Network of Excellence

D4.3 Part II:
Engineering Secure Future Internet Services: A Research Manifesto and Agenda from the NESSoS Community: Final Release
Abstract
This deliverable presents the third and final version of the NESSoS research roadmap in the area of secure service engineering.

Keyword list
Future Internet, Software and Service Engineering, Security, Roadmap
## Document History

<table>
<thead>
<tr>
<th>Version</th>
<th>Type of change</th>
<th>Author(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>V0.1</td>
<td>Table of Contents</td>
<td>Carmen Fernandez (UMA)</td>
</tr>
<tr>
<td></td>
<td>Section 5</td>
<td>Carmen Fernandez (UMA)</td>
</tr>
<tr>
<td>V0.2</td>
<td>Francisco Moyano Carmen Fernández</td>
<td>Francisco Moyano (UMA) Aljosa Pasic (ATOS) Pieter Philippaerts (KUL) Christoph Sprenger (ETH) Benoit Baudry (INRIA)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Format Review</td>
</tr>
<tr>
<td>V0.3</td>
<td>Fabio Martinelli</td>
<td>Comments and contribution in the whole deliverable</td>
</tr>
<tr>
<td></td>
<td>Carmen Fernandez Francisco Moyano</td>
<td>Approved by reviewers</td>
</tr>
</tbody>
</table>
Table of Contents

TABLE OF CONTENTS ............................................................................................................. 3

LIST OF FIGURES .................................................................................................................... 5

LIST OF TABLES ........................................................................................................................ 6

LIST OF ACRONYMS .................................................................................................................. 7

EXECUTIVE SUMMARY ........................................................................................................... 8

1 INTRODUCTION ....................................................................................................................... 9

2 CONTEXT AND SCOPE OF THE NESSoS RESEARCH ROADMAP .................................. 10
  2.1 Future Internet Context ....................................................................................................... 10
  2.2 NESSoS Scope .................................................................................................................... 11
  2.3 NESSoS Boundaries .......................................................................................................... 11
  2.4 NESSoS Research Areas ................................................................................................... 12

3 FUTURE INTERNET APPLICATION SCENARIOS ............................................................... 16
  3.1 eHealth ............................................................................................................................. 16
  3.2 Smart Grid ........................................................................................................................ 18
  3.3 Mapping Research Topics to Scenario Requirements ...................................................... 18

4 TOPICS AND ACTIONS ........................................................................................................ 22
  4.1 Main Contributions of the Roadmap ................................................................................ 22
  4.2 Properties to be Ensured .................................................................................................... 22
  4.3 Two Main Crossing Research Themes of Major Interests .............................................. 23
  4.4 Enabling Methodologies and Technologies to Enhance FI Trustworthiness .................. 29

5 CONCLUSION ........................................................................................................................ 48

CONTRIBUTORS ..................................................................................................................... 49

REFERENCES .......................................................................................................................... 50
List of Figures

Figure 1 Roadmap Topics .................................................................................................................... 15
List of Tables

Table 1 eHealth Requirements Mapping to Research .................................................. 20
Table 2 Smart Grid Requirements Mapping to Research .................................................. 21
### List of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FI</td>
<td>Future Internet</td>
</tr>
<tr>
<td>IAB</td>
<td>Industry Advisory Board</td>
</tr>
<tr>
<td>ICT</td>
<td>Information and Communication Technologies</td>
</tr>
<tr>
<td>IT</td>
<td>Information Technology</td>
</tr>
<tr>
<td>ROI</td>
<td>Return on Investment</td>
</tr>
<tr>
<td>SaaS</td>
<td>Software as a Service</td>
</tr>
<tr>
<td>SDLC</td>
<td>Software Development Life Cycle</td>
</tr>
<tr>
<td>STS-ml</td>
<td>Socio-Technical Security modelling language</td>
</tr>
<tr>
<td>UWE</td>
<td>UML-based Web Engineering</td>
</tr>
</tbody>
</table>
Executive Summary

This document represents the final version of the NESSoS research agenda that identifies the research results that needs to be achieved for tackling the relevant security challenges that the adoption of the FI entails.

These topics were identified in different phases of the project and involved the whole NESSoS community. Indeed, starting from the initial draft set up by the NESSoS core partners, the research agenda has been extended and modified by incorporating the comments of the NESSoS associates and IAB members. Eventually, by means of an on-line survey the whole community had the opportunity to contribute.

First, we identified the topics that could be considered as transversal topics. These topics are assurance and, risk and cost-awareness during the SDLC. We argue that these two topics are central for the adoption of FI.

We also consider enabling methodologies and technologies that contribute to the enhancement of trustworthiness in FI services. Besides these two transversal areas, NESSoS research is focusing on methodologies and technologies for enhancing trustworthiness of the FI, we summarize here the main elements identified:

- Security requirements engineering
- Secure service architecture and design
- Security support in programming environments
- Service composition and adaptation
- Runtime verification and enforcement
- User-centric security
- Security management
- Autonomic security
- Quantitative aspects of security

The target readers of this document include at least researchers and policy makers.
1 Introduction

The Future Internet (FI) brings new vulnerabilities and threats for security, i.e., new security challenges. Thus, reacting to these new aspects will contribute to the success of the adoption of the FI. This will mean the inclusion of security from early stages of the Software Development Life Cycle (SDLC) when applied to software-based services and systems that constitute the FI.

This final version of the NESSoS research agenda collects the contributions of NESSoS members. In particular, with respect to the last version of D4.2 it collects the comments received from the members of the NaLAB and others inside the NESSoS community. This document represents the final contribution of the NESSoS consortium to the definition of a mid-term research agenda and manifesto in the area of secure service engineering.

The topics are organised in different blocks. First, we identified the topics that could be considered as transversal topics to the classical SDLC. These topics are assurance, and risk and cost-awareness during the SDLC. We also consider enabling methodologies and technologies that contribute to the enhancement of trustworthiness in FI services. We start with the methodologies that are usually involved in the SDLC such as requirements engineering, secure architecture and design, secure programming environments, run-time verification and enforcement or service composition. Other topics to be considered also during the life cycle of the software are user-centric security, security management or quantitative aspects of security.

The structure of the document is as follows. Section 2 describes the context where the NESSoS roadmap is framed and its scope within the 2020 vision. Section 3 exemplifies two of the NESSoS scenarios and how they can benefit from the topics identified in this roadmap. These topics are presented in Section 4. They are a review from the topics presented in previous editions, following mainly the comments made by the NESSoS NaLAB members and from the open survey made on the web. Finally, Section 5 concludes this document.

The intended readers of this document include primarily researchers and policy makers. Researchers can find in this document useful insight for further research directions (including young researchers and their PhD studies). Policy makers can find a summary of the main priorities identified by the community and which deserve attention in the research agenda in Section 2.
2 Context and Scope of the NESSoS Research Roadmap

2.1 Future Internet Context

This research roadmap starts with the fundamental assumption that we are in the middle of the largest and most important change in information and communication technology, which is largely based on the service oriented ICT vision, as well as integration of physical and virtual worlds.

In the service oriented vision of ICT, the shift is not only happening in the way the software is designed, composed or delivered: all of these things are already there. The service model enables to continuously “on-the-fly” improve the functionality available to users who require major changes in end-to-end security mechanisms. This new setting will change the way software and service engineering will work, especially to make applications secure. The dynamicity of the applications or the need to a rapid adaptation of them into new settings or requirements might lead to the disappearance of boundaries between design and runtime. Systems will become more open and this will influence the way services are created and adapted and how security will be adopted in them. The need for real time verification and assurance will become more evident in these changing and dynamic environments.

The service-based systems decouple provisioning and management of the infrastructure from use of services, which allows users and service providers to focus on the areas where they are the best positioned to create value. Traditionally, security and dependability have been regarded as secondary aspects in system design and development. They have always been treated as an afterthought and, in many cases, it was taken for granted that security and dependability (S&D) would be provided by external infrastructures. The shift towards service-based systems introduced an opportunity to rethink the way we deal with old ad hoc security and dependability solutions. During the last five years, many research projects already addressed this opportunity through e.g. research on how to provide better support for capturing and incorporating security requirements.

In parallel, the emergence of the so-called Internet of Things and the integration between physical and virtual worlds gave birth to what is sometimes addressed as “future internet services”. Sensors and actuators embedded in physical objects (e.g. machines, roads or street lighting) are linked through networks, producing huge volumes of data that flow to services. What is revolutionary in all this is that these “future internet service-based systems” are already beginning to be deployed, and some of them even work autonomously, that is without human intervention. In the new scenarios, not only systems as a whole but also individual services running in or supported by those systems will have to adapt to dynamic changes of hardware and software, and even firmware configurations, to unpredictable appearance and disappearance of devices, software components or infrastructure resources. Pre-defined trust relationships between components, applications and their system environments can no longer be taken for granted.
2.2 NESSoS Scope

NESSoS efforts focus on the area of secure services development for the FI. NESSoS addresses security at the service level even when the infrastructure level must be considered to detect security requirements and constraints. Security at the hardware level or at the network level are of interest for NESSoS mainly for understanding the limitations imposed by them to higher levels of abstractions for FI services. More details on this are provided in the next section.

The way we envision the secure service development is by the typical process of gathering requirements, followed by the design of an architecture ending on implementation. It is relevant to note that we address security from the early phases of the SDLC, considering that security and privacy by design are main principles of NESSoS. However, all of this might not be enough in the new settings that the FI will bring. We need to ensure that the services created are secure; therefore we foresee security assurance as a transversal topic to be considered of paramount importance. Also, risk and cost awareness during the SDLC is one of the key research directions we foresee as transversal, since it links security concerns with business and decision-making.

Besides these two transversal areas, NESSoS research is focusing on methodologies and technologies for enhancing trustworthiness of the FI [D4.1_II]. We summarize here the main elements identified:

- Security requirements engineering
- Secure service architecture and design
- Security support in programming environments
- Service composition and adaptation
- Runtime verification and enforcement
- User-centric security
- Security management
- Autonomic security
- Quantitative aspects of security

2.3 NESSoS Boundaries

As mentioned above, NESSoS is concerned with how to develop secure software and services, from the early specification and requirements to the final composition and implementation. The FI, however, comprises other entities and mechanisms that are not directly tackled by NESSoS, but which are relevant to consider as they may provide an informative input for the research activities developed within NESSoS.

Sensors and actuators, as well as other embedded devices, present processing and memory limitations that impose constraints on the type of security mechanisms that can be applied or on which security services may be deployed.

Network security is only considered at the higher layers of communication protocol stacks; nevertheless design and implementation of secure cryptographic communication protocols remains a main target of NESSoS.
Hardware and network security are considered up to the extent those are useful to provide high-level aspects of services running through these infrastructures.

Indeed, whereas NESSoS is concerned with how to protect the services hosted in FI infrastructures, how to securely build the infrastructure itself (the system) remains in the boundaries. This is similar for back-end systems and big data infrastructures, as they also present security and privacy challenges, especially when combined with data mining techniques.

Cryptography forms a corpus of knowledge that can enhance existing symmetric and asymmetric encryption mechanisms, digital signatures, hashes and Message Authentication Codes. NESSoS does not concern about the design of these cryptographic primitives, but it relies on concrete specification and implementations of them in order to provide high-level security requirements design and analysis such as confidentiality, integrity or non-repudiation.

