2003

Development of a Security Methodology for Cooperative Information Systems: The CooPSIS Project

Mariagrazia Fugini
Politecnico di Milano, fugini@elet.polimi.it

Mario Mezzanzanica
Universita Statale di Milano, mario.mezzanzanica@unimib.it

Development of a Security Methodology for Cooperative Information Systems: the CooPSIS Project

Mariagrazia Fugini
Dipartimento di Elettronica e Informazione,
Politecnico di Milano Piazza Leonardo da Vinci, 32 I-20133 Milano
Italy
Ph: +39-02-23993400, Fax +39-02-23993411
fugini@elet.polimi.it
http://www.elet.polimi.it/upload/fugini/

Mario Mezzanzanica
Università Statale di Milano-Bicocca, Facoltà di Scienze Statistiche
Via Bicocca degli Arcimboldi 8, 20126 Milano
Italy
Ph:+39 – 02-64487300, Fax: +39-02- 64487330
mario.mezzanzanica@unimib.it

Abstract

Since networks and computing systems are vital components of today's life, it is of utmost importance to endow them with the capability to survive physical and logical faults, as well as malicious or deliberate attacks. When the information system is obtained by federating pre-existing local systems, a methodology is needed to integrate security policies and mechanisms under a uniform structure. Therefore, in building distributed information systems, a methodology for analysis, design and implementation of security requirements of data and processes is essential for obtaining mutual trust between cooperating organizations. Moreover, when the information system is built as a cooperative set of e-services, security is related to the type of data, to the sensitivity context of the cooperative processes and to the security characteristics of the communication paradigms.

The CoopSIS (Cooperative Secure Information Systems) project aims to develop methods and tools for the analysis, design, implementation and evaluation of secure and survivable distributed information systems of cooperative type, in particular with experimentation in the Public Administration Domain. This paper presents the basic issues of a methodology being conceived to build a trusted cooperative environment, where data sensitivity parameters and security requirements of processes are taken into account. The milestones phases of the security development methodology in the context of this project are illustrated.

Keywords

Security, Survivability, Policies, Collaborative Information Systems, Web Services.

1. Introduction

The CoopSIS *(Cooperative Secure Information Systems*) project, framed in the context of National and Regional Projects in Italy regarding e-government and security, aims to develop methods and tools for the analysis, design, implementation and evaluation of "survivable" distributed information systems. Since networks and computing systems are vital components of today's life, it is of utmost importance to provide them with the capability to survive physical and logical faults, as well as malicious or deliberate attacks.

To achieve this target, the project groups together researchers and developers in two areas: distributed information systems for Public Administrations and security. From their marriage, we expect to have a global systematic and efficient approach to help in the design and deployment of available and secure systems. Recently, the widespread use of information technology and the availability of networking services have enabled new types of applications, characterized by several geographically distributed interacting organizations. We refer to the term *Cooperative Information Systems* (*CoopIS*) to denote distributed information systems that are employed by users of different organizations under a common goal (Mylopoulos 1998). In this paper we refer to recent extensions of the concept of *CoopIS*, namely *e-applications* (Mecella 2001), consisting of *e-services* provided by different organizations on the net.

Our approach to the security problem in CoopIS is based on two concepts:

- a *system description* based on formal policy, that is, a formal specification of the security and dependability properties expected for the target system; this yields to a multi-level approach, where the designer can specify both high-level properties or specific fine-grain controls;
- a *formal system description*, that consists of an accurate model of information system components and interconnections in terms of security and dependability features and parameters.

Based on the description of the desired policy and of the target system, it is possible to perform the following actions:

1. To build a model useful to evaluate the global/local survivability of the system, with respect to the desired properties *specified in the system policy*.
2. To automatically or semi-automatically *derive configuration rules for the system* to implement the desired security policy. If derivation is impossible, due to lack of security features in the target system, the missing features will be automatically suggested to the developers by monitoring in real-time the behaviour of the current system and to compare this behaviour with the policy, in order to identify discrepancies from the expected behaviour that could signal either problems in the implementation or unknown/unpredicted attacks or faults.

Since these two are quite difficult tasks to be performed in a generalized way, the methodology we are developing within the CoopSIS project will deal mainly with the following specific application *contexts*, which are of particular importance in cooperative information systems, that is:

- middleware for secure and dependable systems (investigation will explore both CORBA-based paradigms and the more recent SOAP/UDDI approach to distributed applications), also when used to encapsulate legacy or COTS components;
- distributed computing environments of the GRID type;
- classical Internet architectures (web services, DBMS, firewalls, IDS);
- role-based authorization systems for enterprise data access and control;
- protection of XML objects, such as those used for electronic documents or interoperable transactions and data exchanges between different domains.

The paper is organized as follows. After a short review of the methodology phases, presented in Sect. 1.1, in Section 2, we frame our work in the context of research on security distributed systems. In Section 3, we present the requirements of the methodology. Finally, in Section 4 we discuss the results expected from the application of the methodology by presenting its envisioned use in cooperative Information Systems in the Public Administration context.

1.1 Overview of the CoopSIS methodology

The security development methodology is composed of four phases.

1. Phase 1 deals with the *design of* an XML-based *language for formal security policy specification* and formal system description, and with the development of the related support tools (parsers, translators).
2. Based on the provided description of quantitative and qualitative parameters provided by Phase 1, Phase 2 is a *design phase defining models and tools to evaluate global and local system properties*, and to suggest appropriate system (re-)configuration actions when the desired properties cannot be obtained using the available systems. This may range from simple automatic configuration of existing components (e.g., configuring access control lists on a router to block illegal access) to relevant architectural changes (e.g. addition of firewalls or redundant hardware components).
3. Phase 3 deals with *system monitoring*, to detect failures and attacks, to report them and - if desired - to automatically activate appropriate countermeasures and re-design system security.
4. Phase 4 is the *implementation, context-specific phase*. It gives guidelines instantiated on a

specific context, to implement the policies and designed mechanism within the selected application contexts. Contexts studied in our case are Public Administrations and the Italian plan for e-government. More generally, for each context, Phase 4 provides models of the dependability and security properties of the system, with the associated qualitative and quantitative evaluation. Then, Phase 4 develops some of the security mechanisms (including the needed infrastructures) that have been found as missing in a given application context.

Although there are several dependencies between the four phases, there is no strict sequencing of them. Several loops and feedback steps are included among the steps, in order to verify the results of each steps against the requirements provided by the previous ones and to insert correction actions.

