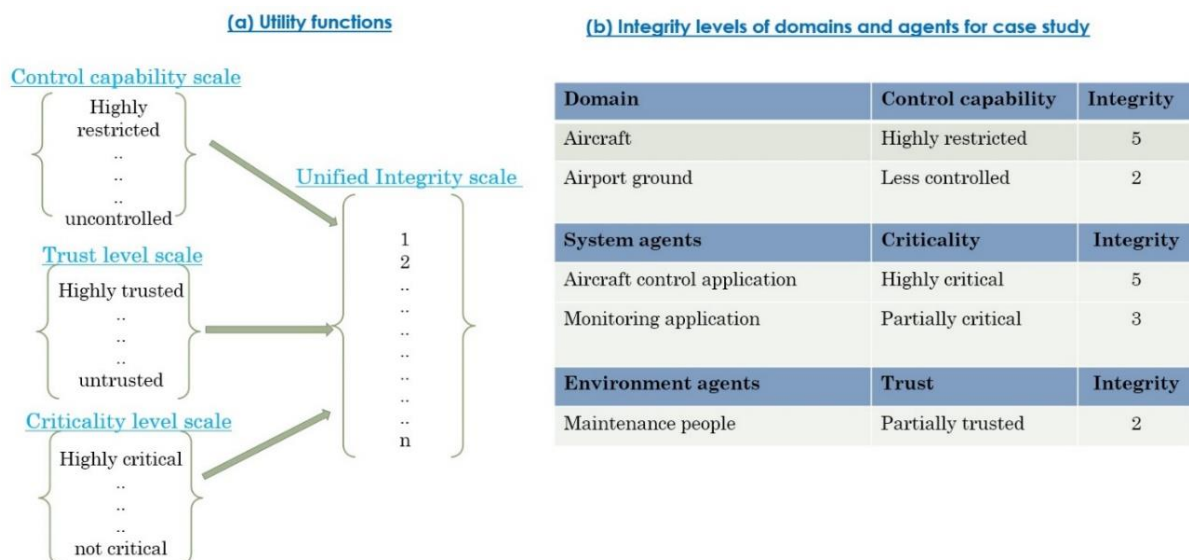


Figure 46: integrity values of domain and agents for use case scenario 1

Similarly, system agents' criticality can be expressed as a required integrity. Critical goals are required to be achieved, hence, it is a required behaviour of the assigned system agent. Criticality corresponds to the minimum integrity value required for a system agent, which conveys the minimum security protection required in worst case. For instance, since aircraft control application is considered highly critical, it requires a high level of integrity. Wherein monitoring application is partial critical it requires relatively a lower level of integrity level. Finally, the integrity of a security domain correlates with the maximum integrity an enterprise can achieve with its capability control. This implies that a security domain cannot guarantee security more than its maturity level.

We assume the existence of some utility functions (Figure 46a) that map the control capability labels of domains, criticality and trust levels of agents into a unified scale of integrity levels. For instance, IEC 61508 [IEC 61508 2010] defines safety integrity levels (SIL) based on controllability of the system from the risk of failures with an SIL scales ranging from A to D. Similarly, Figure 46b shows an example of integrity values that will be used for example use case scenario 1, with an integrity scale range from 1 to 5. These utility functions must be determined based on business risk impact, which is a pre-requisite to define zone model [Province of British Columbia 2012]. The example mapping of utility functions for integrity range 1 to 5 that we have considered in this thesis work is given in Appendix D.

3.1.4 Formal approach of zone modelling methodology

Our zone modelling methodology (see Figure 47) is divided into two main steps: (1) Determining the security zones and integrity validation filters and (2) Identifying data flows integrity requirements and flows access control filters. In step1 the initial input is the set of security domains, the set of agents, the integrity levels of domains and agents, and the data flows between agents. As a result of step1, our process computes the security zones and the integrity validation filters.

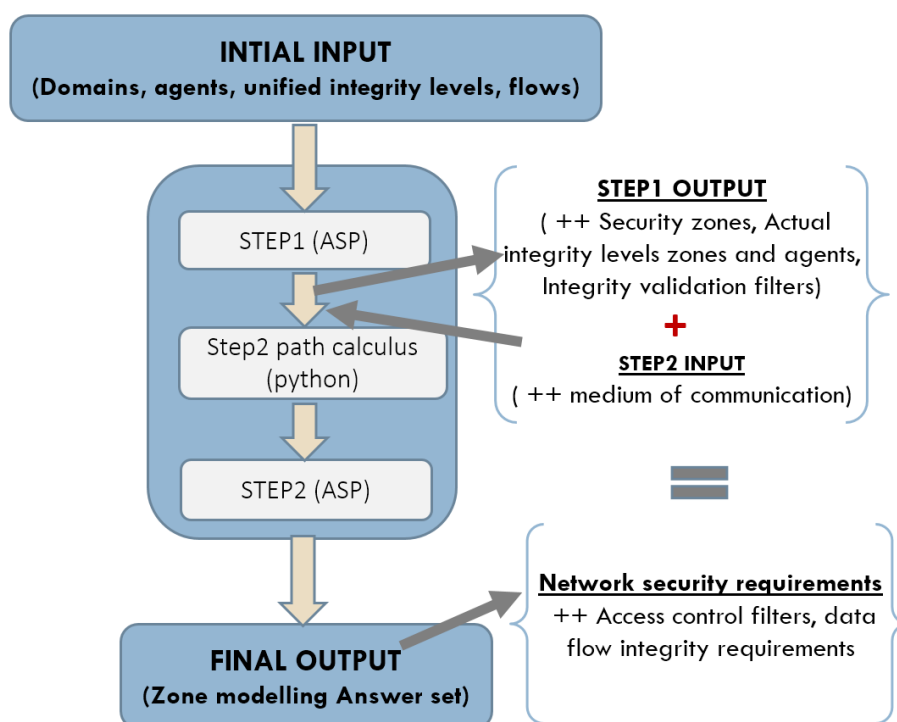


Figure 47: Our zone modelling Methodology approach overview

In step2, the designer needs to provide additional information about the media of communication (i.e., the networks). The final result is a set of network security requirements which are a set security zones, integrity validation filters, agents integrity requirements, access control filters and integrity data flow protection requirements. For the implementation, we formalized step1 and step2 in Answer Set Programming. Answer set programming (ASP) [Gebser et al. 2012] is a declarative logic-based approach that facilitates the solving of difficult search problems by computing answer sets through stable model semantics. We defined rules at each step and the ASP solver determines the set solutions (called answers) that are compliant with the rules and the input. This makes our process traceable and verifiable. In the following, we discuss in detail the modelling rules at step1 and step2. The implementation and the output of our zone modelling solution will be discussed separately in section 3.2, during the illustration of use case scenario 1.

3.1.4.1 Step 1: Specifying zones and filtered flows

The main goal of this step is to specify zones and identify integrity validation filters. We start with a system as a set of domains, zones and agents. We represent it as follows:

$S = \langle \text{DOMAIN}, \text{ZONE}, \text{AGENT}, \text{FLOW}, \text{INSIDE}_Z^D, \text{INSIDE}_A^D, \text{INSIDE}_Z^A, \text{Int}, \text{Int}_{\max}, \text{Int}_{\min}, \text{Int}_{\text{actual}}, \text{Agent}_{\text{Server}}, \text{Agent}_{\text{Client}} \rangle$, Where:

- DOMAIN is the set of security domains.
- ZONE is the set of security zones.
- AGENT is the set of agents, named after entities. $\text{AGENT} = \text{ENV_AGENT} \cup \text{SYST_AGENT}$ with ENV_AGENT and SYST_AGENT being the set of environment and system agents such that $\text{ENV_AGENT} \cap \text{SYST_AGENT} = \emptyset$.
- $\text{Agent}_{\text{server}}: \text{AGENT} \rightarrow \{\text{TRUE}, \text{FALSE}\}$ states if agent is a server (e.g., WEB server).
- $\text{Agent}_{\text{client}}: \text{AGENT} \rightarrow \{\text{TRUE}, \text{FALSE}\}$ states if agent is a client (e.g., browser).
- $\text{FLOW} \subseteq \text{AGENT} \times \text{AGENT}$ is the set of allowed flow of information.
- $\text{INSIDE}_Z^D \subseteq \text{ZONE} \times \text{DOMAIN}$ is a relation that states a zone is in a domain.
- $\text{INSIDE}_A^D \subseteq \text{AGENT} \times \text{DOMAIN}$ is a relation that states an agent is in a domain.
- $\text{INSIDE}_Z^A \subseteq \text{AGENT} \times \text{ZONE}$ is a relation that states an agent is in a zone.
- $\text{Int}: \text{DOMAIN} \rightarrow \mathbb{N}$ returns the integrity level of a security domain which is fixed.
- $\text{Int}_{\max}: \text{ZONE} \cup \text{AGENT} \rightarrow \mathbb{N}$ returns the maximum integrity of a zone or an agent. For environment agents, this value is directly derived from their trust label.
- $\text{Int}_{\min}: \text{AGENT} \rightarrow \mathbb{N}$ returns the minimum integrity level of an agent. For system agents, this value is directly derived from the criticality label.
- $\text{Int}_{\text{actual}}: \text{ZONE} \cup \text{AGENT} \rightarrow \mathbb{N}$ returns the actual integrity of a zone or an agent, which are the final integrity values chosen at the end of the computation.
- $\text{integrity-validation-filter}(a: \text{AGENT}, f: \text{FLOW}, \text{val1}: \text{Int}, \text{val2}: \text{Int})$ states integrity validation requirements such that $\text{integrity-validation-filter}(a, f, \text{val1}, \text{val2})$ describes integrity protection mechanism at agent a must sanitize dataflow f with an integrity level of val1 to achieve a data assurance level of val2 .

In other words, Int , $\text{Int}_{\max}$ and $\text{Int}_{\min}$ represent the integrity utility functions in Figure 46. Accordingly, we define the rules of step1 as follows:

RULE1: Every agent is inside a domain.

$$\forall a \in \text{AGENT}, \exists d \in \text{DOMAIN} \mid (a, d) \in \text{INSIDE}_A^D$$

RULE2: Every security domain contains at least one security zone.

$$\forall d \in \text{DOMAIN}, \text{card}(\{z \mid z \in \text{ZONE}, (z, d) \in \text{INSIDE}_Z^D\}) \geq 1$$

RULE3: The maximum integrity level of a security zone is equal to the integrity level of the domain. This is because, a domain controls zone and therefore we cannot have more assurance on a zone than that of the domain.

$$\forall d \in \text{DOMAIN}, \forall z \in \text{ZONE}, (d, z) \in \text{INSIDE}_Z^D, \text{Int}_{\max}(z) = \text{Int}(d)$$

RULE4: Similar to Rule 3, the actual integrity level of an agent cannot be greater than the integrity level of domain.

$$\forall d \in \text{DOMAIN}, \forall a \in \text{AGENT}, (a, d) \in \text{INSIDE}_A^D, \text{Int}_{\text{actual}}(a) \leq \text{Int}(d)$$

RULE5: The actual integrity of a zone cannot be greater than its maximum integrity.

$$\forall z \in \text{ZONE}, \text{Int}_{\text{actual}}(z) \leq \text{Int}_{\max}(z)$$

RULE6: The actual integrity of agents must be between the maximum and the minimum integrity levels of the agents.

$$\forall a \in \text{AGENT}, \text{Int}_{\min}(a) \leq \text{Int}_{\text{actual}}(a) \leq \text{Int}_{\max}(a)$$

RULE7: The actual integrity levels of an agent is same as that of its residing zone.

$$\forall a \in \text{AGENT}, \forall z \in \text{ZONE}, (a, z) \in \text{INSIDE}_A^Z, \text{Int}_{\text{actual}}(a) = \text{Int}_{\text{actual}}(z)$$

RULE8 - CW-Lite: The actual integrity levels of the interacting agents must adhere to the CW-lite integrity rule. In this way, an agent doesn't access a lower integrity information.

$$\forall a1, a2 \in \text{AGENT}, (a1, a2) \in \text{FLOW} \wedge \neg \text{integrity-validation-filter}(a2, \text{flow}(a1, a2), \text{Int}_{\text{actual}}(a1), \text{Int}_{\text{actual}}(a2)) \Rightarrow \text{Int}_{\text{actual}}(a1) \geq \text{Int}_{\text{actual}}(a2)$$

RULE9 – Principle of complete mediation: Server agents and client agents cannot reside in same zone. Because, as per the zone modelling design principles, intra-zone interactions are usually not analysed. With reference the security design principle known as complete mediation rule, every access to every object must be checked for authority [Saltzer and Schroeder 1975a]. By default, this complete mediation rule is checked for client-server models. Therefore, if server and client reside in same zone there will be a conflict.

$$\forall c, s \in \text{AGENT}, \forall z1, z2 \in \text{ZONE}, (a1, z1) \in \text{INSIDE}_A^Z, (s, z2) \in \text{INSIDE}_A^Z, \text{Agent}_{\text{server}}(s), \text{Agent}_{\text{client}}(c) \Rightarrow z1 \neq z2$$

RULE10 - Principle of least privileges: The least privileges principle aims at minimizing the permissions of users to the minimum required for accomplishing their tasks [Saltzer and Schroeder

1975b]. From a network perspective, a client agent can send flows in a security zone only if he can send flows to all the server agents within that zone. In this way, we map network level privileges to services level privileges. Indeed, agents will be grouped in zones according to users' privileges.

$$\begin{aligned} \forall c, s1, s2 \in \text{AGENT}, \text{Agent}_{\text{client}}(c), \text{Agent}_{\text{server}}(s1), \text{Agent}_{\text{server}}(s2), \forall z1, z2 \in \text{ZONE}, \\ (s1, z1) \in \text{INSIDE}_A^Z, (s2, z2) \in \text{INSIDE}_A^Z, (c, z1) \notin \text{INSIDE}_A^Z, (c, z2) \notin \text{INSIDE}_A^Z, \\ (c, s1) \in \text{FLOW}, (c, s2) \notin \text{FLOW} \Rightarrow z1 \neq z2 \end{aligned}$$

3.1.4.2 Step2: Specifying integrity requirements for the communication medium between zones

At the end of step1, we have the set of zones along with the integrity validation filters. In step2, we address the security issues of inter-zone interactions, i.e., we consider the protection of the flow through the network communication medium (e.g., wired/wireless networks, etc.,) that connect zones. The main goal of this step is to protect the integrity of data flows when traversing untrusted media of communication. Suitably, we complete our system model as follows:

$S = \langle \text{DOMAIN}, \text{ZONE}, \text{AGENT}, \text{FLOW}, \text{MEDIUM}, \text{INSIDE}_Z^D, \text{INSIDE}_A^D, \text{INSIDE}_A^Z, \text{INSIDE}_M^D, \text{CONNECT}, \text{Int}, \text{Int}_{\text{max}}, \text{Int}_{\text{actual}} \rangle$, Where:

- MEDIUM is the set of media of communication.
- $\text{INSIDE}_M^D \subseteq \text{MEDIUM} \times \text{DOMAIN}$ is a relation, which states that a medium of communication is in a domain.
- $\text{CONNECT} \subseteq \text{MEDIUM} \times \text{ZONE}$ is a relation, which states that a zone is connected to a medium of communication.
- $\text{Int}_{\text{max}}: \text{ZONE} \cup \text{AGENT} \cup \text{MEDIUM} \rightarrow \mathbb{N}$ returns the maximum integrity level of a security zone, agent or medium of communication.
- $\text{Int}_{\text{actual}}: \text{ZONE} \cup \text{AGENT} \cup \text{MEDIUM} \rightarrow \mathbb{N}$ returns the actual integrity level of a security zone, agent or medium of communication.
- $\text{PATH} \subseteq \text{FLOW} \times (\text{ZONE} \cup \text{MEDIUM}) \times (\text{ZONE} \cup \text{MEDIUM})$, is a relation that stores where flows are transiting with the constraint that $\forall (f, e1, e2) \in \text{PATH} \Rightarrow (e1, e2) \in \text{CONNECT} \vee (e1, e2) \in \text{CONNECT}$. For instance, $(f, m, z) \in \text{PATH}$ means that flow f transits between medium m to zone z .
- $\text{access-control-filter}(c: \text{CONNECT}, f: \text{FLOW})$ states access control requirements so that $\text{access-control-filter}(c, f)$ means flow f must be permitted at connection point c .
- $\text{dataflow-integrity-protection}(f: \text{FLOW}, e: \text{ZONE} \cup \text{MEDIUM}, \text{value}: \mathbb{N})$ states dataflow protection requirements such that $\text{dataflow-integrity-protection}(f, e, \text{val})$ means some protection mechanism must be applied on dataflow f over zone or medium e to preserve an integrity level of val .

Similar to domains, zones, and agent, the medium of communication $m1$ has two integrity levels: $\text{Int}_{\text{min}}(m1)$, and $\text{Int}_{\text{actual}}(m1)$. Accordingly, we add new rules to include constraints on media of communication:

RULE11: Every zone must be connected to a medium of communication.

$$\forall z \in \text{ZONE}, \exists m \in \text{MEDIUM}, (m, z) \in \text{CONNECT}$$

RULE12 –Access control filter requirements: At each zone, there must be an access control filter that permits allowed flow of information. Not explicitly allowed flows are denied by default.

$$\forall (f, e1, e2) \in \text{PATH}, e1 \in \text{MEDIUM} \Rightarrow \text{access-control-filter}((e1, e2), f)$$

Respectively:

$$\forall (f, e1, e2) \in \text{PATH}, e1 \in \text{ZONE} \Rightarrow \text{access-control-filter}((e2, e1), f)$$

RULE13: The actual integrity level of a medium of communication is the minimum value of the integrity level of its domain, the trust on the medium (i.e., its maximum integrity), and the actual integrity levels of the connected zones.

$$\begin{aligned} \forall m \in \text{MEDIUM}, \text{Int}_{\text{actual}}(m) = \min(\{ \text{Int}(d) \mid d \in \text{DOMAIN}, (m, d) \in \text{INSIDE}_M^D \} \\ \cup \{ \text{Int}_{\text{max}}(m) \} \cup \{ \text{Int}_{\text{actual}}(z) \mid z \in \text{ZONE}, (m, z) \in \text{CONNECT} \}) \end{aligned}$$

RULE14 – dataflow integrity protection requirements: A flow that transits over a medium or a zone, requires an integrity protection, if the integrity level of the medium or the zone is lower than the level of integrity of the flow.

$$\begin{aligned} \forall (a1, a2) \in \text{FLOW}, \forall e1, e2 \in \text{ZONE} \cup \text{MEDIUM} \mid (\text{flow}(a1, a2), e1, e2) \in \text{PATH}, \\ (\min(\text{Int}_{\text{actual}}(a1), \text{Int}_{\text{actual}}(a2)) > \text{Int}_{\text{actual}}(e1) \Rightarrow \\ \text{data-flow-integrity-protection}(\text{flow}(a1, a2), e1, \min(\text{Int}_{\text{actual}}(a1), \text{Int}_{\text{actual}}(a2)))) \end{aligned}$$

Respectively:

$$\begin{aligned} \forall (a1, a2) \in \text{FLOW}, \forall e1, e2 \in \text{ZONE} \cup \text{MEDIUM} \mid (\text{flow}(a1, a2), e1, e2) \in \text{PATH}, \\ (\min(\text{Int}_{\text{actual}}(a1), \text{Int}_{\text{actual}}(a2)) > \text{Int}_{\text{actual}}(e2) \Rightarrow \\ \text{data-flow-integrity-protection}(\text{flow}(a1, a2), e2, \min(\text{Int}_{\text{actual}}(a1), \text{Int}_{\text{actual}}(a2)))) \end{aligned}$$

3.2 SRE methodology illustration

In this section, we illustrate our SRE methodology using the AIRBUS scenario related to aircraft maintenance described in section 2.5.1. The information provided in the example scenario does not respect the abstraction layers as per our methodology. This illustration focuses in describing the separation of concerns at each layered level. It is to note that, since this work is confined to SRE modelling, we consider the information relative to risk analysis as pre-requisites.

3.2.1 Business view (STS modelling)

This view concerns the capture of high-level business needs from the perspectives of business users. What actually the business is? What are the driving objectives to run the business? What are the business needs that are need to be protected from risks? These kind of information is elicited at this view. Relatively, the business needs are considered as assets and are prioritized at this view.

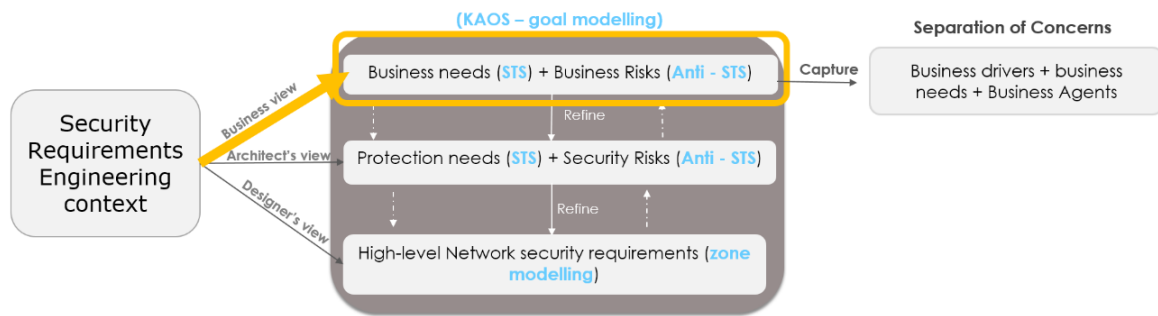


Figure 48: Business view - separation of concerns

3.2.1.1 Scenario Analysis at Business view

Since this view has to capture the perspective of business users, we start by specifying the relation between business needs and the driving objectives of the business, also known as **business drivers**. Business drivers describe the business vision towards the continued success and growth of a business. An organisation can have several business drivers each accorded with a business value. However, there is no standard list available since they differ with type of organization. Some example business drivers that include Reputation, business competitive advantage, customer satisfaction, etc., and are taken reference form [Sherwood 2005; Simplicable 2002].

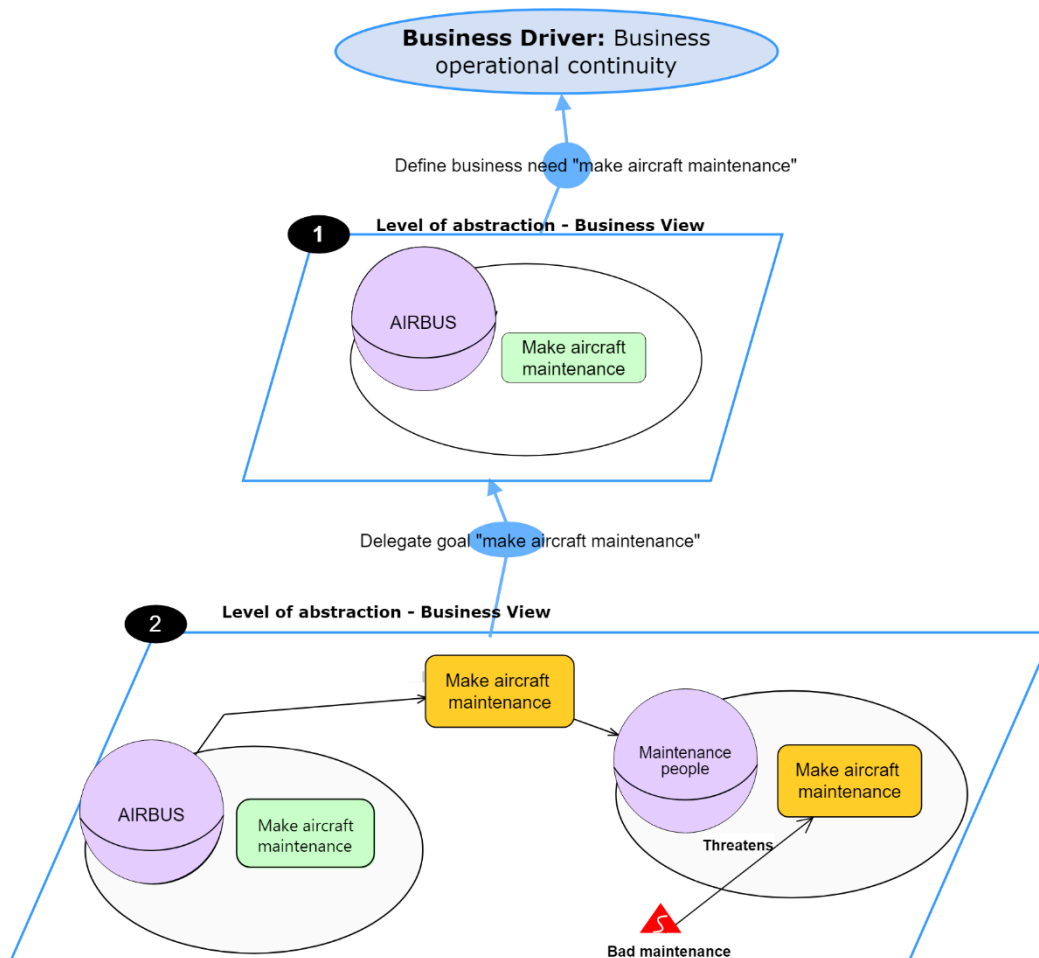


Figure 49: Aircraft maintenance business need linked to business drivers

The use case provided by AIRBUS concerns an aircraft maintenance process, which corresponds to the high-level business objective. We link this business need to an example business driver i.e., business operational continuity because aircraft has to be in good condition to transport the passengers in safely or may incur monetary loss in terms of missed flight trips per day.

The Figure 49 depicts the Business view of the example AIRBUS scenario. The business need “*make aircraft maintenance*” is defined as a responsibility goal within the scope of the role represented by the ‘AIRBUS’ actor. Here, AIRBUS refers to the airline authority having the business need “*make aircraft maintenance*”. Business drivers are represented within our notation by a blue oval form. This implies that business drivers are the root nodes of our entire notation. As described in SABSA, each business driver is associated with some business value expressed in monetary terms. A business need can relate to one or several business drivers and vice versa. The blue frame represents a requirement known as the *composite requirement*. The composite requirements at this Business view comprises of the social actors concerned with the business needs.

The next composite requirement (CR n°2 in Figure 49) describes the delegation of business need to maintenance people. This implies the maintenance people are entrusted to make to decision on whether to permit the aircraft to take the next trip or not. Since maintenance people are human agents, we consider them as environment agents (represented using violet circles with curve). The delegated goal becomes an expectation (represented in yellow colour).

3.2.1.2 Risk analysis at Business view – (Anti-STS modelling)

From risk analysis perspective, “bad maintenance” is a threat event that can disrupt the fulfilment of the responsibility goal i.e., make maintenance. This is expressed using the threat propagation link named ‘threatens’ in STS notation. This does not give any information related to how the threat event can be perceived. Therefore, we extend this threat information using Anti-STS Risk modelling by first specifying the feared threat agents and malicious motivations, see Figure 50.