2.4 NESSoS Research Areas

We briefly list in this section the key recommendations from the NESSoS consortium towards a trustworthy FI. The interested reader may find a more detailed description of these recommendations in Section 4. We split between two traversal areas that were considered of prominent importance since the very beginning from the NESSoS community (described in Section 2.4.1), followed by a set of more specific topics (described in Section 2.4.2).

2.4.1 Two Main Crossing Research Themes of Major Interest

Security Assurance during SDLC: Assurance plays a central role in the development of software-based services to provide confidence about the desired security level. Assurance must be treated in a holistic manner as an integral constituent of the development process. We thus speak about design for assurance, seamlessly informing and giving feedback at each stage of the SDLC that the related models and artefacts satisfy their functional and security requirements and constraints. Since the choice of appropriate assurance methods depends on several factors including the concrete application context and the desired level of assurance, this activity will cover a correspondingly broad range of assurance methods that jointly offer full development cycle support. Quantitative notions of security (including metrics) will allow having systems being able to trade-off among several requirements in a rationale way and considering multiple factors (including energy consumption for the protection mechanisms).

Risk and Cost-aware SDLC: Secure engineering of software services requires security to be an intrinsic aspect of all phases and activities of the SDLC. Crucial in this setting is the continuous identification and assessment of security risks and cost to support the various development activities in reducing vulnerabilities by identifying the adequate security requirements, mechanisms and controls. At the same time, the value of such security solutions and their return on investment (ROI) must be justifiable and clearly demonstrated from a business-oriented perspective. In the setting of the FI SDLC, there are two main challenges that need to be addressed. First, security, risk, and cost assessment need to be embedded as a separate activity of the overall SDLC process, closely interacting with the service engineering activities, in each iteration of the SDLC. Second, the security risk and cost assessment in itself must be
supported by a well-defined process and by assessment methods that are adapted to the characteristics of FI services. Some relevant goals in this respect are achieving traceability between risk and development models, managing evolving risks, and assessing risks at runtime. The definition of more precise economic security measures than what is available today is also necessary, as well as adequate means to assess ROI in security.

2.4.2 Enabling Methodologies and Technologies to Enhance FI Trustworthiness

**Security Requirements Engineering:** The FI applications will consist of a large amount of entities that will bring new security requirements that will need to be addressed such as location-privacy or privacy requirements. Sometimes, each of the entities might have its own security requirements with respect to the application, and it is there when techniques for conflicts resolution should be developed. Special emphasis should be made on requirements related to compliance and privacy requirements. Also, the evolvement of requirements through the whole SDLC and the socio-economic impact of this evolvement should be taken into account.

**Secure Service Architecture and Design:** We need to increase the capability of designing secure software-based service systems for the Internet of the future as well as to analyse security and ensure compliance of the underpinning architectures. The inclusion of identification, assessment and improvement of design principles in order to enhance those architectures in terms of flexibility, modularity and composability are also needed. All of these will facilitate the integration of novel security services as the Future Internet scenarios evolve. The research topics that need to be covered are related to model-driven architecture and security, compositionality of design models and security design patterns for FI. Also, it will be worth to explore the design of new appropriate languages to deal with specific properties such as for example privacy.

**Security Support in Programming Environments:** This research area covers new programming platforms that deliver development and runtime environments for trustworthy application code to be executed in the complex application scenarios depicted by the Future Internet. We will also address language based security, as well as secure coding principles and practices. Research will be based both on language design and implementation, including middleware and run-time environment. Type systems, verifying compilers, support for run-time property verification and enforcement will be addressed here as well. Programming principles and constructs will be investigated in order to ease secure service development and composition for the new application scenarios. Code signatures as well as code instrumentations, aspect oriented and other composition techniques for security and secure execution environments are also in the scope of this area.

**Secure Service Composition and Adaptation:** The integration and interoperability of services in order to tailor and enhance new services require adapting the service interfaces at different levels, including the semantic level. Other aspects to consider include assessing the trustworthiness of composition of services as well as composing security measures.
**Run-Time Verification and Enforcement:** Run-time verification complements programming verification and testing in order to provide assurance that cannot always be delivered. The research approaches we propose for this topic are run-time monitoring of data flow and monitoring usage control properties.

**User-Centric Security:** With the upcoming growth of the FI users are becoming more situational-aware of the risks they are exposed to when using new services. The dynamicity of the FI also makes it difficult to present the information in a clear format to the users. Therefore, the efforts on research should be oriented towards this direction making special emphasis on privacy.

**Security Management:** The security features of any system should be supported by appropriate monitoring and management tools. These tools will aid to implement measures to deal with threats and attacks and also to deal with security incidents. Even though the research topic is not new, we need to deep in it and in its deployment.

**Autonomic Security:** FI services will have to be executed on specific contexts. It would be desirable to choose different security mechanisms depending on certain levels of security depending on the context. New reasoners will therefore be needed in order to exploit service environment information and thus predict security reconfigurations depending on the changes in the environment.

**Quantitative Aspects of Security:** Quantitative security provides a numerical back-up for software artefacts. It is particularly essential for assurance and risk analysis and cost estimation. Some topics that greatly benefit from this area of research are usability and users security awareness. Quantitative security will assist users to become aware of how much personal information, for instance is being leaked in a certain service. Software vulnerabilities can also quantitatively been analysed.

Besides these topics we foresee a set of crucial properties such as compliance, privacy, trust and identity management to be always taken into account. On the top of all of these research topics, and security properties we should consider what threats we will face if the research we propose is not carried out. Figure 1 shows a graphical representation of the topics we listed above.

In the core of this figure we depicted the major transversal topics of interest: security assurance and risk and cost-aware SDLC. Then, the topics highlighted in blue correspond to the software life cycle. The four topics outside the inner box correspond to technologies that are related to the development of the life cycle but have come out as additional topics of research. The properties to be always enhanced and the threats to face are present in all the facets of the life cycle.
Figure 1 Roadmap Topics
3 Future Internet Application Scenarios

As mentioned in Section 2, the view for the future is that a new Internet paradigm will emerge, where new scenarios will arise bringing embedded new security challenges, and in particular, for secure service engineering.

The new FI application scenarios could come from different domains although they are mainly related to scenarios where nomadic users or Internet of Things are involved. The commonality of these scenarios is the need for an environment where multiple parties and organisations have to share resources and re-use services. This brings the need for Software as a Service (SaaS) to be ensured in them. Typical technologies that will be very useful for the kind of new scenarios that will arise are cloud technologies or mobile applications. These technologies will fulfil the needs of FI scenarios that will arise where increasing amount of data and increasing dependency on third parties will be a fact. It is also likely that the users involved in the FI scenarios depend on mobile devices. A trustworthy environment should be guaranteed where users can access and use the provided services in a secure manner.

As examples of application areas, we list here the representatives of the Future Internet Private Public Partnership (FI-PPP) program (an industry-driven innovation effort on Future Internet infrastructures, devices, software, service and media technologies):

- eHealth: electronic Health Records, Patient Monitoring, Patient Consent
- Smart Grid: Smart Home, Micro Grids, and electric Car
- Smart Agri-Logistics
- Environmental ICT
- Multimedia distribution and streaming
- Online games
- Cooperative business networks for transport and logistics
- Service Platform for Transport and Mobility, Urban Services
- Monitoring and public safety and security services

Since the range of FI scenarios is extensive, we selected two scenarios that can be seen as an exemplification of typical FI Internet scenarios. For these scenarios, the composition of services is an essential fact, as well as supervision and control. In particular, for the supervision and control scenario we selected a smart grid scenario whereas for the sharing of resources the chosen one has been an eHealth scenario.

It is worth noticing how these scenarios together embed all the following components of the Future Internet infrastructures:

- IoT (Internet of Things)
- Mobile Personal Devices
- SaaS (Software as a Service)

These scenarios are briefly recalled in the next Subsection. The interested reader is encouraged to read [D11.2], where the scenarios have been described with much more detail.

3.1 eHealth

eHealth (or electronic health) is broadly defined by the World Heath Organization as the “use of information and communication technology for health” [WHO06]. The objective is to
use Information and Communication Technology to improve health care service delivery through the strategic use of technologies such as computers, Internet, satellite receivers, and personal mobile devices.

NESSoS considers the following scenarios.

### 3.1.1 Management of Electronic Health Records

An Electronic Health Record (EHR) includes any information created by, or on behalf of, a health professional in the context of the care of a patient.

We assume that EHR repositories and management systems are managed by GPs (General Practitioners) in their office systems, by a ward or hospital department, or even by a group of hospitals. Indeed, a common trend today is that hospitals group together to build large circles of trust that share, under some regulations and self-imposed rules, patient information, creating a network of hospitals that use a common EHR system.

We are interested not only in EHR management systems but also in their integration with additional services that offer further functionality, which could be provided by external partners. These services may include medical best practices, decision support for the practitioner, analysis of images, billing systems, insurance and financial systems, electronic prescriptions and pharmacy services, etc.

We also assume that there are privacy policies, or access control policies associated with the EHRs that describe what actions are allowed on the data: who is able to read the which EHR, to write it (and how), to delete it (and when). Also, the purpose for which the EHR can be accessed may be stipulated in the policies.

### 3.1.2 Patient Monitoring

This Scenario is related to the application of Internet of Things for Health Care and to mobile Health (mHealth). mHealth [GIT06, IJZ04] broadly encompasses the use of mobile telecommunication and multimedia technologies for the integrated use within health care delivery systems. It can be defined as “mobile computing, medical sensor, and communications technologies for health care” [IJZ04].

The underlying setting include a patient wearing diagnostic sensors that send information about the patient health (e.g. blood pressure) to a server hosted in the hospital. Physicians can query this information at any moment, and can also receive alarms in case of emergency for fast assistance.

### 3.1.3 Patient Consent

Electronic health records should support clinical research [PB05, UGM’06, PPSS06]. But, as discussed in [SSW’06], the Health Insurance Portability and Accountability Act (HIPAA) – and probably other privacy regulation like European Union EC 95/46/EC– complicates the research process. To conduct research, more resources and time are necessary. This is due, among other reasons, to concerns about the security of patient medical information, which makes health organisations more hesitant to let researchers have access to the patient’s information, especially via electronic transfer. Thus a strong assurance that the data has been anonymized or has been aggregated correctly and has been transmitted securely to the intended recipient(s) is necessary.
3.2 Smart Grid

Smart grids use information and communication technology (ICT) to optimize the transmission and distribution of electricity from suppliers to consumers, allowing smart generation and bidirectional power flows – depending on where generation takes place. With ICT the Smart Grid enables financial, informational, and electrical transactions among consumers, grid assets, and other authorized users [Nat09].

NESSoS considers the following scenarios.

3.2.1 Smart Appliances, Smart Metering and Home Area Network

Smart Meters enable frequent billing, make the current and accumulated energy consumption visible to the user, and provide administrative access for the measuring point operator or the network operator. On the other hand, sensors in a house enable appliances to make decisions about changing the state of the devices based on price, time of day, or other parameters designated by the customer.

Those technologies are part of the so-called Advanced Metering Infrastructure (AMI). AMI consists of the hardware and software (including system and data management applications) required to connect, at the different network layers, end systems at customer premises to business and operational systems of utilities and third parties, allowing them to exchange information.

3.2.2 Electro Mobility

Electro mobility consists of the use of electric cars and the further integration of private and public passenger transport. On the one hand, we will need sufficient energy supply to the electric cars. On the other hand, electric cars will be a mobile energy storage system and will contribute to the grid’s ability to compensate fluctuations of distributed generation from renewable energy sources.