The project wherein the methodology is being developed spans over two years, and considers application in the Public Administration domain, using healthcare, welfare, and, in general, e-government applications, where security mechanisms have to be developed for web services-based Portals oriented to providing e-services to citizens and to enterprises distributed on regional areas.

A final demonstrator will be set up, covering the whole chain envisaged by the CoopSIS project (formal description, evaluation, configuration, and monitoring) in one or two specific areas (e.g. web-based ICT architectures, GRID, e-document workflow).

2. The context: security and cooperation

We are witnessing the construction of networks of large-scale, complex and automated information systems, which support day-by-day activities in several vital functions.

For this reason, these systems can be regarded as *critical infrastructures* which need to survive attacks and intrusions as well as failures of components or subsystems. They integrate previously disjoint systems to provide services, which have become critical in our everyday life, such as services to citizens or organizations. Under no circumstances a complete unavailability is acceptable for these infrastructures. Therefore concepts like security and survivability have become crucial requirements of current information systems, and it is foreseeable that their importance will increase in the future at a fast pace.

In the US, this area has been considered to be particularly urgent, even before the events of September 11, as evidenced in the President's Commission on Critical Infrastructure Protection (President 2002). The special issue of IEEE Computer - August 2000 is another symptom of this urgency (Jones 2000). Its importance has also been found to be critical for the success of the Grid initiative, that is, the replacement for the Internet, both in the US and in the EU. The European Community has become convinced of the great need to define and plan a broader and more integrated set of security and dependability-related activities for the IST 6th Framework Program (EU 2002) and than in FWP5 ((Wilikens 1998)). (See the *European Dependability Initiative* forum at http://deppy.jrc.it/, where the term Dependability (Laprie 1995, Laprie 2001) is used by the EU to encompass several system attributes, such as reliability, availability, safety, security, survivability).

In general, the information infrastructure, and thus many of the dependability problems that will exist in years to come, will be global in nature. Thus strong cooperation is envisaged between the EU and US on dependability. Negotiations are being conducted within the framework of the EU-US Science & Technology agreement, via a number of joint workshops. This situation is very well clearly summarized by the welcome message of the DSN-2002 general chair, Dr. Jaynarayan H. Lala: "The traditional concerns of the dependability community (e.g., inadvertent faults, errors, and failures) have now been enlarged by the massive connectivity provided by the Internet to include malicious exploitation of imperfect systems and networks and intentional cyber-attacks on them. How can we build systems that are not vulnerable to such threats - systems that users can depend upon in defence, transportation, financial, and e-commerce sectors? These questions have taken on added importance" (DSN is the IEEE international conference on Dependable Systems and Networks, with an history of more than 30 editions). In order to fully respond to these problems, systematic and integrated approaches are needed to enhance both dependability and security of systems; the MAFTIA project (MAFTIA 2001) is a first example of cooperation between researchers of these areas.

2.1 Formal specification and Modelling

Many modern approaches use a specification language to describe security policies and/or fault tolerant strategies and to build models needed for a-priori dependability evaluation. This is important for application in real-life systems, where it is often impossible to propose a complete rebuild of the whole system. On the contrary, attention must be paid to systems built using Commercial Off-The Shelf (COTS) components. Hence, a proper approach is to formally specify policies and strategies, and later translate them on the target systems.

The elements of the language are basic "mechanisms" that need to be appropriately configured (and eventually reconfigured in presence of a detected misbehaviour) to implement a given policy. A similar experience has been done in the EC project DepAude, and its ancestor Tiran (Botti 1999), where an interpreted language has been defined and implemented to distribute a set of tasks over a set of nodes, with an associated list of reconfiguration actions in the presence of detected malfunctioning.

The availability of a specification document for the security policy and the dependability strategy can also be exploited to simplify the construction of models from the early stages of the design, as advocated, for example, in (Vemuri 1999) and as initially experimented in (Bobbio 2001).

Some efforts have been done to develop a formal model of a target system and apply analytical metrics to assess its weaknesses (Jha2001, Swiler 1998 and Ortalo 1999). Nevertheless, it is clear that no single analysis and modelling approach can successfully cope with the growing complexity of distributed information systems. A unifying methodology is needed to cope with the different aspects involved in information systems security. In particular, when information systems are constructed as federations of local existing systems, a method is needed to integrate existing policies on data and application protection with the smallest impact on the overall system operations and user confidence and trust on the security procedures.

A critical point is that current methods have no control on how many different security components act in combination. In (Helsinger 2001) an approach is described that combines modelling and experimentation but the failure modes of complex networked systems appear not well understood nor adequately modelled yet. A multi-formalism multi-solution (MFMS) approach is very appealing, since a designer may choose the formalism that best suits each part of the overall model and then combine the sub-models in order to obtain one large heterogeneous model. Some results in this direction have been achieved within the Mobius project (Clark 2001).

For security aspects, the successful development of automatic policy systems clearly requires formal languages able to describe:

- multiple involved entities (users, devices, services, ...) as well as their complex interactions
- available mechanisms (protocols, authentication and authorization methods, trust management techniques, ...) and their potentially complex configuration
- required actions in case of attacks or vulnerability detection (e.g., alert, service blocking, automatic system reconfiguration).

Current research in this area include the IETF (Internet Engineering Task Force) working groups IPSP (IP Security Policy) (Blaze 1999,) RAP (Resource Allocation Protocol) (Durham 2000), and POLICY (Moore 2001). The Policy Core Information Model (PCIM) (Moore 2001) and (Moore 2002), defined by the POLICY WG, tries to define a language for general policy description. A relevant alternative is the Ponder (Damianou 2000) language, developed at the Imperial College in London.

Looking at more security-specific languages, the SecPol Sloman (2000) project, for security policy management in distributed systems, and the KeyNote (Blaze 1999) system, focused on trust management, are key examples. Recently, the international research community has outlined the Extensible Markup Language (XML) (Bray 2000) as a promising base to define languages able to fulfil the necessary expressive power. The Extensible Access Control Markup Language (XACML) (Pilz 2001 and (Godik 2002) and the Security Assertion Markup Language (SAML) (Hallam-Baker 2002) projects are among the most interesting examples of XML based languages for security policy definition.

