Towards an Approach for the Security of Information Systems with UML

Salim Chehida
Mustapha Kamel Rahmouni
Towards an Approach for the Security of Information Systems with UML

Salim CHEHIDA
Department of Informatics, University of Mostaganem, Algeria

Mustapha Kamel RAHMOUNI
Department of Informatics, University of Oran Es-Senia, Algeria

Abstract: With the fulgurating growth of the world of telecommunications, pulled by Internet and stimulated by the penetration of transmission technologies, the problems of processes and data security have currently become of paramount importance. Transactions through networks can be intercepted, above all since adequate legislation has not yet been fully enforced on the Internet. Alone, the functional specification of the information systems (IS) is not enough, for the design and the realization of these systems must take into account, in addition to the functional needs, various security requirements. Taking into account, in the modeling process, the various security constraints (Availability, Authentication, Integrity, Secrecy, Non-Repudiation, etc.) constitutes one of the principal challenges for the designer of these systems. UML is the standard language for the modeling of the multiple views of an information system by using its various extension mechanisms. In this paper, we propose new UML-extensions for the modeling of computer security requirements as well as a new development process (the X process) which takes into account the security constraints of an IS in addition to its functional needs, and also the changes and evolution of the technical architecture of the systems.

Keywords: Modeling, UML, UMLsec, Development Process, Security of IS, Software Engineering.

1. Introduction

If the generalization of the Internet connections offers new and promising possibilities, it also involves a number of risks which we should be aware of, weigh their possible consequences, and take adequate measures. A company communicates today with its subsidiaries, its partners, and that induces a massive opening to information. The networks are thus increasingly likely to be the subject of various disturbances such as congestions, malicious accesses and attacks. The number of security problems has recently drastically increased and, unfortunately, this ascending curve certainly would not dip. The companies are confronted to the problems of the secrecy, integrity and availability of information. Adding security solutions to a system that has already been functionally realized is very difficult, and can make the system instable. The security requirements should then be integrated at the design stage, so that they can be identified with the first parts of development process. The a posteriori security of critical systems (Firewall, Antivirus, etc.) does not constitute the best security policy. We think that the development of a security policy must be done at the design stage, and that the final model must integrate, at the same time, the functional and security specifications. The security of the critical systems must start with the development of a “model” which would represent: what are the threats? What do we have to protect? Why? This new approach makes the transformation of the security concept from a posteriori vision to a priori vision (at the development process level). This central activity consists in foreseeing the threats and the vulnerabilities induced by the use of the system.

UML is a standard language that is used to visualize, specify, build and document a software system. This language is not adapted to all the views of the systems: it uses extension mechanisms (Stereotypes, Labels and Constraints) to model various aspects of the system. UMLsec is an extension of UML proposed by J.Jürjens (Munich University of Technology) that includes, at the conceptual level, profiles for secure systems development. After giving the definition of the principal profiles of UMLsec, this paper proposes new extensions (in addition to UMLsec extensions) for the modeling of the security requirements of IS. This paper also presents a new development process allowing the integration of the security constraints (by using UMLsec and the new extensions suggested) in the design of IS as well as the evolution of systems architecture. This process is based on the new approaches of software engineering (MDA, XP, etc), it verifies the characteristics of UP, and it lays down a good policy of iteration as well as good progression tactics.
2. UMLsec Profiles

UMLsec is an extension of UML proposed by J. Jürjens (Munich University of Technology) that includes profiles for secure systems development. Stereotypes\(^1\) are used to formulate the conditions and requirements of security. The tables below show some UMLsec stereotypes with their corresponding labels\(^2\).

<table>
<thead>
<tr>
<th>Stereotype</th>
<th>Description</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secure Dependency</td>
<td>Package to identify the secure dependency relations in the static models</td>
<td>secrecy, integrity, high, fresh</td>
</tr>
<tr>
<td>Secure Links</td>
<td>Package to identify the secure dependency relations between the system’s components</td>
<td></td>
</tr>
<tr>
<td>Data Security</td>
<td>Package to specify the critical objects and the various properties of security on the data</td>
<td></td>
</tr>
<tr>
<td>Fair Exchange</td>
<td>Package to represent the fair exchange scenarios in the electronic transactions</td>
<td>start, stop</td>
</tr>
<tr>
<td>No Down – Flow</td>
<td>Package to secure the information flow</td>
<td>high</td>
</tr>
<tr>
<td>Provable</td>
<td>Package to express non-repudiation in the electronic transactions</td>
<td>action, cert</td>
</tr>
<tr>
<td>Guarded Access</td>
<td>Package to control the access to the objects</td>
<td></td>
</tr>
<tr>
<td>Internet</td>
<td>Internet connection</td>
<td></td>
</tr>
<tr>
<td>Encrypted</td>
<td>Encrypted connection</td>
<td></td>
</tr>
<tr>
<td>LAN</td>
<td>Local area network connection</td>
<td></td>
</tr>
<tr>
<td>Secrecy</td>
<td>Confidentiality of dependence</td>
<td></td>
</tr>
<tr>
<td>Integrity</td>
<td>Integrity of dependence</td>
<td></td>
</tr>
<tr>
<td>Guarded</td>
<td>Guarded object</td>
<td>guard</td>
</tr>
<tr>
<td>LAN</td>
<td>Local area network node</td>
<td></td>
</tr>
<tr>
<td>Smart Card</td>
<td>Smart card node</td>
<td></td>
</tr>
</tbody>
</table>

3. UML Security Extensions

The extensions which we have just proposed concern different views of the system: the secure context model, security cases and critical scenarios for the specification of the security requirements, the secure interactions of objects and the security constraints on the data for the logical view and finally the protected hardware configuration for the technical view of the system. As an illustration, examples of the COMEX system, an information system of Commercial Management for a Harbor Company, will be presented.

