2008 EC-ERCIM Seminar on ICT Security: "Engineering Secure Complex Software Systems and Services"

Brussels, 16 October 2008

Seminar Report

27 November 2008

Table of Contents

1. SCOPE, MOTIVATION AND GOALS

ERCIM (the European Research Consortium on Informatics and Mathematics) and the European Commission jointly organised a Strategic Seminar on “Engineering Secure Complex Software Systems and Services”. The seminar was held in Brussels on October 16th 2008 and is the result of an effort of ERCIM, its Security and Trust Management Working Group, and the European Commission's s DG INFSO Unit F5 “Security”.

The Strategic Seminar aimed at collecting the relevant academic and industrial expertise in secure software engineering and at linking it with industry's best practices in the field. Indeed, in the coming years, security issues related to the continuously evolving and complex nature of the “Internet of the Future” and ICT in general will even grow in importance. The need for assurance of software systems and services demands a set of novel engineering methodologies, tools and techniques in order to ensure secure system behaviour as well as provide justifiable evidence for this status. Security engineering then needs to be effective and cost-efficient, and exploit the rich potential of reducing the design and time-to market costs. There is clearly the urgency and, actually, the opportunity for exploiting synergies of advanced research approaches with industrial best practices in order to reduce the gap between theory and practice in this vital sector of the European scientific, technological and economical scene.

The specific objectives of the Seminar were as follows:

– To present the best practices applied in industry for secure software and system development from several perspectives, including the identification of the main obstacles for achieving robust software assurance processes;

– To discuss latest progress on key research and development initiatives in engineering secure complex software systems and services and in achieving ICT system-level assurance;

– To encourage the dialogue between scientists and industrial players from the field with a view to promoting collaboration; in particular, to discuss the balance between rigorous scientific approaches aiming at achieving provably secure systems and cost-benefit considerations;

– To identify future key research challenges to address in the field, in particular in the context of the evolution towards the Future Internet.

The Seminar comprised sessions on:

1. Industrial best practices and perspectives;
2. Research advances and perspectives and
3. Proposals for the way forward.

In the sequel, the main findings of each panel are described and the way forward is outlined in order to sustain and progress the research activities at the European level in this very relevant topic. The agenda, individual presentations made and list of participants are available at http://www.ercim.org/activity/strategic_seminar.

2. PANEL 1: INDUSTRIAL BEST PRACTICES AND PERSPECTIVES

In this session, the Panellists presented experiences from industrial practices in use for engineering secure software systems and services and discussed initiatives that would help moving ahead the industrial agenda on secure software engineering and software assurance.

More specifically, the Panellists addressed the following subjects: Industrial *best practices* in the field and *IT frameworks, models and tools* required for improving the development of secure software throughout its lifecycle; Creating a sound *business case* for attracting more investments in security; Promoting *software assurance* and putting in place independent measurability and testing procedures for auditing and security compliance purposes; Dealing with the increasing complexity of IT systems; And, *education, professional training and awareness* initiatives.

Best practices In the last few years, there is a growing industrial interest on secure software engineering. For example, many large software companies that are operating in rapidly changing global markets need to quickly customise and adapt their services in a secure way to ever-changing user needs and comply with increasing legal requirements that differ from country to country. To deal with these challenges, software companies are continuously developing and improving their procedures, controls and tools for building and assuring security in their software systems and services. Other industrial sectors like banking or telecommunications are also striving for improving and refining the security of their business and IT processes, and the security of the software they use or produce.

The exact nature and extent to which security must be achieved varies across sectors, industries and organisations depending on the criticality, openness and/or safety requirements. Actually, a shift is observed in a number of directions: Some embedded or safety critical systems start to use more flexible software components, which have to be developed in a shorter and more cost effective lifecycle. In addition, more "consumer-oriented" software components and systems are increasingly being (re)used in a different context, including some critical systems (e.g. browser and plug-ins for e-banking).

A rich set of best practices is thus around in terms of documents and guidelines that ask for strict development process control, supervision, or review. Recently, joint corporate initiatives on secure software were also launched. Examples include SAFECode[1], the UK special interest group on secure software development[2], and others. They clearly demonstrate the great interest that major industrial players and private and public organisations have in cooperating in this field by sharing and promoting pragmatic approaches and proven software assurance practices.