2.2 Performance Evaluation

In the performance evaluation field, the study of the dependability of systems is considered a difficult topic due to the need of modelling together the hardware, the particular hw/sw mechanisms, the interactions among them, the presence of faults and eventually of the error and failure chain, and the need to drop the exponential distribution assumptions for delays and to identify specific ad-hoc indexes. Moreover, although security is considered as one of the attributes of dependability (a mix of integrity and availability with respect to authorized actions, together with confidentiality), performance studies on dependability often do not consider security aspects, and vice versa.

2.3 Monitoring

Continuous and cost-effective monitoring of survivability mechanisms must be performed in order to achieve survivability goals. In fact, the deployment of complex distributed applications based on COTS hardware and software components, as well as legacy subsystems of cooperative environments, seriously affects the possibility of meeting survivability requirements. Actually, these components are usually developed neglecting availability, reliability and security. Therefore, without efficient architectural and methodological solutions, large survivable infrastructures are almost impossible to obtain. In this context, the capability to detect errors or component failures, the ability to correctly diagnose the status or health of components (including entire legacy subsystems), and the availability of quick and intelligent reconfiguration and fault treatment strategies are fundamental issues to be addressed. In particular, fault diagnosis depends on whether the fault is intentional or unintentional. The former case requires intrusion detection techniques (Deswarte 1991, Sobirey 1996) while the latter can be achieved through traditional fault diagnosis techniques (Lin 1990, Iyer 1990, Sosnowski 1994).

2.4 Application Contexts

Developers typically build systems with little or no attention to their security. They try to incorporate these features in the system afterwards. The result of this approach is the adoption of partial and inadequate solutions, which are vendor-specific and poorly integrated with the application.

In general, more efforts have been put in the security field, rather than in the availability one. For example, application security is based either on secure message formats, such as the S/MIME (Ramsdell 1999) format for protection of e-mail messages or electronic documents, or secure channels, such as e-commerce sensitive web traffic protected through the SSL or TLS protocol (Harkins 1998). With respect to network-level security mechanisms, the IPsec protocol (Dierks 1999, Kent 1998) is the standard architecture, used to set up both secure VPN (Virtual Private Network) and end-to-end secure channels at IP level. Integrity of the network infrastructure itself is also of primary relevance: protection and control technologies based on proxies and/or firewalls (Cheswick 1994) and Intrusion Detection Systems (IDS) are the classical solution to avoid unauthorized use of communication resources.

Only recent years have seen efforts towards standard architectural approaches to security. One remarkable project is the Common Data Security Architecture (CDSA) project which aims at defining a middleware infrastructure capable of providing a set of security services in a transparent way. In addition to S/MIME, generic data protection is usually achieved with specific standard formats; here the choice is between Cryptographic Message Syntax (CMS) (Housley 1999) and XML-based standards, such as XML-DSIG (Eastlake 2002) and XML-encryption (still in a draft status).

More articulated efforts have been done in the context of object-oriented distributed systems, such as the CORBA-SEC solution for CORBA architectures. Also the need for fault tolerance in CORBA has been long advocated by the scientific community. This has led to the specification of Fault Tolerant CORBA by OMG, the Object Management Group (CORBA 2000). A number of projects have been activated, both in the academia and in the industry, which aim at developing fault tolerant CORBA platforms. These systems incorporate fault tolerance management in the ORB and/or in additional software layers. Eternal (Narasimhan 2002) is an example of such systems.

Communication between replication groups is done using connection groups. The Eternal system adds fault tolerance to CORBA applications by replicating objects and incorporating a number of mechanisms to maintain replication consistency. For distributed computing environments (such as the GRID initiative) security issues are extremely important (Ferrari 1999, Butler 2000). Such systems generate dynamic aggregations of computational resources that are heterogeneous both in the security architecture and in the user or administrative domain they belong to.

Moreover, reconfiguration (to achieve survivability) is often achieved via distributed plug-ins. Unfortunately this technology introduces a source of complexity in security, by allowing users to change at run-time the set of services provided by the system by means of modules downloaded from local and remote repositories. The Java2 platform defines a security architecture (Gong 1998) that allows developers both to define access policies for pre-existing resources and to add new resources. This architecture leverages symmetric and asymmetric cryptographic standards and is compatible with the X.509 digital certification standard.

2.5 Cooperative Information Systems and Security

In order to integrate features of *data and process security* in a CoopIS/e-application we have to handle:

- security of data handled *within* one e-service;

- security of data exchanged *among* cooperating e-services and therefore of the *cooperation* among individually trusted e-services.

The methodology to be designed needs to extend the traditional waterfall methods by providing a set of steps that, starting from the system and user security requirements and policies, brings to the development of a *secure*, or *trusted*, CoopIS. In the remainder of the paper, we make a set of assumptions about the information system that are now shortly listed (for a complete description of the cooperative information systems assumption we refer to (Fugini 2002)).

1) First, we assume that each e-service has been designed according to internal security requirements and policies and is therefore a *trusted* or secure (set of) *process*(es). The focus of the methodology is on data and process security during the *cooperation* among processes.
2) Secondly, we consider that organizations cooperating in CoopIS/e-applications are of two types:

- *trusted organizations*: data transmission occurs among organizations which trust each other in a network due to organizational reasons (e.g., homogeneous work groups in a departmental structure, or supply-chain relationships among organizations in a virtual enterprise);
- *untrusted/external organizations*: data are transmitted among cooperating entities in general, possibly accessing external data sources. Every time mutual knowledge among organizations participating in CoopIS/e-applications is not given in advance, mechanisms are needed to ensure that mutual trust be established dynamically, during cooperative process executions.

Trust comprises mainly two aspects:

- the sensitivity of data being managed within an organization unit, and
- a secure *information exchange* to guarantee sensitive information.

Sensitivity concerns both correct authentication of cooperating organizations and the process of guaranteeing that only authorized organizations can read, use, and generate data in the cooperative process. To guarantee sensitivity of information, security technologies and mechanisms must be used, e.g., based on the use of digital certificates and signatures, to allow the cooperating organizations to establish a secure communication environment (secure information exchange).

3. The methodology

In this section, we describe the phases of a methodology that integrates CoopIS application requirements with security aspects from the early phases of system design. It considers the foundations on security and survivability overviewed in the previous section and proposes a set of phases specifically oriented to develop secure cooperative systems, prevalently based on web services technology, that exchange data and cooperate in a controlled way.