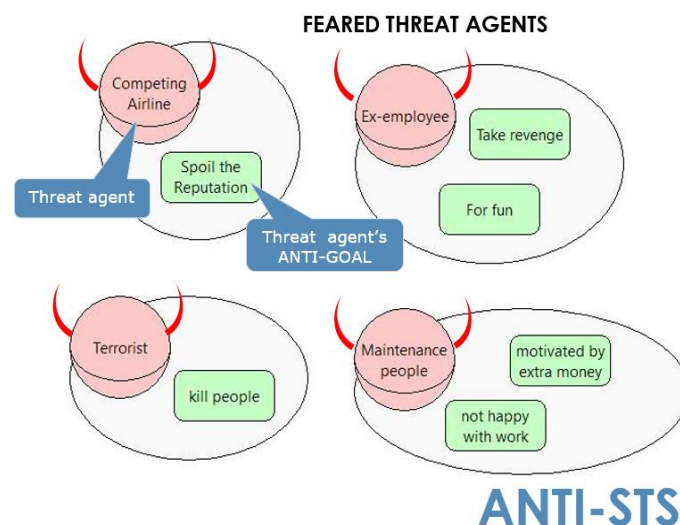


Figure 50: Anti-STS feared threat agents at Business view

For instance, the “*maintenance process*” can either be threatened by an external threat agent such as the “*competing airline*” with a motive to spoil the reputation; or by an internal threat agent “*maintenance people*” who deliberately wants to make bad maintenance. It can also be imagined that, the competing airline and the maintenance people participate together in the execution of a threat scenario. This needs an interaction between the two threat actors (*competing airline* and *the maintenance people*), which explains the social dependency to achieve an anti-goal.

Since the feared threat agents may sometimes refer to internal agents as well (e.g., employees of the enterprise). As consequence, the risk modelling eventually provoke the security experts at Architect's view to analyze and determine the trustworthiness over the expected behavior, which is expressed in terms of unified integrity values. This will help us to move to the security requirement analysis at upcoming layered views i.e., Architect's view and Business view.

3.2.2 Architect's view (STS modelling)

This view concerns about eliciting protection control objectives (i.e., protection needs), that secure the business needs captured at Business view. This is the abstraction layer where one starts considering security and risks of the enterprise networks. What sort of threat scenarios could trigger potential risks to the maintenance process need? What kind of system protection strategy would be necessary to control such risks? These kind of information is elicited at this view. Besides, SABSA describes this layered view acts as a bridge between the business users and the system designers in which the people system architects and security architects work in collaboration with business security analysis and risk analysts.

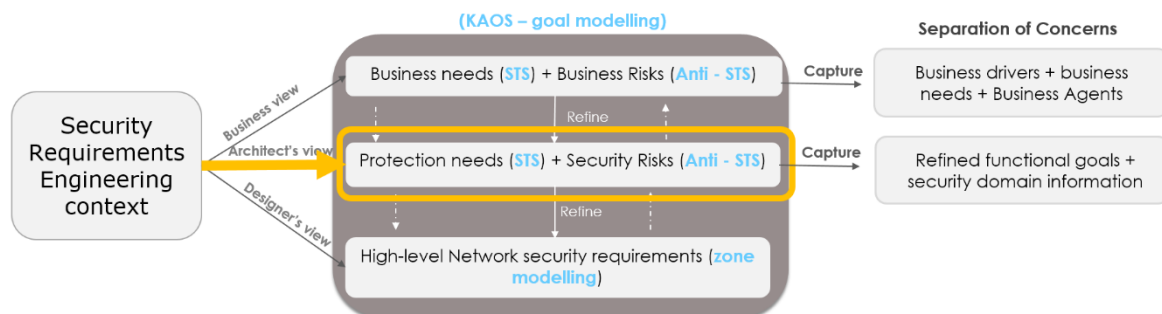


Figure 51: Architect's view - separation of concerns

3.2.2.1 Security analysis Architect's view

We continue the usage of STS and ANTI-STIS for specifying security constraints as well as the security risks. Relatively, the capture of protection needs are expressed in terms of security constraints over goal delegations using STS agent modelling notation and correspond to non-functional security requirements. The capture of risk analysis related information is expressed in terms of unified integrity values of security domains and agents in the enterprise network.

Since this view is a bridge between the business security needs and the system security needs, there will be needing an interaction between the system architects as well. In general, architect is the one who defines high-level functional goals adhering to the high-level business needs. Relatively, the first composite requirement (CR n°3) in Architect's view express the functional goals requirement preceding the capture of security constraints over the agent interactions as shown in Figure 52.

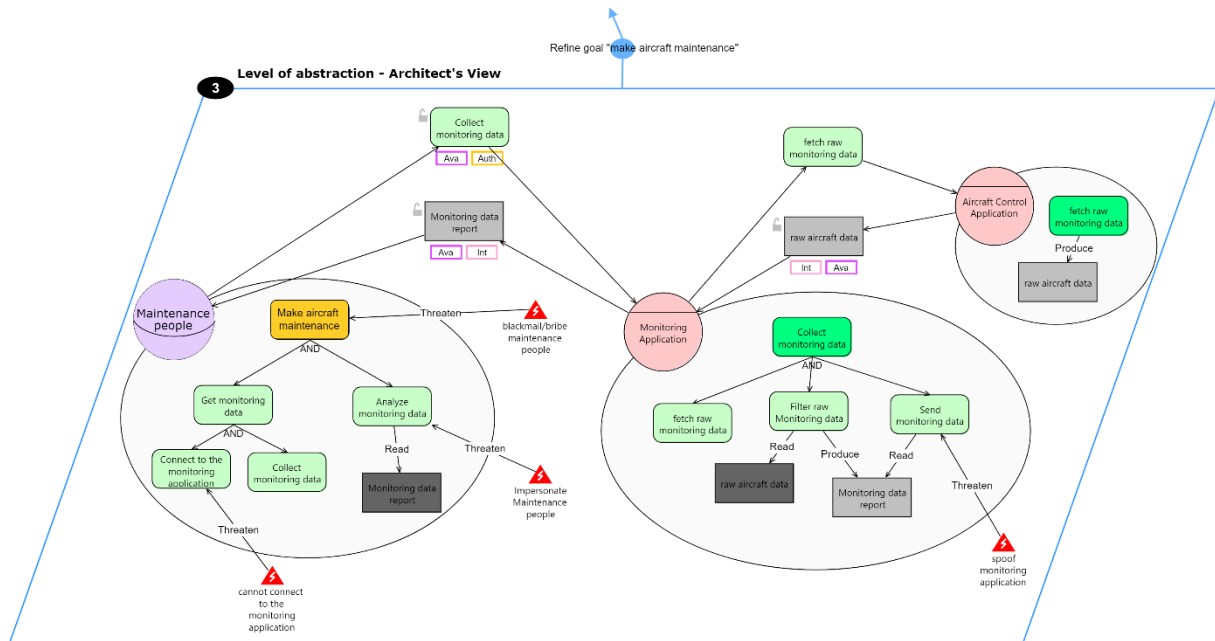


Figure 52: refinement of business functional goals and agents - Architect's view

The composite requirement CR n°3 is similar to a conventional STS (see Figure 34 in section 2.5.1.2), which confines the security requirement and risk analysis in terms of security constraints and threat events. However, it lacks support to integrate risk analysis information of enterprise network. As described in, we propose to security domain concepts and unified integrity levels given in Figure 46 in order to fulfil this gap. Suitably, in the next following composite requirement (CR n°4) express the capture the security domain information. We have hidden the responsibility goals to facilitate the readability. The unified integrity values are expressed in ordinal scale (qualitative) ranging from 1 to 5. It is to note that, the security domain information as well as the qualitative scales and range are part of elicited information from the security domain experts involved at Architect's view.

In the example use case scenario 1, we have considered a total of two domains that are *airport ground domain* (managed by airport authority) and *aircraft domain* (managed by the airline authority). The control capability scales of these domains reflects the perspective of airline authority since it is the business user. By applying the utility function described in Figure 46, these control capabilities are transformed into maximum integrity values of domain. Since these two domains are independent and can have different regulations and governance constraints this will enforce security analysts to take into consideration the inter-domain trust relationship between the airline and the airport authorities, which helps in anticipating security risks [Sherwood 2005]. For example, the interaction dependency between the maintenance people and monitoring application needs attention from security perspective at the receiving end (i.e., monitoring application) since maintained people are operating from less controlled domain and monitoring application is located in highly restricted domain.

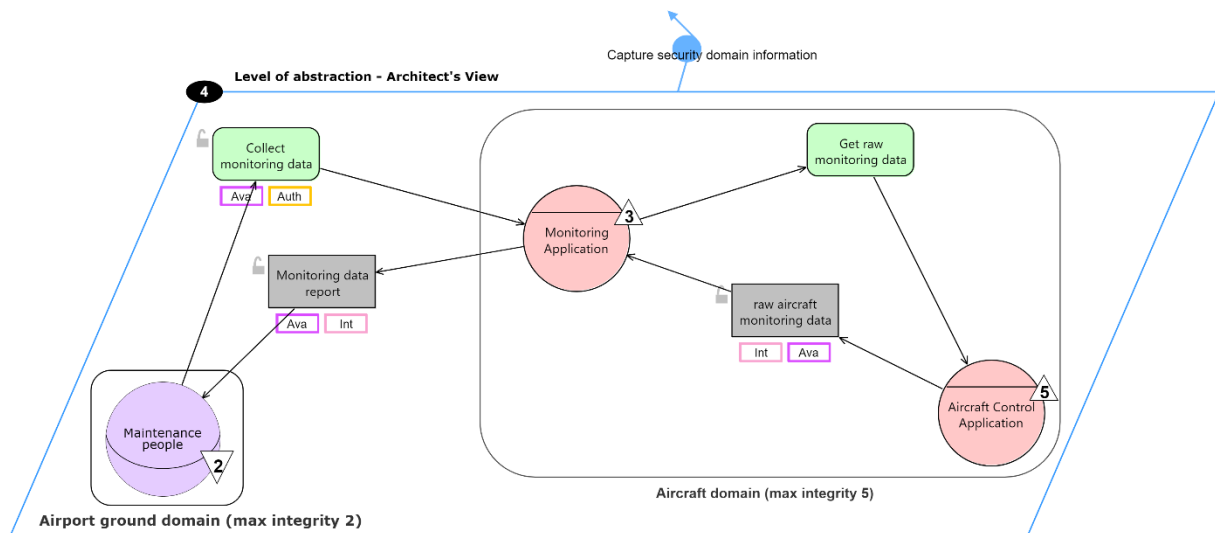


Figure 53: Integrating security domain information (use case scenario 1)

The integrity levels of system agents (depicted in triangular shapes) correspond to the minimum level of integrity that should be maintained in terms of security assurance. Here, the criticality assessment of assets (i.e., system agents) can be influenced by several factors such as the replacement costs of the assets, the degree of involvement in mission critical activities, severity of the consequences of failures (i.e., risk impact) on the business missions in case the security agent becomes non-functional [Kim and Kang 2011].

For instance, the European Commission [European Commission 2006] defines a minimum set of criteria to assess the severity of the risk impact of critical infrastructure as: (a) *Public effect* i.e., number of population affected or loss of life; (b) *Economic effect* i.e., significance of economic loss and/or degradation of products or services; (c) *Environmental effect* i.e., significant impact on public or surrounding environment or public health; (d) *Political effects* i.e., impact on the overall governance; (e) finally, *Psychological effects* i.e., impact on public or people involved (directly/indirectly) in functioning of critical infrastructure. However, in the end it is up to the risk analysis techniques to determine the severity risk impact.

For example, in our example use case scenario, disruption of aircraft control application goals can potentially threaten the safety of human lives; while malfunction of aircraft monitoring application may cost some economic loss. Likewise, through the specification of security domain information at Architect's view, our SRE methodology enforces at Architect's view enforces security experts to consider asset-based risk analysis techniques (such as FMEA/FMECA and Hazop), for analysing security risks and business impact.

Finally, the integrity levels of environment agents (depicted in inverted triangular shapes) correspond to the maximum level of integrity, which is expected as guaranteed. Since environment agents cannot be fully controlled, this compels security analysts to consider the trust assessment in terms of cognitive beliefs (e.g., honesty, willingness towards fulfilling the responsibility goals) [Cristiano Castelfranchi 2002] or technical beliefs (e.g., competence, expertise, experience) or social/technical dependencies [Jonker and Treur 1999]. In our example use case scenario, the maintenance people role is assumed as entrusted with a maximum integrity level of 2 since there is a dependency on laptop that is uncontrolled.

3.2.2.2 Security risk analysis at Architect's view

Security domain information (i.e., asset criticality, trustworthiness, control capability) correspond to asset-based risk analysis concepts. While, the FAIR risk taxonomy in Appendix A refers to threat risk analysis concepts (e.g., threat agents, threat motivation, threat capability, risk impact, control strength). We argue that, an SRE methodology must facilitate to link risk analysis concepts that correspond to both asset as well as threat-based risk analysis, which will eventually allow us to analyse the likelihood and impact of security risks pertaining to critical assets in the enterprise network.

Thus, similar to the functional goals refinement using STS, we need to refine the anti-goals in parallel. For this we continue to refine the malicious objectives of the feared threat agents introduced in our ANTI-STS model at Business view (shown in Figure 50), which shows the social/technical dependencies between the threat agents for fulfilling their malicious objectives. It is to note that, our proposed Anti-STS Risk modelling technique is in preliminary stages. Thus, we limit our illustration to present the conceptual modelling perspectives of Anti-STS with an example threat profile.

We consider a hypothetical threat profile of feared threat agent i.e., competing airline with a motive to spoil the reputation of the airline as shown in Figure 50. The respective Anti-STS Model (in Figure 54) describes that the competing airline wanting to spoil the reputation has delegated the “adverse maintenance” anti-goal to a new threat agent called threat manager. The threat manager is believed to be having some technical skills and capable to plan threat modelling scenarios e.g., STRIDE [Microsoft 2016]. In below, we explain the how Anti-STS Model facilitates to integrate the extended risk related concepts of ISSRM domain model with reference to FAIR risk taxonomy.

The *Pre-requisites* shows that even threat agents have constraints and pre-requisites to launch an attack. For example in Figure 54, the threat manager would need to know when and where the maintenance activity will be taking place for planning the threat scenarios. This information correspond to the constraints of the threat agents in order to initiate an attack or to establish a point of contact with the targeting asset in the enterprise network (e.g., time dependent analysis [Arnold et al. 2014]). In Figure 54, we assume that threat manager gathers this information through the means of a worker at the Airport.

The refinement of the anti-goals using OR constructs shows various possibilities available to a threat agent for satisfying the malicious objectives. For example in Figure 54, advertizing aircraft maintenance can be threatened in several attack methods such as: (a) *bribing maintenance people* (e.g., exploiting the greediness); (b) *physical threatening* (e.g., blackmailing); (c) *get remote access to the laptop* (e.g., privilege escalation); (d) *intrude the interaction* (i.e., man-in-middle attack). Respectively, refinement of each of this refers to cyber threat trees [Ongsakorn et al. 2010], which will enable security experts to determine the likelihood of threat events of successful attacks (e.g., using Bayesian conditional probability distributions [Wu et al. 2012; Buldas and Lenin 2013]).

The *protection needs* (referred as protection events in Figure 44d) threaten the anti-goals i.e., the leaf goals of the attack trees. Protection strategies are the composite security goals pertaining to defend the security design of enterprise network (e.g., multi-layering security, see SABSA [Sherwood 2005]).

Expressing protection needs/strategies as protection events allows to explicitly links the anti-goals and security goals/protection needs (e.g., similar to attack-defence trees [Kordy et al. 2011]).

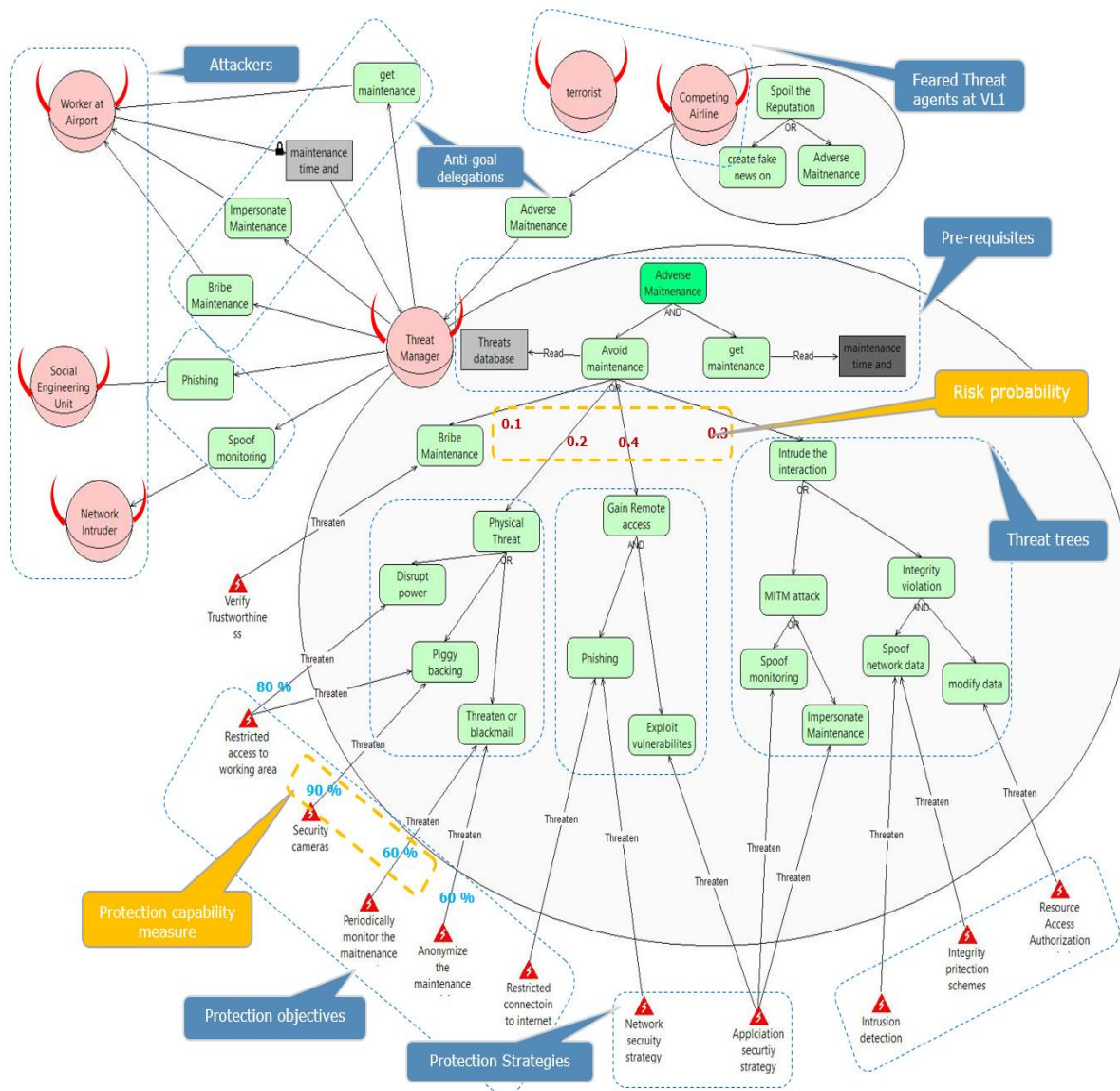


Figure 54: Scenario - Risk analysis using ANTI-STS (sample)

The *protection capability measure* expresses will allow security experts in expressing the anticipated rigor of security control strength needed in order to mitigate the threats based on risk probability. For instance, in Figure 54, if we consider the example threat event concerning *bribing the maintenance people*. The protection objective demands the verification of trustworthiness of the maintenance people. Which enforces security experts to determine trust level (referred as security domain information at Architect's view). In case the maintenance people are unconditionally trusted then the likelihood of this is negligible. Consequently, the security protection need relative to the verification of trustworthiness will become obsolete. Likewise, one can prioritize the threat events relative to likelihood of risk (e.g., similar to the fuzzy logic-based threat prioritization technique [Singhal and Banati 2013]). Finally, similar to STS which promote threat propagation, the propagation of protections event allows the security experts to trace to the malicious motives (e.g., spoil reputation) of feared threat agents identified at Business view.

3.2.3 Designer's view (zone modelling)

This view concerns with realising the conceptual protection strategies (application/network protection strategies as shown in Figure 54) into logical system design elements. Since our thesis work is about network security requirement engineering, we confine this view to integrity the network security zoning, which is a known defence-in-depth strategy for network security design. Relatively, in below we implementation of our proposed zone modelling methodology (described in section 3.2) in the context of example use case scenario 1. At the end of this view, we capture the security zone information along with integrity validation filter requirements, access control requirements and data flow integrity requirements, see Figure 55.

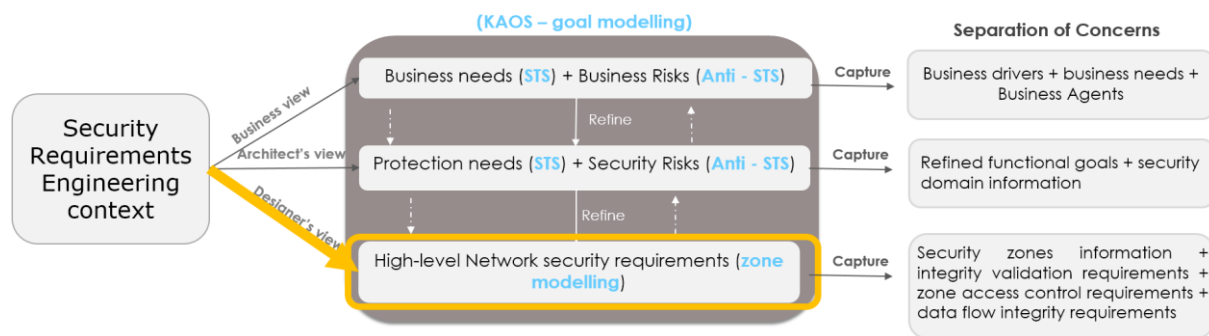


Figure 55: Designer's view - separation of concerns

3.2.3.1 Step1 – Eliciting security zones and integrity validation filter requirements

As shown in Figure 47, the initial knowledge on the system concerning the agents, domains and their integrities needs to be provided as input. We derive this knowledge from the elicited information at Architect's view (see the composite requirement 4 in Figure 52). In addition, the initial input also needs to include the list of permitted data flows between agents with regards to the business objectives. For this, we derive the permitted flows from the interactions of the agents. ASP specification of input flow are listed in Table 7. In our case we have only three flows defined. For example, role maintenancePeople must be able to send data flows to the agent monitoringApplication and vice versa.

Table 7. Step 1 input – sample of permitted data flows

Interacting agents	List of flows
Maintenance people, monitoring application	Flow(maintenancePeople,monitoringApplication)
	Flow(monitoringApplication,maintenancePeople)
	Flow(aircraftControlApplication,monitoringApplication)

This derivation of flows from interactions is inspired from BPMN (Business process modelling notation), which uses a flowcharting technique (similar to activity diagrams [Schmidt et al. 2016]) to visually depict the sequence of activities (i.e., task goals) and information flows between system components. Respectively, we use the dashed line with open arrowhead pointing the direction of message flows (taken from BPMN notation), see the composite requirement 6 in Figure 56. We refer these flows as network data flows that are bi-directional. The arrowhead points to the passive agent. The other end of the line (with circular end) refer to active agents who initiates the communication.

Therefore, in our example scenario, aircraft control application initiates the interaction to end data to monitoring application and vice versa is not possible.

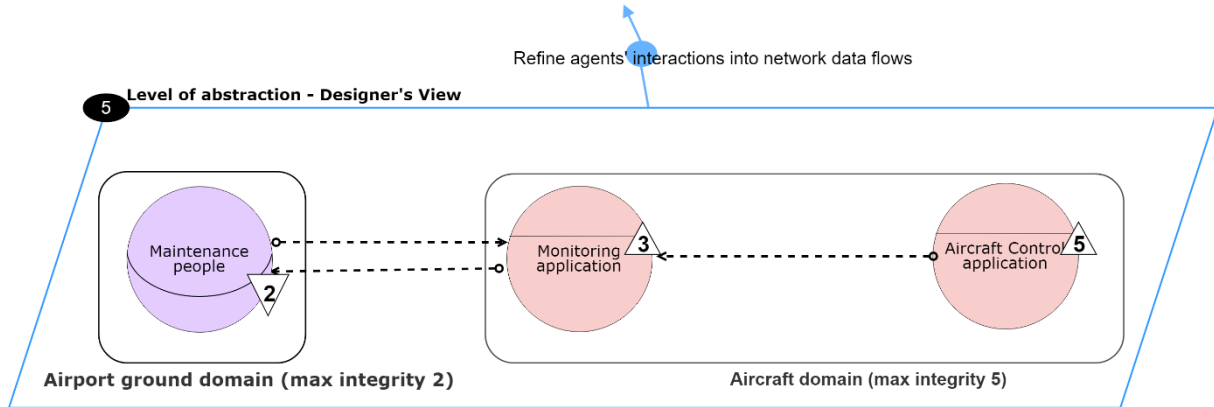


Figure 56: Refinement of network data flows

Given the input information from the composite requirement 5, our ASP-based tool automatically identifies the security zones and the integrity validation filters according to the rules listed in section 3.1.4.1. We express the step1 output in the composite requirement 6 in Figure 57.

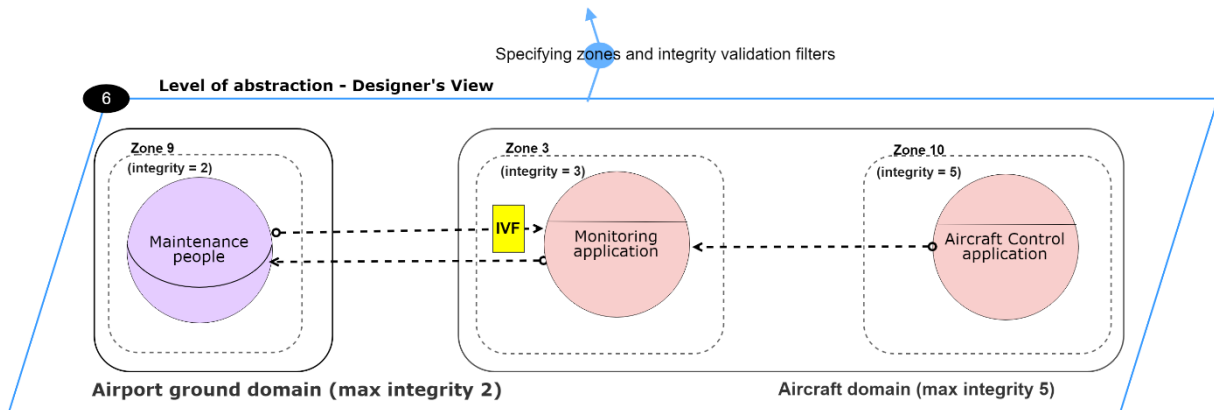


Figure 57: Zone modelling STEP1 – output (use case scenario 1)

Indeed, the Clingo ASP solver can produce many solutions (i.e. zone models), which can equally satisfy all the rules. However, their costs of implementation vary. From a broad view, the implementation cost is the summation of security management costs (to preserve the actual integrity levels of each zones) and the cost of security verification costs to uphold the integrity verification filtering requirements. Respectively, our tool computes the optimized solution using the ASP optimization statement *minimize*, to find the minimal cost of solution, see Figure 58.

```
14 % Optimisation
15 #minimize {L,A:integrityActual(A,L);DSTL,filter(A,F,SRCL,DSTL):filter(A,F,SRCL,DSTL)}.
```

Figure 58: Cost-based optimization rule

In our case, the Clingo solver searches for an optimal answer set simply with the least sum of weights of all the literals (i.e., actual integrity value L of agent A with respect to the filter at a flow F). However, the cost calculus can vary differently for different organisations based on their logistics study similar to the varying cost effects of the Design Assurance Levels [PRICE Systems 2005]. Thus, other and more

complex cost computation formulas can be specified. Our zone model is abstract and design independent therefore does not restrict the design solutions. The obtained zone solution at step1, subsumes a total of three zones (i.e., zone9,) with two zones (i.e., zone3 and zone 10) inside aircraft domain and one zone (i.e., zone9) in airport ground domain. However, our tool cannot classify the zone types like DMZ, restricted, etc. In addition, we have retained the random zone numbering as such generated by ASP solver as it does not allow to customize it.