3.2.3 Local Market Automation

A particular way of creating a micro-grid with local billing capacity is via the use of distributed software agents. Such entities are adaptive, self-aware, self-healing, and semi-autonomous control elements that rapidly respond at a local level, without the direct intervention of a centralized control system or humans. The agents are combined to form a multi-agent system, capable of negotiating policies and reaching agreements. If the design is properly done and the incentives are correct, the agreements are near to optimal goals for the system as a whole.

3.3 Mapping Research Topics to Scenario Requirements

The goal of this section is to relate the requirements of the aforementioned scenarios to the research interests topics in Topics and Actions presented in Section 4. [D11.2] listed a set of security requirements that should be met by the scenarios.

Some of these requirements are out of the scope of NESSoS as standard mechanisms exist to tackle them. For example, confidentiality, integrity, non-repudiation and authentication,
which are common requirements to both scenarios, may use existing mechanisms such as symmetric and asymmetric encryption, message authentication codes and keyed hashes.

Single sign-on may use standard procedures, like SAML or Kerberos, whereas availability can be enforced with mechanisms that are relevant only after the system has been designed and implemented, like redundancy or firewalls.

Section 3.3.1 describes the alignment between the eHealth scenario requirements and the proposed topics and actions, whereas section 3.3.2 proceeds similarly with the requirements of the smart grid scenario.

### 3.3.1 eHealth Security Requirements

As the assets to protect in the eHealth scenario are sensitive personal identifiable information about patients, one of the primary requirements is privacy. In this direction, research has been conducted in order to analyse privacy threats during requirements analysis by using problem frames and a UML profile for privacy requirements elicitation. More research in the future is proposed in the area of analysis of privacy requirements that will aid the existing analysis already performed by NESSoS (see Section 4.4.1.2)

Non-disclosure is another requirement directly related to privacy that is pursued. In particular, the Avantssar’s Ochestrator is capable of automatically orchestrate the communication between a client (a patient or a physician by delegation) and an expert. The orchestrator generates a mediator between the two entities that satisfies certain constraints, such as the non-disclosure requirement.

Also, STS-ml is a goal-oriented language that allows capturing an information system from different views. It is also a requirements specification language and allows determining whether a requirement is fulfilled by the current specification of the system or not. In particular, compliance with the non-disclosure requirement is checked.

Another important security requirement is authorisation through both declarative and dynamic access control policies. This line of research is covered by the ActionGUI modelling language, which allows generating a security-aware application by decoupling the data model from the security and GUI model. The security model captures the roles and permissions of the entities with regards to the data model and the view only shows information over which the logged entity has permission to see or update.

Another line of research related to authorisation is UWE (UML-based Web Engineering), which allows modelling the logic and security requirements of a web application. Concretely, UWE defines a content diagram where user and roles are defined in the context of the application, and the basic rights of each role are further defined at different levels of granularity (e.g. access a complete EHR or only a part of it). Roles are also used to define the navigation paths through the web application that users can take after authentication.

Related to authorisation, revocation and delegation of authorisation are also important requirements that are already being tackled by the research carried out by partners. In particular, a formalism is provided in order to verify certain related properties such as whether a patient is left unattended after a physician is retired or on vacations. Research on run-time verification and enforcement, and assurance proposed in this roadmap will be also used in these cases. In line with the research to be carried out in these cases, the support for the emergency case is also an important requirement that is in line with the research of partners. Concretely,
the auto-delegation for health care formalizes a ‘break-the-glass’ access control policy that allows less qualified physicians to access patients’ information in case of an emergency, that is, if the potential damage caused by no action is big enough.

Table 1 summarises the relationships between requirements and related research in the eHealth scenario.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Related Research</th>
</tr>
</thead>
<tbody>
<tr>
<td>Privacy</td>
<td>Problem frame-based privacy threats identification</td>
</tr>
<tr>
<td></td>
<td>Avantssar’s orchestrator</td>
</tr>
<tr>
<td></td>
<td>STS-ml</td>
</tr>
<tr>
<td>Authentication¹ and Authorisation</td>
<td>ActionGUI</td>
</tr>
<tr>
<td></td>
<td>UWE</td>
</tr>
<tr>
<td></td>
<td>Revocation and delegation of authorisation formalism</td>
</tr>
<tr>
<td>Support for emergency case</td>
<td>Auto-delegation policy</td>
</tr>
</tbody>
</table>

Table 1 eHealth Requirements Mapping to Research

### 3.3.2 Smartgrid Security Requirements

As it was the case in the eHealth scenario, privacy becomes a requirement of paramount importance in smart grid due to the sensitivity of the information that it manages: consumption of energy patterns, location of the car, etc. This requirement conflicts with another important requirement: secure bootstrapping, as the more privacy-aware this process should be, the more difficult and less inexpensive it becomes. A bootstrapping model is formalised in the smart metering context by using Aslan++, the Avantssar specification language. In this language, the messages among entities are modelled and analysed in order to find possible attacks. The proposed model lack attacks assuming no brute force attacks on cryptographic primitives and the secure storage of local keys.

Some privacy concerns have been addressed also at the infrastructure level by protecting wireless sensor networks that supervise the smart grid, or more concretely, the base station of such networks. The idea is to make it harder for attackers to find where the base station is by injecting fake traffic to homogenise the transmissions from a node to its neighbours.

The Electro mobility scenario presents several requirements. Concerning privacy, it is important to design solutions that prevent the Electric Vehicle Supply Equipment from learning the prices at which a given energy supplier is selling or buying electricity to a particular customer. In this direction, run time verification and enforcement will aid to develop a method to verify whether non-interference can capture confidentiality information.

¹ The authorisation requirement usually has the authentication requirement as a precondition, even when some of the research simply assume a traditional user-password login scheme.
Other requirements of this scenario include authentication, authorisation and audit. In this direction and in the context of secure charging of electric vehicles in public parking, a complete information system that fulfils these requirements is being developed. Authentication is implemented by a web service that receives public credentials about the person, the car or the battery of the car signed with the private credentials. Authorisation determines whether the person has permission to charge his car within a given context, and takes several policies specified in XACML, returning an authorisation decision. The audit service logs every action (charges, authentication attempts, etc) providing proof of integrity and origin.

Table 2 provides a summary of the relationships between requirements and related research for the smart grid scenario.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Related Research</th>
</tr>
</thead>
<tbody>
<tr>
<td>Privacy</td>
<td>Non-interference methods</td>
</tr>
<tr>
<td>Privacy at the infrastructure level</td>
<td>Homogeneous injection for sink privacy</td>
</tr>
<tr>
<td>Secure bootstrapping</td>
<td>Aslan++ formalisation of bootstrapping protocol</td>
</tr>
<tr>
<td>Secure logging / Authentication / Authorisation</td>
<td>Information system for public parkings</td>
</tr>
</tbody>
</table>

*Table 2 Smart Grid Requirements Mapping to Research*
4 Topics and Actions

4.1 Main Contributions of the Roadmap

The previous versions of the NESSoS roadmap [D4.1_II, D4.2_II] identified the main topics of research that the NESSoS consortium considered to be worth to be researched in the mid-term. Here we will describe them considering the progress on the State of the Art and the knowledge and understanding we have acquired during the last year of work in NESSoS.

We identified different groups of topics (see Sections 4.3 and 4.4): two main transversal topics and a series of enabling technologies and methodologies to enhance trustworthiness of the FI. On the top of these two groups there is a set of common properties to be always ensured in the FI.

4.2 Properties to be Ensured

Compliance, Privacy, Trust, and Identity protection. These topics are currently becoming more and more important due to the increase of interoperability and heterogeneity concerns within the FI. The increase in privacy protection should become a goal where research efforts should be directed. Particularly, it is worth to mention that location-privacy will be crucial in FI scenarios. NESSoS partners are doing some efforts towards this in a specific scenario that is used during the project, that is the smart grid scenario. In this sense, advances have been driven towards providing techniques for location-privacy for one of the most important elements of a SmartGrid, which are Wireless Sensor Networks. In particular, how to hide the location of the base station from attackers [RIO12]. Similar efforts have to be made in other scenarios and cases of the FI.

Also, the “right to be forgotten” is challenging. New schemas for identity management are required, since identity is emerging as an additional Internet layer with high impact within the FI. Recently, companies consider identity management as a handicap. This becomes more of an issue when companies tend to operate in the cloud and try to concentrate their operations in these settings. It is then when Identity-as-a Service could be desirable (IaaS) [CSA11]. This concept may comprise many services that provide entities with identity such as Policy Enforcement Points (PEP-as-a-service), Policy Decision Points (PDP-as-a-service), Policy Access Points (PAP-as-a-service). A similar approach could be useful for the FI scenarios. Identity of the many heterogeneous entities conforming the FI could be provided by the services mentioned earlier. Moreover, this involves the need for developing new or adapted mechanisms that enable managing users identities. This means that identity management mechanisms that are able to deal with the interoperability and heterogeneity of the FI are needed.

Trust should be considered as a property on the top of security mechanisms. In particular, trust management among boundaries will become of paramount importance in order to gain a tight integration of things, services and humans in an efficient and useful FI. At the moment, inter-boundaries trust management is not being considered by the research community but it will be essential for FI. Not only ways to assess reliability and good behaviour of entities in a
system are needed but also how this can be translated for a setting of different heterogeneous systems and entities. This calls for a better integration of trust and reputation models.

Note that we are emphasizing on compliance and privacy from the requirements engineering point of view in Section 4.4.1, including data protection.

4.3 Two Main Crossing Research Themes of Major Interests

4.3.1 Security Assurance

The main objective is to enable assurance in the development of software based services to ensure confidence about their security level. Our core goal is to incept a transverse methodology that enables to manage assurance throughout the service and software development life cycle (SDLC). The methodology is based on several strands: A first sub-domain covers early assurance at the level of requirements, architecture and design. A second sub-domain includes the more conventional and complementary assurance techniques based on implementation. Additionally, we consider also quantitative notions of security (e.g. security metrics) that have their own separate topic.

4.3.1.1 Assurance during the Early Stages of the SDLC

Early detection of security failures in Future Internet applications reduces development costs and improves assurance in the final system. This first strand aims at developing and applying assurance methods and techniques for early security verification. We consider several such methods:

- **Stepwise refinement.** One main area of research is step-wise refinement of security, by developing refinement strategies, from policies down to mechanisms, for complex protocols, services, and systems. This involves the definition of suitable service and component abstractions (e.g., secure channels) and the setup of the corresponding reasoning infrastructure (e.g., facts about such channels). We need to extend the refinement framework to cope with compositional techniques for model-based secure service development.

- **Formal model extraction.** There is an increasing demand of models and techniques to allow the formal analysis of secure services. The objective is to develop methodologies, based on formal mappings from constraint languages, to other formalisms for which theorem proving and/or (semi-) decision procedures are available, to support formal (and, when possible, automated) reasoning about the security policy models.

- **Algorithmic protocol verification.** We also study methods and develop tools for the automatic verification of FI protocols. This requires several extensions with respect to traditional protocol verification. For example, we need to deal with more complex primitives and security properties and we need to extend the standard Dolev-Yao attacker to include new attack possibilities. In this area, we can record substantial progress along several lines. In [ACRT11], we have studied attacker models that are relevant for Future Internet applications, namely, multiple non-communicating attackers and a class of attacks that arise from the XML representation of messages, called XML rewriting attacks. We give a decision procedure for these cases based on a generalized form of intruder constraints. In [FT11, FT12], we have proposed an integrated formal
language for specifying declarative distributed authorisation policies that communicate through insecure asynchronous media. We need to concentrate efforts towards increasing the expressiveness of our protocol models and extending the scope of the related methods and tools (e.g., to handle larger classes of authorisation policies).