First of all, let us introduce the new scenario that involve web services (Curbera 2002) taking into account a conceptual architecture proposed for web services named Service Oriented Architecture (SOA) illustrated in Fig.1, where three main actors appear:

- Service provider: who creates and holds the service;
- Service directory: who *publishes*, on service provider's request, the service in order to publicize it;
- Service requestor: who needs for a service and uses the service directory to *find* the best service matching its requisite. Once found, the service requestor directly connects to the service provider in order to *invoke* (and so *use*) the service.

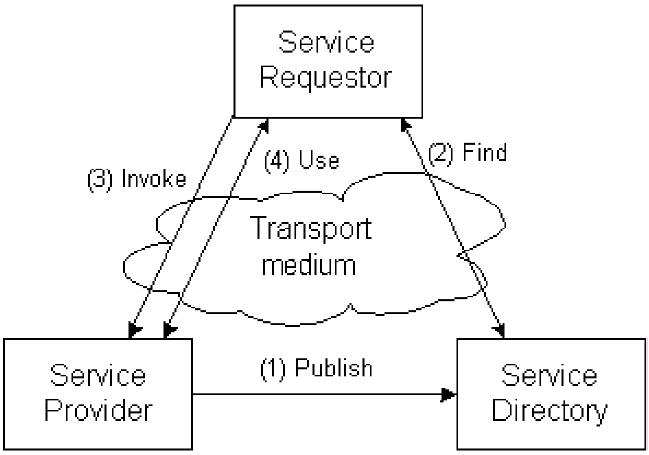

Fig. 1 Service oriented architecture

In this scenario, Web Services can be invoked by an application, a component, and another web-component. In such a way, several web-components cooperate to perform an application, e.g., to search all the data regarding a citizen's curriculum when supporting him/her to find a job over a set of territorial information systems (the Provinces systems, and the regional system). This means that a web-component that needs the cooperation of another web-component has to cooperate with another one considering several features: availability of a channel, convenience, reliability and security.
Security problems regard authentication of the partners that cooperate (e.g., the province and the regional system that exchange regarding a citizen), data base security, network security, and also node survival in order to preserve the transactional properties over the distributed execution of a long web transaction.

3.1 Phase 1- Foundations: Design of a Description Language for Formal Policies

The first phase of the methodology consists in laying down the foundations and policies needed by the planned development activities of the cooperative system. This phase targets the following objectives:

a) Analyze in detail the dependability and security problems and mechanism of the application contexts:
   - middleware for secure and dependable systems (both CORBA-based paradigms and the more recent SOAP/UDDI approach to distributed applications), also when used to encapsulate legacy or COTS components;
   - distributed computing environments of the GRID type, augmented with security and reconfiguration features;
   - classical Internet architectures (web services, DBMS, firewalls, IDS), including role-based authorization systems for enterprise data access;
   - protection of XML objects, such as those used for electronic documents, interoperable transactions or data exchanges between different domains.

b) Build on current knowledge and results of item a) to define two multilevel languages, one for system description (behavioural and topological) and one for policy specification (description of general high-level properties as well as context-specific fine-grain controls).

c) Develop the languages to describe and to evaluate the dependability and security of the systems, and design/identify architectures and mechanisms supporting the implementation of policies.

d) Define the foundations of the basic mechanisms needed to monitor the system activity and detect failures (that is, violation of the policies).

The phase is divided into *areas* according to the specific application contexts to which it is related.

TRUSTED AND RELIABLE COOPERATIVE ENVIRONMENTS

- Identification of security requirements and policies of a CoopIS, where the sensitivity of user and system data (mainly databases and business data, such as customer directories or collections of documents) is strictly classified and a core-business data is individuated.

- Investigation of the survivability properties of various architectures for the integration of security services and cooperation infrastructures based on Web Services, with the target to create a Secure Service Oriented Architecture (SSOA). The SSOA must be designed specifically to support a Business Community defined as a "virtual community" of users that share a common business area and need to interact digitally. Such a Business Community can be instantiated in many different contexts ranging from Public Administrations to private organizations. In addition to dependability and security, the requirements of a SSOA are the pervasivity, interoperability, openness, extensibility and service provider independence. To accomplish this task the phase studies existing architectures, standards and technologies for access control to local and wide area services. This study includes the analysis of the models and of the existing solutions and the analysis of the match between such solutions and the formal requirements specified by the security policies.

DISTRIBUTED COMPUTING GRID-TYPE ARCHITECTURES

- Evaluation of the survivability features of different grid-types environments; for example, the HARNESS system seems to be the most attractive one, due to its easy dynamic reconfigurability. This evaluation requires in-depth study of the security and dependability problems for grid computers with dynamic reconfiguration capabilities built on top of the Java2 platform and Web Services technologies.
- General evaluation and modeling of the features of security and dynamic reconfiguration mechanisms existing in some environments, such as in Java2/JSSE.

XML DATA PROTECTION

- Evaluation and modeling of secure document management services, with reference to structured documents, possibly containing multimedia components. Analysis of possible document transformations that change the content and that allow for different levels of security is also included.
- Features and applicability of the XML Digital Signature and Encryption formats for generic data protection.

CLASSICAL INTERNET APPLICATIONS

- Analysis of the features of the most common Internet protocols, devices and services, at different abstraction levels. This study provides the functional and quantitative models of the survivability of Internet architectures by analyzing the features of routers, firewalls, and application servers, as well as those of standard security protocols and formats (such as IPsec, SSL, S/MIME).

To exploit a useful system description language and formal policy definition language, the methodology has to accomplish the following tasks:

- Selection of a policy definition language, focused on expressiveness and completeness; in particular, languages such as XML-based languages, XACML (Extensible Access Control Markup Language) and SAML (Security Assertion Markup Language), can be considered, since they are particularly attractive for their extensibility.
- Evaluation of the emerging models in the security field, in order to have a complete background to use a proper language syntax - multilevel policy extensions to take into account aspects typical of various contexts (network, application, and middleware layers). For example, this would allow also specification of policies for document exchanges at the application level for secure document management services.
- Definition of a language that includes security policies and dependability aspects. A particular attention has to be paid to the definition of performance, dependability and security parameters. Considering for example the development of XML dialects, we have to evaluate the required

extensions or modifications to an existing policy language, or its development ex-novo.
- Definition of a language to describe topology and features of distributed architectures for the target application contexts, with respect to those aspects relevant for survivability analysis.

3.2 Phase 2 – Design

In the design phase, *system security* data (such as password, security files, log files, authentication data and so on) are designed, and ways for exchanging data in a secure way are identified. This phase has to deal with the design of the security infrastructure model for interoperability among elements of the cooperative information system. The following areas have to be tackled with.