Furthermore, the calculated integrity levels attached to the zones also apply to the agents within the zones. These integrity levels have to be interpreted as the pre-requisite security requirements. The future security design implementing these requirements must maintain these integrity levels at minimum. Indeed, in practice, there already exist formally accepted approaches, which specify the profoundness of security verification required; for instance, the design assurance levels (DALs) in aircraft systems [Bieber et al 2011]. The higher the DAL is, the higher the assurance activities and design verification methods are demanded. In our methodology context, the actual integrity levels of zones and system agents exhibit similar characteristics of DALs.

Additionally, our tool identifies the integrity validation filters (IVF) attached to agents, listed. In our modelling notation we depict them in yellow squares, see Figure 57. These integrity validation filters represent validation processes implemented either by the target agent (e.g., by some specific validation code) or some external security mechanisms (e.g., deep inspection mechanisms). In example scenario 1, our tool specifies that the data flow between the maintenance people (zone 9) and the monitoring application (zone 3) must be validated.

Table 8: Step1 - IVF rules (use case scenario 1)

<pre>IVF (monitoringApplication, flow (maintenancePeople, monitoringApplica tion), sanitize (2, 3))</pre>

In Table 8 the specification describes that there is an IVF defined at monitoring application agent, for the *flow(maintenanceProple, monitoringApplication)*, with *sanitize (2, 3)*. Here, *sanitize (2, 3)* specifies that an incoming data flow having an integrity level of 2 (this value is inherited from the integrity level of the maintenance people) must be sanitized in order to conform to the integrity level 3 (of monitoring application). Interpretation of such integrity validation requirement, i.e. what means validation to conform integrity level of 3, can be carried out on the basis of dedicated documents such as the specification for data assurance levels by EUROCONTROL [EUROCONTROL 2012]. Suitably, *IVF with sanitize (2, 3)* might be implemented by a security mechanism such as a web application firewall that checks for SQL injection/viruses/etc. Likewise, at the end of Step1, the security architect obtain the list of zones, the integrity values of agents and zones and integrity validation filters.

3.2.3.2 STEP 2 Implementation Step2 input information

Before running step2, the security architect needs to complete the output from step1 by providing additional information about the media of communication (see Figure 47). The input information concerning medium of communication includes the integrity values of the medium, the domain in which the medium belongs to, and finally, the zones connected to it (the white clouds and the black lines in

Figure 59). In this regard, in our example scenario, we assumed three medium of communication: *Aircraft access* and *Aircraft Internal Access*.

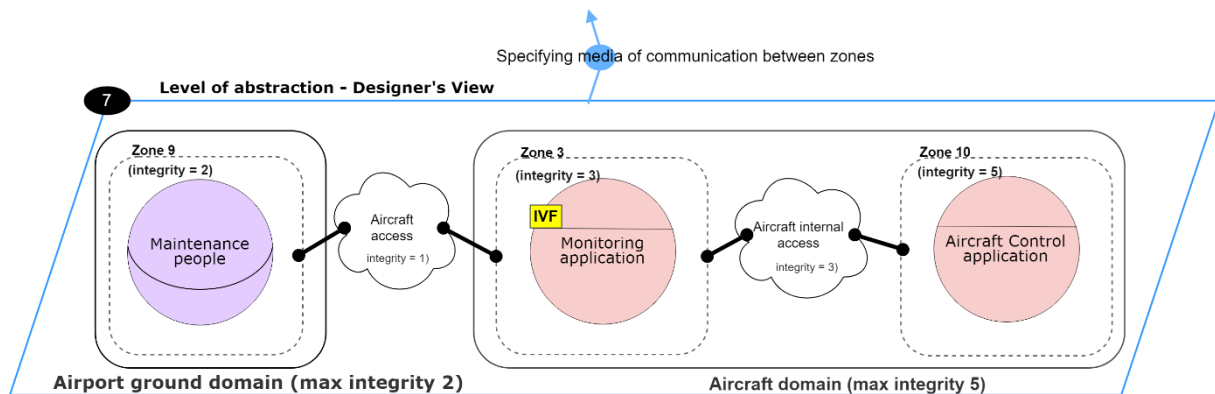


Figure 59: Zone modelling STEP2 – input (use case scenario 1)

The integrity value attached to a medium of communication represents the level of trust one can have about the packets transmitted by the medium. As consequence, this integrity value depends on the assurance level that only known users can connect to it. The integrity value will be calculated as the minimum between the integrity value of the domain and the connected zone. We also allow the security architect to specify an initial integrity value that represents the risk that unexpected users have access to the medium. For instance, it is easier to access a wireless network than a wired network. In this way, our methodology can easily integrate a second risk analysis phase dedicated to how easy it is to connect to networks. Figure 60 shows an example of the integrity levels of the media for example use case scenario 1, calculated based on the restricted levels of the medium.

Medium	level of Control + openness to access		Restricted level	Integrity
Aircraft Internal Access	Controlled	Restricted connection only to the agents inside the aircraft	Restricted	3
Aircraft Access	Uncontrolled	Not applicable - public network medium	Cannot be restricted	1

Figure 60: medium integrity levels example (use case scenario 1)

The access control filter rules (ACF) are defined (from RULE11) at the interfaces of each zone in order to control all the inter-zone communications. Respectively, the filter rule can either permit or deny a flow through a medium.

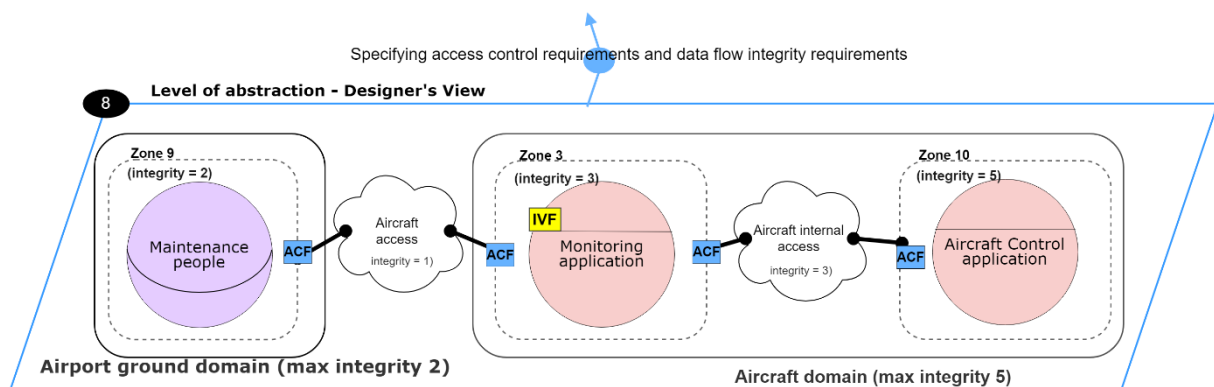


Figure 61: Step2 output –FINAL

These requirements describe the list of permitted flows that are given as input at step1. If not explicitly stated in an ACF requirement, data flows are denied. Table 9 shows a sample of the generated ACF requirements for the use case scenario 1. For instance, the requirement statement *ACF(connectedTO(aircraftAccess,9), flow(maintenancePeople,monitoringApplication))* indicates that there must be an ACF at the interconnection point between medium of communication *aircraftAccess* and *security zone9* that permits data flows from the *maintenancePeople* to *monitoringApplication*. Depending on the security design choices, these ACF may be implemented by one or more access control mechanisms such as firewalls, application gateways, etc.

Table 9: Step2 – ACF rules (use case scenario 1)

```
ACF(connectedTO(aircraftAccess,9),
    flow(maintenancePeople,monitoringApplication))
ACF(connectedTO(aircraftAccess,9),
    flow(monitoringApplication,maintenancePeople))
ACF(connectedTO(aircraftAccess,3),
    flow(maintenancePeople,monitoringApplication))
ACF(connectedTO(aircraftAccess,3),
    flow(monitoringApplication,maintenancePeople))
ACF(connectedTO(aircraftInternalAccess,3),
    flow(aircraftControlApplication,monitoringApplication))
ACF(connectedTO(aircraftInternalAccess,10),
    flow(aircraftControlApplication,monitoringApplication))
```

Finally, the tool produces data flow integrity requirements (from RULE13) representing security protection needs attached to data flows while transiting over a medium or a zone (see Table 10). They are depicted by grey squares in Figure 61). For instance, the requirement *dataFlowIntegrityRequirement(flow(maintenancePeople,monitoringApplication), aircraftAccess,2)* states that the data flows from *maintenancePeople* to *monitoringApplication* must be protected over medium of communication *airportAccess* to ensure the data flows maintain integrity level 2. Different integrity mechanisms such as digital signatures can implement these requirements.

Table 10: Step2 – data flow integrity requirements (use case scenario 1)

```
dataFlowIntegrityRequirement(
    flow(maintenancePeople,monitoringApplication), aircraftAccess,2)
dataFlowIntegrityRequirement(
    flow(monitoringApplication,maintenancePeople), aircraftAccess,2)
```

3.3 Chapter Summary

In this chapter, we presented our layer based SRE methodology built on SABSA abstraction layers. It subsumes three layered views i.e., Business view, Architect's view and designers view. Each view represents the separation of concerns of stakeholders working at different abstraction layers, see Figure 62. We introduced the notion of composite requirement, which merges modelling feature of STS and secure KAOS in order to accommodate the abstract levels at each view. We illustrated the SRE methodology with the user case scenario 1 concerning aircraft maintenance process. Appendix D provides a complete view of the SRE specification for the AIRBUS use case scenario 1.

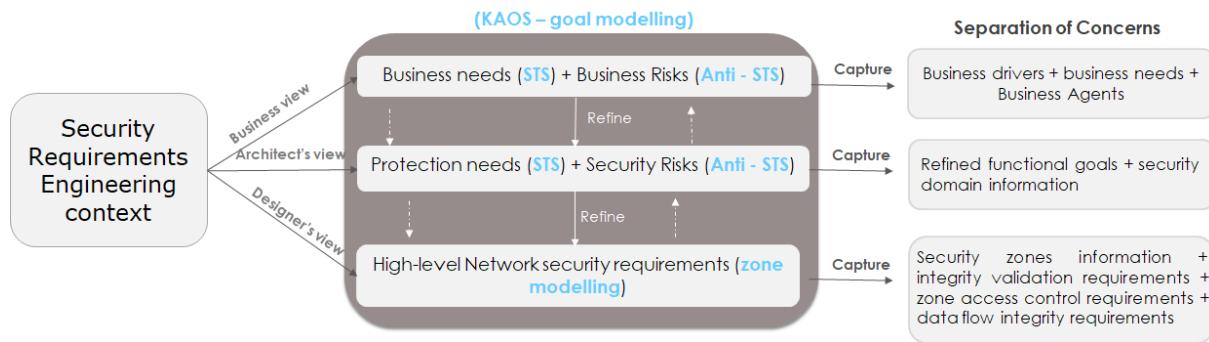


Figure 62: Layered SRE methodology - complete view

Business view introduces the business context of enterprise network. It encompasses identifying the business agents representing the enterprise itself (e.g., AIRBUS) and their respective business needs (e.g., make aircraft maintenance).

Architect's view introduces the global security context of enterprise network system. It concerns the refinement of business functional goals and system actors in STS modelling; capture of protection needs with reference to the protection events realized through Anti-STS Risk modelling. In addition, security domain information (i.e., criticality/trust levels of agents and control capabilities of security domains) are captured in terms of unified integrity values, which permits to express enterprise security risks information.

Designers view introduces the network security design context of enterprise network system. We proposed a zone modelling methodology based on three security principles: complete mediation, least privileges and Clark-Wilson lite formal model. We defined a set of formal rules as well as the list of initial integrity levels values computed based on risk impact, which makes our methodology approach traceable and verifiable. The whole process has been implemented in ASP to automate the security zones computation. It produces a set of network security requirements: security zones, integrity validation filters, access control filters, and data flow integrity requirements.

CHAPTER 4 EVALUATION OF PROPOSED SRE METHODOLOGY

In this chapter, we validate our layer based SRE methodology using the evaluation criteria derived in chapter 2. We repeat the step 3 of our proposed evaluation methodology using two use case scenarios. The first scenario concerns the e-commerce enterprise networks taken from ANSSI website. The next scenario concerns the AIRBUS use case scenario 2 from IREHDO2 project (described in section 2.5.2). It is to note that, since Anti-STS for risk modelling has not been fully developed, we confine our evaluation to their layered framework.

The actual evaluation process as described in section in chapter 2 includes two iterations. In the first iteration we authors created the SRE specification and presented the model to the real stakeholders at AIRBUS. In the second iteration, we let the real stakeholders create the SRE specification. In this thesis work, our evaluation analysis and results are confined to the first iteration alone since we have not yet developed the tool. Respectively, the evaluation results that we provide in this chapter correspond to the initial feedback obtained from the security experts. However, we could not test all of the criteria (e.g., **R^M2.1.3**), the details of which will be provided in the discussion part.

We first brief the e-commerce scenario and the corresponding network SRE context (section 4.1). Then we present the SRE specification for the scenario that describes composite requirements specified at *Business*, *Architect's* and *Designer's* views (section 4.2). In this respect, we describe the security domain information that expresses the risk assessment security domains and agents in terms of integrity values. Section 4.3 briefs the implementation of airbus scenario 2 (described in section 2.5.2). Finally, we report our experience in creating the specifications for the two scenarios using our SRE methodology. In this respect, we highlight its pros and cons with reference to the verification methods of the quality criteria detailed in Appendix C. At the end, we discuss the threats to validity of our evaluation process (section 4.4).

This chapter consolidates the following contributions [Bulusu et al. 2018c; Laborde et al. 2019].

4.1 E-Commerce case study description

The scenario e-commerce enterprise network from the reference [ANSSI 2017] provides us with an initial network architecture. It comprises of server components such as WEB server, DNS server, Application server, Database server, and Accountability server. The employees are distinguished as administrators and standard users, who can connect to the network through LAN or WIFI. If the employees are outside the enterprise, they can remotely connect to the enterprise network. This will differentiate local users and remote users. Finally, when the clients visit the enterprise (noted as visitors), they are allowed to connect to the web through a dedicated WIFI network.

The accountability server is described as highly critical as it manages the financial information of the company (e.g., salaries of employers). The web server hosts the e-commerce web site. Therefore, it is also critical because a denial of service attack will highly affect the business of the company. Finally, the web server requires interactions with the application and database servers to provide the e-commerce service. Consequently, they are also highly important assets for the business especially the database for which data integrity is primordial.

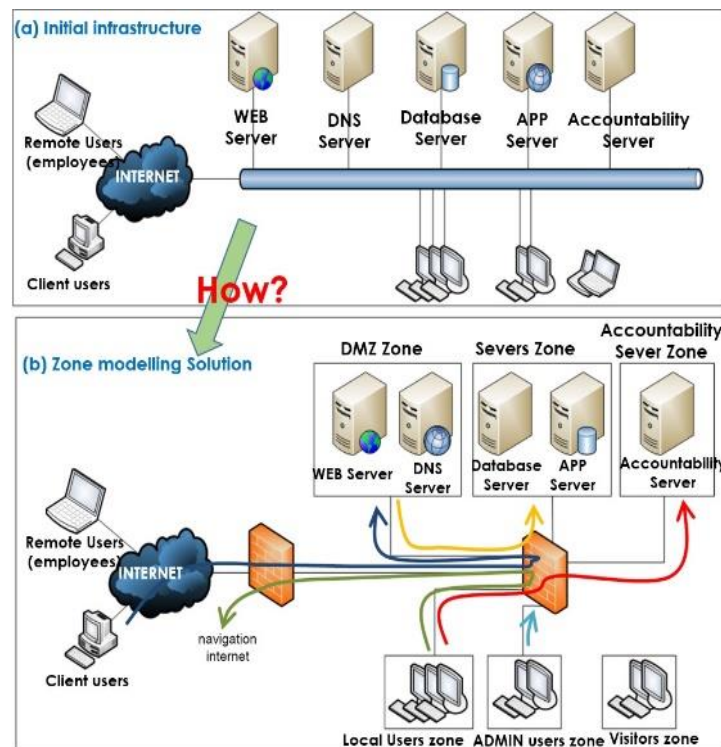


Figure 63: e-commerce example case study [ANSSI 2017]

There can be several ways to improve the security of this enterprise network. The network security requirements engineers propose the example security architecture in Figure 63b (and described in [ANSSI 2017]). The example solution reflects some best practice guidelines by defining some zones such as DMZ zone, user's zones, etc. For instance, the accountability server is isolated in a separate zone as it is highly critical. In addition, total of two firewalls have been defined one to monitor the internet traffic flows and the other firewall to monitor the internal traffic of the e-commerce enterprise. In this respect, some advocated list of security measures are as follows:

- (a) Provide the access interface to the internal network from the internet, usually via a VPN tunnel;
- (b) Check the authorization of the connection requests from remote users;
- (c) Check the security level of the users before allowing the connection (e.g., using patches and anti-virus update in particular);
- (d) If everything is OK, then allow streams to the inner zones (and only those that are needed for the communication), must always going through the firewalls.

However, it is not clear on how the architects concluded to this zoning solution as well as derive the security measures. This approach requires additional arguments to demonstrate the solution meets the elicited risks. For instance, with reference to the example requirement (d), how to verify if the traffic

flows are needed for communication or not. Similarly, how to ensure that the proposed network zoning solution is correct and cost-effective? How to ensure that no network security requirement is missing or irrelevant? Here, a formal approach justifying the transition from the problem to the solution is required for a traceable and verifiable security zone specification process.

In below, we describe the first iteration of the evaluation, which concerns the implementation of the aforementioned scenario using the layer based SRE model. The following section 4.2, we first consolidate our observations and comments on the SRE modelling experience. While presenting our specification at each layer, we highlight the relative evaluation criteria (from Appendix C) in parenthesis wherever required so that it facilitates link our observations to the verification of the evaluation criteria. At the end in section 4.4, we provide the consolidated results of our evaluation in tabular format, where we provide the overview of our initial analysis results and observations.

4.2 E-commerce scenario implementation

In this section, we implement the e-commerce scenario using our layer based SRE model. It is to note that the information provided in the example scenario does not respect the abstraction layers as per our methodology. Therefore the scenario is discussed incrementally with increasing level of details, respecting the abstraction views. Accordingly we use the information provided in the scenario that is relevant to each view. In some cases, where information provided in the scenario seemed not sufficient. For instance, since the scenario describes the network architectural components but do not provide details on their respective goals and the interactions needs specific to e-commerce business. Therefore, in order to facilitate the expression of scenario at different abstraction layers, we had to consider some hypothetical information related to functional goals wherever needed. We first describe the specification in three layers, then we provide our evaluation analysis.

Since we did not have any tool, we principally used two tools to create our SRE specification. First, STS-tool [Paja et al. 2014] to model STS diagrams, then we used Draw.io [JGraph Ltd] , an online diagramming software tool to model our layer based SRE specification, which consumed significant amount of time (corresponds to the evaluation criterion **R^M2.1.2**).

4.2.1 Business view

In our methodology, we start by specifying the relation between business needs and the driving objectives of the business, also known as business drivers. Business drivers are the objectives that are vital for the continued success and growth of a business. An organisation can have several business drivers each accorded with a business value.

Figure 64 depicts the e-commerce scenario specification at Business view. The specification is targeted to express the business context of business users whose objective is to manage e-commerce business. In our specification we considered them as an environment agents since they directly refer to the business people of the organization are expressed using the role “e-commerce”, see the composite requirement (CR) n°1. The business driver concerns the operational continuity of the e-commerce

business. The refinement of link of business driver into identifying the high-level business need we refine the “manage e-commerce business” goals into high-level functional goals (see CR n°2).

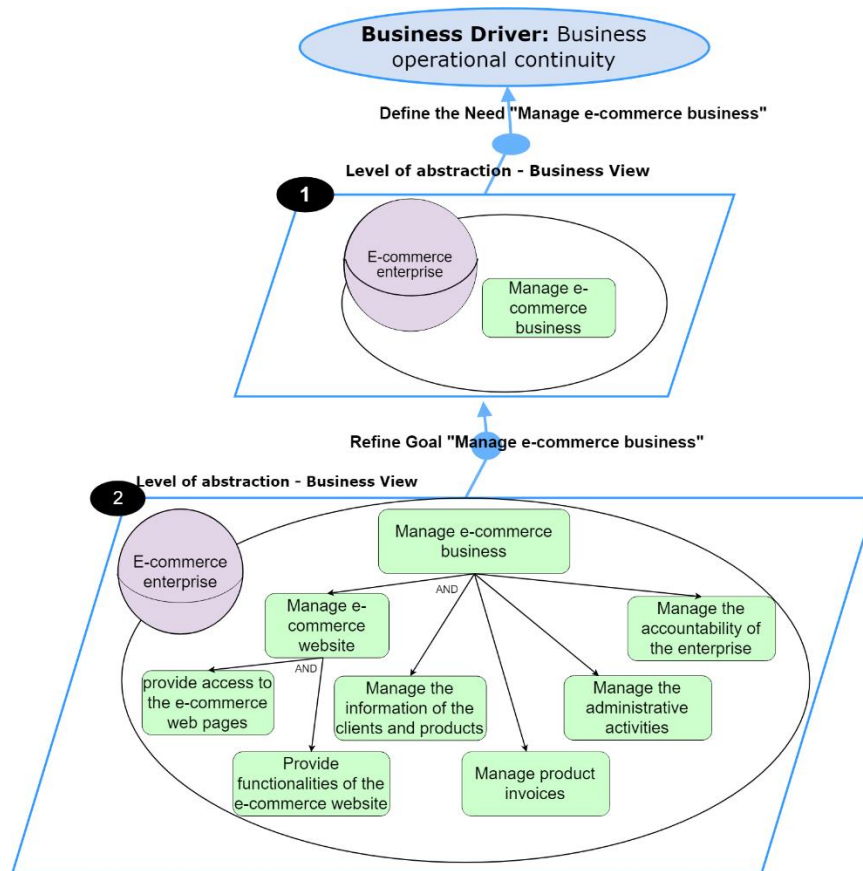


Figure 64: E-commerce scenario specification - Business view

Altogether, the Business view confines to the elicitation and refinement of ‘WHAT’ aspects of the business needs from the business users. This strictly leaves out the concerns of the ‘HOW’ aspects, which concerns the further refinement of functional goals (corresponds to the criterion $R^{M3.1}$). In addition, since we use STS notation to express the business agent and respective goals, we inherit the comprehensibility characteristics of STS notation relative to the quality criterion $R^{M2.1.1}$.

4.2.2 Architect’s View

At Architect’s view, where we start identifying the roles and responsibilities of the actors involved in the e-commerce business process. The refined business goals at Business view are therefore delegated to the respective agents. The composite requirement n°3 in Figure 65 depicts the initial step of election process at Architect’s view. This will permit us to introduce the architectural components described in the e-commerce scenario that include WEB server, Application server, Database server, Accountability server and the E-commerce employees who include admin user as well.

In the specification, we express the delegation as well the refinement of functional goals in 6 sub-composite requirements (i.e., CR n°3A to CR n°3F). Thanks to the AND-refinement link of composite requirement (described in section 3.1.1), enables to use divide and conquer approach [Knuth, 1998]. This eventually reduces the complexity of the elicitation process of STS modelling, thereby allowing the security experts to independently work with the refinement of each functional goals.

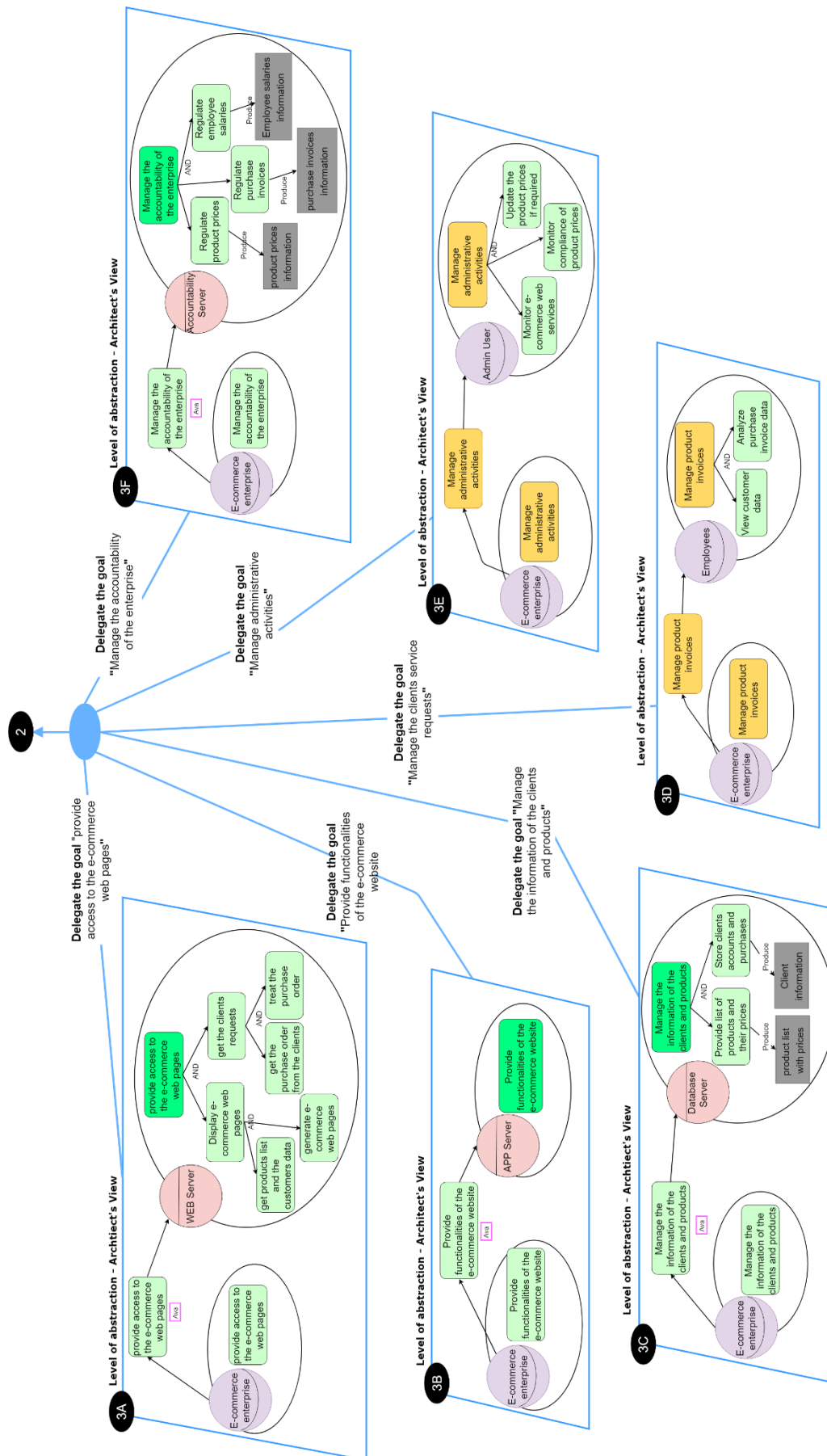


Figure 65: E-commerce scenario - refining business goals

In addition, this type of refinement also helps in tracing the refinement context (otherwise ‘HOW’ aspects) of each of the functional goal elicited at Business view as well as the inclusion of new agents identified as architectural components (corresponds to the criterion $R^{M3.2}$). Whenever the agents identified as environment agents (e.g., administrator/employees), the functional goals is transformed into an expectation (expressed in yellow colour). However, at some point of time these architectural components need to interact with each other for fulfilling some functional goals, which requires the transmission/exchange of information between multiple agents. Suitably, the CR n°4 in consolidates the individual composite requirements (i.e., CR n°A to CR n°F) in order to facilitate the expression of the interaction needs of the architectural components, see Figure 66.