4.3.1.2 Security Assurance in Implementation

Several complementary assurance techniques are available to ensure the security at the level of an implementation. Security policies can be implemented correctly by construction through a rigorous secure programming discipline. Internet applications can be validated through testing and debugging. It is possible to develop test data generation that specifically targets the integration of services, access control policies or specific attacks. Moreover, implementation can be monitored at run-time to ensure that they satisfy the required security properties.

Several activities are related to secure programming. This strand addresses a comprehensive solution for program verification, with a particular focus on session management in concurrent and distributed service compositions.

- Programming for verifiable security: The Future Internet will reinforce the prominence of highly distributed and concurrent applications, increasing the need for methodologies that prevent security holes that exploit the computational infrastructure. The objective is to develop a discipline of secure programming based on verifiable security, using program analyses and verification methods. We need to develop enforcement mechanisms that combine different verification methods and allow enforcing a wide range of policies (of information flow and resource usage).

- Support for secured session management: A specific issue for web service security is the proper processing of the message stack. For instance, a BPEL process may have several instances or sessions running concurrently and among these instances several might be accepting some incoming message at the same time. Hence for the run-time reliability of the system it is crucial to assign the right message to the right session. This dispatching operation needs to be carefully designed from security requirements.

In addition, an important set of research activities concerns testing and debugging. Testing is the process of finding faults in an application after it has been implemented, but before it is deployed to the final users. The main goal of testing techniques is to produce inputs that manifest faults in the application. Debugging is the process of giving an input that manifests a fault, gathering enough information about the fault so that it can be fixed. This typically requires understanding the fault and its causes.

Testing and debugging complement techniques that happen earlier in the SDLC such as early assurance techniques used during the design of the application and secure programming techniques used during its implementation. Even when early assurance and secure programming techniques have been used, testing is still needed to identify faults that may have been introduced during the implementation of the application or during the integration with external systems, while debugging is needed to fix those faults. Traditionally, testing and debugging have been slow and costly processes requiring a large amount of manpower. In the highly dynamic environment of the modern and Future Internet, where applications need to be quickly updated and new applications are constantly being developed, it is of fundamental
importance to develop automatic techniques and tools that reduce the time needed for testing and debugging, while assuring the safety and reliability of the deployed application.

Here are three aspects that although not comprehensive, are characteristic for service-oriented applications in the Future Internet.

- **Model-based penetration testing.** Penetration testing consists in evaluating the robustness of a (current or future) Internet application to well-known attacks. We need to investigate models to guide a penetration-testing tool for interactive testing, which should be automated as much as possible. Given an implementation, and a model for a common type of attack or risk, penetration testing will be useful to see if this implementation suffers from any vulnerability. We assume the models are transition systems. These inputs are fed into a penetration-testing tool to guide the user in interactive model-driven penetration testing.

- **Automatic generation of test data for web applications.** The input domain of a web application can be modelled by an XML schema. In that case a test data is an XML file conforming to the schema. The major issue for test generation is that there can be an infinite number of XML files that conform to the schema. We need to explore several approaches for automatic test data generation. A first approach is based on the partitioning of the input domain and then automatically to generate instances in each range of the partition. A second approach consists of sampling the input domain through combinatorial testing techniques.

- **Debugging.** The complexity of FI applications is reflected in the debugging process: source code may often be unavailable, non-deterministic behaviour makes it hard to reproduce the faulty behaviour, and the use of legacy languages without proper memory management prolongs the existence of dangling pointer vulnerabilities. Current research focuses on two different areas for early detection. Differential slicing is an efficient technique for debugging security vulnerabilities based on binaries [JCCM+11]. Early detection [CGMN12] is a runtime approach for finding and diagnosing use-after-free and double-free vulnerabilities. While previous work focuses on the creation of the vulnerability (i.e., the use of a dangling pointer), early detection shifts the focus to the creation of the dangling pointer(s) at the root of the vulnerability. Other aspects of debugging security vulnerabilities such as those related to distribution require further investigation.

4.3.1.3 Towards a Traverse Methodology

Security concerns are specified at the business-level but have to be implemented in complex distributed and adaptable systems of FI services. We need comprehensive assurance techniques in order to guarantee that security concerns are correctly taken into account through the whole SDLC. A chain of techniques and tools crossing the above areas is mandatory.

4.3.1.4 Quantitative Security for Assurance

Measurements are essential for the objective analysis of system security. Metrics can be used directly for computing risks (e.g., probability of threat occurrence) or indirectly (e.g., time between antivirus updates). Security metrics in Future Internet applications become increasingly important (although still very challenging). Service-oriented architectures demand
for assurance indicators that can explicitly measure the quality of protection of a service, and hence indicate the effective level of trustworthiness. These metrics should be assessable by and communicable to third parties. Clients want to make sure that their data is well protected when it is outsourced to other domains over which the client has only limited or no control. We need to define formal metrics and measurements that can be practically calculated. Compositional calculation approaches will be studied in this context. Many of the proposed metrics will be linked to and determined by the various techniques in the Engineering process. This topic emerged as a central one and it is also discussed more broadly in the rest of the document.

4.3.1.5 Threats

We envisage several threats if the proposed research agenda is not followed by the scientific community.

The increasing complexity of services demands not only the ability to build secure systems, but also to be able to prove and certify service security levels to third parties that will then utilize these services for their own purposes (new services). This is a complex task, requiring a chain of proofs and assurance cases and techniques during the whole SDLC. Indeed, it is important to stress that the assurance techniques associated with the different phases of the SDLC such as design verification, testing and debugging, and runtime verification and enforcement are complementary yet all necessary. This means that each of these techniques independently contributes towards improving security assurance. Therefore, we have to invest further research efforts in order to improve all of these techniques and make them fit to face the challenges posed by the Future Internet. More generally, software with vulnerabilities presents a threat not only to European competitiveness but also to European economy and critical infrastructures. We will witness situations where the vendor of software might not provide a patch in time to minimize the risks of attacks, which take advantage of the given vulnerability. Issues related to security assurance (together with other phases of secure service engineering) also address the crucial need to evaluate software coming from unknown or not well-known European sources.

4.3.2 Risk and Cost Aware Service Development Life Cycle (SDLC)

Risk management is the set of coordinated activities to direct and control an organisation regarding risk. It involves risk assessment and the identification of treatment options for unacceptable risks [ISO09]. Our research agenda for a risk and cost aware SDLC highlights challenges that are FI characteristic and for which traditional methods for risk identification, risk modelling and risk analysis provide little specialized support. Generally, there is the need for a methodology to support a risk and cost aware SDLC for secure FI services. Such a life cycle model aims to ensure the stakeholders’ return of investment when implementing security measures during various stages of the SDLC. We can envision several aspects of this kind of SDLC:

- **Process:** The methodology for risk and cost aware SDLC should be based on an incremental and iterative process that is accommodated to an incremental software development process. While the software development proceeds through incremental phases, the risk and cost analysis will undergo new iterations for each phase. As such the results of the initial risk and cost analyses will propagate through the software
development phases and become more refined. In order to support the propagation of analysis results through the phases of the SDLC one needs to develop methods and techniques for the refinement of risk analysis documentation. Such refinement can be obtained both by refining the risk models, e.g. by detailing the description of relevant threats and vulnerabilities, and by accordingly refining the system and service models.

- **Aggregation**: In order to accommodate to a modular software development process, as well as effectively handling the heterogeneous and compositional nature of Future Internet services that also involves the perspective and requirements of several competing stakeholders, one needs to focus on a modular approach to the analysis of risks and costs. In a compositional setting, also risks become compositional and should be analysed and understood as such. This requires, however, methods for aggregating the global risk level through risk composition. Building on recent work on modular risk assessment [BRS10], some progress has been made on component-based risk assessment [BRS12], where risks are identified, analysed and documented at the component level. The objective is to support composition of risk assessment results.

- **Evolution**: The setting of dynamic and evolving systems furthermore implies that risk models and sets of chosen mitigations are dynamic and evolving. Thus, in order to maintain risk and cost awareness, there is a need to continuously reassess risks and identify cost-efficient means for risk mitigation as a response to service or component substitution, evolving environments, evolving security requirements, new stakeholders emerging, etc., both during system development and operation. Based on the modular approach to risk and cost analysis one needs methods to manage the dynamics of risks. In particular, the process for risk and cost analysis is highly iterative by supporting updates of global analysis results through the analysis of only the relevant parts of the system as a response to local changes and evolvements. Recent work in this direction includes a generalisation of the ISO 31000 risk analysis process [ISO09] to provide specialized support for change management [LSS11]. In this work, method and language support is also provided to facilitate the modelling and traceability of changes from the target of analysis to the risk models, as well as the explicit modelling and analysis of the resulting changes to the risk picture. In [SS12], the need for tool support to handle change and evolution is discussed, and a prototype tool to automate traceability and detection of risk changes is presented.

- **Run-time**: The dynamic and evolving nature of the FI need to be handled also at run-time. For this purpose various means for risk and cost monitoring can be utilized to support continuous assessment of security risks and costs in which assessment results are continuously updated. One such research strand is on risk- and cost-based methods for run-time enforcement of access and usage control policies. For highly dynamic FI services and systems, attributes that are required for access decisions change frequently, and their value may be uncertain. In [KLM+10] an approach to risk-aware usage decision is proposed, where the uncertainty is used to assess the involved risk of access or usage requests. Recent progress has been made in [KLM+12] where cost is explicitly taken into account such that policy enforcement not only ensures an acceptable level of risk, but also justifiable cost. Another research strand is on dynamic risk assessment by
indicator monitoring. The use of indicators and metrics for security monitoring is well-established, but for organisations to utilize monitoring there is a need for methods to identify adequate indicators and to assess their validity with respect to what they intend to measure. Such a method is presented in [LRS12], and can be used to facilitate automated and dynamic updates of risk analysis results. Tool-support for this is discussed in previous work [LRS+11], where also an architectural pattern for enterprise level monitoring tools is presented.

- **Interaction:** The methodology of this strand spans the orthogonal activities of security requirement engineering, secure architecture and design, secure programming as well as assurance and the relation to each of these ingredients must be investigated. During security requirements engineering risk analysis facilitates the identification of relevant requirements. Furthermore, methods for risk and cost analysis offer support for the prioritisation and selection among requirements through e.g. the evaluation of trade-off between alternatives or the impact of priority changes on the overall level of risks and cost. In the identification of security mechanisms intended to fulfil the security requirements, risk and cost analysis can be utilized in selecting the most cost efficient mechanisms. The following architecture and design phase incorporates the security requirements into the system design. The risk and cost models resulting from the previous development phase can at this point be refined and elaborated to support the management of risks and costs in the design decisions. Moreover, applying cost metrics to design models and architecture descriptions allows early validation of cost estimates. Such cost metrics may also be used in combination with security metrics for the optimisation of the balance between risk and cost. The assurance techniques can therefore be utilized in providing input to risk and cost analysis, and in supporting the identification of means for risk mitigation based on security metrics.

### 4.3.2.1 Other Challenges

A formal foundation for risk management may serve as a basis for rigorous analysis and reasoning about risk by means of formal methods. The first challenge in risk analysis and management from a formal methods perspective is to find a formal semantic foundation of risk that is sufficiently general and expressive for the task. The second one, and possibly a harder challenge, is to utilize the formal foundation, together with the principles of formal methods, to define methods to support risk analysis and surrounding activities.