IDENTIFICATION AND CONFIGURATION OF SURVIVABILITY MECHANISMS

The tasks to be performed in this area are as follows:
- Identification of procedures and strategies for correct reconfiguration of the infrastructure that allows for a better behaviour in reaction to failures of components, attacks or overloads. For example, various reflective systems and fault tolerance structures are evaluated, having the capability to dynamically adapt their behaviour based on information system user requirements and fault conditions.
- Monitoring and reconfiguration strategies compared against a given survivability level- definition of strategies for correct reconfiguration of the infrastructure and validation of such strategies.

MODELLING AND MODEL EVALUATION
- Use of formalisms (such as Workflow Models) to express security features, constraints, and policy choice support.
- Extension of formalisms to deal with the possibility of importing and exporting features.
- Derivation of a *security evaluation schema* for cooperative systems described according to the defined language. By schema, we mean the identification of the main security modelling components, the definition of their relations into the aggregation of the components into the overall system model, and the identification of the security indexes of interest to define an integrated approach for the compositional construction of security models, leading to the development of a set of security classes in an object-oriented language.

SURVIVABLE SECURE ARCHITECTURES
- Development of a model and a methodology for secure web-based information systems, including trusted data exchange in cooperative environments.
- Definition of an architecture for the integration of a Public Key Infrastructure with a Service Registry, leading to the identification of those PKI components that can be integrated in a service registry such as UDDI (Universal Description, Discovery, and Integration), and organization of the components interaction in such a way to preserve the chain of trust that ensures the correct work of PKIs.
- Definition of an Architecture for the protection of transport protocols for Service Oriented Architectures, to enable authentication, encryption and fault tolerance mechanisms in the transport protocols of Service Oriented Architectures, such as SOAP (Simple Object Access Protocol).
- For grid-types system, design of plug-in repositories that exploit PKI and secure channel technology to certify the source and guarantee the integrity of downloaded plug-ins, as well as a bridge to permit controlled integration of non-secure modules in the secure grid architecture.

3.3 Phase 3 – System Monitoring

System monitoring means the observation of the cooperative system during its operation in order to find the best and most efficient solutions to security of data, devices, and communication channels. Different monitoring approaches have to be studied and then reconciled during an experimentation phase.

In general, this activity is related to identification of mechanisms for error detection and accurate diagnosis of malfunctions of (COTS or legacy) components of distributed infrastructures. This includes also intrusions, overload and congestion detection. A reasonable possibility consists in starting from the threshold-based mechanisms that have been defined in other contexts, such as for diagnostic purposes in embedded systems. Despite diagnosis in the context of large COTS and legacy-based distributed systems is a different problem, "threshold" is a good approach since it allows to establish a set of *minimal requirements* to be fulfilled by the security mechanisms. The threshold can be established more efficiently if system observation is performed along time.

A different approach can be studied. By building on well-known intrusion-detection principles (sniffing, log analysis, and so on), the methodology can monitor the security of a system. In current intrusion-detection systems (IDS), this is a difficult task because the IDS module has to detect illegal or unusual system behaviour, with no formal definition of what is legal and normal. In our methodology, the formal policy specification allows one to be very accurate and to detect misbehaviours by comparing the actual activity with the one specified in the policy. Obviously, the more specific a policy (e.g., accept only ICMP packets from node X to node Y) the more effective the monitoring effort.

In the monitoring phase of the methodology, the work concentrates on the fast matching problem, that is how to quickly find if some data (log, network packet) match a set of rules. Approaches based on BDD (Binary Decision Diagrams) seem to be very promising in this respect

3.4 Phase 4 – Context-specific Implementation

Implementation and experimentation of techniques and architectures designed in the previous are implemented and instantiated on specific contexts. A policy-based framework has to be developed to simplify this task and provide a consistent background for all the considered facets and application contexts.

This framework will provide the basic functions for actual system implementation, and will generate a common base format to interface the tools developed for various purposes inside the project. In fact various tools can be employed, ranging from system simulation and evaluation engines to automatic configurators, from monitors to specific dependability and security modules.

A common framework to specify and manage system description and policy definition has to be developed. This requires the following steps:

- Implementation of the framework core engine. This mainly includes all necessary parsing functions for formal system description and policy description languages specified in the first steps, and the interfaces required to integrate and coordinate all the specific computational modules (evaluation, configuration and monitoring). After parsing and checking the policies and the descriptions, they will be translated to an intermediate internal format, to be used by all others modules of the distributed framework.
- Design and development of the interfaces needed to integrate and coordinate security modules developed by the other development groups for specific aspects of the application domain. Java and/or Corba technologies will be used in the development of the framework, thanks to their modularity and dynamic behaviour.

In general, the second step experiments the configuration and evaluation strategies identified in phase 1 of the methodology. For example, this will permit to define a composite validation strategy based on different and complementary analytical, simulation and experimental methods for dealing with large distributed systems. As a result, tools will be developed that permit to measure (in a quantitative or - at least - qualitative way) the survivability of a system, given the policy defined for it.

A specific task is related to automatic configuration of security aspects. Starting from the formal system and policy descriptions, tools have to be employed to automatically create the configuration rules for elements studied in the first phase (e.g., ACL for routers, filters for firewalls and proxies, authorization sets for web/DBMS servers), to be later applied automatically or manually. This is an important task, because it is well-known that part of the security failures are due to human errors in system configuration.

Based on the results of the steps, one or more monitoring systems are developed. Thanks to the formal policy specification, a high level of accuracy can be achieved in misbehaviour detection by comparing the actual activity with the one defined as acceptable in the system policy.

Specific activities are summarized according to the application area that they are related to.

TRUSTED AND RELIABLE COOPERATIVE ENVIRONMENTS

- Definition of mechanisms to manage redundancies, like redundant sensors for system monitoring, for accurate diagnosis, for fast reconfigurations and state restoration.
- Implementation of the model developed for secure web-based information systems including trusted data exchange in cooperative environments.
- Implementation, deployment and test of the conceptual architecture of a Security Server. The techniques developed by the researchers involved in the definition of the multi-formalism approach and the mechanisms developed by the project researchers involved in the monitoring activity will be used to evaluate the survivability properties of the system.
- Implementation and evaluation of a pilot Secure Service Oriented Architecture building a prototypal implementation of the architectures defined in the previous phase in such a way to evaluate the feasibility and the performance of the solutions adopted.
- Implementation and evaluation of a prototypal architecture for user access to services across organizations. The system will be integrated with the architecture developed in the previous item.
- Implementation of a security platform made of selected products and solutions for authentication, access control, audit, communication to manage user data and system security data
- Implementation of a system based on COTS components to experiment and evaluate the results of the research activities.