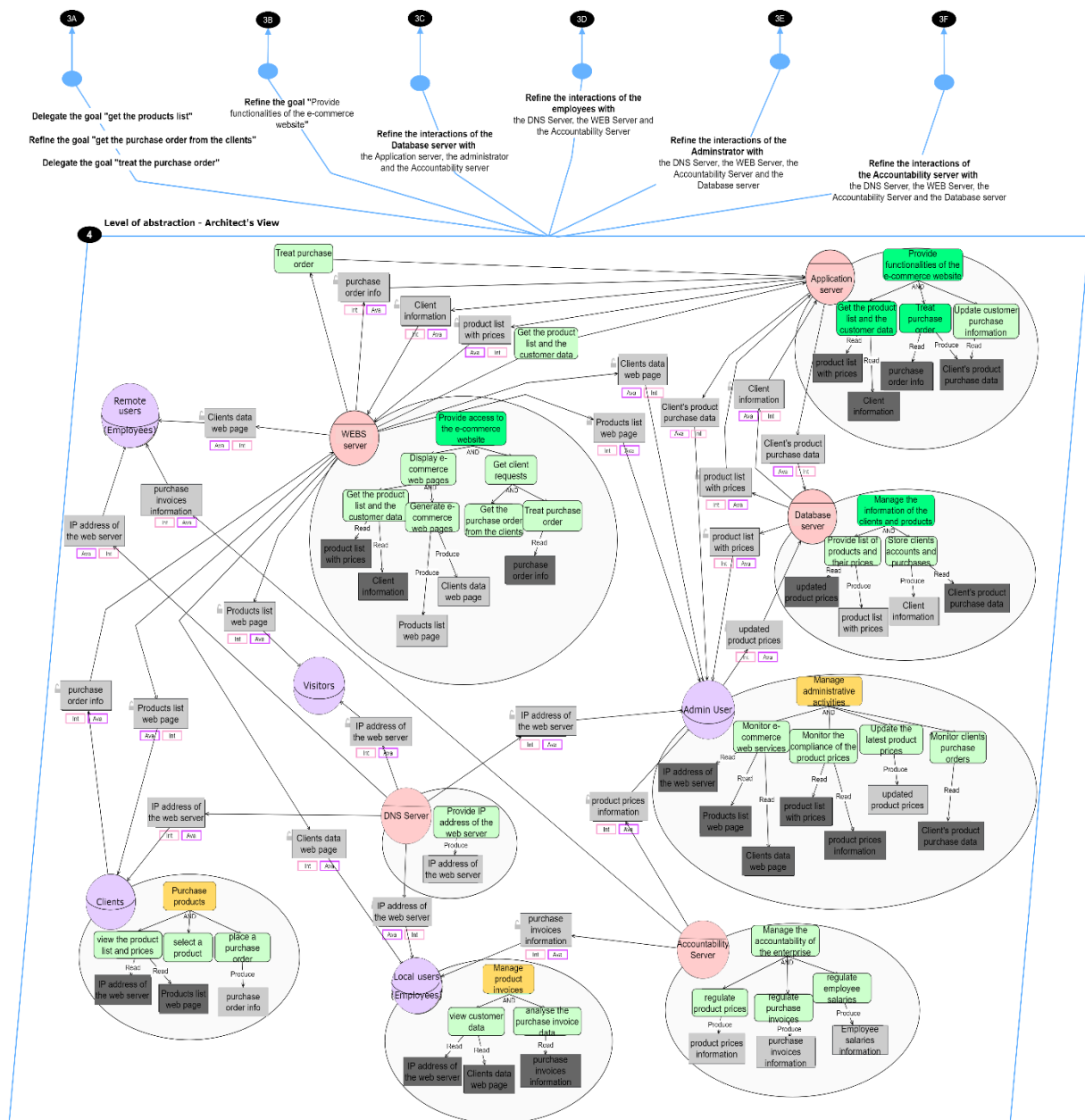


Figure 66: E-commerce scenario - refining agent interaction needs

In addition, the CR n°4 specification also allows to express the security constraints over the interaction needs (characteristics of STS notation). For instance, the commonly identified security constraints express the conservation of integrity and availability properties of the transmitted data.

Nevertheless, not all agents' interaction needs may require same level of security protection. Therefore, we need to further integrate the risk analyses process to facilitate the assessment of risks related to the data resources and objectives assigned to agents. Thanks to the security domain information that allows to assess risks related to security domains and agents to provide domain control capability, trust of environment agents and criticality of system agents (corresponds to the evaluation criterion $R^M3.2$).

For security domain information, we assume the existence of some utility functions that maps the control capability labels of domains, criticality and trust levels of agents into a unified scale of integrity values (see Figure 67).

Domain	Control capability	Integrity
Inside enterprise sub- domain	Highly restricted	5
Outside enterprise sub- domain	Less controlled	2
internet	uncontrolled	1
System agents	Criticality	Integrity
WEB server	Partially critical	3
DNS server	Partially critical	3
APP sever	Partially critical	3
Database server	Critical	4
Accountability server	highly critical	5
Environment agents	Trust	Integrity
Admin user	highly trusted	4
Local user	trusted	3
visitor	Partially trusted	2
Remote user	Partially trusted	2
Client user	not trusted	1

Figure 67. Integrity values of domains and agents for the example case study

The integrity values assigned to domains specify the maximum level of integrity that can be reach within the domains since they express the maturity of an enterprise to deploy security controls and/or its capability to control a given environment. In our e-commerce scenario, we identify two domains: the enterprise domain and the rest of the world named internet. The enterprise domain is divided into two sub-domains: the internal sub-domain and external sub-domain. The internal sub-domain consists in the assets within enterprise premises that are physically secured. We consider also that the enterprise is mature with well-trained employees. Thus, this sub domain is well controlled. The outside sub domain consists in employees working from their home using laptops provided by the office. Then, it is less controlled. The internet domain being outside the control of the enterprise is uncontrolled.

The integrity levels of environment agents express their trust levels. They are assessed based on the the degree of the trustworthiness over their expected behavior in a given context. These levels correspond to the maximum level of integrity, which is expected as guaranteed (depicted in inverted triangular shapes). For instance, in our use case, the administrator is highly trusted because this person is well-known and qualified. Local employees are only trusted because this role refers to more people. Visitors are partially trusted because they are known people and the reception staff verifies the visitor's identity card which must be surrendered in exchange for a wearable badge.

Finally, the integrity levels of system agents express their criticality levels. They are assessed based on the sensitivity to threats and their risk impact of system agents' goals on the overall business. These levels (depicted in triangular shapes) correspond to minimum integrity level of integrity that should be maintained in terms of security assurance. For instance, 'providing access to the e-commerce web site' is critical for the business of the enterprise. Thus, the web server, which is assigned to this goal, is critical too. The DNS and the application servers have the same criticality as the web server because the risk is the same if one of them cannot achieve its associated goal. The database server is highly critical. On one side, its goal 'provide list of products and their price' is critical only. On the other side, the goal 'store clients' accounts and purchase' is highly critical because a threat on this goal will have a strong impact on the reputation of the enterprise. Finally, the accountability server is vital because if the enterprise can't manage its accountability leads to bankruptcy. The Figure 68 depicts the CR n°5 that express the security domain information captured for e-commerce scenario.

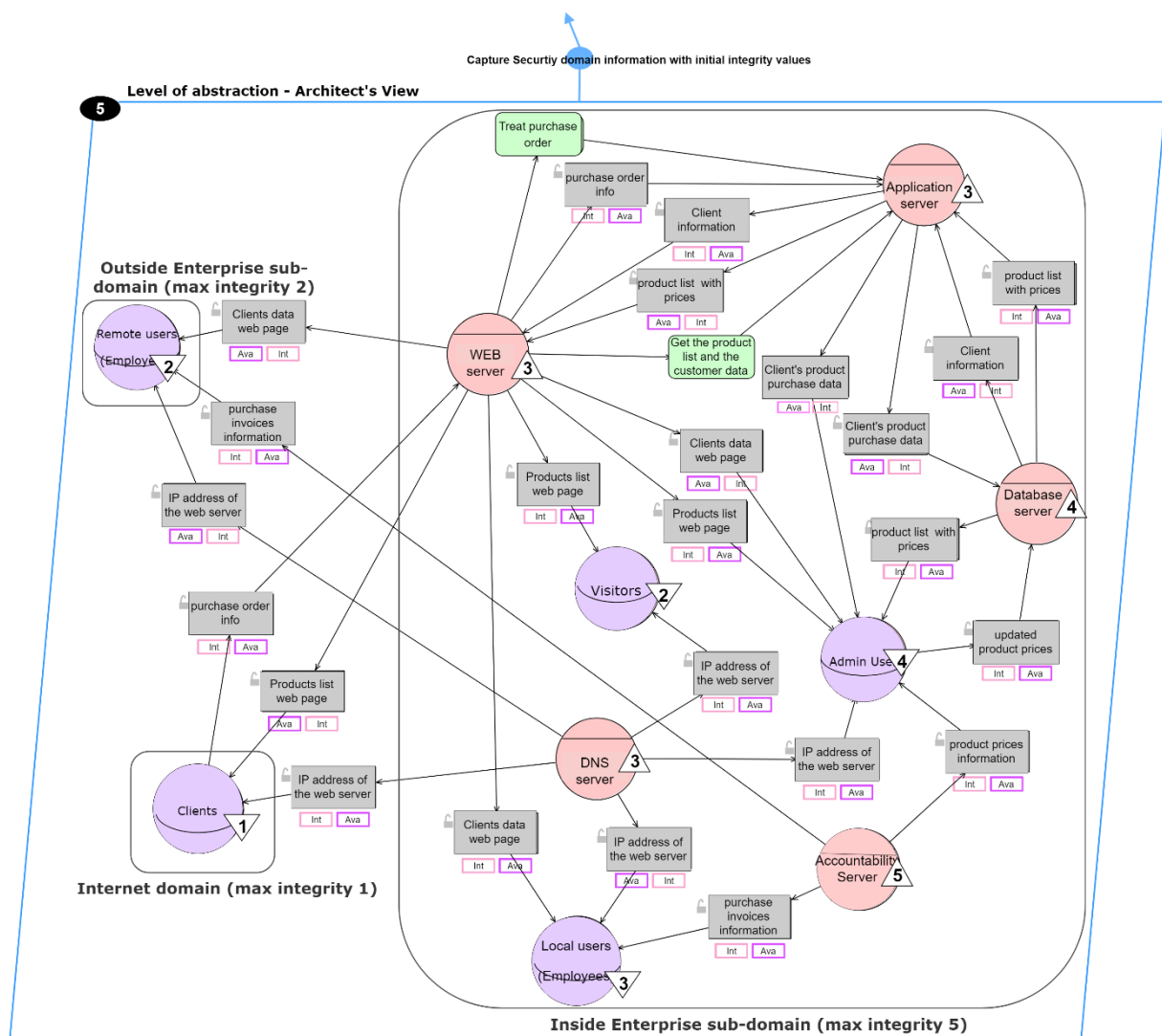


Figure 68: E-commerce scenario - expressing the security domain information

Altogether, the elicitation at this view takes care of two aspects while eliciting. First one is to identify and specify all the concerning actors as well as their respective goals and interaction needs. The second one is to specify the security domain information of these actors, which allows specifying the expected trust over the security management activities in order to protect the critical assets (e.g., system agents,

information resources). At the end of this view, the refined composite must comprise of all the actors that play active role in fulfilling the business needs as well as their anticipated security protection needs.

4.2.3 Designer's view

At design view, we start considering the network security context, where we implement our two-step security zone modelling methodology (described in section 3.1.4). The zone modelling rules (described in section) are implemented in ASP using the Clingo solver [Gebser et al. 2014] and Python2.7.

The first step takes in as input the security domain information expressed in CR n°5 at Architect's view (corresponds to the evaluation criterion $R^M6.2$). In addition, it also requires set of permitted data flows are derived from the resource transmission interactions identified at architect's view, which must be support by the communication. At this point, the security designers has to determine direction of the data flow with reference to the type of flows (e.g., TCP/UDP traffic flows). For instance, interaction flows between users and webserver are bi-directional since they correspond to TCP traffic flow wherein the users are client agents and webserver is the server agent. The set of permitted data flows defined for the e-commerce scenario are given in Table 14 in Appendix D.

The composite requirements specifications CR n°6 and CR n°7 correspond to the step 1 of our zone modelling methodology (see Figure 70). The composite requirements (CR n°6) express the initial input to the first step zone modelling methodology. In e-commerce scenario we have a total of 10 agents and 38 data flows specified at the beginning of the design view.

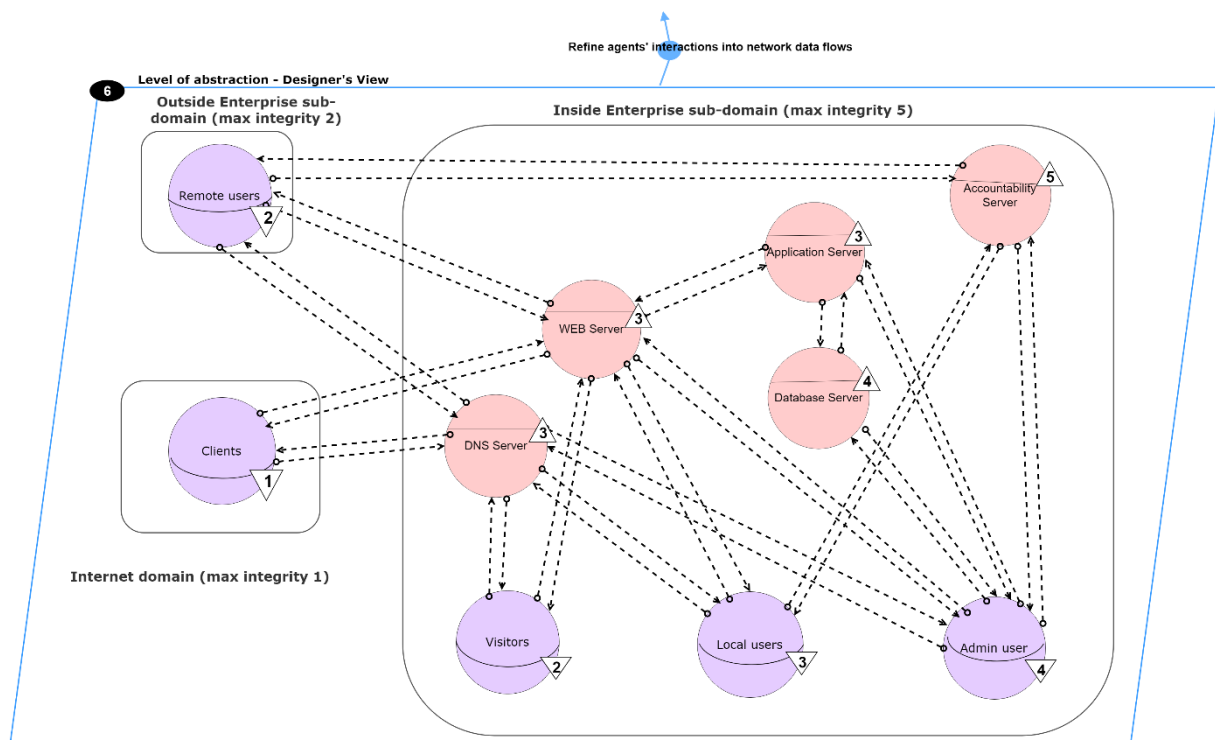


Figure 69: E-commerce scenario - zone modelling step1 (input)

At the end of step1, the computed zone solution subsumes a total of eight zones (i.e., zone1 to zone7 and zone9). Wherein, six zones (i.e., zone1, zone2, and zone4 to zone7) are specified within the inside enterprise subdomain, and the remaining two zones (i.e., zone9 and zone 1) are specified within the

outside enterprise sub-domain and the internet domain respectively. The composite requirement (CR n°7) express the zone modelling output of step1 which specifies the zones and their integrity levels along with the integrity validation filtering requirements (IVF).

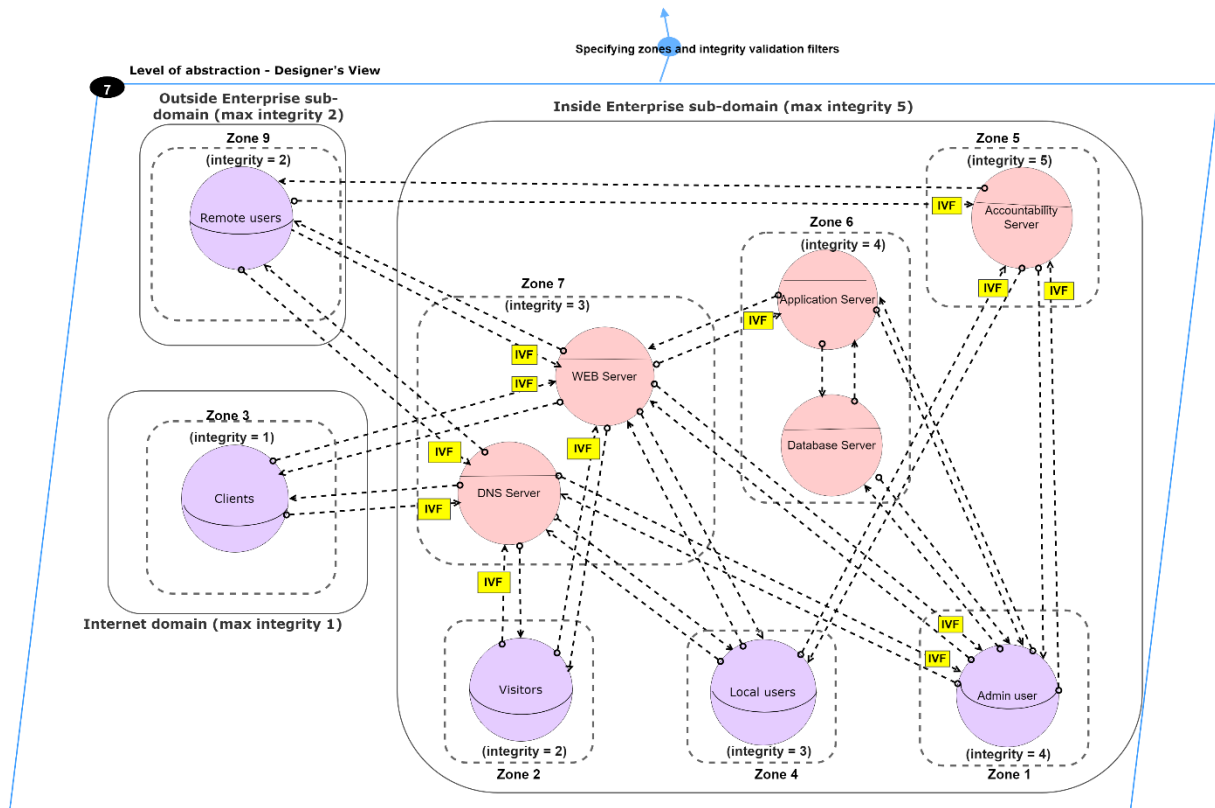


Figure 70: E-commerce scenario - zone modelling step1 (output)

In addition, it provides us with a total of 12 integrity validation filtering requirements as given in Table 11. The refinement link between CR n°6 and CR n°7 permits us to explicitly trace the link to the security zone modelling information (corresponds to the evaluation criterion $R^M 6.1$).

Table 11: E-commerce scenario - IVF rules (step1)

Agents	Integrity Validation Filter (IVF) Rules
DNS Server	IVF(dnsServer, flow(clients, dnsServer), sanitize(1, 3)) IVF(dnsServer, flow(remoteUsers, dnsServer), sanitize(2, 3)) IVF(dnsServer, flow(visitors, dnsServer), sanitize(2, 3))
WEB Server	IVF(webServer, flow(clients, webServer), sanitize(1, 3)). IVF(webServer, flow(remoteUsers, webServer), sanitize(2, 3)) IVF(webServer, flow(visitors, webServer), sanitize(2, 3))
APP Server	IVF(appServer, flow(webServer, appServer), sanitize(2, 4))
Accountability Server	IVF(accountabilityServer, flow(adminUser, accountabilityServer), sanitize(4, 5)) IVF(accountabilityServer, flow(localUsers, accountabilityServer), sanitize(3, 5)) IVF(accountabilityServer, flow(remoteUsers, accountabilityServer), sanitize(2, 5))
Administrator	IVF(adminUser, flow(dnsServer, adminUser), sanitize(2, 4)) IVF(adminUser, flow(webServer, adminUser), sanitize(2, 4))

At step2, our ASP tool takes in two sets of information that includes: the result of step1, and the media of communication and respective integrity values elicited from the security architects. In our example scenario, we assumed three medium of communication: *Private access*, *Public Access* and *Internet Access*. The private and public access medium are controlled by the enterprise wherein the former handles the traffic within the inside enterprise sub-domain and the later handles traffic between the two inside/outside sub-domains of the enterprise. Internet access is uncontrolled since it refers to public access medium. Figure 71 shows the integrity levels of the medium calculated based on the restricted levels of the medium considered in our example scenario. As described in section 3.2.2, these integrity values permit security architects to express security risks when unexpected users or attackers have access to the medium (corresponds to the evaluation criterion $R^{M6.2}$).

Medium	level of Control + openness to access		Restricted level	Integrity
Private access	Controlled	Permitted access only to the employees within the inside enterprise sub-domain and to all the e-commerce server agents	Restricted	3
Public access	Controlled	Permitted access to only WEB/DNS servers and to the visitors	Partially Restricted	2
Internet access	Uncontrolled	Not applicable - public network medium	Cannot be restricted	1

Figure 71: Example Integrity levels for the medium

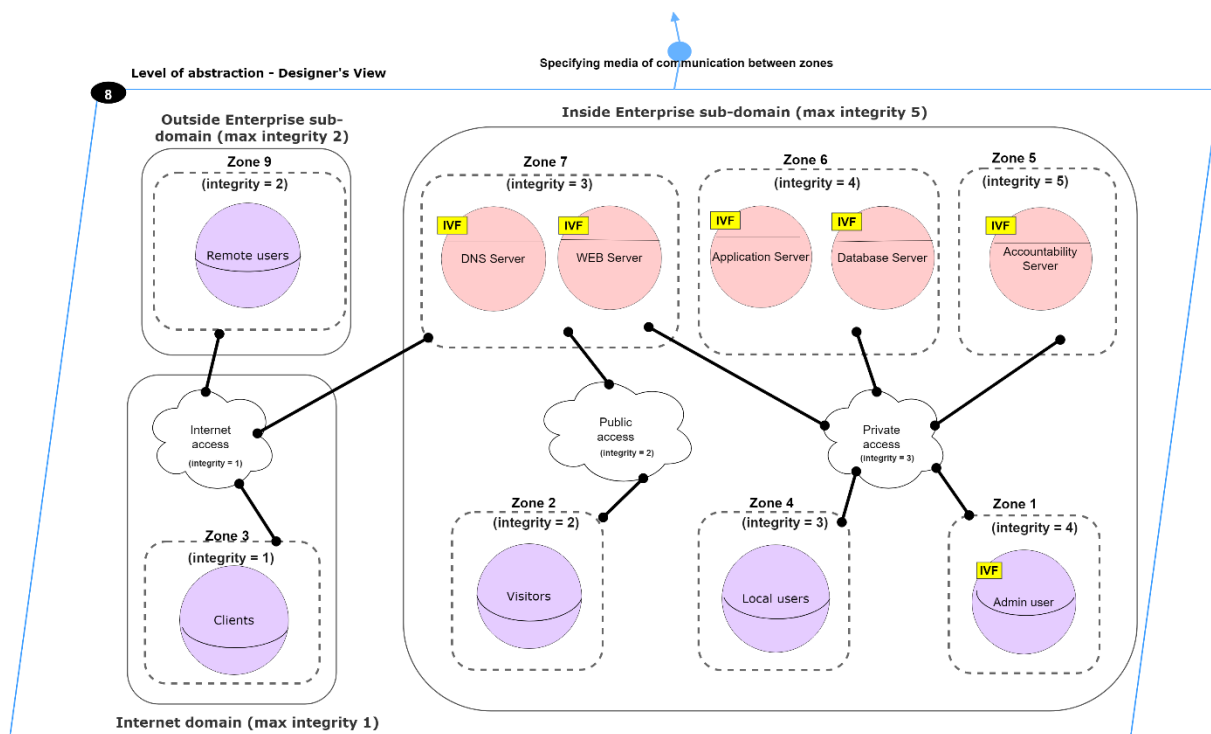


Figure 72: E-commerce scenario - zone modelling (step2) (input)

Respectively, the composite requirements specifications CR n°8 and CR n°9 correspond to the step 2 of our zone modelling methodology (see Figure 73). Wherein, the composite requirements (CR n°8) express the step2 input that aggregates the step1 output with the newly elicited information on the media of communication (given in Figure 71). The composite requirement (CR n°9) express step2 output which corresponds to the final output of our ASP tool (given in Figure 72). It results us with a total of 85 access

control filtering requirements (ACF) 10 data flow integrity validation filtering requirements as given in Table 15 and Table 16 in Appendix D.