Automated support for enforcement of best practices and the ability to reason about the business impact of security are key issues to manage security related efforts in an economically feasible way. Thus, further progress in the field would require stakeholders to jointly embark on new initiatives aiming at: (i) developing adequate IT frameworks, models and tools (ii) investing into the "economics" of security; (iii) building appropriate risk management and risk assessment frameworks; and, (iv) increasing the levels of awareness, education and training in secure software.

Novel IT frameworks, models and tools during all the phases of the software lifecycle Robust IT frameworks, methodologies, processes, models and tools that enforce best practices are fundamental to capture security requirements and ensure and rigorously demonstrate their validity in the software. Industry is lacking a common framework for developing and delivering secure systems. Software security should be an integral part of every phase of the software lifecycle, from design and development to deployment, integration, operations, monitoring and auditing. All phases need to be taken into account, in particular long support and maintenance phases. For example, a software system may be used for many years through successive upgrades. It is fundamental to capture rigorously the coverage and requirements in terms of 'security' – for instance, which assets to protect from which risks – and assess whether the software system is able to meet those needs and evolve with future requests for change.

The existence of common IT development and execution frameworks enforces the use of best practices and fosters collaborative work towards their further improvement for achieving higher levels of secure software.

Formalising and describing how the many possible processes and their security requirements have to be organised into an application or system is essential to software industry. Good models to use are 'a must' for guiding the pursuit of a balanced conceptual solution. Modelling should provide the right abstract schemes to make possible the description, explanation, comparison, and assessment of alternative scenarios and thereby for achieving a balanced secure software solution.

Organisations are realising how crucial it is for their business to get the appropriate IT tools that support security in the software they produce or use. To the extent possible, such tools need to be platform- and programming language-agnostic. While researchers continue to debate the specific role and contribution of

[1] http://www.safecode.org/
[2] http://www.ktn.qinetiq-tim.net/groups.php?page=gr_securesoft

security-based languages versus formal methods versus tools, industry requires tools that encapsulate specialised knowledge by translating underlying theoretical foundations into concrete secure software development practices. Such tools have to be well integrated into development environments and be easy to use by non-experts.

Creating the business case for security Despite this accrued interest of industry on secure software practices, overall, IT security has to compete with several other investment priorities. For example, a restricted budget needs to be distributed among security and other non-functional system requirements like performance, quality or usability. While the value and underlying development cost for addressing the latter is never questioned because of immediate impact on customers' business operation, the value of security and its return on investment (ROI) are always an issue and need to be clearly demonstrated since visible effects only occur in case of an incident. With squeezing IT budgets and ever-shorter times to market, how much do managers need to spend on IT security to achieve *enough security* and when *secure* is *secure enough*?

There is a need to research the construction of business cases for secure software throughout the whole lifecycle of secure software products, linking expected cost benefits, (adverse) business impact and risk mitigation to different levels of investment. Understanding the value (tangible and intangible) that investments on secure software can add through the product value chain is vital for business and IT managers taking decisions on spending money on security:

– Managers need to understand how much risk their company is ready to take for a given threat and manage that risk accordingly. In particular, they need to understand how serious a threat is, how much it would affect the value of the assets the threat is targeting and how much impact it will have on the company's business but also on the corporate image (in case of faulty products, vulnerable services or poor service provisioning). Consequently, companies could better align business with security requirements.

– There is also a risk of substantial economic losses when companies do not develop software with built-in security while their competitors move onto that direction (within a context of increasing globalisation of software development).

In many companies, there is a huge gap between IT architects, security people and the business part of the organization. This disparity creates severe and costly obstacles in day-to-day operations. It may also affect strategic decisions that relate to mergers and acquisitions or when designing and controlling outsourced business processes. For example, nowadays, financial institutions outsource vast parts of their IT infrastructure, services and operations. Then it is crucial for all involved business and IT people to speak the same 'security' language as well as maintain a tight control on all security aspects related to the IT components outsourced to those third parties.

... While dealing with assurance, measurability and testing Understanding the value of security and assessing and managing risks implies putting in place an appropriate set of "controls" at different levels, business, technology or processes. Such a control framework would allow prevention of vulnerabilities and monitoring *compliance* with internal or external security requirements, including legal compliance. That requires, however, putting in place an appropriate set of independent *measurement and testing procedures* as well as *metrics* for collecting data, auditing performance and, ultimately, proving/ensuring security by measuring it.