Risk methodology validation and integration are crucial issues as well. The former is required to analyse and compare objectively the strengths and weaknesses of different risk methodologies. The latter allows combining different risk frameworks in order to leverage the strengths of the frameworks and suppressing their limitations.

Some efforts have already been made on achieving higher levels of abstraction for software security and risk and cost-awareness, on dynamic risk monitoring and on common semantics for development and risk analysis models, in order to achieve traceability amongst both models. Nevertheless, other solutions are necessary. Run-time risk assessment is required since risk changes very frequently and static risk analysis performed once or twice a year is not enough. Run-time re-configurability of security based on risk management requires further research. Other areas include achieving modularity, identifying common causes of failures,
analysing inconsistencies and handling changes in risk analysis documentation, evolving risks, allowing composition and decomposition of risk models and dynamic contract-based risk sharing via certification.

4.3.2.2 Threats

We envisage several threats towards the development and provisioning of secure FI services should the scientific community at large fail to pursue the above-described agenda. In particular:

- Industry and service providers need to ensure properties such as compliance, privacy, trust and identity protection while making business. Without methods to ensure return on investment (ROI) in security, security may fail the competition with other business priorities. The ROI in security during the SDLC must therefore not only be ensured, but also clearly demonstrated at a business level.

- Risk and cost management in the FI setting must be supported by methods, techniques and tools in order to handle the highly dynamic, compositional and heterogeneous nature of FI. Traditional risk analysis methods are largely monolithic in the sense that systems are understood and analysed as a whole. Without a modular approach to risk analysis, a full analysis may have to be conducted anew whenever services or systems are recomposed. There is hence a risk that traditional methods become too heavy and costly for the FI setting, and that risk models and risk analysis results quickly become invalid and out of date.

- Related to the latter is the lack of methods and techniques to handle change and evolution. When systems and services change and evolve, so do risks and should be modelled and analysed as such. With traditional methods, previous analysis results may become out of date and risk analysis efforts can be in vain. Moreover, lack of appropriate means for run-time risk assessment of the frequently changing FI services may be highly insufficient for continuously keeping the security risk picture up to date.

4.4 Enabling Methodologies and Technologies to Enhance FI Trustworthiness

Besides the transversal technologies that we identified as relevant for the SDLC in this section we will describe the technologies and methodologies that are related to the SDLC and that are needed in order to achieve trustworthiness for FI.

4.4.1 Security Requirements Engineering

The main focus of this research strand is to enable the modelling of high-level requirements that can be expressed in terms of high-level concepts such as compliance, privacy or trust. These can be subsequently mapped into more specific requirements that refer to devices and to specific services. A key challenge is to support the participation of an unprecedented multitude of autonomous stakeholders and devices – probably one of the most distinguishing characteristics of the FI.

The need for assurance in the Future Internet demands a set of novel engineering methodologies to guarantee secure system behaviour and provide credible evidence that the
identified security requirements have been met from the point of view of all stakeholders. The
security requirements of Future Internet applications will differ considerably from those of
traditional applications. The reason is that Future Internet applications will not only be
distributed geographically, as traditional applications are, but they will also involve multiple
autonomous stakeholders, and may involve an array of physical devices such as smart cards,
phones or RFID sensors that are perpetually connected and transmit a variety of information
including identity, bank accounts or location. Some of these transactions might even happen
transparently to the user; for example, a person’s identity could be seamlessly communicated
by a personal device to the store she is entering to do the shopping. Addressing concerns about
identity theft, unauthorized credit card usage, unauthorized transmission of information by
third-party devices, trust or privacy are critical to the successful adoption of FI applications.

Service-orientation and the fragmentation of services (both key characteristics of FI
applications) imply that a multitude of stakeholders will be involved in a service composition
and each one will have its own security requirements. Hence, eliciting, reconciling, and
modelling all the stakeholders’ security requirements become a major challenge [BGG+04].
Multilateral Security Requirements Analysis techniques have been advocated in the state of the
art but substantial research is still needed. In this respect, agent-oriented and goal-oriented
approaches such as Secure Tropos [GMZ06] and KAOS [DLF93] are currently well recognized
as means to explicitly take the stakeholders’ perspective into account. These approaches will
represent a promising starting point but need to be uplifted in order to be able to cope with the
level of complexity put forward by FI applications. New requirements frameworks and
languages that take legislative constraints, as well as socio-technical and economic aspects into
account, are needed in order to manage data through multiple domains. Indeed, it is important
that security requirements are addressed from a higher-level perspective, e.g., in terms of the
actors’ relationships with each other. Unfortunately, most current requirements engineering
approaches consider security only at the technological level. In other words, current approaches
provide modelling and reasoning support for encryption, authentication, access control, non-
repudiation and similar requirements. However, they fail to capture the high-level requirements
of trust, privacy or compliance. It is also essential to analyse how security may impact on other
functional and non-functional requirements, including Quality of Service/Protection (QoS/P),
both at design-time and at run-time.

This picture is further complicated due to the vast number and the geographical spread of
smart devices stakeholders would deploy to meet their requirements. Sensor networks, RFID
tags, smart appliances that communicate not only with the user but also with their
manufacturers, are examples of such devices. Such deployments inherit security risks from the
classical Internet and, at the same time, create new and more complex security challenges.
Examples include illicit tracking of RFID tags (privacy violation) and cloning of data on RFID
tags (identity theft). Applications that involve such deployments typically cross organisation
boundaries.

Security requirements are only considered by a small fraction of software engineering
professionals in practice. A recent survey was conducted with 374 software professionals
[EYLL11]. The survey considers if eliciting, analysing and documenting of security
requirements occurs in real-world practice. The survey stated that around 59% of software
engineers consider security requirements just implicitly. Only 9% consider security
requirements explicitly and 31% did not consider them at all. These results further show that knowledge about security requirements engineering is sparse in industry and that guidelines and methods are required that help incorporate the available security knowledge into security requirements engineering in practice.

The need for easy access to knowledge about security requirements is recently addressed by different ontologies [SSCW12]. In addition, Fabian et al. [FGH+10] present a conceptual framework and terminology for security requirements engineering approaches. A study also shows how existing ontologies cover aspects of security requirements definition. It reveals that a large easy access common body of security knowledge is still not present. Hence, today software engineers have to extract pieces of knowledge from different sources.

A study on research in security requirements engineering for evolving software systems [NNY10] revealed challenges for security requirements engineering research. The study revealed that software evolution studies focus on the system-level and ignore the organisational level. The visions and goals of organisations are often the driving factor for software evolution. A deeper understanding of the development of such goals over time could reveal important knowledge. This knowledge can drive software lifecycles.

In addition, software systems often break, because of the changes caused by evolution. We need requirements engineering approaches that withstand these changes. Hence, requirements engineering methods should support the development of systems to make this possible.

Moreover, systems have to fulfil their security requirements even if their operating conditions change. Requirements engineering approaches are needed that consider the environment a system is in, and adapts its security requirements and solutions accordingly. Especially, considering FI applications that provide this elasticity becomes an increasingly important challenge.

Another challenging aspect in the security requirements engineering realm is related to how to evaluate the effectiveness of these methods. In fact, the effectiveness of security requirements methods is uncharted territory. We do not know to what degree the methods yield to the identification of security requirements. A significant number of methods have been proposed to elicit and analyse security requirements, but there are few empirical comparative evaluations to select a method over another. With the notable exception of [OS09, MP12] most works only report evaluations where the method’s effectiveness is assessed by the methods’ own inventors. It is thus really important to understand if these methods are effective and why they are or are not. The challenging question is not only which security requirements methods work, but also what makes them work. It is necessary create an empirical ground for comparing security requirements methods and help researchers to improve their methods to shorten the path for the adoption of these methods in real practice.

In light of the challenges and principles highlighted above the research objectives we proposed here are based on the evolution and the trends in the security requirements engineering area. They are detailed below:

- Definition of techniques for the identification of all stakeholders (including attackers), the elicitation of high-level security goals for all stakeholders, and the identification and resolution of conflicts among different stakeholder security goals.
• Refinement of security goals into more detailed security requirements for specific services and devices.
• Identification and resolution of conflicts between security requirements and other requirements (functional and other quality requirements).
• Transformation of a consolidated set of security requirements into security specifications.
• Provision of easy access to security requirements engineering knowledge for practitioners.
• Creation of an empirical ground to evaluate the effectiveness of security requirements methods.
• Creation of security requirements engineering approach for evolving and context aware systems.

The objectives listed above obviously remain generic by nature, one should bear in mind though that the forthcoming techniques and results will be applied to a versatile set of services, devices and stakeholder concerns.

4.4.1.1 Legal Compliance

Identifying relevant compliance regulations for a software system and aligning it to be compliant is a challenging task. The construction of software systems that meet compliance regulations, such as laws, is considered to be difficult, because it is a cross-disciplinary task in law and software and systems engineering [BMT87]. Otto et al. [OA07] conclude in their survey about research on laws in requirements engineering that there is a need for techniques to identify and analyse laws, and to derive requirements from laws. Massey et al. [MOH+10] conducted a case study that illustrated the challenges of interpreting legal texts and integrating these into security and privacy requirements for software systems. The authors concluded that software engineers lack the necessary training to interpret legal texts.

Pattern-based approaches capture the knowledge of domain experts for re-use. Beckers et al. [BFK+12] proposed a pattern-based approach for identifying and analysing laws. However, the identification and analysis of a relevant law alone is not sufficient for software engineers. They require a structured method that uses this approach to derive software requirements and further implementable software specifications. Different approaches exist that analyse the structure of laws and find different ways to integrate these into a given software engineering process. Breaux et al. [BA08] present a framework that covers analysing the structure of laws using a natural language pattern. This pattern helps to translate laws into a more structured representation. Siena et al. [SPS08] describe the differences between legal concepts and requirements. Álvarez et al. [AOP02] describe reusable legal requirements in natural language.

The following research objectives are the focus of compliance research:
• Identifying relevant laws for a given software engineering project, based upon the functional as well as quality requirements (e.g. security and privacy) of the system.
• Developing computer-aided or even automated approaches to identify relevant laws.
• Supporting the interpretation of legal texts and aligning these with the requirements of a software system.
• The mapping of legal texts to requirements is a challenge, because the legal domain and the software engineering domain use different terminologies.
• The increasing scale of software systems requires that laws of different countries are considered in software projects. For example, cloud-computing systems often store their data distributed in different countries.
• Providing training measures to teach software engineers how to evaluate whether their software system is compliant and how to interpret legal texts.

4.4.1.2 Privacy Requirements Engineering

Privacy concerns have increased recently, due to an increase in large-scale information systems that process or store personal information in FI applications. A number of guidelines for privacy are available. The Fair Information Practice Principles (or short FIPs) [OECD80] are widely accepted, which state that a person’s informed consent is required for the data that is collected. Collection should be limited for the task it is required for and data should be erased as soon as this is not the case anymore. The collector of the data shall keep the data secure and shall be held accountable for any violation of these principles. In the European Union, the EU Data Protection Directive, Directive 95/46/EC does not permit processing personal data at all, except when a specific legal basis explicitly allows it or when the individuals concerned consented prior to the data processing [EU95]. The U.S. have no central data protection law, but separate privacy laws, e.g., the Gramm-Leach-Bliley Act for financial information, the Health Insurance Portability and Accountability Act for medical information, and the Children’s Online Privacy Protection Act for data related to children [HSA08]. These legal guidelines must be implemented by any given software system for which the guidelines apply. However, in order to comply with these guidelines, the privacy requirements for a given software system have to be elicited. In addition, the privacy requirements of the users of a system might extend beyond the legal demands. These should also be considered, to make it acceptable to its users. For example, an internet-based distributed medical database will not be accepted by users if the privacy interests of the patients are not considered.