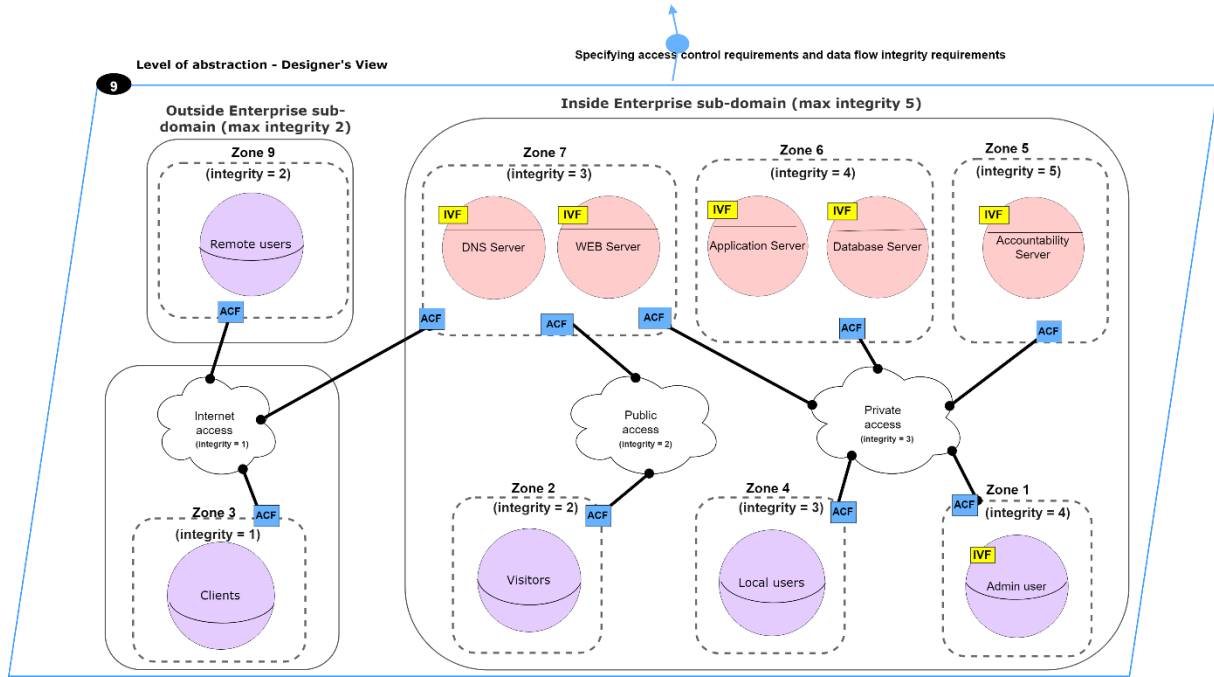


Figure 73: E-commerce scenario - zone modelling (step2)

Altogether, at the end of design view, our ASP tool facilitates to derive high-level network security requirements which can be traced back to high-level business objectives (corresponds to the evaluation criterion $R^M 4.2$).

4.3 AIRBUS scenario 2 implementation

In this section, we implement the e-commerce scenario2 (described in section 2.5.2) using our layer based SRE model. We have employed this scenario for our evaluation iteration 2 in chapter 2. As mentioned in section 2.5.2, the experimentation has provoked the discussion and has eventually helped in developing the scenario details, (see Figure 37) but still miss the abstraction details. In below, we will not explain the SRE specifications in detail like the e-commerce scenario, instead we highlight the important observations unique to this scenario2 implementation. For this, we consider the model given in Figure 37 as an input and we express is in our layer based SRE model.

Unlike e-commerce scenario or AIRBUS scenario 1, this scenario consists in 2 business needs (see Figure 36 in section 2.5.2). Therefore, our SRE model must be capable to express the network security requirements that addresses these two needs.

The first business need concerns the safety of the aircraft. In order to ensure the safety of the aircraft, the aircraft is updated with specific load information. This implies that, update aircraft is a high-level objective that is defined with keeping the safety business driver in mind. The second business need concerns with the provision of inflight services to the passengers in order to enhance the customer satisfaction. Here, we consider the safety and customer satisfaction as the business drivers. The first

business need is expressed as the responsibility goal “update aircraft” within the scope of AIRBUS actor. Here, by saying AIRBUS actor, we refer to some responsible team within AIRBUS who takes care of it. But at this level we do not know the details of the responsible team. Similarly, the second need is expressed as the responsibility goal “Provide entertainment service”.

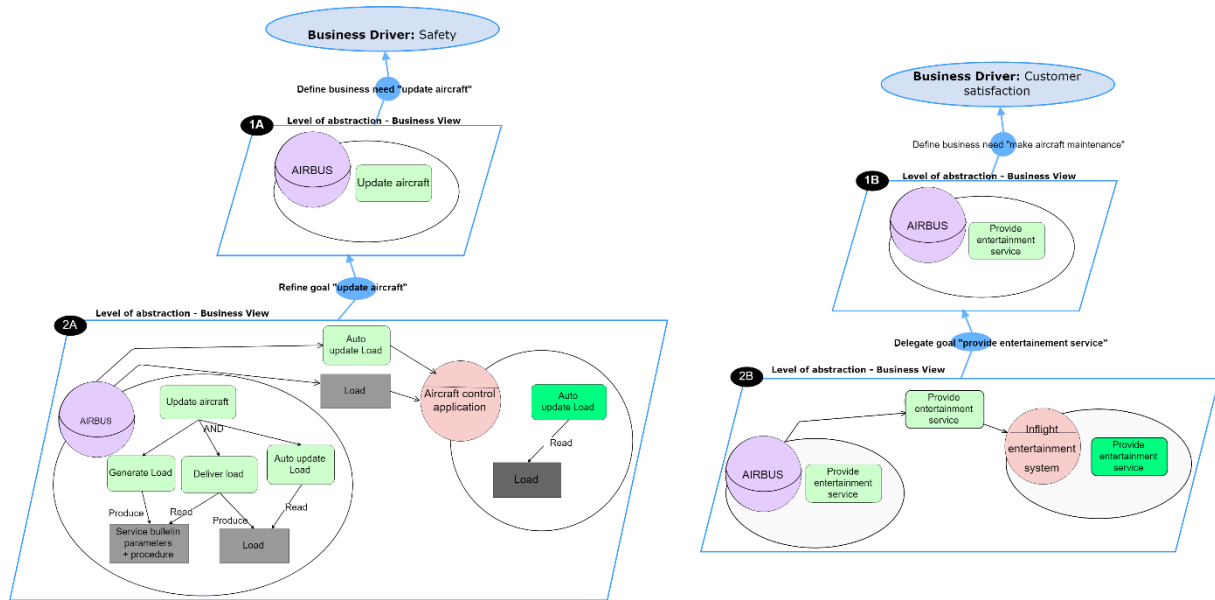


Figure 74: AIRBUS scenario 2 - Business view

Since these two needs are independent to another, we express the two needs in separate layered specifications at business and Architect’s views. Figure 74 and Figure 73 express the respective composite requirements specifications at business and Architect’s views.

For business need1, at Business view, we know that the load information has to be generated by AIRBUS and transmitted to the aircraft. Only after refining the “update aircraft” goal in CR n° 2A at the Architect’s view, we realize that the load information is not directly transmitted by AIRBUS actor to the aircraft. Instead, AIRBUS depends on several other actors and there is a process for generating the load. This information is traced using the refinement link between CR n° 1A and CR n° 2A that expresses delegation of the goal “generate load”.

Similarly, the CR n° 2C describes the detailed process for providing the entertainment service. Figure 89 in Appendix D present the respective security domain information that is expressed in the composite requirements CR n° 4A and CR n° 4B. This entails the security experts to ensure that the elicited security domain information must not conflict with each other. For instance, the aircraft control application is identified as common in both the processes, the receptive integrity level must be same. This allows to verify the consistency of the security domain information.

However, when it comes to the design view, the network security zoning must consider all the components in the scenario. Thanks to the AND refinement link of the composite requirements, that allows us to integrate the security domain information of these two needs and compute a consolidated security zoning solution, see Figure 76.

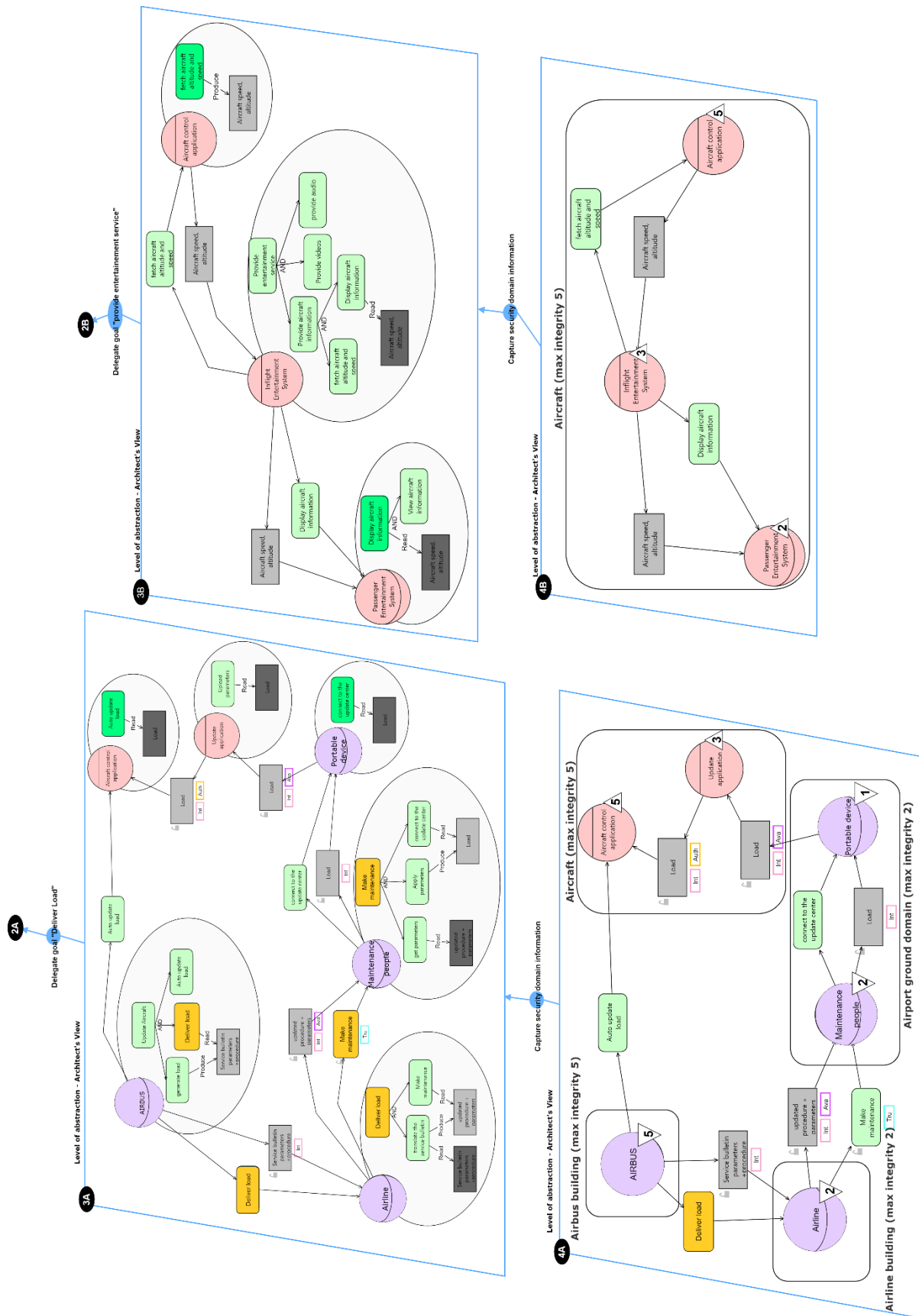


Figure 75: AIRBUS scenario 2 - Architect's view

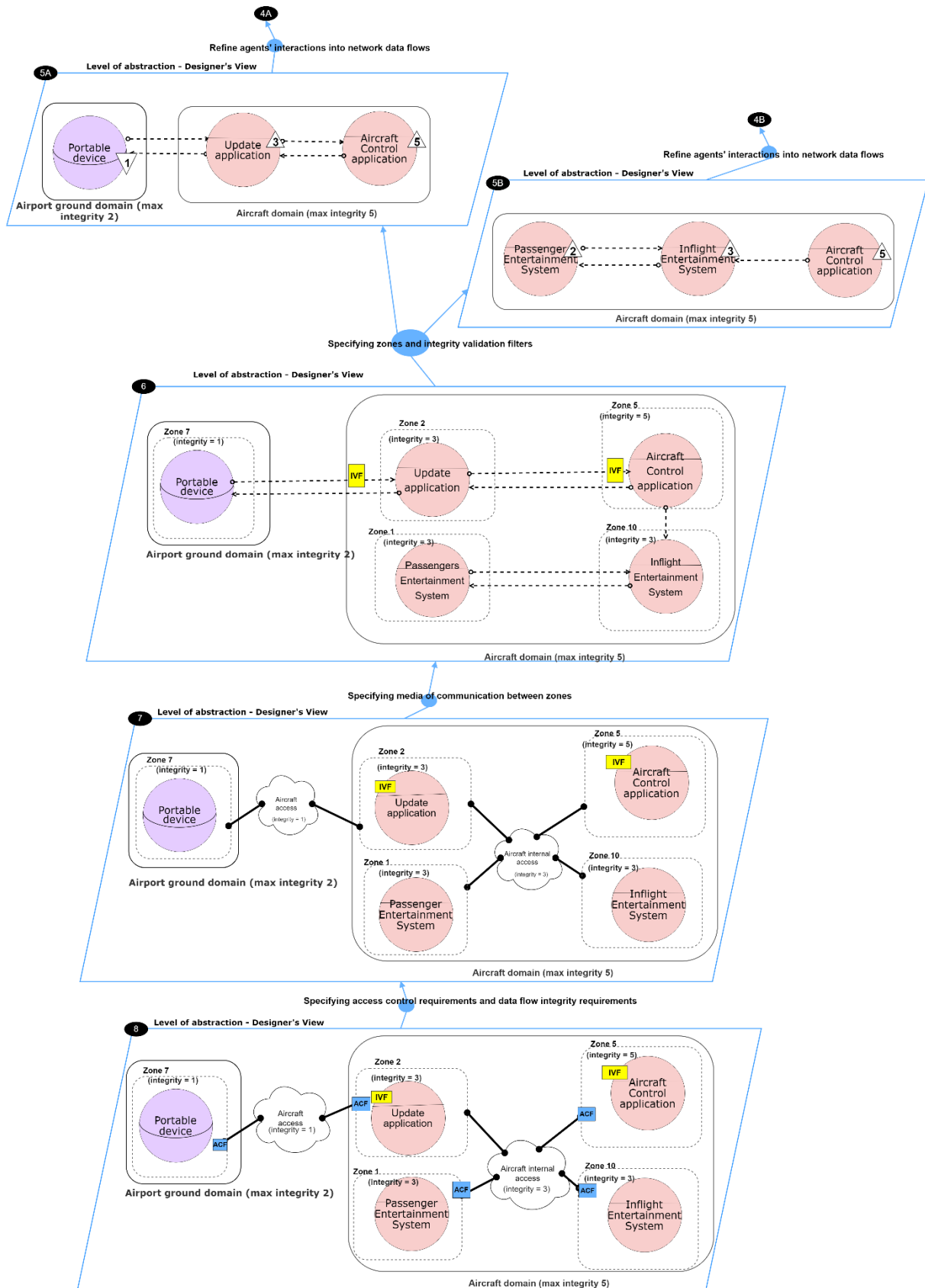


Figure 76: AIRBUS scenario 2 - Design view - zone modelling

4.4 Discussion and Evaluation – SRE methodology

This section is dedicated to present our arguments relative our analysis on the performance of our layer based network SRE methodology with reference to the 10 evaluation criteria derived from the 6 quality criteria. Note that, by saying evaluation criteria we are referring to refined anticipated characteristics of good network SRE methodology (i.e., leaf nodes in figure) from the perspective of security experts involved in IREHOD2 project. They include $R^{M2.1.1}$, $R^{M2.1.2}$, $R^{M2.1.3}$, $R^{M3.1}$, $R^{M3.2}$, $R^{M4.1}$, $R^{M6.1}$, $R^{M6.2}$, $R^{M6.3}$ and $R^{M6.4}$. Wherein, by saying quality criteria we are referring to quality characteristics of good requirements (i.e., rood nodes in figure). We considered the following 6 quality criteria as our initial analysis. They include: R^M1 : adequacy (C10), R^M2 : comprehensibility (C11), R^M3 : abstract (C6), R^M4 : traceability (C5), R^M5 : feasibility (C3) and R^M6 : metadata (C21).

In section 4.4.1, we present our evaluation results of SRE methodology with reference to the evaluation criteria. In the end in section 4.4.2, we present our evaluation comments specific to the zone modelling methodology.

4.4.1 Discussion on the SRE methodology evaluation with reference to the evaluation criteria

In **Table 6**, we resumed the evaluation results of the proposed SRE methodology. Firs column displays the set of evaluation criteria (refer Figure 31 in section). Second column displays respective measurement scales for each of the three SRE methods with respect to the lists the verification methods given in Appendix C. Since we integrate the modelling concepts of STS and Secure KAOS in SRE model, therefore we inherit some characteristic of their characteristic features analysed in chapter 2, section 2.3. Out of listed 10 evaluation criteria, we could not test the some criteria (i.e., $R^{M2.1.3}$, and $R^{M6.4}$). In below, we present our arguments related to our evaluation results.

Table 12: Evaluation results of our proposed SRE methodology

Evaluation criteria (R^M)	Layer based network SRE methodology
$R^{M2.1.1}$: The SRE methodology modelling language terminology must be easily understood to the user	high
$R^{M2.1.2}$: The cost of the SRE methodology training should be minimal	low
$R^{M2.1.3}$: The time taken to learn the SRE methodology should be minimal	--
$R^{M3.1}$: Should support the need for users to easily communicate ideas and feedback respecting the abstraction needs	high
$R^{M3.2}$: Should support the need for users to clearly define the number of abstraction levels of refinement	high
$R^{M4.1}$: The methodology should facilitate to trace the network security requirements back to business objectives	high
$R^{M6.1}$: Must facilitate to specify and link network security zone information	high
$R^{M6.2}$: Should allow to annotate each requirement with risk attributes	high

R^M6.3: Should allow to annotate each requirement with priority information	High
R^M6.4: Must facilitate to specify the implementation costs of network security requirements	--

Our layered based SRE methodology is largely inspired from SABSA layered framework. It consists in three abstraction layers i.e., Business view, Architect's view, and designers view. At design view, our SRE specification express the output of our ASP tool in graphical format. The principle idea to develop any new modelling is the comprehensibility factor. Since our methodology constitutes different layered views involving different stakeholders, this impelled the necessity in choosing rightful SRE methodologies that are suitable in regards with the comprehensibility level of the stakeholders. So that it becomes easier to establish a good communication with the stakeholders while eliciting and evaluating the requirements (**R^M3.2**). Respectively, we use STS notation at Business view and Architect's view, since our analysis (presented in section 2.5) shows that STS concepts are easy to use (**R^M2.1.1**). At design view, we implement ASP tool.

However, since we do not have tool we had to first create the model in STS specification, then we used an online graphical tool (Draw.io) to customize our specification at Architect's view (**R^M2.1.2**). This process consumed some time and also allowed manual errors (e.g., grammatical/typo errors) that I had to work several times for some big models (e.g., CR 4 in Figure 66). It is to note that the complexity of creating the specification is not evenly distributed and varied for different views. From our experience, we observed that the specification at Architect's view consumed us more time comparatively the specifications at remaining two views. However, since we did not run the iteration 2 of our evaluation, we could not test the evaluation criteria concerning the learnability from the perspective of security experts (**R^M2.1.3**).

The next issue we faced while creating the specifications is regarding the information gathering (i.e., elicitation process). By the end of Architect's view, we had to identity all the agents and agent interactions. For AIRBUS scenarios, this job was easier since we had contact with the security experts. However, for e-commerce scenario, all we had was the document from ANSSI website, so we had to imagine the information relating to the interaction needs. This information is important because at design view we derive the permitted information flows from these interactions. However, it is to note that, we do not presume this issue as a shortcoming of our SRE methodology. Instead we consider it as strong characteristic that enforces the involvement of security experts while creating the SRE specification, which eventually creates a platform to have effective communication (**R^M3.1**). This reflects both the forward and backward traceability of the network security needs. This allows the security analysis to verify the refinement of these high-level network security requirement. The integrity levels express the rigor of security protection required.

The utility functions that we introduce at the Architect's view of our SRE methodology, allows to express the risk assessment of the agents, domains in terms of integrity values. This permits to link the risk analysis information to network security requirements (**R^M6.2**). However, it is to note that logic behind the utility functions on how the mapping is made can vary with respect to the risk mangement process within an organization. For example, in order to justify that DNS servers will always have an

integrity level of 3, it is first necessary to elicit what characteristics of DNS server should one look to attribute a certain integrity level. Likewise, risk analysis techniques like attack graphs can be helpful to analyse and measure these characteristics more objectively.

The ASP tool implementing the formal rules of our zone modelling methodology allows us create zone specification based on the security domain information as well as the list of initial integrity levels values computed based on risk impact elicited at Architect's view. In the end, the tool provides us with a set of network security requirements: security zones, integrity validation filters, access control filters, and data flow integrity requirements. Thanks to our composite requirements with refinement links, that facilitates in expressing this network security zoning information in a way that is traceable back to business objectives (**R^M4.1** and **R^M6.1**). In addition, the rationale *attribute* accorded to the refinement links of the composite requirement helps to reason the context of refinement, which we refer to the input and output of the zone modelling steps.

Furthermore, the actual integrity values of the zones help in prioritizing the network security requirements with reference to the rigor of security validation required (**R^M6.3**). For instance, the IVF requirements state that there must be a data flow integrity validation procedure at an agent if the incoming data flows do not satisfy the CW-lite rule (see RULE 8 in section 3.2.3.1). Here, the sanitization function helps to increase the confidence on the information flows.

For example, the IVF filter at Accountability server is specified for the incoming flows from all the employee agents (i.e., local users, remote users and the administrator users). The respective sanitize functions describe the rigor of integrity validation needed to check these data flows to guarantee they conform to constraints of integrity level 5. Since the integrity level of the zones that these agents residing in differ, the rigor of integrity validation checking the incoming data flows to guarantee they conform to constraints of integrity level 5, vary accordingly. Respectively, this will help in selecting the security measures such as a web application firewall that checks for SQL injection/viruses or a VPN connection for remote connections etc. Even though, the elaboration of the sanitization function depends on the domain environment as well as the type of information flows we are dealing with. We displayed the example prioritization of IVF requirements at accountability server in Table 13.

Table 13: Prioritizing IVF requirements at accountability server

IVF requirements at Accountability Server	Difference in the integrity levels of interacting zones	Rigor of security validation anticipated
IVF(accountabilityServer, flow(adminUser, accountabilityServer), sanitize(4, 5))	1	Low
IVF(accountabilityServer, flow(localUsers, accountabilityServer), sanitize(3, 5))	2	Medium
IVF(accountabilityServer, flow(remoteUsers, accountabilityServer), sanitize(2, 5))	3	Very high

However, it is to note that, the qualitative scale for prioritization depends on the scale of integrity levels considered by utility functions. In this thesis, we have considered an example scale of integrity levels (i.e., 1 to 5). The difference in integrity scales will give us the three possible values (i.e., 1: low, 2: medium, 3: high). This scaling permits in prioritizing requirements based on risk impact. That means,

the lower the difference in integrity values means the lower the risk impact regardless of internal/external threats.

Finally, the cost of network security requirements is a cumulative function of the cost of security zoning plus the cost of network security solutions to implement the high-level network security requirements based on the rigor of anticipated security verification. The optimization functions of our ASP tool enables the security architects to verify and control the costs of security zoning with the help of optimization function. In this work, we presented an example optimization rule presented in Figure 58, which computes the optimal solution by minimizing the actual integrity levels of agents and zones. However, for learning the cost of security measures, we need the further refinement the high-level network security requirements, which is not covered in this thesis work. As consequence, we could not test the evaluation criteria **R^M6.4** since we our tool partially addresses the cost requirements.

4.4.2 Discussion on the evaluation of Zone modelling methodology

This ASP tool is developed by keeping the requirement engineers in mind. To them, network security concepts related to security zoning are novel since they do not have enough background to integrate the network security practice to SRE process. On the other hand, the network security practitioners who are well familiar with network security zoning and partitioning will not have enough understanding on the business communication needs and risk impact. Therefore, our tool facilitates the security architects in eliciting high-level network security requirements with reference to the business interaction needs identified at Architect's view.

We have chosen Answer Set Programming (ASP) because it is a lightweight language for knowledge representation and reasoning that has been developed in the field of reasoning and logic programming [Grasso et al. 2013]. The optimization function 'minimize' of ASP solver allows to compute optimal solutions. Note that, in this thesis we do not claim to provide optimal zone solution, for the measurement attributes of cost heuristics may differ from organization to organization. Instead, we intend to help security architects to be able try different attributes for characterizing the cost heuristics. The cost optimization rule (see figure) is an example considered in our experimentation. Therefore, our tool is not subjected to evaluate the novelty or optimality of zone solution obtained.

One way to test the performance of the tool by checking its compilation time. Because, the compilation time is taken to compute the optimized solution using the optimization statement minimize, to find the solution with minimal cost. In our experimentation for the e-commerce scenario, we executed our tool 30 times on a Mac Book Pro (2.8GHz Core i7, 16GB RAM) and configured the solver to use 4 threads in parallel. The execution times to calculate the optimal solution based on our cost optimization rule varied between 5.8 seconds and 7.6 seconds. However, it is to note that, the execution time varies with number of agents, data flows as well as the cost optimization rule (described in 3.2.1.1). In addition, the completion time also depends on the number and scaling range of integrity levels considered in the utility functions. In our experimentation, we have considered an integrity scale range from 1 to 5. Since we confined the utility functions to an assumption, we could not test the accurate performance of the tool.

4.5 Chapter Summary

This chapter principally explores the pros and cons of our SRE methodology with reference to the evaluation criteria (i.e., $R^{M2.1.1}$, $R^{M2.1.2}$, $R^{M2.1.3}$, $R^{M3.1}$, $R^{M3.2}$, $R^{M4.1}$, $R^{M6.1}$, $R^{M6.2}$, $R^{M6.3}$ and $R^{M6.4}$) captured in chapter 2. To validate our observations, we implemented two use case scenarios. One scenario describes the IREHDO2 context. The other scenario describes the SRE context of an e-commerce enterprise network. The results of the evaluation study show that the layer based SRE approach of our SRE methodology is easier to apply to network SRE context.

First, the three views (i.e., Business view/Architect's view/Designer's view) facilities to integrate the security problem context of the stakeholders working in varying layers of abstractions in network development cycle. The design view enables to capture network security requirements through network security zoning. This confirms our evaluation related to the comprehensibility and abstraction needs (i.e., $R^{M2.1.1}$, $R^{M2.1.3}$, $R^{M3.1}$, $R^{M3.2}$ and $R^{M6.1}$). Second, the modelling notation of composite requirements with the refinement links permits to us to trace the network security requirements back to business objectives. This confirms our evaluation related to the traceability aspects (i.e., $R^{M4.1}$).

Finally, our security zone modelling methodology at design view computes the actual integrity levels of zones using this security domain information, which permits to integrate the risk assessment of the domains and agents in the enterprise network. Eventually, the actual integrity levels of the security zones facilitates in prioritizing the security requirements based on the risk impact. This confirms our evaluation related to the traceability as well as the prioritization aspects (i.e., $R^{M4.1}$ and $R^{M6.3}$). However, we could not test some criteria related to the learnability of SRE modelling ($R^{M2.1.2}$) and to the facility to integrate the costs of network security requirements ($R^{M6.4}$).