Measurability, security monitoring and testing procedures need to be put in place at every phase of the software lifecycle from analysis and design to development and production. Common criteria are useful but rather heavy to use and suffer from inflexibility with respect to system or product evolution. Commonly accepted certification procedures for secure software are thus needed, together with commonly agreed adequate metrics, measurement procedures, benchmarks and empirical data.

… and with increasing levels of complexity of software systems Presently, the complexity is rapidly increasing when moving from the secure engineering of isolated application components to that of software systems that mix various infrastructure resources with application functionalities. Such software systems are usually built incrementally, on top of legacy systems, resulting in "systems of systems" with functionality often different from what their underlying components were designed for. Their scope becomes thus much larger. Moreover, they increasingly rely on real-time dynamic composition involving third-party software components and services. Under these circumstances, achieving secure systems and secure software products is a huge challenge and key business success factor.

Business organisations need to have the right IT tools to enable them to create sustainable value including dealing with the security dimension of complex and critical software programmes and systems as well as of critical equipment controlled by software.

Promoting education and awareness, the spine of software systems security Software security is an integral component of every phase in the software development lifecycle and demands security conscious and well-educated software architects and software developers. There is no liability though on software companies for likely damage they may cause due to software vulnerabilities of their products.

Education and training for developing secure software is costly. Present awareness initiatives and investments on higher-level education, professional and on-the-job training are insufficient, because they mainly focus on secure IT setup, operations and management rather than development aspects. Worldwide-recognised certifications on secure software engineering for individuals and organisations could boost the number of software specialists in secure software. First initiatives on secure software certification are taking off. Overall, though, more awareness creation initiatives are needed, which would stress the importance of secure software within senior management, line managers, software architects, programmers and users.

3. Panel 2: Research Advances and Perspectives

During this session, the Panellists discussed several promising research directions for engineering secure complex software systems. They also analyzed the limitations of current technologies and outlined possible links with industrial best practices in the field.

More specifically, the Panellists addressed the following topics: Research in security requirements engineering, as a crucial building block of the software development process; the creation of a set of model-based techniques and automated tools for the development of complex secure software systems; the design and implementation of methods based on programming language techniques for secure coding and programming; the recent advances on methodologies and tools for the verification and validation of specifications and code; and finally, the role of risk in the creation of secure complex “systems of systems”.

Security requirements engineering Several security weaknesses originate in the incomplete or conflicting nature of security requirements of software code. In fact, managing security requirements during the whole software development lifecycle is a challenging task. This step is often overlooked or performed by non-experts and is thus frequently resulting in a mismatch between the interpretation of these requirements during the other software development phases and the original intent.

Specific expertise, methods and tools should be devoted to this task. A step-by-step refinement procedure (e.g. model-based requirements design) would help to improve the process from requirements elicitation to analysis. As for other phases of the software development process lifecycle, automated tools are needed to assist the security requirements engineer and keep track of them during the subsequent steps. Also, mechanisms able to pass from negative-form requirements (common in security) to more operational ones (as for functional requirements) should be envisaged. Some initial efforts on extending UML with security requirements are on place and should be further investigated.

As a whole, security requirements engineering is an area where progress is possible and potentially useful in order to answer common software industry needs, including considering those which will arise from the evolution towards the Future Internet.

Models for Secure Software Engineering The software development process needs several models to deal with domain specific aspects. Each model is useful for the designer to investigate certain features and identify the correct security solutions to adopt. The models are then refined / translated from high level to lower level ones. During these model transformation steps, one needs to ensure that the security of the final product is kept across the different layers. For example, the interactions of a software system with its environment and their specific security dimensions need to be appropriately modelled, analyzed and considered in the secure system design and be integrated in the software engineering process. For this, specific techniques such as model-driven design could be applied. Other related developments worth considering include security patterns (cf. also architecture patterns) already deployed in several projects as well as case modelling and analysis of “uses” and “abuses”.

Process description and model checking techniques could be used to validate specific solutions at a given design stage, e.g. for validating requirements. Techniques for design should at least involve component-based approaches in order to allow modular verification. In fact, compositionality is a major security challenge that is related with the increasing complexity of ICT systems and their scalability. Security properties are often not compositional though ad-hoc techniques can be found on a case-by-case basis. In addition, when dealing with system composition, system boundaries/interfaces are usually weak links that need to be carefully taken into account in order to avoid to introduce further security holes. Abstraction/model transformation/code generation methods have also been mentioned here.