The need for privacy in the Future Internet demands a set of novel approaches to guarantee privacy protection for all stakeholders involved. The privacy requirements of these stakeholders have to be elicited and ensured. This problem is difficult to address in Future Internet applications like clouds, because of their complexity caused by the numerous stakeholders and technical systems involved. Hence, privacy requirements engineering approaches have to be able to take this complexity into account. For example, the PriS method [KKG08] elicits privacy requirements in the software design phase and models privacy requirements as abstract organisational goals. Deng et al. [DWS+11] generate a threat tree for privacy based upon the threat categories e.g. linkability. Spiekermann and Cranor [SC09] develop a framework for considering privacy in software systems. The authors execute a privacy requirements analysis via stating important stakeholder expectations. Their method also requires an analysis that checks if the system processes personal information and if this is necessary for a system. If not, these shall not be acquired by the system. The authors state that software engineers lack awareness for these issues.

Selecting appropriate controls to ensure privacy is difficult. Privacy controls can be chosen as mechanisms that are introduced into the software. For example, the method from Deng et al. maps privacy threats to parts of a system. The method supports the selection of privacy controls appropriate for that particular part of the system. Another approach is to change the system in
order to preserve privacy. For example, the PriS method focuses on processes in software. Privacy solutions result in changes of these processes. Hence, depending upon the preferred kind of solution, software engineers have to choose a fitting method.

The selection of privacy controls requires also knowledge about existing privacy controls. Danzies and Gürses investigated current privacy related technologies in order to classify their origin into a larger context of privacy [DG10]. The authors classified privacy enhancing technologies (PETs), e.g., anonymizers, and derived three areas of privacy research: privacy as confidentiality, privacy as control, and privacy as practice. Privacy as confidentiality considers privacy within an autonomous digital sphere. In this sphere data about persons is protected from unauthorized access. Privacy as control is about concealing personal information, but at the same time control what happens to it. Privacy as practice concerns the transparency of data flow. The focus is on the way personal data is collected, aggregated, analyzed and used. The transparency of these actions shall help people to make informed decisions about their personal data. Researchers have to develop engineering methods that support software engineers in choosing the right privacy control for a given problem.

Regarding the research approaches and challenges described above, we propose the following research objectives for the evolution of the privacy requirements engineering area:

• Identify a common abstraction level for discussing about privacy for requirements engineers, lawyers, cryptographers, etc.
• Develop computer-aided support for identifying privacy threats based upon requirements or architectures.
• Provide reasoning methods for choosing the right privacy enhancing technologies for a privacy requirement.
• Integrate privacy requirements engineering into existing software engineering processes to raise the awareness of software engineers for these issues.
• Support the different views of privacy in different legislations around the world.

4.4.1.3 Threats

There are several threats in the case the previous research strand is not properly addressed. In particular, business risks are increasingly related to operational risk and eventually to IT security risks. The lack of the capability to properly embed specific FI security requirements in the SDLC and the lack of analysis capability when considering conflicting multi-stakeholder interests will dramatically limit the effectiveness and impact of new FI services.

4.4.2 Secure Service Architecture and Design

By 2020 the Future Internet (FI) will be both laid out as public infrastructures and dynamically created by the objects connecting to one another [HNP09]. Consequently, FI services and applications will run open and dynamically in heterogeneous environments. Model-driven engineering provides (i) abstractions and tools to reason about functionality and security despite heterogeneous platforms, as well as (ii) generative techniques to tailor the security mechanisms to specific environments. In this section, we explore the challenges and opportunities triggered by FI characteristics for model-driven techniques.
The design phase of FI applications is a timely moment to enforce and reason about these security mechanisms, since by that phase one must have already grasped a thorough understanding of the application domain and of the requirements to be fulfilled. Meanwhile, the dynamic, open and heterogeneous nature of FI prevents from completely grasping all constraints and expectations about the application. The fundamental challenge is thus to leverage model-driven techniques to (i) reason about trade-offs between security and other concerns, at design time and at runtime, and (ii) to explore and capitalize diverse strategies to face unexpected situations.

4.4.2.1 Reasoning about Security in Multi-concern Design Models

Separation of concerns is an essential principle of software design that aims at managing the growing complexity of software intensive systems [GEL08, PAR72]. Since the early 2000’s this software engineering principle has been integrated in model-driven engineering, through a large research and tooling effort to support aspect-oriented modeling (AOM) [RGF06, WJE09]. The main motivation of AOM is to allow modeling different viewpoints separately (e.g., functionality and access control), analyze them separately and then compose them in a global system design that can be refined towards implementation. In this perspective model composition tools [FFR07] are particularly important for model-driven security design.

The principle of separation of concerns for model-driven security is currently developed in two major frameworks that were acknowledged to be influential papers, 10 years after, at the MODELS’12 conference: UMLsec [JUR02] and SecureUML [LBD02]. Both approaches extend UML to integrate access control and privacy modeling, for the design of secure web applications. With the emergence of the FI these approaches to separation of concerns need to be extended. Indeed, FI applications will be much more integrated with heterogeneous execution environments than current web applications. This means that, in addition to security and functionality, designers of FI applications will have to consider energetic concerns (since data acquisition or monitoring services will run on resource-constrained execution nodes) or volatility concerns. Designers will also have to reason at early stages about conflicting concerns such as CIA vs. usability [CRG05] or privacy vs. personalisation [AGK11, GEL08].

The emergence of novel concerns for the design of FI applications thus requires (i) formalisms to model these concerns; (ii) model composition mechanisms that integrate security with these concerns; and (iii) automated reasoning techniques that allow exploring trade-offs between concerns, e.g. using meta heuristic search algorithms. Novel approaches in NESSoS contribute to the state of the art on transformation verification to support the sound integration of security concerns [BEC12].

4.4.2.2 From Design Diversity to Model-driven Diversity Synthesis

Previous works have shown that the low level of diversity in software systems is a major risk for resilience and security [STA04, FSA97]. This relies on the observation that a vast majority of applications, even sensitive ones, run on a small number of operating systems, communication protocols or hardware infrastructures. Consequently, if attackers find a way to exploit a breach on one system, this knowledge can serve to attack a very large number of systems or to disseminate the attack through the network. In the face of this threat, the systems, security and distributed systems communities have investigated multiple strategies to introduce
diversity in software systems, to improve security. These solutions are inspired by the long tradition of design diversity for reliability [ALR04], and range from artificial immune systems [HOF00] to distributed algorithms that optimize diversity [ODS04, CKS08].

All existing techniques for introducing software diversity are essential for security. However, current approaches hardly consider the extremely dynamic and open nature of FI applications. Indeed, they all consider either a centralized architecture or a stable set of functionalities, which allow designing the diversification mechanisms before deployment. However, FI applications will be decentralized and constantly evolving, calling for novel mechanisms to handle dynamic software diversity. Here, model-driven engineering transformation and generative technologies, in combination with testing techniques, should be considered as the foundations for innovative adaptive design principles that explore and increase the global diversity in FI applications. The major challenge here lies in the production of meaningful diversity, at runtime, through automatic exploration techniques.

The shift from design diversity to model-driven runtime diversity synthesis for increased security in FI applications requires: (i) model-driven generative techniques to inject diversity in software modules; (ii) model-based reasoning mechanisms to select and propagate useful diversity; (iii) adaptive runtime mechanisms to establish diversity in a decentralized fashion. The DIVERSIFY FET project has started investigating these issues in 2013, in direct collaboration with ecologists who will provide expertise in biodiversity dynamics.

4.4.2.3 Support for Model-driven Security Dynamic Adaptation

The open and dynamic nature of FI applications and their environment prevents system architects and designers to foresee all situations an application will have to face. This leads to the emergence of innovative techniques to evolve the architecture and design models while the system is running, fading the boundary between design time and runtime [GHE11].

The support for runtime security policies will require adapting policy enforcement architectures to new architecture models that emerge for the Future Internet. As an example, the common architecture for access control enforcement, based on a PEP, PDP and PIP usually deployed on client-server architectures, hardly fits pervasive architectures that are highly distributed, decentralized and heterogeneous, and that include computation nodes that do not have enough resources to run a PEP, PDP and PIP. Consequently, it will be necessary to leverage the massively distributed monitoring capacities of pervasive architectures to devise novel security enforcement mechanisms. In certain cases these mechanisms should adopt an optimistic approach to security enforcement [POV99], relying on opportunistic audits rather than on systematic preventive checks.

These novel developments for secure design modelling, should leverage the recent advances in the ‘models@runtime’ paradigm, which consists in keeping an abstract representation of a running system in order to reason about changes and drive dynamic reconfiguration [MBJ09].

---

2 www.diversify-project.eu
4.4.2.4 Integrate Security Modeling in Domain-specific Modeling Languages

The increasing number of concerns integrated in FI applications leads to the emergence of multiple domain-specific languages. These languages allow stakeholders with heterogeneous backgrounds to model their concerns in the early development phases. This reduces the cognitive distance between the abstract formal concepts and domain experts’ knowledge, reduces the risk of errors in requirements elicitation and can drastically improve the quality of the final application. However, by opposition to UML, these new languages do not have a native support to express access control or privacy policies.

In order to face this novel situation, model-driven secure design needs novel abstractions to devise generic formalisms for the definition of security policies. The syntax of these formalisms should be adaptable to the domain-specific language, and the generic formalisms should offer ‘hooks’, i.e., domain-specific actions can be integrated in the analysis of policies. As an example, recent work integrated security concerns in a business modeling language to let project managers and company executives reason on security issues on models expressed in concepts they can apprehend [GOF12].

Other works include access control policy enforcement mechanisms generated automatically from high-level, requirements models. The policies need to be submitted to checks in order to ensure that security aspects being modeled are preserved in the code [BKM12]. This work is developed within the ASCENS EU project.

4.4.2.5 Reusable Architectural Know-how

Another research focus is on design patterns and on reusable architectural know-how. A design pattern is a general repeatable solution to a commonly occurring problem in software design. Design patterns, once identified, allow reuse of design solutions that have proved to be effective in the past, reducing costs and risks usually arising by uncertainty, leveraging risk and cost-awareness. There are large catalogues and surveys on security patterns, however the FI applications yet to come and the new scenarios enabled by FI need to extend and tailor these catalogues. In this context, the first step is studying the patterns currently available and, what is more important, to analyze the relationships amongst them, identifying those which may be useful for FI scenarios.

4.4.2.6 Threats

We distinguish two critical areas for the design and architecture of secure services:

• Concern interaction analysis. If we do not use effective methods and tools to deal with security concerns in design models the major threat we will face will be the increasing cost to deploy, fix and maintain security mechanisms. Indeed, security concerns are tightly related to other concerns such as functionality or human interfaces. If we can design abstract models for these concerns then, the cost to analyze interactions and fix the models until reaching a satisfactory trade-off between the concerns can be kept reasonable. On the other hand, if these analysis cannot be performed on abstract models, they are much more difficult to understand at the code level, and even more

3 www.ascens-ist.eu
difficult to fix and maintain, because of all the technical details that must be introduced in the implementation.