THESIS CONCLUSION

Thesis Background and Motivation

Considering the pace of the emerging network security threats, protecting enterprise networks has never been more important. Several business-critical activities and operations continue to rely on the functionalities of network technologies, disruption of which may incur severe losses to business or even damage company reputation. Network security principally aims at ensuring legitimate access to the business data assets by preventing intrusion, destruction, modification or misuse. However, to achieve network security, several aspects have to be taken into consideration such as network segmentation (i.e., security zoning); security of network devices connecting the communicating end-user systems; and security of the information being transferred across the communication links[ISO/IEC 27033 2009].

Even though, security architects contemplate the implementation of security solutions (e.g., IDS/IPS, VPN, IP Firewall, and Web Application Firewalls, DMZs) adapting them to the network architectures can be challenging. It requires a comprehensive understanding of the business environment and its operational context before determining adequate security practices pertinent to the business objectives. Otherwise, an inadequate network security design can lead to data loss in spite of the monitored traffic, and security incidents handling. Besides, any lateral changes to the network security design (e.g., adding firewalls/proxies), post deployment, can turn to be a real overhead in terms of time, effort, and costs. In order to evade these aforementioned problems, it is essential to consider network security at earlier stages i.e. right from the gathering of security requirements for architecting networks, which otherwise refers to the ‘security by design’ principle.

In this regard, the Security Requirements Engineering (SRE) process has been gaining significant attention from both academic research and industrial sectors since it helps to capture and document security requirements by analysing the business risk impact of potential threats on business objectives. A large variety of SRE approaches (e.g., such as agent-oriented, goal-oriented and problem frames oriented) propose varying perspectives of formal modelling in order to enhance the quality and experience of the SRE process. However, they render no support to capture security requirements from the network security perspective. While one reason is being their emphasis confined to system/software security alone, the other reason is being their lack of support to consider security problem context of stakeholders (e.g., business users, security architects, network security designers, developers) involved at different layers of abstraction in the network security development life cycle. Therefore, a methodology for network security is promptly desired. This shortcoming has coerced us to probe for an alternative solution (i.e., a SRE methodology) to foster the implementation of the “security by design” principle from network security perspective.

Thesis Project and Research Questions

This thesis work is realized as a part of IREHDO2 project that concerns future generation aircraft networks. Our work is done in collaboration with the security experts at AIRBUS who are interested in enhancing their SRE practices in order to increase the assurance on the final security solution enforced on their aircraft networks. Our main objective in this project is to propose an SRE methodology that allows us to capture, express and analyse the network security requirements, and that facilitates the refinement into network security and monitoring configurations (TOP/DOWN approach).

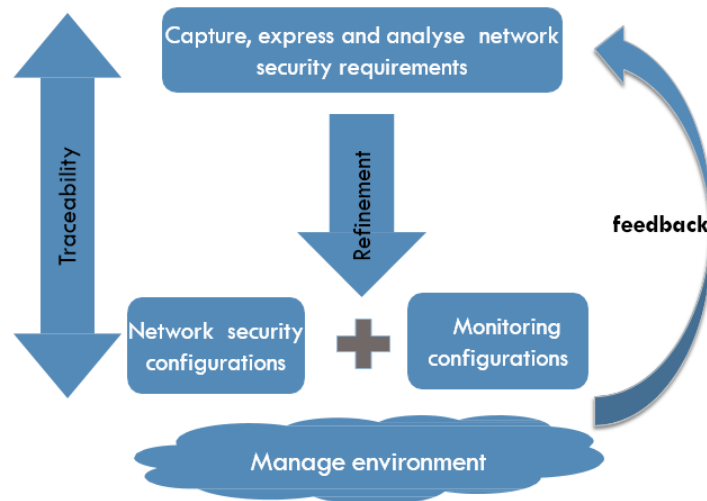


Figure 77: Our goal in IREHDO2 project

Security Requirements engineering (SRE) is a complex process. On one hand, the security experts (e.g., security requirement engineers, business analysts, security architects) have to discover the business assets, associated security needs and constraints (e.g., costs, regulatory constraints). How to be sure if security all security needs of all the stakeholders are identified? On the other hand the other group of security experts (e.g., security risk analysis, security design engineers) have to ensure that the all security threats, are addressed with some counter security measures. How to be sure if some threat is not forgotten? How to be sure if the defined security measures are important and rightly address the threats? A SRE methodology is considered as a tool that facilitates the security experts in eliciting network security requirements while considering the aforementioned aspects in a given SRE context.

Therefore, this thesis work fosters the review of some important aspects related to the characteristics of good SRE methodology that ensure the effectiveness of SRE process in network context. To this end, our principle research questions of the thesis are defined as:

RQ1: How to evaluate SRE methodologies in a network SRE context?

RQ2: How to define a good SRE methodology that facilitates the network security requirements elicitation and refinement i.e., from high-level business objectives into low-level network security zoning requirements?

Considering the **RQ1** study, the detailed review of the formal SRE modelling aspects of widely acknowledged SRE approaches (i.e., goal-oriented, agent-oriented and problem frames based) has

allowed us to explore varying philosophies of eliciting security requirements. In brief, goal-oriented approaches (e.g., Secure KAOS) emphasize on capturing security requirements by linking to goals to intentional obstacles (a.k.a. anti-goals) hierarchy resembling an attack trees; agent-oriented SRE approaches (e.g., STS) emphasize on capturing security requirements over the interaction dependencies of the agents; finally, problem frames based approaches (e.g., SEPP) emphasize on capturing security requirements by characterizing the constrained behaviour of system-to-be components as security problems. Furthermore, we were able to study in detail the varying characteristics of these approaches, their pros and cons with reference to the anticipated characteristic features of network SRE methodology (such as comprehensibility of modelling concepts, traceability of security requirements, etc.).

Considering the **RQ2** study, we have investigated the network security zoning principles that help security architects in eliciting and analysing early network security requirements. Furthermore, the exploring of SABSA, an enterprise architecture driven security requirements framework, has provided us insights on the abstraction layers and the abstraction needs of different stakeholders involved within the development cycle of enterprise networks.

Synthesis of contributions and Future perspectives

While the challenges inherent to our research topic are numerous, we succeeded to consolidate several independent areas of research works that together contribute to the successful implementation of SRE process. As a result, the literature study of this dissertation presented in **CHAPTER 1** consists in the background theory and concepts of security risk management methods and risk analysis (RA) techniques; SRE process and the three SRE methods studied in this thesis (i.e., STS, Secure KAOS and SEPP); Network security zoning; and SABSA framework. While, **CHAPTER 2** consolidates our premier contributions relative to RQ1 that facilitate the specification of good characteristics of SRE methodology in network SRE context; **CHAPTER 3** presents our core contributions of this work related to RQ2 that concern our proposed SRE methodology.

In below, we enlist the claimed contributions and limitations realized during this dissertation.

A RE based SRE Evaluation methodology

Problem concerned: Having observed the varying perspectives of the three renowned SRE approaches (i.e., goal-oriented, agent-oriented and problem frames oriented), we had to decide on which one of these approaches suites best to the network security context of IREHDO2. Existing comparative studies did not help us in this task since their evaluation criteria is ad hoc and does not consider the working context of the security requirement engineers.

Proposed solution: To address this issue, we have proposed a generic evaluation methodology built on the classical idea of requirements engineering. Our proposed methodology differentiates its strategy from previous comparative studies for two reasons. First, we involve the security experts that are the SRE end-users in the whole process. Secondly, identifying SRE evaluation criteria corresponds to identifying the characteristics of a good SRE methodology. The refinement of evaluation criteria was

done using goal modelling notation (KAOS). The root goal nodes correspond to the quality characteristics of good requirements. The leaf nodes express the evaluation criteria, each defined with objective verification methods.

Using this methodology, we have derived 10 evaluation criteria in the network SRE context of IREHDO2 project. These evaluation criteria are refined with reference to the six quality criteria of good security requirements (i.e., adequacy (C10), comprehensibility (C11), traceability (C5), abstract (C6), feasibility (C3) and metadata (C21)). We have employed the criteria to evaluate three SRE methodologies (KAOS, STS and SEPP), the results of which shows that none of them suites to capture network security requirements (e.g., security zoning).

Limitations: The main limitation of this evaluation methodology is that its implementation requires the participation of security experts. For instance, iteration 2 requires that the security experts master the concepts of SRE methodology under evaluation, which raises concerns in terms of time and training costs. In practice, once a SRE methodology is chosen, a lot of time and money is put to train the users and it is very unlikely that one would switch to new methodology soon. Therefore, from the industrial usage perspective choosing the best suitable SRE methodology at earlier stages reduces overhead and saves time. Our evaluation methodology helps in this by enabling its users to think like a requirements architect who plans, designs and reviews the derivation of security requirements well before selecting an SRE methodology.

Future perspectives: The 10 evaluation criteria derived with reference to the six quality criteria of good security requirements. However, as per our weaving methodology there 15 more quality criteria that need to be considered. Therefore, currently we are working on capturing the anticipated characteristics of SRE methodology with reference to all the 20 quality criteria. The respective verification methods permit us to synthesize and document the meta-information/ measurement metrics that assist in addressing the validity threats of evaluation process.

On the other hand, we would like to apply our evaluation approach to other requirements engineering contexts (e.g., cloud computing). The objective to be able to determine which of the evaluation criteria are generic and which are specific to security context. In the long term, we intend to build a common repository to maintain the evaluations carried out in each scenario context so that they can used as a reference for any similar evaluations. Furthermore, the verification methods and the measurement metrics can be reused on other contexts or at least can provide a consolidated evaluation prepositive of other security experts. 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.

Layer-based SRE methodology for network security

Problem: The evaluation experiment confirms that currently a SRE methodology for network security is missing. From the derived evaluation criteria, we deduce that the anticipated SRE methodology for network security must facilitate to express the abstraction needs of different stakeholders involved in the network development life cycle.

Solution: To address this requirement, we proposed a layer-based SRE methodology inspired from SABSA framework. Our methodology consists of three layered views i.e., Business view, architect's view and Designer's view. The Business view concerns the elicitation of high-level business needs. The architect's view concerns the elicitation of security needs over the interaction needs of the actors; the elicitation of security domain information (i.e., criticality/trust levels of agents and control capabilities of security domains) through integrity utility functions, which permits to integrate risk analysis process. Finally, the Designer's view concerns the elicitation of high-level network security requirements through security zoning. We use express business and architect's view in STS modelling, while at Designer's view we proposed a security zone modelling methodology. To facilitate the linking of all these three abstract views, we proposed a new SRE modelling notation called composite requirements that merges STS and KAOS modelling concepts.

Limitations: The first limitation of our SRE methodology is it lacks a tool support. As consequence, it consumes significant time in creating the SRE specification. Second limitation concerns our critical assumption relating to the utility functions. We presume that they help to integrate risk assessment information through mapping the trust/criticality levels of agents as well as the control capability of domains to the unified integrity levels. Indeed, the integrity levels plays vital role the specification of security zones. To explain the logic behind this mapping, it would require a rigorous research on how to analyse the trust / criticality levels based on risk impact similar to works.

Future perspectives: For future works, we intend to work towards developing a graphical tool for representing the modelling notation of our layer-based SRE methodology. This will facilitate us to evaluate our methodology for more complex scenarios of network SRE context. In addition, the tool can be used to evaluate the learnability factor of our proposed methodology. In the long term, we would like to extend our study on developing the functional logic behind the utility functions. For this, we need to look for the Meta information that may help us in characterizing the mapping of integrity scales to the trust/criticality and control capability levels.

Anti-STS, a multi-agent risk modelling technique

Problem concerned: We argue that, an SRE methodology must facilitate the risk analysis that corresponds to both asset as well as threat-based risk analysis, which will eventually allow us to analyse the likelihood and impact of security risks pertaining to critical assets in the enterprise network. Security risk analysis consists of two types: asset-based and threat-based risk analysis. The utility functions expressing the integrity levels of security domains and agents correspond to asset-based risk analysis concepts. Therefore, we needed also to integrate a based risk analysis process. Our evaluation study shows that STS notation does not provide adequate support to integrate threat-based risk analysis concepts. Furthermore, existing attack modelling techniques do not help in expressing complex attacks like APTs (advanced persistent threats) in which multiple attacking agents act collaboratively to infiltrate enterprise networks.

Solution: To address this issue, we proposed a threat profile oriented risk-modelling technique that integrates the social modelling concepts from STS modelling and the Anti-goal modelling concepts from Secure KAOS. We name it as Anti-STS. We argue that attackers are not lonely agents and they too have

to rely on some malicious users or malicious programs in order to launch an attack. Our Anti-STS Model permits to analyse the social and technical dependencies between the threat actors (i.e., individual or group, system element or software program, etc.) will help in anticipating the factors of likelihood and threat capabilities.

Limitations: Since our principal objective of this thesis is to propose SRE methodology for network security, we had allotted limited time to develop our proposed Anti-STS Model into a full-fledged risk modelling technique. We retain this task as our future perspectives.

Future perspectives: Although, we introduced Anti-STS modelling to be considered as a threat-based risk-modelling technique, we have identified yet another use of Anti-STS, which is to be used as a threat intelligence tool to represent existing APT attacks. The threat intelligence reports on APT attacks are generally shared as a list of documents that are written in natural language, like *APT notes* on github¹. Our perspective is to reuse the modelling concepts of Anti-STS Technique to express explicitly the multiple attacking agents and their malicious behaviour in a graphical model. This will help us in towards developing a common platform to formally represent the threat intelligence on notable APT like attacks. As an initiative, we have already tried our hands on modelling using *carbanack* attack in [Bulusu et al. 2017b].

In future, we intend to work in developing a tool to represent the APT attacks. We had prepared a proposal to Master students to help us in developing this tool. The tool must facilitate to express the composite attack scenario of an APT in terms attacking agents and their interaction dependencies. This will enable us to study and analyse the social and technical dependencies between the attackers and the technical systems used during the execution of the APT attack. Eventually one can deduce the list of events that can enable to detect the malicious behaviour of the APT attack. Some example events can include the creation of malicious configuration files in the library or untimely execution of legitimate library files etc. In the end, such events can be transformed into formal rules which can be fed directly to the security monitoring systems such as intrusion detection systems (IDS) in order to monitor the real traffic against such malicious behaviour. The results of this work can also aid in situation based security management frameworks such as [Laborde et al. 2018].

In long term, we want to use this tool to compare and distinguish varying attack patterns employed in APT campaigns. This aspect corresponds to the Req8 given in Section II. For example, the regular attack patterns and social engineering styles that are frequently employed or different malicious capabilities of malware programs or different skillsets that the attackers exhibited in order to infiltrate the target systems. Such information corresponds to learning the threat profile. This will enable us to model and compare the threat profiles of multiple APT groups.

Logic-based zone modelling methodology

Problem concerned: Network zoning (a.k.a. network segmentation) is a key defence-in-depth strategy that segregates and protects key company assets and limits lateral movements of attackers across corporate network in case of intrusion. Therefore, determining security zones and respective trust levels

¹ <https://github.com/kbandla/APTnotes>

is a preliminary step for security architects to derive other network security requirements such physical access, dataflow control and protection, and end-systems protection [Sedgewick 2014; Secure Arc 2009]. Nonetheless, in the literature, there has been limited focus in this regard and no rigorous approach formally support this process.

Solution: To address this issue, we proposed a zone modelling methodology based on three security principles: complete mediation, least privileges and Clark-Wilson lite formal model. We defined a set of formal rules as well as the list of initial integrity levels values computed based on risk impact, which makes our methodology approach traceable and verifiable. The whole process has been implemented in an ASP tool to automate the security zones computation. It produces a set of network security requirements: security zones, integrity validation filters, access control filters, and data flow integrity requirements. We illustrated the use of this methodology using an e-commerce use case scenario. The tool also facilitates calculating cost-optimal security zone models.

Limitations: The current methodology does not support in the further refinement of high-level network security requirements. For instance, refinement of Integrity validation filtering requirements (IVF) need to be further refined in order to be able to identify security measures such as a web application firewall that checks for SQL injection/viruses or a VPN connection for remote connections etc. Another limitation of this work is it does not automate the prioritization of elicited network security requirements.

It is to note that, our ASP tool deliberately do not guarantee optimal zone solution since as we deliberately focused on enabling security experts to customize the attributes that allow to compute optimal zone solution. Therefore, we do not consider it as a limitation.

Future perspectives: As future works, we want to first investigate the refinement of the high-level network security requirements produced by our current work. As an example, IVF attached to agents may require to be refined due to design constraints. For instance, it might be impossible to enforce the IVF on the *accountabilityServer* (from the e-commerce case study) for technical constraints. In this case, the initial security requirements needs to be refined by introducing new security agents (e.g., network security proxies) to achieve the IVF, similar to the final zone model in the e-commerce case study.

In parallel, we would like to extend our security zone modelling approach to consider the confidentiality and availability requirements. Access control filters, defined by our methodology, partially address confidentiality requirements only. We intend to explicitly integrate formal confidentiality models. In long term, we intend to consider the security zoning aspects of dynamic context, in which the agents, their integrity levels, and attack scenarios can change easily from one situation to another, increasing the challenge of creating new zones.

BIBLIOGRAPHY

- AHMAD, S. 2012. *Measuring the Effectiveness of Negotiation in Software Requirements Engineering*. University of Western Australia.
- AMYOT, D., HORKOFF, J., GROSS, D., AND MUSSBACHER, G. 2009. A lightweight GRL profile for i* modeling. *International Conference on Conceptual Modeling*, Springer, 254–264.
- ANDERSON, R.J. 2010. *Security Engineering: A Guide to Building Dependable Distributed Systems*. John Wiley & Sons.
- ANSSI. 2010. EBIOS — Expression des Besoins et Identification des Objectifs de Sécurité | Agence nationale de la sécurité des systèmes d’information. <http://www.ssi.gouv.fr/guide/ebios-2010-expression-des-besoins-et-identification-des-objectifs-de-securite/>.
- ANSSI. 2017. Sensibilisation et initiation à la cybersécurité. https://www.ssi.gouv.fr/uploads/2016/05/cyberedu_module_4_cybersecurite_organisation_02_2017.pdf.
- ANTON, A.I. 1996. Goal-based requirements analysis. *Requirements Engineering, 1996., Proceedings of the Second International Conference on*, IEEE, 136–144.
- ARNOLD, F., HERMANN, H., PULUNGAN, R., AND STOELINGA, M. 2014. Time-dependent analysis of attacks. *International Conference on Principles of Security and Trust*, Springer, 285–305.
- BIBA. 1977. *Integrity considerations for secure computer systems*. DTIC.
- BIEBER ET AL. 2011. DALculus—theory and tool for development assurance level allocation. *International Conference on Computer Safety, Reliability, and Security*, Springer.
- BRESCIANI, P., PERINI, A., GIORGINI, P., GIUNCHIGLIA, F., AND MYLOPOULOS, J. 2004. Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems* 8, 3, 203–236.
- BSI. 2012. *Threats Catalogue – Elementary Threats*. .
- BULDAS, A. AND LENIN, A. 2013. New efficient utility upper bounds for the fully adaptive model of attack trees. *International Conference on Decision and Game Theory for Security*, Springer, 192–205.
- BULUSU, S.T., LABORDE, R., WAZAN, A.S., BARRÈRE, F., AND BENZEKRI, A. 2016. Towards the weaving of the characteristics of good security requirements. *International Conference on Risks and Security of Internet and Systems*, Springer, 60–74.
- BULUSU, S.T., LABORDE, R., WAZAN, A.S., BARRÈRE, F., AND BENZEKRI, A. 2017a. Which Security Requirements Engineering Methodology Should I Choose?: Towards a Requirements Engineering-based Evaluation Approach. *Proceedings of the 12th International Conference on Availability, Reliability and Security*, ACM, 29.

- BULUSU, S.T., LABORDE, R., WAZAN, A.S., BARRÈRE, F., AND BENZEKRI, A. 2017b. Describing advanced persistent threats using a multi-agent system approach. *Cyber Security in Networking Conference (CSNet), 2017 1st*, IEEE, 1–3.
- BULUSU, S.T., LABORDE, R., WAZAN, A.S., BARRÈRE, F., AND BENZEKRI, A. 2018a. A Requirements Engineering-Based Approach for Evaluating Security Requirements Engineering Methodologies. In: *Information Technology-New Generations*. Springer, 517–525.
- BULUSU, S.T., LABORDE, R., WAZAN, A.S., BARRÈRE, F., AND BENZEKRI, A. 2018b. Applying a Requirement Engineering Based Approach to Evaluate the Security Requirements Engineering Methodologies. *ACM SAC RE 2018*.
- BULUSU, S.T., LABORDE, R., WAZAN, A.S., BARRÈRE, F., AND BENZEKRI, A. 2018c. A Logic-Based Network Security Zone Modelling Methodology. *SECUREWARE2018*.
- CCTA, C. 2001. Risk analysis and management method, CRAMM user guide, Issue 2.0. *United Kingdom Central Computer and Telecommunication Agency (CCTA)*.
- CHEN, P., DESMET, L., AND HUYGENS, C. 2014. A study on advanced persistent threats. *IFIP International Conference on Communications and Multimedia Security*, Springer, 63–72.
- CHRISTIAN, T. 2010. *Security requirements reusability and the SQUARE methodology*. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.
- CHUNG, L., NIXON, B.A., YU, E., AND MYLOPOULOS, J. 2012. *Non-Functional Requirements in Software Engineering*. Springer Science & Business Media.
- CLARK, D.D. AND WILSON, D.R. 1987. A comparison of commercial and military computer security policies. *Security and Privacy, IEEE Symposium on*, IEEE.
- CLEGHORN, L. 2013. Network defense methodology: A comparison of defense in depth and defense in breadth. *Journal of Information Security* 4, 03, 144.
- CRISTIANO CASTELFRANCHI, Y.-H.T. 2002. The role of trust and deception in virtual societies. *International Journal of Electronic Commerce* 6, 3, 55–70.
- CYBEDU. 2017. Sensibilisation et initiation à la cybersécurité-consortium. https://www.ssi.gouv.fr/uploads/2016/05/cyberedu_module_4_cybersecurite_organisation_02_2017.pdf.
- DARDENNE, A., VAN LAMSWEERDE, A., AND FICKAS, S. 1993. Goal-directed requirements acquisition. *Science of computer programming* 20, 1, 3–50.
- DAVID LYNAS. SABSAC Foundation courses training - David Lynas Consulting Limited. *SABSACourses*. <https://www.sabsacourses.com/course-schedule/>.
- DONALD FIRESMITH. 2007. Common Requirements Problems: Journal of Object Technology by Donald Firesmith.pdf. http://www.jot.fm/issues/issue_2007_01/column2.pdf.
- DUBOIS, É., HEYMANS, P., MAYER, N., AND MATULEVIČIUS, R. 2010. A systematic approach to define the domain of information system security risk management. In: *Intentional Perspectives on Information Systems Engineering*. Springer, 289–306.
- ELAHI, G. AND YU, E. 2007. A goal oriented approach for modeling and analyzing security trade-offs. *International Conference on Conceptual Modeling*, Springer, 375–390.

- EUROCONTROL. 2012. Specification for Data Assurance Levels. <http://www.eurocontrol.int/sites/default/files/publication/files/20120315-adq-dal-spec-v1.0.pdf>.
- EUROPEAN COMMISSION. 2006. COMMUNICATION FROM THE COMMISSION - a European Programme for Critical Infrastructure Protection. <https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX:52006DC0786>.
- FABIAN, B., GÜRSER, S., HEISEL, M., SANTEN, T., AND SCHMIDT, H. 2010. A comparison of security requirements engineering methods. *Requirements engineering* 15, 1, 7–40.
- FIRESMITH, D. 2003. Specifying good requirements. *Journal of Object Technology* 2, 4, 77–87.
- FIRESMITH, D. 2007. Common Requirements Problems, Their Negative Consequences, and the Industry Best Practices to Help Solve Them. *Journal of Object Technology* 6, 1, 17–33.
- GEBSER, M., KAMINSKI, R., KAUFMANN, B., AND SCHAUB, T. 2012. Answer set solving in practice. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 6, 3, 1–238.
- GEBSER, M., KAMINSKI, R., KAUFMANN, B., AND SCHAUB, T. 2014. Clingo= ASP+ control: Preliminary report. *arXiv preprint arXiv:1405.3694*.
- GEPHI.ORG. Gephi 0.9.2 - The Open Graph Viz Platform. <https://gephi.org/>.
- GONTARCZYK, A., MCMILLAN, P., AND PAVLOVSKI, C. 2015. Cyber Security Zone Modeling in Practice. *Proceedings of the 10th International Conference on Information Technology and Applications (ICITA)*, Sydney, Australia.
- GOVERNMENT OF CANADA, COMMUNICATIONS SECURITY ESTABLISHMENT. 2007. Baseline Security Requirements for Network Security Zones in the Government of Canada. <https://www.cse-cst.gc.ca/en/node/268/html/15236>.
- GRASSO, G., LEONE, N., AND RICCA, F. 2013. Answer set programming: language, applications and development tools. *International Conference on Web Reasoning and Rule Systems*, Springer, 19–34.
- HATEBUR, D., HEISEL, M., AND SCHMIDT, H. 2007a. A pattern system for security requirements engineering. *ARES 2007. the second international conference.*, IEEE.
- HATEBUR, D., HEISEL, M., AND SCHMIDT, H. 2007b. A pattern system for security requirements engineering. *Availability, Reliability and Security, 2007. ARES 2007. The Second International Conference on*, IEEE, 356–365.
- HATEBUR, D., HEISEL, M., AND SCHMIDT, H. 2008. A formal metamodel for problem frames. *International Conference on Model Driven Engineering Languages and Systems*, Springer, 68–82.
- HOGGANVIK, I. 2007. A Graphical Approach to Security Risk Analysis. .
- HOO, K.S., SUDBURY, A., AND JAQUITH, A. 2001. *Tangible ROI Through Secure Software Engineering*. .
- HØYLAND, A. AND RAUSAND, M. 2009. *System reliability theory: models and statistical methods*. John Wiley & Sons.