Dynamic change of systems and code needs to be addressed, especially for systems that are maintained, functionally extended and operated for several years. The capability to cope with dynamic evolution of system functionalities should also be considered when dealing with system validation and certification. These processes are indeed time and resource consuming.

When applicable, formal methods seem to be able to guarantee an increased robustness of software. Nevertheless, the cost of their application and in particular the bulk of knowledge engineers that need to use them is yet one of the main obstacles to their deployment at a full industrial scale. Indeed, one of the research directions with major impact would be to embed formal methods in automated development tools in a transparent way with regard to the user. These tools should be user friendly and able to support also to the decision making process.

Methods for measuring the trustworthiness of the software systems is yet another area of importance for industry, where major research efforts are necessary. Metrics for software security do not seem currently able to cope with *absolute* security. Researchers seem more oriented towards the realistic approach of measuring different characteristics and properties of software that can be interpreted as indicating the *relative* security of that software, when compared either with itself operating under different conditions, or with other comparable software operating under the same or different conditions.

Language-based Security Embedding security mechanisms inside the programming environments is one effective way to reduce software vulnerabilities. Language-based security techniques and specific type systems allow indeed verifying at compilation time the absence of (certain) vulnerabilities. They also constrain the run-time execution of applications in order to ensure that those fulfil security policies. These methods are transparent to the user and often with minimal overhead. They allow transferring the burden of ensuring the security of the final code from the programmer of the application to the programming environment developers. Often, language based security offers the possibility to third parties to verify the security of the application developed, by allowing more complex certification scenarios (e.g. for mobile or outsourced code).

There is room for several improvements in this area of research. For instance, there are several efforts ongoing for embedding information flow management techniques in programming languages such as Java, or for embedding security mechanisms in Business Process Execution Languages used for composing complex services. One advantage of working with the latter kind of languages is their relatively higher abstraction level, which contributes to make language-based security and verification techniques even more effective. A new promising area of research is the application of these techniques for proving complex properties of cryptographic algorithms as well as provably correct implementations.

In summary, language-based security is regarded as the backbone of secure software engineering.

Advances in security verification and validation Given that software-intensive systems are prone to vulnerabilities, their detection and prevention prior to code deployment and operation is a fundamental step. Several rigorous techniques have been developed for checking system specifications, such as model checking (almost fully automated) and theorem proving (still needing significant human efforts). These formal methods, when supported by automated tools implementing them, are definitely successful in checking the specification of security protocols and automatically finding bugs. Some of these tools are also deployed at industrial level. However, there are several limitations that must be addressed for a wider deployment of such tools in industrial software development environments: In particular, they do not seem to scale-up and thus the real issue is the need to cope with the increasing complexity of software-intensive systems.

In addition, as mentioned before, security is a very peculiar non-functional property, where one needs to take into account the uncertainty about the behaviour of the system components (e.g. malicious software) as well as external threats. Fixing a threat model and making assumptions on it is often an error prone process. Indeed, often security protocols are not really flawed, but the programmer adopts them in an environment that does not enjoy the appropriate execution assumptions of these protocols. A main issue here is still in fact to take into account several attack scenarios and threat models and the corresponding so-called "most-general attackers" scenarios. As for other areas, the notions of composition, modularity and abstraction come to help in order to master complex ICT systems.

In summary, while at the specification level several automated analysis tools are already deployed at industrial level, with significant success, at the executable code level more research efforts are necessary to make these tools usable in practice during software development at industrial scale.

Advances in risk assessment for systems of systems Risk is a crucial notion in security and its role in the design of complex systems of systems needs to be further investigated. Advanced risk assessment will particularly become important for such systems as we move towards the Future Internet. There are several issues to address, in particular the complexity and the (cyclic) interdependencies inherent in ICT systems, often composed of several parts developed by different parties. Change is also part of the lifecycle of systems of systems and assessing its overall risk needs to take into account this process and make it cost effective, for instance by avoiding to repeat unnecessary steps. The impact of change is indeed a major factor in software development and maintenance. A possible solution is again to consider compositional risk-assessment methodologies, although this requires methods and mechanisms to deal with interfaces and again dependencies among components. It seems that embedding risk in an explicit manner in all the steps of the software development lifecycle could help to reduce the cost and make the improvements in software engineering more concrete.