- Model-driven adaptation. Because of the highly dynamic nature of FI applications, reconfiguration and adaptation are essential mechanisms for the success of FI. Thus, we need rigorous and systematic techniques for composing models to analyze interactions before reconfiguration. The major threat if we cannot achieve safe and effective model composition will be limited abilities for adaptation and thus increased risks of failures in front of major changes in the environment.

### 4.4.3 Security Support in Programming Environments

Security Support in Programming Environments is not new; still it remains a grand challenge, especially in the context of FI services. Securing FI services is inherently a matter of secure software and systems. The context of FI services sets the scene in the sense that (1) specific service architectures will be used, that (2) new types of environments will be exploited, ranging from small embedded devices (‘things’) to service infrastructures and platform in the cloud, and (3) a broad range of programming technologies will be used to develop the actual software and systems.

The search for security support in programming environments has to take this context into account. The requirements and architectural blueprints that will be produced in earlier stages of the software engineering process cannot deliver the expected security value unless the programs (code) respect these security artefacts that have been produced in the preceding stages. This sets the stage for model driven security in which transformation of architecture and design artefacts is essential, as well as the verification of code compliance with various properties [BCE11]. Some of these properties have been embedded in the security specific elements of the software design; others may simply be high priority security requirements that have to be articulated, such as the appropriate treatment of concurrency control and the avoidance of race conditions in the code, as a typical FI service in the cloud may be deployed with extreme concurrency in mind.

Supporting security requirements at the programming code level requires a comprehensive approach. At least two essential facets must be covered:

- The service creation means must be improved and extended to deal with security needs. Service creation means aggregating as well as composing services from pre-existing building blocks (services and more traditional components), as well as programming new services from scratch using a state-of-the-art programming language. The service creation context will typically aim for techniques and technologies that support compile and build-time feedback. One could argue that security support for service creation must focus on and enable better static verification.

- The service execution support must be enhanced to deal with hooks and building blocks that facilitate effective security enforcement at run-time. Dependent on the needs and the state-of-the-art this may lead to interception and enforcement techniques that “simply” ensure that the application logic consistently interacts with underpinning security mechanisms such as authentication or audit services. Otherwise, the provisioning of the underpinning security mechanisms and services (e.g. supporting
mutual non repudiation, attribute based authorisation in a cloud platform etc.) will be required as well for many of the typical FI service environments.

It is crucial to improve upon these two facets in order to lift the current state of practice to a higher degree of quality. Having the right compilation tools will not only reduce the number of bugs and help find them quicker, but it will cut down on the attack surface of an application by avoiding common programming vulnerabilities. Additionally, if an attacker succeeds in exploiting a service, the monitoring tools and policies will be able to mitigate the attack by constraining the access of the attacker to the system.

In the remainder of this section, we further elaborate on the needs and the objectives of community wide research activities, in order to deal effectively with the grand challenge sketched above.

4.4.3.1 Middleware Aspects

The research community should re-investigate service-oriented middleware for the Future Internet, with a special emphasis on enabling deployment, access, discovery and composition of pervasive services offered by resource-constrained nodes. The most relevant ones are Quality-of-Service aware dynamic service discovery and composition, in particular accounting for properties related to security, privacy and trust. In order to ensure that published security properties of FI services are correct, monitoring business compositions must be done and analysed. Monitoring infrastructures for several platforms including Java and BPEL must be developed. Another important facet in this respect is information flow analysis for business process languages. The increasing usage of IT systems in practical business logic execution entails the need for high quality and reliability. Business workflows frequently act on behalf of multiple parties having potentially differing interests, thus malfunction can lead to the compromise of sensitive business data. Therefore, the analysis whether the business process conforms to corporate information security policies is of high priority. Note that contemporary languages and technologies lack this capability.

Regarding this, it may be worthwhile to explore up to which extent model-driven security engineering techniques and methodologies can be used in the context of secure business processes, where separation and binding of duties policies as well as access control policies constraint the execution of tasks by specific parties [VAL12].

4.4.3.2 Secure Service Programming

Many security vulnerabilities arise from programming errors that allow an exploit. Future Internet will further reinforce the prominence of highly distributed and concurrent applications, making it important to develop methodologies that ensure that no security hole arises from implementations that exploit the computational infrastructure allowed by Future Internet. The research community must further investigate advances over the state-of-the-art in fine-grained concurrency to enable highly concurrent services of the Future Internet, and will improve analysis and verification techniques to verify, among others, adherence to programming principles and best practices.
4.4.3.2.1 Verifiable Concurrency

Lock-free wait-free algorithms for common software abstractions (queues, bags, etc.) are one of the most effective approaches to exploit multi-core parallelism. These algorithms are hard to design and prove correct, error-prone to program, and challenging to debug. Their correctness is crucial to the correct behaviour of client programs. Research should now focus on building independently checkable proofs of the absence of common errors, including deadlock, race conditions, and non-serializable ability [JPS’08].

4.4.3.2.2 Adherence to Programming Principles and Best Practices

Programming support must include methods to ensure the adherence of a particular program to well-known programming principles or best practices in secure software development. Emphasis will be put on language extensions that guarantee adherence to best practices, and verified design patterns that can be used during development. Also, it is necessary to consider that new features in programming languages may result in new features in modelling languages. Moreover, new programming constructs will arise to deal with several security properties and include disciplined programming techniques.

The research community might investigate and re-visit methods from language-based security, in particular type systems, to enforce best-practises currently used in order to prevent cross-site scripting attacks and similar vulnerabilities associated with web-based distributed applications. Obviously, the logical rationales underlying such best practises must be articulated, enabling he development of type systems enforcing these practises directly – thus allowing users to deviate from rigid best practices while still maintaining security.

4.4.3.3 Platform Support for Security Enforcement

Future Internet applications span multiple trust domains, and the hybrid aggregation of content and functionality from different trust domains requires complex cross-domain security policies to be enforced, such as end-to-end information flow, cross-domain interactions and usage control. In effect, the security enforcement techniques that are triggered by built-in security services and by realistic in the FI setting, must address the challenge of complex interactions and of finely grained control [HMS06]. Research should therefore focus on enforcing cross-domain barriers in the interaction among different cross-domains, and on the enforcement of fine-grained security policies via execution monitoring.

Secure Cross-Domain Interactions

Web technology inherently embeds the concept of cross-domain references, and applications are isolated via the Same-Origin-Policy (SOP) in the browser. From a functional perspective, the SOP poses limitations on composability and cooperation of different applications, and from a security perspective, the SOP is not strong enough to achieve the appropriate application isolation. New technologies are required to on one hand enforce strictly defined policies, but on the other hand not hamper composability too much.

Finely grained execution monitoring

Trustworthy applications need run-time execution monitors that can provably enforce advanced security policies [GBJ06][BLW05] including fined-grained access control policies,
usage control policies and information flow policies [SM03]. These topics are clearly related also to the area of run time verification and enforcement.

**Supporting Security Assurance for FI Services**

Assurance will play a central role in the development of software-based services to provide confidence about the desired security level. Assurance must be treated in a holistic manner as an integral constituent of the development process, seamlessly informing and giving feedback at each stage of the software life cycle by checking that the related models and artefacts satisfy their functional and security requirements and constraints. Obviously the security support in programming environments that must be delivered will be essential to incept a transverse methodology that enables to manage assurance throughout the software and service development life cycle (SDLC). The assurance aspects are a key part of this roadmap (see Section 4.3.1).

**4.4.3.4 Threats**

The OWASP, project (www.owasp.org), top ten list for web application security, clearly shows how coding issues as injection, cross scripting and generally speaking wrong programming practices are the major issues to be tackled. Indeed, reliable programming environments and proper coding techniques are crucial to minimize the presence of exploitable vulnerabilities in software-based services. Lack of specific research activities in the previous topics poses a risk in the increase in errors in those programming environments, that will be also propagated to the final programmer code. Similarly, failing to recognise that new high level service execution languages introduce new potential threats (in addition to the level of system programs) as well as the need of middleware for run-time monitoring risks to contribute to expand the possibility of attacks on web services.

**4.4.4 Secure Service Composition and Adaptation**

Future Internet services and applications will be composed of several services (created and hosted by various organisations and providers), each with its own security characteristics. The business compositions are very dynamic in nature, and span multiple trust domains, resulting in a fragmentation of ownership of both services and content, and a complexity of implicit and explicit relations among the participants. Service composition support is required, in terms of the composition languages such as BPEL, as well is in terms of the underpinning in middleware platforms.

One of the challenges for the secure service composition is the need for new formalisms to specify service requests (properties of service compositions) and service capabilities, including their security policies, and tools to generate code for service compositions that are able to fulfil these requirements based on the available services. In addition to complying with the requested functional and quality-of-service characteristics, composition languages must support means to preserve at least the security policies of those services being composed.

As a matter of fact, dynamic adaptation will play a major role in FI applications to ease service composition, paying special attention to the semantic level adaptation, which is left aside in most of the nowadays proposals. Security contracts should be used during the whole life of software and will be exposed in the composition of services, not only in single services,
and their dynamic evolution should be managed. Furthermore, the existence of an open market for composable services with well-defined security properties is required, and service customisation should not come at the cost of security. As a consequence of the reasons mentioned above, service composition should be an easy, secure, and commonly performed task.

Given that the outcome of the composition of two secure services might not be a secure bigger service, it is required to assess the risk of a service composition. Also, it would be very interesting to have a test-bed for comparing Service Adaptation by Contract approaches. Other topics to address include quantifying and control information sharing in service composition, and developing automatic risk reduction capabilities when recruiting services for compositions. Unifying the different Aspect-Oriented Modelling (AOM) techniques for model composition poses a gap. Also, the research community needs to consider the cases where only partial or inadequate information on the services is available, in such a way that the composition will have to find compliant candidates or uncover the underspecified functionality.

In order to achieve the integration and interoperability of services, some ongoing solutions are based on semantic annotations and secure adaptation contracts, as well as on decentralized secure composition and on distributed component models. However, further solutions are required. First, services and components need to be more open, with clearer open interfaces and need to be easily accessible from known repositories. Moreover, it is required to research on how to efficiently compose security measures.

4.4.4.1 Threats

Service composition is one of the main distinguishing features of the Future Internet service paradigm. The capability to achieve trustworthy secure composition is thus paramount. Failing in constructing such a framework would harm the whole concept. Building secure services, that cannot be further composed is an inherent obstacle that need to be removed and would make fragile the whole architecture.

4.4.5 Run time Verification and Enforcement

Run-time verification complements programming-level verification and testing in order to provide the assurance that the latter cannot always deliver, be it for scientific and technological reasons, be it for reasons of organisational complexity. The latter may frequently occur in a multi-organisational context, typical for service compositions in Future Internet. We need to research on approaches for run-time monitoring of data flow, as well as technologies for privacy-preserving usage control.

- Run-time monitoring of data flow. Electronically and autonomously executed business logic plays a crucial role in today’s practice. Since these systems may possibly have access to sensitive data of different parties with potentially contradicting interests, information flow policies may need to be enforced. Lately, it has been shown that information flow controlling run-time monitors can assure the same level of termination insensitive non-interference as the original Denning-style static checking procedure, while providing the advantage of being able to be more permissive. We need to develop the theoretical foundations of a run-time monitor, which is suited for the enforcement of
information flow policies in an environment, where complex hierarchic data is manipulated (such as for instance in BPEL).