XML DATA PROTECTION

- Evaluation of the effects of faults derived from chosen transformations during documents exchanges.
- Assessment of the capabilities of the designed security policy for electronic document security.
- Design and possible development of a policy-based e-document middleware, to take into account policy requirements in the whole life of a document, including its exchange with foreign domains.

4. Results of the methodology

Upon completion of the methodology, the following results are expected for the considered application contexts:

- creation of a framework (based on the languages defined in the first phase) to describe a cooperative distributed system and to formally specify its security and dependability policy;
- availability of tools (integrated in the framework of the previous point) for automatic system configuration or reconfiguration, for evaluation of its properties, and for the monitoring of its activities;
- availability of specific modules to augment the security and reliability of the target CoopIS architecture.

A test bed will be set up to demonstrate the whole workflow of the framework (specification, evaluation, configuration and monitoring) for a specific application domain. A prototype of applications augmented with survivability features will be developed and tested as depicted in Fig. 2.

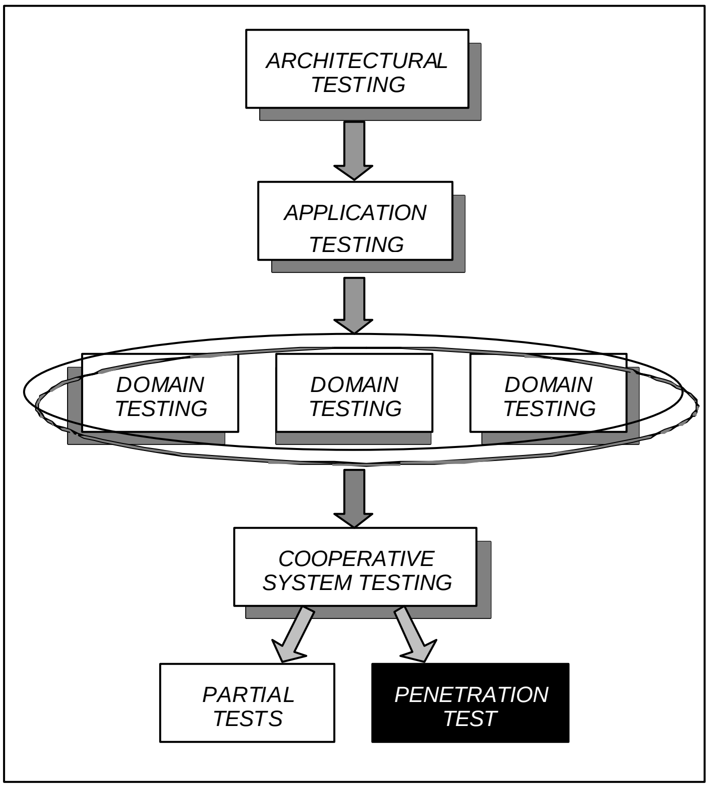

Fig. 2 CoopIS Testing

The methodology phases undergo a continuous feedback (Fig. 3) in order to evaluate the objectives actually reached, as defined for each phase. More specifically, the languages, the models and the architectures have to be documented in technical manual of the methodology, while the prototypes of the software components (translators, evaluation and monitoring systems, ...) must be integrated into a demonstrator in the form of a suite of tools able to specify, evaluate, configure and monitor a distributed architecture in the customer's application context.

This methodology is being experimented in a National Project regarding the integration of web portals of local public administration (Provinces, Regions, Chambers of Commerce, Ministry of Welfare) in Italy in some thematic areas such as the healthcare domain and the Welfare and Work Support Policies.

In fact, the development of ICT structures in enterprises and in Public Administrations (PA) requires specialized security measures. The first tools able to reach this goal are the Security Policies, that is, planning and technical documents written to set up in a formal way the behavior to be followed and the measures to be undertaken in order to protect information systems. Whatever the type of activity and of the system, large or small, public or private, security issues can not be ignored at no level: hence, security issues have been accepted at the normative level by laws of the Italian State as well as at the local (regional and provinces) level, as well as at the international level - where common criteria and standards have been set up to evaluate system and application security.

The implementation, that is, the decision about the best way to design and realize a plan able to guarantee adequate security levels, is still an open problem. Various guidelines exist trying to establish core principles for professionals and managers; however, the compilation of a security plan always goes through a set of basic steps: i) the identification of assets and data to be protected, classified under suitable risk classes; ii) the identification of goals, that is, the level where an organization wants to pursue a policy of securization for its assets, accepting risk degrees and a certain effort to follow such policy; iii) the implementation and control phase, where the identified procedures are established according to the existing laws and rules, according to the configuration and nature of the system, and to the enterprise organizational or administrative structure.

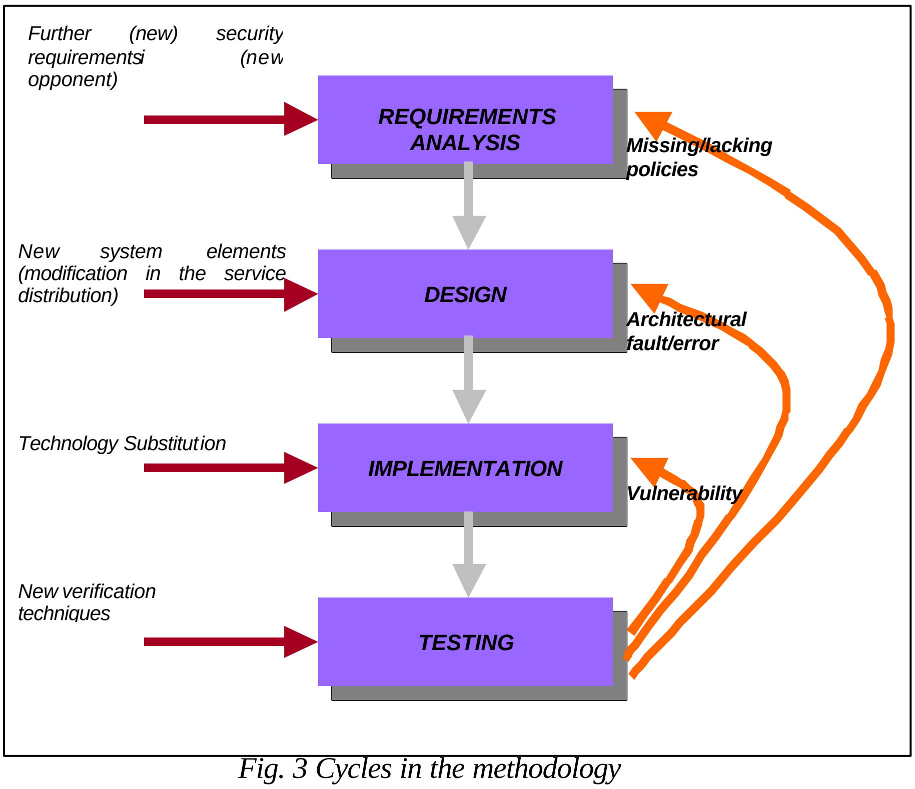
Further (new) security requirementsi (new opponent)
REQUIREMENTS ANALYSIS
Missing/lacking policies
New system elements (modification in the service distribution)
DESIGN
Architectural fault/error
Technology Substitution
IMPLEMENTATION
Vulnerability
New verification techniques
TESTING