4. Panel 3: The way forward

In the beginning of the session, the findings from the two previous sessions were briefly summarised. The Panellists were then asked to comment upon these findings and possibly provide some complementary views. The discussions that followed focused on work required for bridging the gap between today's industrial practices and research advances and on promising ways ahead.

The main issues that were brought up during the discussions can be grouped as follows: (a) enabling methodologies and tools for building secure complex systems and services; (b) revisiting the business case for security; (c) software liability aspects; and (d) standardisation, education and other relevant issues for the field.

Enabling methodologies and tools for building secure complex software systems Industrial software is not built from scratch anymore. It is often built on top of legacy systems and/or outsourced. This increases the need for having means to verify the security properties and performance of legacy systems and/or third party software. The composition environment is a critical component, as it should permit to control the security properties of composed software, whether at the design phase or dynamically, at run time. Compositionality is a big challenge. Even if a software system is built from individually trusted components, the overall system may not be trusted. Modular verification of smaller modules may prove to be a good solution in large complex systems. Virtualization may significantly reduce the level of complexity and facilitate security assessment of legacy systems. It is also possible to build trustworthy systems by using untrusted components or components having unknown security properties.

Today, there is a lack of integration of security engineering and software engineering methodologies and platforms. The general (wrong) perception is that software engineering is dealing with construction of correct software, while security engineering is dealing with the deployment of software. The software architecture should be the starting basis. Security, manageability and scalability should be the main drivers for the software architects. Security requirements to consider include also ethical aspects and societal implications (e.g. on the privacy of individuals).

Software systems are hosted in specific environments. A wide range of different environments may host different instances of the same system. This diversity introduces a higher level of complexity with respect to ensuring the correct functioning of all security properties of the system.

Producing secure code should not be an expensive process. Industry needs usable and efficient methodologies and tools that automate the security of software code. Formal methods have proven to be useful for checking security specifications but not really software implementations. In general, industry considers formal methods as complex and expensive to use. Different cases such as software for critical systems require different methodologies. For example, in many particular situations, insufficient secure software may affect human lives (e.g. uploading insecure software in pace makers) as security and safety are linked together. In such critical software systems, the use of heavy formal methods is to be envisaged as a means to assure higher levels of safety. Overall though, there is still no consensus on which are the most promising formal method solutions to use for building the tools industry needs. It is therefore urgent to undertake further work for bridging the gap between fundamental theories (e.g. heavy formal methods) and pragmatic approaches for industry to use.

Automated mechanisms are needed for testing not only the security properties of the software but also the results of its interaction with the environment. Finally, further research is needed for building systems of the Future Internet that learn from their failures and their interactions with the environment (e.g. handling or operating under malware attacks), and are able to self-reconfigure and evolve.

Revisiting the business case for security At present, organisations are not truly using business cases for security. Many people see the need for having business cases for security as a way to show the return on investment. Others consider it as indispensable to have integrated frameworks for assessing business and technology risks and complying with regulations:

– In several products and services, security mainly means reliability and consequently becomes part of their core requirements; as such, it is no longer perceived as a separate attribute to consider.

– Since several companies are engaged in maintaining and supporting their software products for 8-10 years after their first release, it makes sense to consider security from the outset and addressing it through a business case.

As it is not possible to invest equally in all aspects of security in software at once, priorities must be set up. Risk management frameworks can help in doing this by analysing the probability for a given risk to happen as well as its impact on business. Boundaries can be defined at the level of the business case by assessing security in a holistic way aiming to mitigate risks and undesirable impacts. Risk assessment can sometimes identify security needs to be addressed and sorted out. Modelling tools can be very useful for risk assessment because they enable the description, explanation, discussion and assessment of alternative scenarios and the evaluation of inherent risks. This may lead to refining business cases and making them show the costs and benefits of the secure solution.

Nowadays, cloud computing is an interesting new business case for large software companies and their third party providers. New business models are emerging. However, the impact and implications of cloud computing on security of dynamically composed services, on privacy and on protection of personal data or confidential business data are still to be seen.

Software liability For the moment, software companies in general and those companies in particular offering packaged software services or Service Oriented Architecture (SOA)-based applications and services are not liable for the likely damages they may cause due to software vulnerabilities of their products. As liability may change with time, it is important for companies to adopt best practices quickly.