- HULL, E., JACKSON, K., AND DICK, J. 2010. *Requirements Engineering*. Springer Science & Business Media.
- ICS-CERT. 2016. Recommended Practice: Improving Industrial Control System Cybersecurity with Defense-in-Depth Strategies. https://ics-cert.us-cert.gov/sites/default/files/recommended_practices/NCCIC_ICS-CERT_Defense_in_Depth_2016_S508C.pdf.
- IEC 61025. 2006. Fault tree analysis (FTA). <https://webstore.iec.ch/publication/4311>.
- IEC 61165. 2006. Application of Markov techniques. <https://webstore.iec.ch/publication/4721>.
- IEC 61508. 2010. Functional safety of electrical/electronic safety-related systems - Part 1: General requirements. <https://webstore.iec.ch/publication/5515>.
- IEC61882. 2016. Hazard and operability studies (HAZOP studies) - Application guide. <https://webstore.iec.ch/publication/24321>.
- IEEE 830. 1998. IEEE 830-1998 - IEEE Recommended Practice for Software Requirements Specifications. <https://standards.ieee.org/findstds/standard/830-1998.html>.
- ISO 13335. 2004. ISO/IEC 13335-1:2004 - Information technology -- Security techniques -- Management of information and communications technology security -- Part 1: Concepts and models for information and communications technology security management. http://www.iso.org/iso/catalogue_detail.htm?csnumber=39066.
- ISO29148:2011. ISO/IEC/IEEE 29148:2011 Systems and software engineering – Life cycle processes – Requirements engineering. .
- ISO31010, I. 2009. ISO/IEC 31010:2009 - Risk management – Risk assessment techniques. .
- ISO/IEC 27033. 2009. IT network security standard, retrieved from <http://www.iso27001security.com/html/27033.html>. .
- ISO/IEC27000. 2018. ISO/IEC 27000:2018 Information technology — Security techniques — Information security management systems - Overview and vocabulary (5th edition). .
- JACKSON, M. 2001. *Problem Frames: Analysing and Structuring Software Development Problems*. Addison-Wesley, New York.
- JGRAPH LTD. Draw.io, an open platform to create and share diagrams. <https://about.draw.io/>.
- JONKER, C.M. AND TREUR, J. 1999. Formal analysis of models for the dynamics of trust based on experiences. *European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, Springer, 221–231.
- JÜRJENS, J. 2002. UMLsec extending UML for secure development systems. 412–425.
- KAR, P. AND BAILEY, M. 1996. Requirements Management Working Group: Characteristics of Good Requirements. *INCOSE International Symposium*, Wiley Online Library, 1225–1233.
- KARPATI, P., SINDRE, G., AND OPDAHL, A.L. 2011. Characterising and analysing security requirements modelling initiatives. *Availability, Reliability and Security (ARES), 2011 Sixth International Conference on*, IEEE, 710–715.

- KHWAJA, A.A. AND URBAN, J.E. 2002. A synthesis of evaluation criteria for software specifications and specification techniques. *International Journal of Software Engineering and Knowledge Engineering* 12, 05, 581–599.
- KIM, A. AND KANG, M.H. 2011. *Determining asset criticality for cyber defense*. NAVAL RESEARCH LAB WASHINGTON DC.
- KLETZ, T.A. 1999. *HAZOP and HAZAN: identifying and assessing process industry hazards*. IChemE.
- KORDY, B., MAUW, S., RADOMIROVIĆ, S., AND SCHWEITZER, P. 2011. Foundations of attack–defense trees. In: *Formal Aspects of Security and Trust*. Springer, 80–95.
- KORDY, B., PIÈTRE-CAMBACÉDÈS, L., AND SCHWEITZER, P. 2014. DAG-based attack and defense modeling: Don't miss the forest for the attack trees. *Computer science review* 13, 1–38.
- KOTONYA, G. AND SOMMERVILLE, I. 1996. Requirements engineering with viewpoints. *Software Engineering Journal* 11, 1, 5–18.
- LABORDE, R., BULUSU, S.T., WAZAN, A.S., BARRÈRE, F., AND BENZEKRI, A. 2019. Logic-based methodology to help security architects in eliciting high-level network security requirements. *SAC 2019*.
- LABORDE, R., OGLAZA, A., WAZAN, A.S., BARRÈRE, F., AND BENZEKRI, A. 2018. A situation-driven framework for dynamic security management. *Annals of Telecommunications*, 1–12.
- VAN LAMSWEERDE, A. 2004. Elaborating security requirements by construction of intentional anti-models. *26th International Conference on Software Engineering, 2004. ICSE 2004. Proceedings*, 148–157.
- LIN, L., NUSEIBEH, B., INCE, D., AND JACKSON, M. 2004. Using abuse frames to bound the scope of security problems. *Requirements Engineering Conference, 2004. Proceedings. 12th IEEE International*, 354–355.
- LIN, L., NUSEIBEH, B., INCE, D., JACKSON, M., AND MOFFETT, J. 2003. Introducing abuse frames for analysing security requirements. *Requirements Engineering Conference, 2003. Proceedings. 11th IEEE International*, 371–372.
- LIU, L AND YU, E. GRL - Goal-oriented Requirement Language. <http://www.cs.toronto.edu/km/GRL/>.
- LODDERSTEDT, T., BASIN, D., AND DOSER, J. 2002. SecureUML: A UML-based modeling language for model-driven security. In: *<<UML>> 2002—The Unified Modeling Language*. 426–441.
- LUND, M.S., SOLHAUG, B., AND STØLEN, K. 2010. *Model-Driven Risk Analysis: The CORAS Approach*. Springer Science & Business Media.
- MAR, B.W. 1994. Requirements for development of software requirements. *INCOSE International Symposium*, Wiley Online Library, 34–39.
- MAREK, P. AND PAULINA, J. 2006. The OCTAVE methodology as a risk analysis tool for business resources. *International Multiconference Computer Science and IT, Hong Kong*.
- MARIUSZ STAWOWSKI. 2009. Network Security Architecture - ISSA Journal. .
- MASSACCI, F., MYLOPOULOS, J., AND ZANNONE, N. 2010. Security requirements engineering: the SI* modeling language and the secure tropos methodology. In: *Advances in Intelligent Information Systems*. Springer, 147–174.

- MATULEVICIUS ET AL. 2012. Syntactic and semantic extensions to secure tropos to support security risk management. .
- MAYER, N. 2009. Model-based management of information system security risk. <https://tel.archives-ouvertes.fr/tel-00402996/>.
- MCDERMOTT, J. AND FOX, C. 1999. Using abuse case models for security requirements analysis. *Computer Security Applications Conference, 1999.(ACSAC'99) Proceedings. 15th Annual*, IEEE, 55–64.
- MEAD, N.R. 2007. *How to compare the Security Quality Requirements Engineering (SQUARE) method with other methods*. DTIC Document.
- MEAD, N.R. AND STEHNEY, T. 2005. *Security quality requirements engineering (SQUARE) methodology*. ACM.
- MELLADO, D., BLANCO, C., SÁNCHEZ, L.E., AND FERNÁNDEZ-MEDINA, E. 2010. A systematic review of security requirements engineering. *Computer Standards & Interfaces* 32, 4, 153–165.
- MELLADO, D., FERNÁNDEZ-MEDINA, E., AND PIATTINI, M. 2007. A common criteria based security requirements engineering process for the development of secure information systems. *Computer Standards & Interfaces* 29, 2, 244–253.
- MICROSOFT. 2016. *The STRIDE Threat Model*. .
- MICROSOFT. Windows Vista integrity mechanism and earlier integrity models. <https://msdn.microsoft.com/fr-FR/library/bb625957.aspx>.
- MOURATIDIS, H. AND GIORGINI, P. 2007. Secure tropos: a security-oriented extension of the tropos methodology. *International Journal of Software Engineering and Knowledge Engineering* 17, 02, 285–309.
- MUÑANTE, D., CHIPRIANOV, V., GALLON, L., AND ANIORTÉ, P. 2014. A review of security requirements engineering methods with respect to risk analysis and model-driven engineering. *International Conference on Availability, Reliability, and Security*, Springer, 79–93.
- NETWORKX DEVELOPERS. 2016. NetworkX 2.1 Python package. <https://networkx.github.io/documentation/stable/#>.
- NHLABATSI, A., NUSEIBEH, B., AND YU, Y. 2009. Security requirements engineering for evolving software systems: A survey. .
- NIELSEN, D.S. 1971. *The Cause/Consequence Diagram Method as a Basis for Quantitative Accident Analysis*. Danish Atomic Energy Commission, Risoe. Research Establishment.
- NIST. 2013. NIST (SP) 800-53 Revision 4_ Security and Privacy Controls for Federal Information Systems and Organizations retrieved from <https://doi.org/10.6028/NIST.SP.800-53r4>. <https://doi.org/10.6028/NIST.SP.800-53r4>.
- NIST. 2014. Framework for Improving Critical Infrastructure Cybersecurity- National Institute of Standards and Technology. .
- NIST 800–160. 2014. NIST 800–160 - System Security Engineering: An Integrated Approach to Building Trustworthy Resilient Systems. National Institute of Standards and Technology. .

- ONGSAKORN, P., TURNEY, K., THORNTON, M., NAIR, S., SZYGENDA, S., AND MANIKAS, T. 2010. Cyber threat trees for large system threat cataloging and analysis. *Systems Conference, 2010 4th Annual IEEE*, IEEE, 610–615.
- OSA, O. 2016. *Open Security Architecture Landscape*. .
- OWASP. 2017. OWASP Top 10 Most Critical Web Application Security Risks. https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf.
- PAJA, E., DALPIAZ, F., AND GIORGINI, P. 2014. Sts-tool: Security requirements engineering for socio-technical systems. In: *Engineering Secure Future Internet Services and Systems*. Springer, 65–96.
- PHILLIPS, C. AND SWILER, L.P. 1998. A graph-based system for network-vulnerability analysis. *Proceedings of the 1998 workshop on New security paradigms*, ACM, 71–79.
- PRICE SYSTEMS. 2005. True Planning guidance for estimating the cost impacts of ARP-4754, DO-254 and DO-178b/c certification. <http://www.pricesystems.com/Portals/1/Blog/02-05-15-A/DO-178bc%20and%20DO-254%20TP%20Modeling%20Guidance%20-%20DRAFT.pdf>.
- PROVINCE OF BRITISH COLUMBIA. 2012. Enterprise IT Security Architecture Security Zones: NETWORK SECURITY ZONE STANDARDS - Office of Chief Info Officer. www.cio.gov.bc.ca.
- RANNENBERG, K., PFITZMANN, A., AND MÜLLER, G. 1999. IT security and multilateral security. *Multilateral Security in Communications–Technology, Infrastructure, Economy*, 21–29.
- REHMAN, S. AND GRUHN, V. 2018. An Effective Security Requirements Engineering Framework for Cyber-Physical Systems. *Technologies* 6, 3, 65.
- RESPECT-IT. KAOS Tool - Objectiver: HomePage. <http://www.objectiver.com/index.php?id=25>.
- ROSENSCHEIN, J.S. 1988. Synchronization of multi-agent plans. In: *Readings in Distributed Artificial Intelligence*. Elsevier, 187–191.
- ROSS, D.T. AND SCHOMAN, K.E. 1977. Structured analysis for requirements definition. *IEEE transactions on Software Engineering* 1, 6–15.
- RUBINSTEIN, R.Y. AND KROESE, D.P. 2011. *Simulation and the Monte Carlo method*. John Wiley & Sons.
- SALNITRI, M., PAJA, E., AND GIORGINI, P. 2015. *From socio-technical requirements to technical security design: an sts-based framework*. Technical report, DISI-University of Trento.
- SALTZER, J.H. AND SCHROEDER, M.D. 1975a. The protection of information in computer systems. *Proceedings of the IEEE*.
- SALTZER, J.H. AND SCHROEDER, M.D. 1975b. The protection of information in computer systems. *Proceedings of the IEEE* 63, 9, 1278–1308.
- SANS. 2015a. Infrastructure Security Architecture for Effective Security Monitoring. .
- SANS. 2015b. Creating a Threat Profile for Your Organization - SANS Institute. <https://www.sans.org/reading-room/whitepapers/threats/paper/35492>.
- SANS. 2017. Securing Against the Most Common Vectors of Cyber Attacks. .

- SANS INSTITUTE. 2003. Information Security: Managing Risk with Defense in Depth. .
- SAP. 2013. DEFENSE IN DEPTH AN INTEGRATED STRATEGY FOR SAP SECURITY. .
- SCHMIDT, R., GUÉDRIA, W., BIDER, I., AND GUERREIRO, S. 2016. *Enterprise, Business-Process and Information Systems Modeling: 17th International Conference, BPMDS 2016, 21st International Conference, EMMSAD 2016, Held at CAiSE 2016, Ljubljana, Slovenia, June 13-14, 2016, Proceedings*. Springer.
- SCHNEIER, B. 1999. Schneier, B. Attack Trees. *Dr. Dobb's Journal*. *Dr. Dobb's journal* 24, 12, 21–29.
- SCHUMACHER, M. 2003. *Security engineering with patterns: origins, theoretical models, and new applications*. Springer.
- SECURE ARC. 2009. Logical Security Zone Pattern. http://www.securearc.com/wiki/index.php/Logical_Security_Zone_Pattern.
- SEdgeWICK, A. 2014. *Framework for improving critical infrastructure cybersecurity, version 1.0*. .
- SHANKAR, U., JAEGER, T., AND SAILER, R. 2006. Toward Automated Information-Flow Integrity Verification for Security-Critical Applications. *NDSS*.
- SHERWOOD, N.A. 2005. *SABSA (Sherwood Applied Business Security Architecture) - a business-driven approach*. CRC Press.
- SIMPLICABLE. 2002. 29 Strategic Drivers. <https://simplicable.com/new/strategic-drivers>.
- SINDRE, G. AND OPDAHL, A.L. 2001. Capturing security requirements through misuse cases. *NIK 2001, Norsk Informatikkonferanse 2001*, <http://www.nik.no/2001>.
- SINGHAL, A. AND BANATI, H. 2013. Fuzzy logic approach for threat prioritization in agile security framework using DREAD model. *arXiv preprint arXiv:1312.6836*.
- SOLHAUG, B. AND STØLEN, K. 2013. TheCORASlanguage—Whyitisdesigned the way it is. *Safety, Reliability, Risk and Life-Cycle Performance of Structures and Infrastructures, Proceedings of 11th International Conference on Structural Safety and Reliability (ICOSSAR'13)*, 3155–3162.
- SOUAG, A., MAZO, R., SALINESI, C., AND COMYN-WATTIAU, I. 2015. Reusable knowledge in security requirements engineering: a systematic mapping study. *Requirements Engineering*, 1–33.
- STAMATIS, D.H. 2003. *Failure Mode and Effect Analysis: FMEA from Theory to Execution*. ASQ Quality Press.
- STEVENS, S.S. 1946. On the theory of scales of measurement. .
- TELMON, C. AND CISA, CI. 2012. Haruspex—Simulation-driven Risk Analysis for Complex Systems. .
- UZUNOV, A.V., FERNANDEZ, E.B., AND FALKNER, K. 2012. Engineering Security into Distributed Systems: A Survey of Methodologies. *J. UCS* 18, 20, 2920–3006.
- VAN LAMSWEERDE, A. 2009. *Requirements engineering: from system goals to UML models to software specifications*. .
- VAN LAMSWEERDE, A., BROHEZ, S., DE LANDTSHEER, R., AND JANSSENS, D. 2003. From system goals to intruder anti-goals: attack generation and resolution for security requirements engineering. 3.

- WALIA, G.S. AND CARVER, J.C. 2009. A systematic literature review to identify and classify software requirement errors. *Information and Software Technology* 51, 7, 1087–1109.
- WEBER, P., MEDINA-OLIVA, G., SIMON, C., AND IUNG, B. 2012. Overview on Bayesian networks applications for dependability, risk analysis and maintenance areas. *Engineering Applications of Artificial Intelligence* 25, 4, 671–682.
- WIERINGA, R.J. 1996. Requirements engineering: frameworks for understanding. .
- WU, J., YIN, L., AND GUO, Y. 2012. Cyber attacks prediction model based on Bayesian network. *Parallel and Distributed Systems (ICPADS), 2012 IEEE 18th International Conference on*, IEEE, 730–731.
- YU, E.S.K. 2011. *Social Modeling for Requirements Engineering*. MIT Press.

Appendix A. FAIR Risk Taxonomy

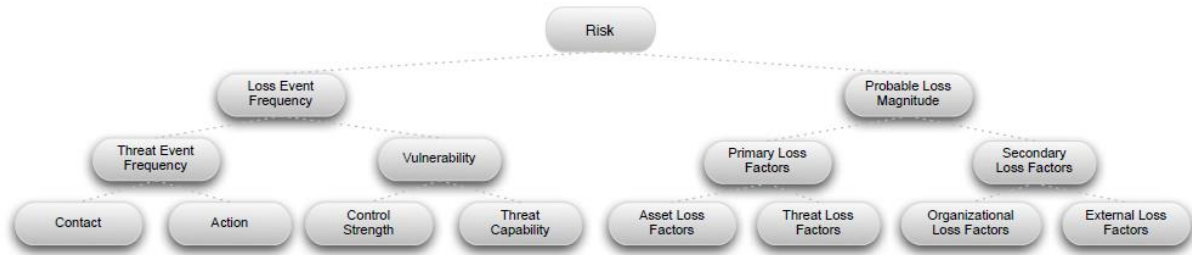


Figure 78: Risk taxonomy hierarchy

NAME	Definition
Loss Event Frequency (LEF)	The probable frequency, within a given time-frame, that a threat agent will inflict harm upon an asset
Control Strength (CS)	Defined as the strength of a security control or mitigation control as compared to a standard measure of force or threat capability of a threat agent
Threat Capability (TC)	Defined as the capability (skill set, knowledge) of a threat agent to cause harm to an asset.
Threat Event Frequency (TEF)	The probable frequency occurrence of thereat events within a given time-frame, when a threat agent act against an asset
Vulnerability	The probability that the control strength of an asset fails to resist the malicious actions of a threat agent
Contact	Occurs when a threat agent establishes a physical or virtual (e.g., network) connection to an asset
Action	An act taken against an asset by a threat agent. Requires first that contact occur between the asset and threat agent.
Probable loss magnitude (PLM)	The probable magnitude of loss occurred as a result of loss events
Asset Loss Factors	Criticality, Cost, Sensitivity (Reputation, Competitive Advantage, Legal), etc.
Threat Loss Factors	Action, Misuse, Disclosure, Modify, Deny Access, etc.
Organizational Factors	Timing, Detection, Response, Containment, Repudiation, Recovery, etc.
External Factors	Competitors, Legal, Media, Stakeholders, etc.

Appendix B. Graphical tool for analyzing the semantic dependencies of the quality characteristics

The key aspect of this graph tool is to identify and categorize the concepts for each of the definitions and represent them in a graph format. The method subsumes following stepwise approach Figure 64.

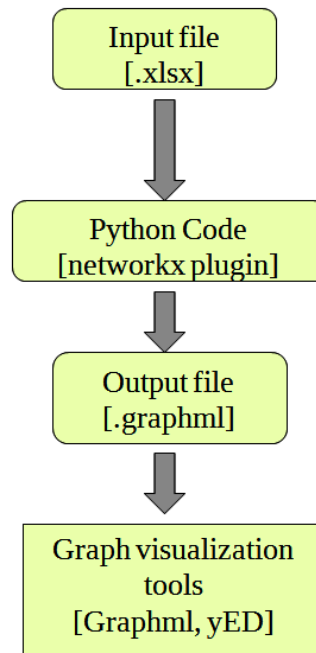


Figure 79: Graph creation approach

First step, we manually extracted all the 185 criteria definitions into a word file and then for each definition, we had further extracted the associated concepts as proposed by respective sources. We then categorized the criteria and identified concepts as well as the defined relation between them in an excel file as shown in figure 65.

In the next step, we built a graph, using python parser, where nodes represent criteria and concepts, and edges represent the defined relation between as per criterion definitions. We used python graph module *networkx* [NetworkX developers 2016] to create the graph in *graphml* format, a standard, comprehensive and easy-to-use file format available for graphs. This graphml file format is based on XML standard and hence ideally supported by many graph visualization tools. The final step of our analysis method is to analyse this graph with the help of graph visualization tools. We used two graphml viewer tools, Gephi and yEd.

NODES - 212

ID	COLOR	TYPE	DESCRIPTION
135	grey	def	Understandable and usable by all stakeholders
136	grey	def	Unnecessary constraints
137	grey	def	User can understand and apply the methods from available data
138	grey	def	Users and customers scope and evaluation
139	grey	def	Users from marketing department
140	grey	def	Vague pronoun (it, that, ...)
141	grey	def	Verify if all changes are reflected with up to date information
142	grey	def	(What) rather than (how)
143	red	cr	ABSTRACT (AB)
144	green	cr	ACHIEVABLE (ACH)
145	green	cr	ADAPTABILITY (ADB)
146	purple	cr	ADEQUACY (ADQ)
147	aqua	cr	AFFORDABLE (AF)
148	green	cr	ALLOCATED (ALC)

EDGES - 278

Src Node	Dest Node	Edge value
1	Definition of terms	WS2
2	logical links between concepts related	WS2, WS9
3	WELL STRUCTURED (WS)	WS13
4	WELL STRUCTURED (WS)	WS13
5	Standard constructs	WS12, WS14
6	Specific words	WS12, WS14, UA3, UA
7	Standard constructs	WS12, WS14
8	Cross-referencing dependency links	T1, T11, T15, ATR1
9	Group related requirements	CREF9, ATR1
10	Identical Reqs	CREF9, ATR1
11	Performance level of a requirement	ATR1
12	Dependencies attributes between Rec	CREF10
13	Group related requirements	CREF11
14	Abstraction or elaboration of Reqs	MIN10, CL15
15	Minimal explanation	MIN10, U9, CS14

PYTHON PARSER

```

def create_graphml():
    wb = openpyxl.load_workbook('DATA_v4.xlsx', data_only = True)
    # Fetch nodes data
    sheet1 = wb.get_sheet_by_name('NODES')
    len1 = 0
    len2 = 0
    sheet2 = wb.get_sheet_by_name('EDGES')
    for row in range(2, len1 + 1):
        node_color = sheet1.cell(row, 1).value
        node_type = sheet1.cell(row, 2).value
        node_id = sheet1.cell(row, 3).value
    # Fetch edges data
    for row in range(2, len2 + 1):
        edge_attr_name = sheet2.cell(row, 1).value
        edge_attr_value = sheet2.cell(row, 2).value
        edge_id = sheet2.cell(row, 3).value
    # Create graph
    G = nx.Graph()
    # Add nodes
    for node_id, node_color, node_type in zip(nodes.keys(), nodes.values(), node_types.values()):
        G.add_node(node_id, color=node_color, type=node_type)
    # Add edges
    for edge_id, edge_attr_name, edge_attr_value in zip(edges.keys(), edges.values(), edge_attr_values.values()):
        G.add_edge(edge_id, edge_attr_name, edge_attr_value)
    # Save graph
    Gml = nx.GraphMLWriter().write(G)
    with open('graphml_v4.xml', 'w') as f:
        f.write(Gml)

```

GRAPHML FILE

```

<?xml version="1.0" encoding="utf-8"?>
<graphml xmlns="http://graphml.graphdrawing.org/xmlns" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://graphml.graphdrawing.org/xmlns http://graphml.graphdrawing.org/xmlns/graphml.xml" >
  <key attr.name="color" attr.type="string" for="node" id="d1" />
  <key attr.name="type" attr.type="string" for="node" id="d2" />
  <key attr.name="type" attr.type="string" for="edge" id="d3" />
  <key attr.name="value" attr.type="string" for="edge" id="d4" />
  <node id="1" label="Definition of terms" color="grey" type="def" />
  <node id="2" label="logical links between concepts related" color="grey" type="def" />
  <node id="3" label="WELL STRUCTURED (WS)" color="grey" type="def" />
  <node id="4" label="WELL STRUCTURED (WS)" color="grey" type="def" />
  <node id="5" label="Standard constructs" color="grey" type="def" />
  <node id="6" label="Specific words" color="grey" type="def" />
  <node id="7" label="Standard constructs" color="grey" type="def" />
  <node id="8" label="Cross-referencing dependency links" color="grey" type="def" />
  <node id="9" label="Group related requirements" color="grey" type="def" />
  <node id="10" label="Identical Reqs" color="grey" type="def" />
  <node id="11" label="Performance level of a requirement" color="grey" type="def" />
  <node id="12" label="Dependencies attributes between Rec" color="grey" type="def" />
  <node id="13" label="Group related requirements" color="grey" type="def" />
  <node id="14" label="Abstraction or elaboration of Reqs" color="grey" type="def" />
  <node id="15" label="Minimal explanation" color="grey" type="def" />
  <edge id="1" source="1" target="2" type="WS2" value="WS2" />
  <edge id="2" source="2" target="3" type="WS2, WS9" value="WS2, WS9" />
  <edge id="3" source="3" target="4" type="WS13" value="WS13" />
  <edge id="4" source="4" target="5" type="WS13" value="WS13" />
  <edge id="5" source="5" target="6" type="WS12, WS14" value="WS12, WS14" />
  <edge id="6" source="6" target="7" type="WS12, WS14, UA3, UA" value="WS12, WS14, UA3, UA" />
  <edge id="7" source="7" target="8" type="WS12, WS14" value="WS12, WS14" />
  <edge id="8" source="8" target="9" type="T1, T11, T15, ATR1" value="T1, T11, T15, ATR1" />
  <edge id="9" source="9" target="10" type="CREF9, ATR1" value="CREF9, ATR1" />
  <edge id="10" source="10" target="11" type="CREF9, ATR1" value="CREF9, ATR1" />
  <edge id="11" source="11" target="12" type="ATR1" value="ATR1" />
  <edge id="12" source="12" target="13" type="CREF10" value="CREF10" />
  <edge id="13" source="13" target="14" type="CREF11" value="CREF11" />
  <edge id="14" source="14" target="15" type="MIN10, CL15" value="MIN10, CL15" />
  <edge id="15" source="15" target="16" type="MIN10, U9, CS14" value="MIN10, U9, CS14" />
  </graphml>

```

Figure 80: Graph creation approach input and output files

Following Figure 52 reflects all the criterion nodes and their defined reference links. However, it is important to note that this graph reflects our first analysis of the semantic dependencies. Different interpretation of criterion definitions and the semantic dependencies, can impact the number and nature of these edges. We coloured the nodes to highlight some patterns in dependencies. Therefore, a rigorous analysis concerning validity threats requires a proper research.