- Monitoring Usage Control Properties. Usage control (e.g. see [LMM10]) extends traditional access control with policies and mechanisms to control the usage of data after it has been accessed. Therefore, usage control addresses central privacy-related security issues, which are raised by Future Internet applications and for which only partial solutions exist nowadays. Advances on the state of the art should be made in observing and controlling the usage of sensitive data in Future Internet applications. We need to develop methods that monitor the use of data and ensure that usage conforms to the intended purposes for which the data was collected. Based on previous work on usage control and monitoring of security policies, we need to adapt and extend run-time verification techniques for checking the adherence of data consumers to usage control policies. Furthermore, we need to study the integration of these monitors into Future Internet applications that report on or, where possible, prevent the misuse of sensitive data.

Even though there has been already substantial progress in these directions [BHFZ11a, BHFZ11b, BJKZ12] there are number of remaining limitations that should be addressed by further research. One current limitation in monitoring security policies is the amount of data that can be efficiently processed with the existing monitoring techniques. We need to consider specific techniques such as parallel processing to address this issue. Another limitation concerns the handling of incomplete or inconsistent input data. Moreover, security policies are typically formulated at high levels of abstraction whereas the monitors observe low-level system events that are scattered to different network nodes or to different layers of the system stack. The management of distributed enforcing mechanisms also deserves further study.

4.4.5.1 Threats

Run-time verification and enforcement offer the possibility to overcome some limitations of other security techniques such as static analysis and permits continuous and fine-grain enforcement of policies. Lack of these technologies will limit the control capabilities that user may exercise on systems/services/data, as well as harm the possibility to enable compliance techniques with legislations, organisational policies, and business rules.

4.4.6 User-Centric Security

The growth of the FI is making security management on the user side more complex since they have to be more situational-security aware, with the increased knowledge on the threats and risks they are exposed to. A number of security-related decisions, that a user is exposed to, is also increasing. In parallel, the so-called user empowerment is an important trend in several security areas, such as for example identity management or transparency enhancing.

Challenges include, for example, how to present clear information to the user about changes in security contexts and concerns. At the same time, usability, privacy controls or security configuration are increasingly context-aware. There are already projects and papers dealing with context-driven security and the issue of taking the burden of security responsibility a step away from the user [FIA]. Further advances could involve the use of semantic technologies adopted for a wide range of security mechanisms, policies and solutions,
in addition to a visual representation of security states for awareness and analysis. The Posecco⁴ project lists a number of publications relevant for security policy configuration or automated conflict resolution.

There are also notable efforts related to new generation of RBAC (role-based access control) and enforcing rules that are dependent on runtime parameters. Location-aware RBAC can be used for example to implement location dependent access control and also other security enhancing solutions on the Future Internet devices. Purpose based access control allows users using some data for a certain purpose with conditions. Extensible context-aware security mechanisms, such as access control, do not help only users, but also enable administrators to specify more precise and fine-grain authorisation polices for any application. However, further research is needed in contextual feature models as a means to explicitly represent and reason about the interplay between variables in one or different contexts.

Location, WiFi and Bluetooth traces, for example, provide rich context information. They can be used for predicting the movement of users, to identify meaningful places like home and work, identify “friends” or newcomers in interaction patterns, the degree of familiarity etc. Mobile devices could utilize user behavior to identify the users and make security decisions on their behalf, such as for example selection of one out of many authentication methods.

Privacy settings could also profit from similar technologies, while user-controllable policy learning could be applied where the user and system have to refine a common or default policy.

4.4.6.1 Threats

If we lose the opportunity to perform research in this direction, users would be overwhelmed with complexity of security configurations, choice of mechanisms etc. User trust could decrease while it would allow the spread either of services with limited security mechanisms, thus attack-prone, or disincentive user from using services altogether.

4.4.7 Security Management

The horizontal activities in the area of full life cycle support for secure services are predominantly addressing the construction, and to a lesser extent the potential evolution and adaptation of secure services for the FI. It is clear that such secure services –especially the security features and related subsystems - should be supported with appropriate monitoring and management support in order to observe the “quality of protection” in production systems at run time, and in order to implement the necessary measures for dealing with new threats and attacks, and possibly also with security incidents that require modification of the service implementation and/or its deployment environment. This type of activity (security monitoring and management) is not new in the domain of monitoring security infrastructures and secure systems, yet it is still limited in the space of service provisioning and deployment.

It should be noticed that research on risk management and assurance, as sketched in earlier sections, would be instrumental to this additional challenge. Security monitoring and management for new services will be essential to limit and control the total cost of ownership in the corresponding services business. This topic should also be considered in light of the

“autonomic security” challenge, which may offer additional advantages that contribute to the business case of securing FI services.

4.4.7.1 Threats

Security management is always a main concern. The new FI services demand for new management techniques to ease system and services administrator tasks, in particular during security incidents, allowing them to manage and maintain secure systems. In particular, new research has to focus on building a security management infrastructure for common (non-security) services.

4.4.8 Autonomic Security

In the future, due to a high number of events, devices, users, services etc., automated customisation at run-time of specific security mechanisms will be of paramount importance, although it could lead to new potential attacks in which the attacker tries to put down this automated reconfiguration mechanism. New techniques and methods for automation of secure service engineering would be desirable that allow for a risk reduction in the engineering process as well as a cost saving for companies when adopting them.

Autonomic security assumes security decisions as autonomous and spontaneous acts of the system, considering the possibility to take appropriate actions, based on self-capabilities, as self-monitoring and self-protection. These properties can benefit from the entrance of such big data explosion into the IT mainstream. Every FI component is able to generate a stream of information that is both valuable and ever changing. It is becoming insufficient to simply store the data for later analysis and modelling. Large data stream processing is becoming a commodity. Further research is however needed to see how this can be used in the area of autonomic security.

Another area where interesting research is being developed is trustworthiness of data used for decision making in autonomic security.

In this ambit, predictive analysis of security problems that can be used in order to anticipate and rely on a good decision support is challenging. The key issues for this are twofold, creating a smart reasoner that makes rapid and relevant reconfiguration decisions, and deciding which data is really valuable for feeding the reasoner.

Secure dynamic adaptive architectures is another challenging field, more specifically how to integrate security concerns in these adaptive architectures, and how to consider specific features of FI (e.g., monitoring geo-localisation to adapt security according to the location of a mobile device). Understanding the effect of architecture reconfiguration on the application is a gap to be filled, being required verification based on concern interaction analysis.

Solutions should focus on new suitable monitoring mechanisms to feed the reasoners. In order to tackle evolution of FI environments, adaptive configuration of policies and countermeasures, as well as dynamicity of mechanisms to respond to vulnerabilities, are required. Enforcement gateways with reacting components could take a more relevant role as well. Establishment of security contexts under which a service executes are also relevant, since reconfiguration decisions will be based on these contexts. The research directions in this area are twofold: development of contextual frameworks for security, and verified reusable components for certain contexts.
4.4.8.1 Threats

The main risk of not having autonomic security is to fall again in the well-known problem of having security detached from system development. In that case, security is just considered as an afterthought, with all the costs deriving from system re-design/development and deployment. As a matter of fact, considering autonomic security helps to consider security as built-in aspects of any ICT product. However, it remains to be seen what the impact of autonomous systems will have on the security. Depending on the capabilities of the system, it must be ensured that the security of a system can only be strengthened by the autonomic framework, in order not to add new potential attack vectors.

4.4.9 Quantitative Aspects of Security

Quantitative security entails that the claims about the security of a software artefact (and the quality of the software methods used to produce it) can be backed up by objective, numerical evidence.

The quantitative aspects of security provide support for the properties mentioned in Section 4.2 and the topics discussed in Section 4.4. In particular, quantitative assessment of security becomes essential for assurance and risk and cost analysis and estimation. In this latter point, security economics play an important role as an emerging field of research [AND01]. However, there are other topics that, even though at first sight could seem unrelated to this issue, could greatly benefit from this area of research. We are referring to usability and user security awareness [ING10]. The former would benefit as quantitative assessment supports numerically studying the interaction of users with security and privacy policies, whereas the latter assists users to become aware of, for instance, how much privacy is being leaked while using a certain application. This latter information could be gathered by applying formal quantification of security properties [BAC09].

Other areas that may use these quantitative aspects include security metrics, and concretely the prediction of software vulnerabilities [SMW11], and empirical methods [YSK12] where it would be possible, for example, to quantitatively analyse the interaction between developers and a secure programming standard.

Evidence-based research in secure software engineering is becoming of strategic importance for many funding agencies. For instance, the "Science of Security" initiative has emerged in the US, fostering empirical and formal methods in the field of security. In this context, the NSA has awarded three top US universities (CMU, NC State, Urbana-Champaign) with a large grant (http://www.iti.illinois.edu/research/projects/science-security-sos-lab/). In Europe, the 7th Framework Programme was calling for projects on "metrics and tools for quantitative security assessment and predictive security" in its Trustworthy ICT Objective (ICT-2011.1.4).

Quantitative security is an inter-disciplinary research area that requires the pooling of expertise from security, software engineering, artificial intelligence, formal methods, social and cognitive sciences, as well as economics. This strategic research line need to be fostered in Europe in a more substantial and coordinated way.
4.4.9.1 Threats

Quantitative Security provides a more accurate way for users and developers to understand whether the applications they are using work in the expected way. Failing to research in this area will mean a more vague assessment on the use of software and applications.
5 Conclusion

This deliverable presents the final version of the NESSoS roadmap. Its contents are an update of the previous releases of the roadmap delivered during the first two years of the project. This final version has been mainly updated with the comments and suggestions from the NaLAB members and the NESSoS community.

The NESSoS roadmap aims to establish the research lines to be considered in the area of secure service engineering in the context of FI. In this new paradigm new security challenges will arise for services and applications. This means that new solutions, mechanisms and technologies must be developed in order to provide security to all of them. Thus, the NESSoS consortium has proposed along the three editions of the roadmap the topics that they considered are needed for research in secure service engineering in the future. The roadmap has been supported by the NESSoS community in several stages: in the first place by the NESSoS IAB and associate members and during the last year by the members of the NaLAB. The document has been a public document during this last year and has received comments from the community interested in secure service engineering.

In the future we expect this document to be used by the secure service engineering community and to drive the research agenda in the area.
Contributors

We list here the contributors of this document (cumulative also considering the previous versions):

Martin Abadi
Mohammed Achemlal
Benoit Baudry
Kristian Beckers
Marianne Busch
Manuel Clavel
Jorge Cuellar
Lieven Desmet
Marina Egea
Sandro Etalle
Carmen Fernandez
Dieter Gollmann
Maritta Heisel
Paola Inverardi
Erno Jeges
Wouter Joosen
Jan Jürjens
Raimo Kantola
Sokratis Katsikas
Nora Koch
Yves Le Traon
Javier Lopez
Mass Lund
Fabio Martinelli
Fabio Massacci
Ron van der Meyden
Francisco Moyano
Bashar Nuseibeh
Federica Paci
Aljosa Pasic
Pieter Philippaerts
Alexander Pretschner
Peter Ryan
Pierangela Samarati
Riccardo Scandariato
Bjønar Solhaug
Christoph Sprenger
References

[BAC09] M. Backes, B. Koepf, A. Rybalchenko, Automatic Discovery and Quantification of Information Leaks (Oakland '09), 2009.


[D4.1_II] Engineering Secure Future Internet Services: A Research Manifesto and Agenda from the NESSoS Community.

[D4.2_II] Engineering Secure Future Internet Services: A Research Manifesto and Agenda from the NESSoS Community.

[D11.2] Selection and Documentation of the Two Major Application Case Studies.