Should software companies become liable, they would need to become in full control of all the products, applications and services they sell, including all the underlying technology components supporting such products. In practice though, that would mean that liability would have to be based on best efforts, using state-of-the-art methods and tools, since the current state-of-the-art does not allow to build completely fault-less products. It would also mean that companies would need to be able to establish clear assignment of responsibility in the supporting technology chain, in case a problem occurs because of a deficient product. Establishing however, who is responsible and liable for this may prove to be a very complex and costly task. Indeed, such a product may be the result of complex compositions of different software components or services belonging to many different players or third parties. Moreover, modern services (software as a service) are often compiled or executed at the premises of a service provider, which is linked to a customer via a service level agreement (SLA) that defines the quality of protection the customer receives. However, to achieve a SLA, many different infrastructures may be intervening, each belonging to a different owner.

Therefore, a prerequisite for solving software liability is solving the compositionality problem. Moreover, mitigating risks signifies being able to reason about their probability to happen. Virtualisation technologies may also prove to be useful as many different legacy applications may run in parallel under one single platform but isolated from each other, with individual monitoring capabilities associated to each of them.

Standardisation, education and other relevant issues Currently there is a lack of sufficient standardisation in software security, while it is well known that standardisation fosters interoperability and competition. Among the new fields under standardisation are privacy and security in RFIDs. In some cases, clear specifications are available at a certain level of abstraction, but implementations of standards are often not completely in line with these specifications. Robust tools for testing and validating such implementations are necessary.

Often there is a gap between the methodologies that secure software engineers are taught in Universities and the knowledge they need when working in industry. A closer and more productive cooperation is required between industry and academia in order to produce more adapted curricula dealing with both foundational knowledge principles and industrial reality.

Finally, it was mentioned that more research is needed on the links between law and security in software systems and how to possibly increase security with new legal instruments. The example of Digital Rights Management (DRM) was cited, where researchers need to explore the vulnerabilities and weaknesses of existing DRM systems (in France, this is actually forbidden by law), in order to yield more secure DRM systems.

5. Concluding remarks

The overall discussion and significant participation of both industry and research/academia representatives at the event is further evidence of the relevance of the topics addressed in the Seminar.

The business and industrial communities have a lot of motivations for adopting best practices for secure software engineering and generally increasing the security of their ICT systems, products and services.

The scientific community can already bring several methodologies and tools in this arena, although targeting specific priorities as some of the ones identified in this report would likely help to close the gap between foundational and practical work. There is definitely the need for research and efforts that would effectively permit to integrate security and software engineering in one coherent framework.

As complexity of ICT systems increases, easy-to-use software tools that encapsulate highly intensive specialised knowledge need to be conceived and developed through research and industrial partnerships. In order to ease this process, industry and academia should indeed share similar expertise and adopt the same language and terminology. Therefore, a framework for common understanding and cooperation should be established in the near future in order to overcome the current landscape.

Raising current levels of education and awareness in the field is another main issue discussed during the Seminar. It must encompass not only software professionals but also politicians, policy makers, lawyers, business managers and auditors.

Finally, special attention must be given to new forms of emerging IT infrastructures such as cloud computing, “the internet of things” and, more broadly, the Future Internet, that bring new issues and challenges for secure software as well as new opportunities for industry and business organizations. Indeed, to a large extent, functionality in the Future Internet will be provided by means of services (both in a technical meaning, as ubiquitously available infrastructure for integration across domains as, for instance, Web Services do provide, and a business meaning, as service offerings can be leveraged through the Future Internet towards a new dimension of Business Ecosystems). From a secure software engineering perspective, this puts a strong emphasis on decoupling related functionality from business functionality to overcome security and trust silos and provide the same level of flexibility as it is required from the business logic.

Acknowledgements

ERCIM and the European Commission's s DG INFSO Unit F5 “Security” would like to warmly thank the Organising Committee of this Seminar, namely: ***Javier Lopez*** (U. of Malaga), ***Volkmar Lotz*** (SAP Research), ***Fabio Martinelli*** (IIT-CNR), ***Aljosa Pasic*** (Atos Origin), ***Dimitris Plexousakis*** (ERCIM), ***Manuel Carvalhosa*** and ***Thomas Skordas*** (EC, DG INFSO-F5). We would also like to particularly thank all the speakers and participants of the seminar for their contribution to the success of this event.