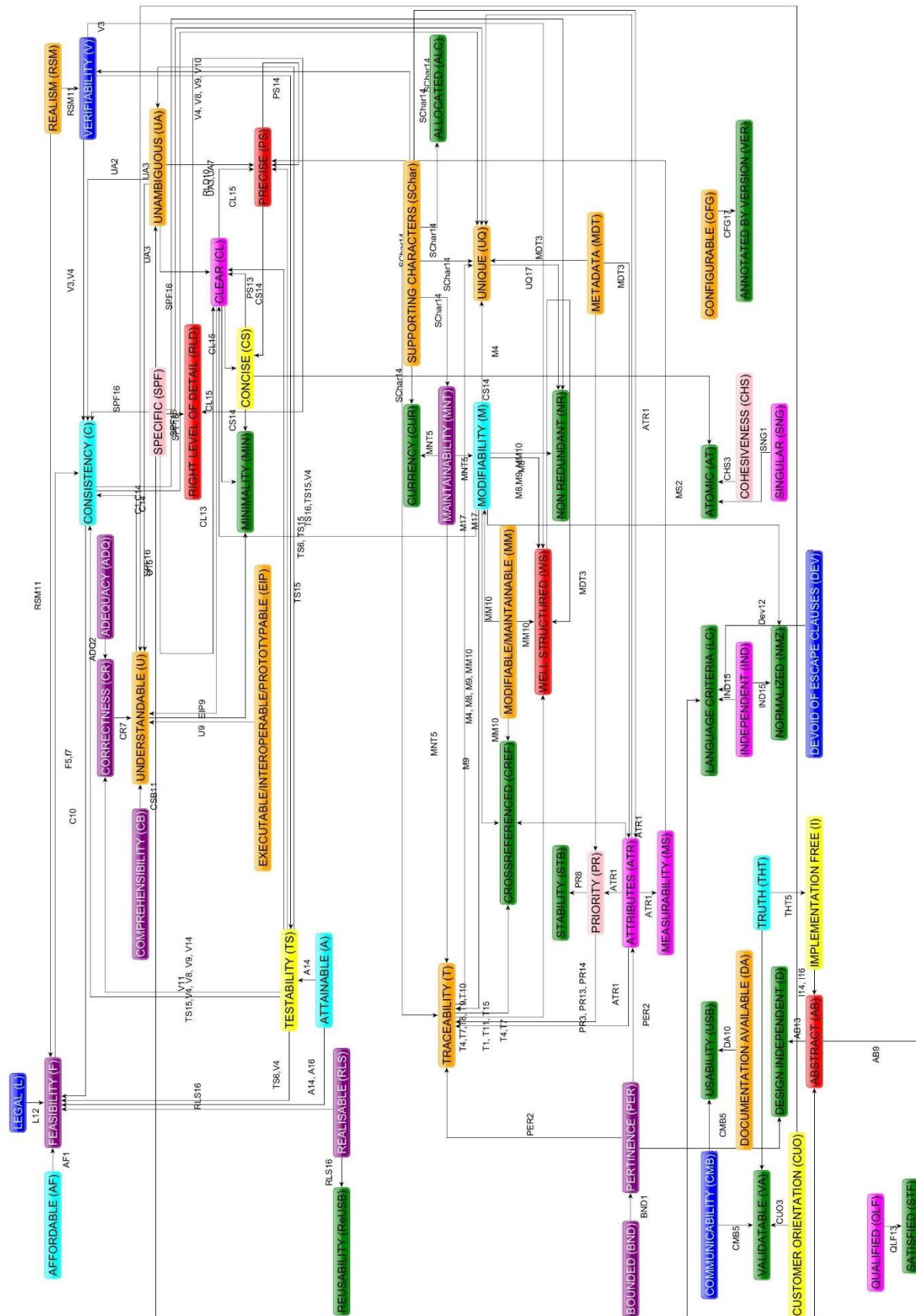


Figure 81: Overview of Criteria graph

Many variations were found in the propositions of these reference links between the criterion and concept nodes resulting in different pattern of definitions. For example, a reference can be observed

between a criterion and a concept, or either between two criteria or two concepts. Figure 0-82 depicts an example pattern. These criterion graph patterns greatly help in evaluating the efficiency of requirements elicitation and evaluation process employed by RE methodology. Depending on the context definition and type of reference link, the patterns vary differently.

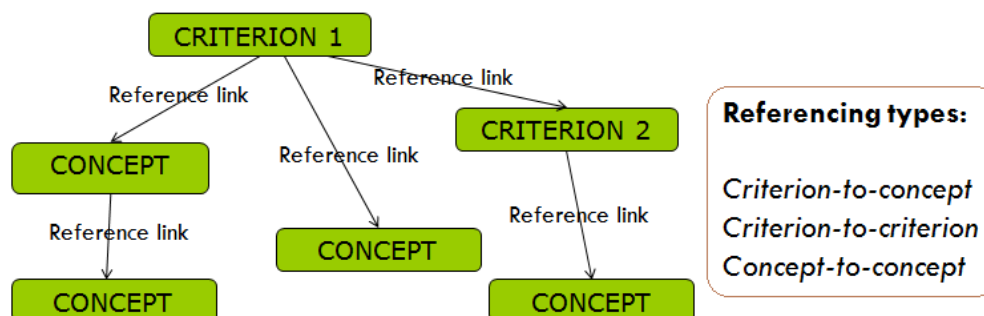


Figure 0-82: Criteria graph example pattern

Accordingly, we have highlighted some observed patterns by colouring the nodes.

Appendix C. Verification methods for the elicited evaluation criteria

Nominal scaling: uses numerals (a.k.a. labels) towards verifying the determination of equality

Ordinal scaling: uses rank ordering (e.g., less/more or small/big) towards verifying the determination of greater or less

Interval scaling: uses quantitative numeric values that express the exact difference between the values in measurement units (e.g., duration, length, weight).

Ratio scaling: uses quantitative numeric values that express the exact difference between the values in fraction of units (e.g., hours per day)

R^M2.1.1: The SRE methodology modelling language terminology must be easily understood to the user	
Ordinal scaling: The degree of support needed to understand the concepts.	
Verification method	Performance measure
Easy to understand	high
Easy to understand with some help	medium
Need some training to understand the concepts	low

R^M2.1.2: The cost of the SRE methodology training should be minimal	
Ordinal scaling: The degree of openness to tool support	
Verification method	Performance measure
Freely available tool	high
Tool available on free trial version	medium
No tool available, need manual effort	low

The verification of criteria **R^M2.1.3** is relative to **R^M2.1.1** and **R^M2.1.2**.

R^M2.1.3: The time taken to learn the SRE methodology should be minimal	
Ratio scaling: Learnability time in number of weeks	
Verification method	Performance measure

Easy to adapt the concepts in one week	high
Need at least two weeks	medium
Need at least one month	low

R^M3.1: Should support the need for users to easily communicate ideas and feedback respecting the abstraction needs	
Nominal scaling: network security requirements	
Verification method	Performance measure (nominal)
Facilitates to express the network security zoning information	high
No support to define abstraction levels	nil

R^M3.2: Should support the need for users to clearly define the number of abstraction levels of refinement in network security development cycle i.e., from business security problem context till network security problem context.	
Interval scaling: definite number of abstraction levels	
Verification method	Performance measure (ratio)
Allows to define the abstraction levels with clear separation of concerns	high
No support to define abstraction levels	nil

R^M4.1: The methodology should facilitate to trace the network security requirements back to business objectives	
Ordinal scaling: The degree of openness to trace security requirements	
Verification method	Performance measure (ordinal)
Tracing of (network) security requirements is explicit (i.e., direct)	high
Tracing of (network) security requirements is implicit (i.e., indirect)	medium
Minimal support for cross-referencing	low

The verification of criteria **R^M6.1** is relative to **R^M3.1**

R^M6.1: Must facilitate to specify and link network security zone information	
Nominal scaling: link security zoning information	
Verification method	Performance measure
Explicit linking of security zoning information to business risk impact	high
No support to link security zoning information	nil

R^M6.2: Should allow to annotate each requirement with risk attributes	
Nominal and Ordinal scaling: The degree of support to link risk attributes explicitly	
Verification method	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

R^M6.3: Should allow to annotate each requirement with priority information	
Ordinal scaling: link goal/threat with priority	
Verification method	Performance measure
Allows to prioritize security goals/requirements	high
Does not facilitate to prioritize security goals/requirements	nil

R^M6.4: Must facilitate to specify the implementation costs of network security requirements	
Nominal scaling: link implementation costs	
Verification method	Performance measure
Allows to annotate requirements with implementation costs	high
Does not allow to give information on implementation costs of requirement	nil

Appendix D. SRE methodology complete view

Utility functions example mapping – Integrity scale 1 to 5

Control capability	Criticality	Trust assumption	Integrity
Highly restricted	highly critical	Very highly trusted	5
restricted	Critical	highly trusted	4
controlled	Partially critical	trusted	3
Less controlled	Less critical	Partially trusted	2
uncontrolled	Uncritical	not trusted	1

Figure 83: utility functions example mapping

Control capability	Control	Maximum Assurance over the maturity of the security management	Integrity level
Highly restricted	Managed by the enterprise	Can enforce rigorous security verification process that is thoroughly verified	5
Restricted		Can enforce moderate security verification process that can be verified on regular basis	4
Controlled		Can enforce security measures but cannot be verified on regular basis	3
Less controlled	Managed by third-party	Externally managed by the third party. Security measures enforcement is on agreed terms. (i.e., Trust assumption on the capability of the third party)	2
uncontrolled	Cannot be managed	Cannot enforce any security measure.	1

Figure 84: Example Control capability / integrity level mapping

Criticality	Risk impact	Integrity
highly critical	Public effect	5
Moderately Critical	Economic/Environment effect	4
Partially critical	Economic effect	3
Less critical	Psychological effect	2
Uncritical	No effect	1

Figure 85: Example Criticality / integrity mapping

Trust level	Training and qualification	Integrity
Very highly trusted	Unconditionally trusted	5
highly trusted	Well trained employees qualified for highly critical operations	4
trusted	Well trained employees qualified for critical operations	3
Partially trusted	Well trained employees qualified partially/less critical operations	2
not trusted	Untrained employees/users	1

Figure 86: Example Trust/integrity level mapping

AIRBUS scenario – Aircraft maintenance SRE model

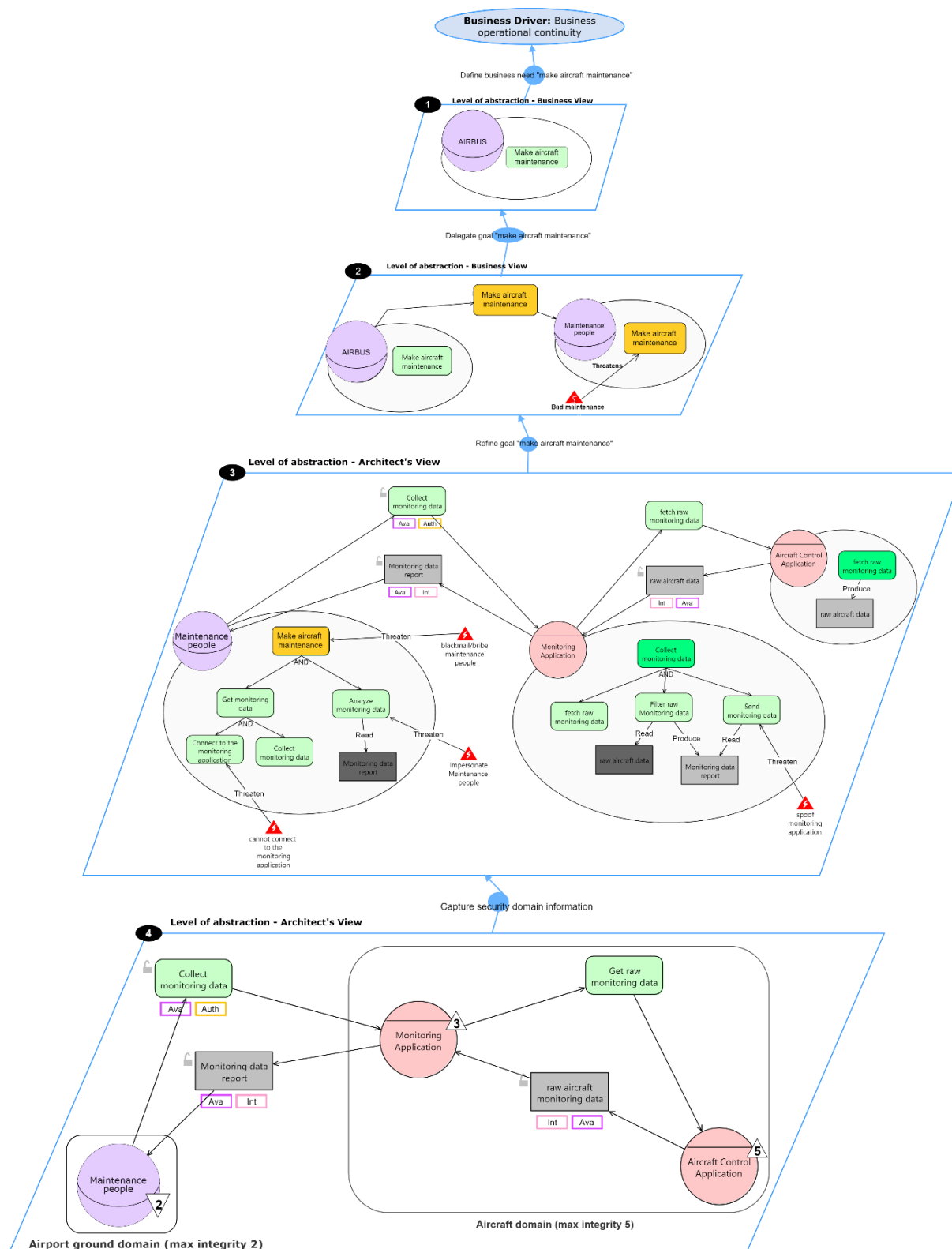


Figure 87: Use case Scenario 1 - Business view + Architect's view

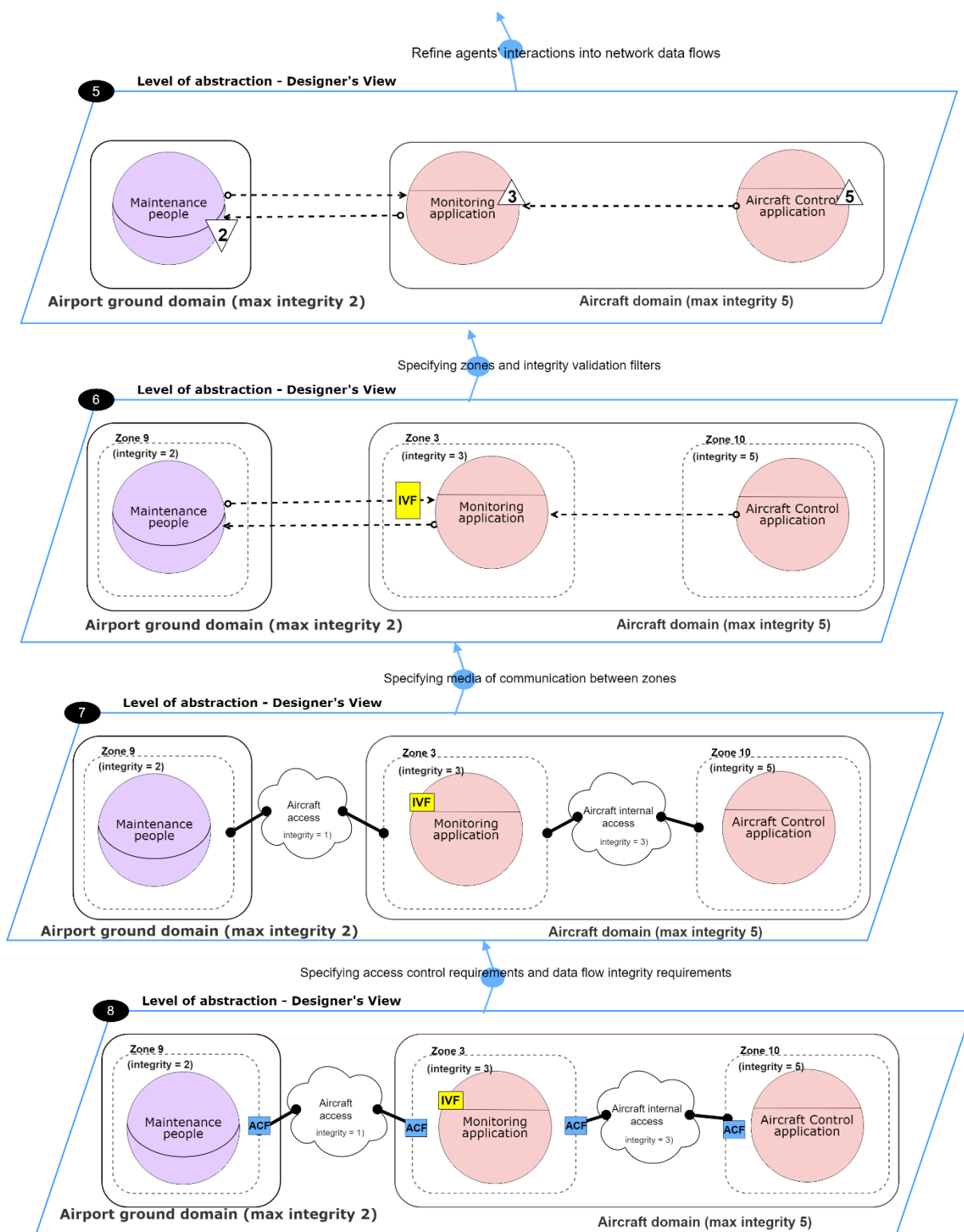


Figure 88: Use case Scenario 1 - Designers view

AIRBUS Scenario 2 – Security domain information

Domain	Control capability	Integrity
Aircraft	Highly restricted	5
Airbus building	Highly restricted	5
Airline building	Less controlled	2
Airport ground	Less controlled	2
System agents	Criticality	Integrity
Aircraft control application	Highly critical	5
Inflight entertainment system	Partially critical	3
Update application	Partially critical	3
Passengers entertainment system	Less critical	2
Environment agents	Trust	Integrity
AIRBUS	Very highly trusted	5
Airline	Partially trusted	2
Maintenance people	Partially trusted	2
Portable device	untrusted	1

Figure 89: AIRBUS scenario 2 - Security domain information

E-commerce scenario – Network security requirements

Table 14: E-commerce scenario - permitted data flows (step1)

Traffic between	List of data flows
Users and the DNS Server	<code>flow(adminUser,dnsServer) .</code> <code>flow(localUsers,dnsServer) .</code> <code>flow(visitors,dnsServer) .</code> <code>flow(remoteUsers,dnsServer) .</code> <code>flow(clients,dnsServer) .</code> <code>flow(dnsServer,adminUser) .</code> <code>flow(dnsServer,localUsers) .</code> <code>flow(dnsServer,visitors) .</code> <code>flow(dnsServer,remoteUsers) .</code> <code>flow(dnsServer,clients) .</code>
Users and the WEB Server	<code>flow(adminUser,webServer) .</code> <code>flow(localUsers,webServer) .</code> <code>flow(visitors,webServer) .</code> <code>flow(remoteUsers,webServer) .</code> <code>flow(clients,webServer) .</code> <code>flow(webServer,adminUser) .</code> <code>flow(webServer,localUsers) .</code> <code>flow(webServer,visitors) .</code> <code>flow(webServer,remoteUsers) .</code> <code>flow(webServer,clients) .</code>

Employees and the Accountability Server	<code>flow (adminUser,accountabilityServer) .</code> <code>flow (localUsers,accountabilityServer) .</code> <code>flow (remoteUsers,accountabilityServer) .</code> <code>flow (accountabilityServer,adminUser) .</code> <code>flow (accountabilityServer,localUsers) .</code> <code>flow (accountabilityServer,remoteUsers) .</code>
Administrator and the Database Server	<code>flow (adminUser,databaseServer) .</code> <code>flow (databaseServer,adminUser) .</code>
APP Server and Database Server	<code>flow (appServer,databaseServer) .</code> <code>flow (databaseServer,appServer) .</code>
The WEB Server and the APP Server	<code>flow (webServer,appServer) .</code> <code>flow (appServer,webServer) .</code>
Admin user and the APP Server	<code>flow (adminUser,appServer) .</code> <code>flow (appServer, adminUser) .</code>

Table 15: Access control filter requirements (step2)

ZONES	Access medium	ACF rules
Zone1	Private access	<code>ACF (connectedTO (privateAccess,1) ,</code> <code> flow (databaseServer,adminUser))</code> <code>ACF (connectedTO (privateAccess,1) ,</code> <code> flow (dnsServer,adminUser))</code> <code>ACF (connectedTO (privateAccess,1) ,</code> <code> flow (accountabilityServer,adminUser))</code> <code>ACF (connectedTO (privateAccess,1) ,</code> <code> flow (webServer,adminUser))</code> <code>ACF (connectedTO (privateAccess,1) ,</code> <code> flow (adminUser,databaseServer))</code> <code>ACF (connectedTO (privateAccess,1) ,</code> <code> flow (adminUser,dnsServer))</code> <code>ACF (connectedTO (privateAccess,1) ,</code> <code> flow (adminUser,accountabilityServer))</code> <code>ACF (connectedTO (privateAccess,1) ,</code> <code> flow (adminUser,webServer))</code>
Zone4	Private access	<code>ACF (connectedTO (privateAccess,4) ,</code> <code> flow (dnsServer,localUsers))</code> <code>ACF (connectedTO (privateAccess,4) ,</code> <code> flow (accountabilityServer,localUsers))</code> <code>ACF (connectedTO (privateAccess,4) ,</code> <code> flow (webServer,localUsers))</code> <code>ACF (connectedTO (privateAccess,4) ,</code> <code> flow (localUsers,dnsServer))</code> <code>ACF (connectedTO (privateAccess,4) ,</code> <code> flow (localUsers,accountabilityServer))</code> <code>ACF (connectedTO (privateAccess,4) ,</code> <code> flow (localUsers,webServer))</code>
Zone5	Private access	<code>ACF (connectedTO (privateAccess,5) ,</code> <code> flow (accountabilityServer,adminUser))</code> <code>ACF (connectedTO (privateAccess,5) ,</code> <code> flow (accountabilityServer,localUsers))</code> <code>ACF (connectedTO (privateAccess,5) ,</code> <code> flow (accountabilityServer,remoteUsers))</code> <code>ACF (connectedTO (privateAccess,5) ,</code> <code> flow (adminUser,accountabilityServer))</code> <code>ACF (connectedTO (privateAccess,5) ,</code> <code> flow (localUsers,accountabilityServer))</code> <code>ACF (connectedTO (privateAccess,5) ,</code>

		flow(remoteUsers, accountabilityServer))
Zone6	Private access	ACF (connectedTO (privateAccess, 6), flow (databaseServer, adminUser)) ACF (connectedTO (privateAccess, 6), flow (appServer, webServer)) ACF (connectedTO (privateAccess, 6), flow (webServer, appServer)) ACF (connectedTO (privateAccess, 6), flow (adminUser, databaseServer))
Zone7	Private access	ACF (connectedTO (privateAccess, 7), flow (appServer, webServer)) ACF (connectedTO (privateAccess, 7), flow (dnsServer, adminUser)) ACF (connectedTO (privateAccess, 7), flow (dnsServer, localUsers)) ACF (connectedTO (privateAccess, 7), flow (accountabilityServer, remoteUsers)) ACF (connectedTO (privateAccess, 7), flow (webServer, appServer)) ACF (connectedTO (privateAccess, 7), flow (webServer, adminUser)) ACF (connectedTO (privateAccess, 7), flow (webServer, localUsers)) ACF (connectedTO (privateAccess, 7), flow (adminUser, dnsServer)) ACF (connectedTO (privateAccess, 7), flow (adminUser, webServer)) ACF (connectedTO (privateAccess, 7), flow (localUsers, dnsServer)) ACF (connectedTO (privateAccess, 7), flow (localUsers, webServer)) ACF (connectedTO (privateAccess, 7), flow (remoteUsers, accountabilityServer))
Zone7	Public access	ACF (connectedTO (publicAccess, 7), flow (dnsServer, userVISITOR)) ACF (connectedTO (publicAccess, 7), flow (webServer, userVISITOR)) ACF (connectedTO (publicAccess, 7), flow (userVISITOR, dnsServer)) ACF (connectedTO (publicAccess, 7), flow (userVISITOR, webServer))
Zone7	Internet access	ACF (connectedTO (internetAccess, 7), flow (dnsServer, remoteUsers)) ACF (connectedTO (internetAccess, 7), flow (accountabilityServer, remoteUsers)) ACF (connectedTO (internetAccess, 7), flow (webServer, clients)) ACF (connectedTO (internetAccess, 7), flow (webServer, remoteUsers)) ACF (connectedTO (internetAccess, 7), flow (clients, dnsServer)) ACF (connectedTO (internetAccess, 7), flow (clients, webServer)) ACF (connectedTO (internetAccess, 7), flow (remoteUsers, dnsServer))

		ACF (connectedTO (internetAccess, 7), flow (remoteUsers, accountabilityServer)) ACF (connectedTO (internetAccess, 7), flow (remoteUsers, webServer)) ACF (connectedTO (internetAccess, 7), flow (dnsServer, clients)) ACF (connectedTO (internetAccess, 7), flow (webServer, clients)) ACF (connectedTO (internetAccess, 7), flow (webServer, remoteUsers)) ACF (connectedTO (internetAccess, 7), flow (clients, dnsServer)) ACF (connectedTO (internetAccess, 7), flow (clients, webServer)) ACF (connectedTO (internetAccess, 7), flow (remoteUsers, dnsServer)) ACF (connectedTO (internetAccess, 7), flow (remoteUsers, accountabilityServer)) ACF (connectedTO (internetAccess, 7), flow (remoteUsers, webServer))
Zone2	Public access	ACF (connectedTO (publicAccess, 2), flow (dnsServer, userVISITOR)) ACF (connectedTO (publicAccess, 2), flow (webServer, userVISITOR)) ACF (connectedTO (publicAccess, 2), flow (userVISITOR, dnsServer)) ACF (connectedTO (publicAccess, 2), flow (userVISITOR, webServer))
Zone3	Internet access	ACF (connectedTO (internetAccess, 3), flow (dnsServer, clients)) ACF (connectedTO (internetAccess, 3), flow (webServer, clients)) ACF (connectedTO (internetAccess, 3), flow (clients, dnsServer)) ACF (connectedTO (internetAccess, 3), flow (clients, webServer))
Zone9	Internet access	ACF (connectedTO (internetAccess, 9), flow (dnsServer, remoteUsers)) ACF (connectedTO (internetAccess, 9), flow (accountabilityServer, remoteUsers)) ACF (connectedTO (internetAccess, 9), flow (webServer, remoteUsers)) ACF (connectedTO (internetAccess, 9), flow (remoteUsers, dnsServer)) ACF (connectedTO (internetAccess, 9), flow (remoteUsers, accountabilityServer)) ACF (connectedTO (internetAccess, 9), flow (remoteUsers, webServer)) ACF (connectedTO (internetAccess, 9), flow (dnsServer, remoteUsers)) ACF (connectedTO (internetAccess, 9), flow (accountabilityServer, remoteUsers)) ACF (connectedTO (internetAccess, 9), flow (webServer, remoteUsers))

		ACF (connectedTO (internetAccess, 9), flow (remoteUsers, dnsServer)) ACF (connectedTO (internetAccess, 9), flow (remoteUsers, accountabilityServer)) ACF (connectedTO (internetAccess, 9), flow (remoteUsers, webServer))
--	--	--

Table 16: Data Flow integrity requirements (step2)

Private access medium	flowIntegrityRequirement (flow (accountabilityServer, adminUser), privateAccess, 4). flowIntegrityRequirement (flow (adminUser, accountabilityServer), privateAccess, 4). flowIntegrityRequirement (flow (adminUser, appServer), privateAccess, 4). flowIntegrityRequirement (flow (adminUser, databaseServer), privateAccess, 4). flowIntegrityRequirement (flow (appServer, adminUser), privateAccess, 4). flowIntegrityRequirement (flow (databaseServer, adminUser), privateAccess, 4).
Internet access medium	flowIntegrityRequirement (flow (accountabilityServer, remoteUsers), internetAccess, 2). flowIntegrityRequirement (flow (dnsServer, remoteUsers), internetAccess, 2). flowIntegrityRequirement (flow (remoteUsers, accountabilityServer), internetAccess, 2). flowIntegrityRequirement (flow (remoteUsers, dnsServer), internetAccess, 2). flowIntegrityRequirement (flow (remoteUsers, webServer), internetAccess, 2). flowIntegrityRequirement (flow (webServer, remoteUsers), internetAccess, 2).