Fig. 3 Cycles in the methodology

The methodology has to consider the context of Security Policies for administrative structures with phases and tools able to collect security requirements and rules in the form of checklists. In particular, the aim of work to be performed specifically in the e-government plan is to create a set of rules, laws and standards regarding security, trying to compose different backgrounds (of business, administrative, and technical type) to provide an overall view of the security matter useful to the creation of a set of effective policies.

PA are actually a particular context since most of services provided by PA have a deep dependence from, and impact on, legal aspects of citizens' lives and on the usual operation and performance of companies and organizations. Besides the security requirements of a large company, PA have to preserve their image and have to rule and govern the behaviour of their employees as well as of their users in the use of their networked information systems. Hence, PA have to ensure an operativity adherent to rules and laws, tackling problems such as the correct use of the digital signature and certificates since the administrative acts need to have legal value according to current laws, have to activate authentication and non repudiation systems for administrative transactions, to have to protect sensitive data and so on. We just mention the forthcoming application regarding the Electronic Identity Card and its security implications, that deserves a careful study of policies, attacks, mechanisms and interoperability measures necessary to perform all identification processing correctly and in a secure and under transaction survivable protocols.

5. Concluding remarks and discussion

The methodology presented in this paper has to consider the context of Security Policies for administrative structures with phases and tools able to collect security requirements and rules in the form of checklists. In particular, the aim of work to be performed in practical applications has to balance benefits and drawbacks and this will be done by setting up two activities: 1) a lab where security is evaluated in Cooperative Information Systems in the field of mobile and multichannel systems; 2) within the context of the Italian E-Government plan in the context of the WorkFair Virtual Network being developed in some Italian Areas.

The first activity consists in examining the security requirements of a CoopIS in networked PA environments also based on wireless connections. A lab is being created to simulate a set of e-services in the context of emergency, real-time systems and in financial applications to study the dynamic execution of e-services in these contexts and in particular the problems of security in cooperation over multichannel platforms.

The second activity consists in applying the methodology to the set of Web Services being developed within the *Portals for Employment* constituting a support to the job e-market place being designed in some Italian Regions for citizens and enterprises. The security aspects in such context are bound to the interpretation of rules, laws and standards regarding security, trying to compose different backgrounds (both organizational and technical) to provide an overall view of the security matter useful to the creation of a set of effective policies.

Public Administrations (PA) are actually a particular context since most of services provided by PA have a deep dependence from, and impact on, legal aspects of citizens' lives and on the usual operation and performance of companies and organizations. Besides the security requirements of a large company, PA have to preserve their image and have to rule and govern the behaviour of their employees as well as of their users in the use of their networked information systems. Hence, PA have to ensure an operativity adherent to rules and laws, tackling problems such as the correct use of the digital signature and certificates since the administrative acts need to have legal value according to current laws, have to activate authentication and non repudiation systems for administrative transactions, to have to protect sensitive data and so on. We just mention the forthcoming application regarding the Electronic Identity Card and its security implications, that deserves a careful study of policies, attacks, mechanisms and interoperability measures necessary to perform all identification processing correctly and in a secure and under transaction survivable protocols.

Acknowledgments

The authors are thankful to Italian partners in the COFIN projects on security, and in particularly to prof. Antonio Lioy of Politecnico di Torino and Pierluigi Plebani of Politecnico di Milano for common work and ideas.

References

Avizienis (2001) A.Avizienis, J.-C.Laprie, B.Randell, "Fundamental Concepts of Dependability", T.R.739, pp.1-21, Dept. of Computing Science, Univ. of Newcastle upon Tyne, 2001.

Blaze (1998) M.Blaze, A.Keromytis, M.Richardson, L.A.Sanchez, IPsec Policy Architecture, 1998.

Blaze (1999) M.Blaze, J.Feigenbaum, J.Ioannidis, A.Keromytis, The KeyNote Trust Management System version 2, RFC-2704, Sep.1999.

Bobbio (2001) A.Bobbio, A.Horváth. "Model checking time petri nets using NuSMV", Proc. 5th Int. Workshop on Performability Modeling of Computer and Communication System, Erlangen, Germany, Sep.2001.

Botti (1999) O.Botti, V.De Florio, G.Deconinck, et al. "TIRAN: Flexible and Portable Fault Tolerance Solutions for Cost Effective Dependable Applications", Proc. 5th Int. Euro-Par Conference on Parallel Processing, Toulouse, France, 1999.

Bray (2000) T.Bray, J.Paoli, C.M.Sperberg-McQueen, Extensible Markup Language (XML) 1.0, W3C Recommendation, Oct.2000

Butler (2000) R.Butler, et al. "A National-Scale Authentication Infrastructure", IEEE Computer, 33(12):60-66, 2000.

Cheswick (1994) W.R.Cheswick, S.M.Bellovin, Firewalls and Internet Security, Addison-Wesley, 1994.

Clark (2001) G.Clark, et al. "The Mobius modeling tool", Proc. 9th Int. Workshop on Petri Nets and Performance Models, Aachen, Germany, Sep.2001, pp.241-250.

Curbera (2002) F., M Duftler., R Khalaf., W Nagy., N Mukhi., S Weerawarana, “Unraveling the Web Services Web, An Introduction to SOAP, WSDL and UDDI”, IEEE Internet Computing March/April 2002

Damianou (2000) N.Damianou, et al.: A Language for Specifying Security and Management Policies for Distributed Systems - The Language Specification Version 2.3, Imperial College Research Report DoC 2000/1, Oct.2000.

Deswarte (1991) Y.Deswarte, L.Blain, J.C.Fabre "Intrusion tolerance in distributed computing systems", Proc. IEEE Symp. on Research in Security and Privacy, Oakland (USA), May 1991, pp.110-121.

Dierks (1999) T.Dierks, C. Allen, The TLS Protocol Version 1.0, RFC-2246, Jan.1999.

Durham (2000) D.Durham, J.Boyle, R.Cohen, et al., The COPS (Common Open Policy Service) Protocol, RFC-2748, Jan.2000.

Eastlake (2002) D.Eastlake 3rd, J.Reagle, D.Solo, "XML-Signature Syntax and Processing", RFC-3275, Mar.2002.

EC (2001) "MAFTIA: Conceptual Model and Architecture", EC Project IST-1999-11853, Universidade de Lisboa: Technical Report DI/FCUL TR-01_10, Oct. 2001.

EU (2002) European Commission, "Information Society Technologies: 2002 Work Programme".

Ferrari (1999) A.Ferrari, et al., "A Flexible Security System for Metacomputing Environments", Proc. of HPCN 1999, Amsterdam (NL).

Fugini (2002) M.G.Fugini, P.Plebani, “A Methodology for Development of Trusted Cooperative Information Systems”, Proc. Information Resource Management Association Conf., Prague, May 2002.

Godik (2002) S.Godik, T.Moses, OASIS eXtensible Access Control Markup Language (XACML), April 2002.

Gong (1998) L.Gong, "Java Security Architecture", October 1998.

Hallam-Baker (2002) P.Hallam-Baker, E.Maler, “Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML)”, April 2002

Harkins (1998) D.Harkins, D.Carrel, The Internet Key Exchange, RFC-2409, November 1998.

Helsinger (2001) A.Helsinger, W.Ferguson, R.Lazarus. "Exploring large-scale, distributed system behavior with a focus on information assurance", Proc. DARPA Information Survivability Conf. 2001, pp.273-286.

Housley (1999) R.Housley, "Cryptographic Message Syntax", RFC-2630, June 1999.

Kent (1998) S.Kent, R.Atkinson, Security Architecture for the Internet Protocol, RFC-2401, Nov.1998

CORBA (2000) Fault Tolerant CORBA, OMG Technical Committee Document ptc/00-04-04, Object Management Group, March 2000.

Iyer (1990) R.Iyer, L.Young, P.Iyer. "Automatic Recognition of Intermittent Failures: An Experimental Study of Field Data", IEEE Trans. Computers, vol.39, pp.525-537, 1990.

Laprie (1995) J.-C.Laprie, "Dependable Computing: Concepts, Limits, Challenges", 25th Int. Symp. On Fault-Tolerant Computing, pp.42-54, June 1995.

Lin (1990) T.-T.Y.Lin, D.P.Siewiorek, "Error Log Analysis: Statistical Modeling and Heuristic Trend Analysis", IEEE Trans. Reliability, vol.39, pp.419-432, 1990.

Jha (2001) S.Jha, J.M.Wing. "Survivability Analysis of Networked Systems", Proc. of the Int. Conf. On

Software Engineering, Toronto, May 2001.

Jones (2000) A.Jones, "The Challenge of Building Survivable Information-Intensive Systems", IEEE Computer, pp.39-43, August 2000.

Mecella (2001) M. Mecella, B. Pernici, “Designing Wrapper Components for e-Services in Integrating Heterogeneous Systems”, VLDB Journal, Special Issue on e-Services, 2001.

Moore (2001) B.Moore, E.Ellesson, J.Strassner, A.Westerinen, Policy Core Information Model -- Version 1 Specification, RFC-3060, Feb.2001.

Moore (2002) B.Moore, L.Rafalow, Y.Ramberg, et al. "Policy Core Information Model Extensions", draft-ietf-policy-pcim-ext-07.txt, Feb.2002.

Mylopoulos (1997) J. Mylopoulos J., Papazoglou M. (eds.), “Cooperative Information Systems”, IEEE Expert Intelligent Systems & Their Applications, vol. 12, no. 5, Sept./Oct. 1997.

Narasimhan (2002) P.Narasimhan, L.E.Moser, P.M.Melliar-Smith, "Strong Replica Consistency for Fault-Tolerant CORBA Applications", J. of Computer System Science and Engineering, 2002.

Ortalo (1999) R.Ortalo, I.Deswarte, M.Kaaniche. "Experimenting with Quantitative evaluation Tools for Monitoring Operational security", IEEE Trans. on Soft. Eng., vol.25, n.5, Sep.1999.

Pilz (2001) G.Pilz, XACML Domain Model version 3, July 2001.

President (2002) President's Commission on Critical Infrastructure Protection, Technical Report.

Ramsdell (1999) B.Ramsdell, S/MIME Version 3 Message Specification, RFC-2632, June 1999..

Sloman (2000) M.Sloman, N.Dulay, B.Nuseibeh, SecPol: Specification and Analysis of Security Policy for Distributed Systems, Imperial College http://www-dse.doc.ic.ac.uk/Projects/secpol/SecPol-overview.html .

Sobirey (1996) M.Sobirey, B.Richter, H. Konig "The intrusion detection system AID architecture, and experiences in automated audit analysis", Proc. of the IFIP Int. Conf. on Communications and Multimedia Security, 278-290, Sep.1996.

Sosnowski (1994) J.Sosnowski. "Transient Fault Tolerance in Digital Systems", IEEE Micro, vol.14, pp.24-35, 1994.

Swiler (1998) L.P.Swiler, C.Phillips, T.Gaylor, "A Graph-Based Network-Vulnerability Analysis System", Sandia Report SAND97-3010/1, Sandia National Laboratories, Jan.1998.

Vemuri (1999) K.Vemuri, J.Dugan, K.Sullivan "Automatic Synthesis of Fault Trees for Computer based Systems", IEEE Trans. on Reliability, vol.48, no.4, Dec.1999.

Wilikens (1998) M.Wilikens, et al. "Defining the European Dependability Initiative: A Strategy Document", JRC, 1998.