3.1. Secure Context Model

Many authors, like G. Booch in [8] or more recently P. Roques and F. Vallee in [18], recommended the use of collaboration diagrams to represent, in a synthetic manner, the various functional requirements of a system. After the definition of security conditions, we can present the various security requirements on a diagram, which can be called secure context model. This model consists in defining the various expected security services of the system considered as a black box. The collaboration diagram is used in the following way:

- The system is represented by a central object; this object is surrounded by other objects symbolizing the various actors.
- The objects are connected by bonds; on each bond are shown output messages which represent the various security services provided by the system.

---

\(^1\) The stereotypes make it possible to extend the semantics of the modeling elements and to define new UML elements classes.

\(^2\) A label or marked value is a pair (name, value) which adds a new property to UML modeling element.
3.2. Security Cases Model

In this model, we are interested in the specification of the system requirements in terms of security. To do that, we use the use cases in a different manner by introducing the concepts of security cases and security cases model. The security cases model is used to structure the security services provided by the system (always considered as a black box) for the various actors as a set of security cases. A security case represents a security service returned by the system for one or more actors. For example: to verify the identity of an user, to ensure the integrity and the secrecy of the exchanged information, to ensure the non-repudiation of transactions, etc.

A security case specifies an awaited system behavior to meet security needs without imposing the realization mode of this behavior. It makes it possible to describe what the future system will have to do in terms of computer security without defining how to do it. Security cases are distinct from use cases: they do not produce a functional added value but they indeed cover all security services that a user needs. Figure 2 presents an example of a security cases model.

We used three constraints\(^3\) for the interactions between system and actors:

- The \{secrecy\} constraint to ensure the secrecy of the interactions.
- The \{integrity\} constraint to ensure the integrity of the interactions.
- The \{identity\} constraint to ensure the identity of the parties during the execution of an action of interaction between an actor and the system.

3\(^3\) A constraint is a semantic relation between UML modeling elements. Each constraint is indicated between braces and is placed close to the element (stereotyped or not).

3.3. Critical Scenarios

The critical scenarios consist in describing and representing the critical interactions or actions using the various services of security specified by the security cases. A critical scenario represents a particular succession of sequences (interactions between the actors and the system) which involves a risk in terms of computer security. To underline this risk, we will associate the various constraints of security on the interactions between the system considered as a black box and the various actors. For example: the scenarios which ensure the non-repudiation in the electronic transactions and the scenarios which specify the interactions with exchange of critical information. For the description of the critical scenarios, we used the diagram of sequence which makes it possible to better visualize the interactions. Figure 3 presents an example of a model which emphasizes critical exchanges.

3.4. Secure Interactions of Objects

After the identification of the classes and objects of the system (the Static Model), we now replace the system by a collaboration of objects. A scenario of secure interactions of objects represents an ordered set of messages exchanged by objects (instances of classes and actors) with the specification of the security constraints on these messages. A message represents the specification of a one-way communication between objects which transports information and whose goal is to generate a reaction from the receiver. It can include parameters which transfer values from the transmitter to the receiver. [21] For the representation of secure interactions of objects, we used the sequence and the collaboration diagrams of the UML. Figure 4 presents an example of a model of secure interactions of objects.

- The \{secrecy\} constraint to ensure the secrecy of the messages;
- The \{integrity\} constraint to ensure the integrity of the messages;
- The \{identity\} constraint to ensure the identity of the transaction parties.
3.5. Data Security

The set of security cases discovered through the specification of security constraints guides all the dynamic views, by representing the critical scenarios, the collaborations and the interactions of objects with the sequence diagrams. In order to benefit from the security analysis phase, it is necessary to update the class diagram by adding security constraints on the data. The class diagram is viewed as the most important diagram in the object methods. After having developed the class diagram, we will define security constraints on the attributes and the operations starting from the critical scenarios represented on message flows between objects.

- The \{secrecy\} constraint specifies the data being confidential.
- The \{integrity\} constraint is used to ensure the integrity of the data.
- The \{identity\} constraint indicates that only the authorized parts can reach the data.

3.6. Protected Hardware Configuration

The protected hardware configuration model consists in expressing the implementation constraints at the physical level represented by nodes and connections, which are the various types of machine connected by various means with the integration of the prevention tools (Firewall, IDS, etc) to implement the security constraints. This model also allows representing the types of connections (LAN, VPN, etc) between the various nodes.

The deployment models and hardware configuration models are both expressed by using a deployment diagram. However, they do not quite express the same description level. The hardware configuration model is used to express the constraints of implementation at the physical level. It consists of the nodes and the physical connections of the system. On the other hand, the deployment model expresses the physical distribution of the system’s functions and permits to justify the localization of the data bases and working environments. Figure 6 presents an example of a protected hardware configuration model.
4. Development Process

The X process is a new method of modeling which we have just proposed in this paper. "X" means literally that the process follows two ways in top and two ways in bottom what gives the shape in X, the two ways top are used for the parallel specification of the functional needs and the security constraints, and the two ways of bottom correspond to the two axes of the changes imposed on the information system: it is of the logical or design view which describes the aspects static and dynamic of system in terms of classes and objects and the technical view which is concerned with specification of the technical architecture of the system.

4.1. Axioms Founders

The X process is based on the two following axioms:

1) Functional specification of IS is not sufficient to solve the various problems related to the integration of the networks in these systems, like security problems. Design and the realization of IS must hold account, in addition to the functional needs, of the various constraints of security. We can thus proceed to a parallel specification, along a left axis “functional” and an axis right “of security”. With evolutions of the functional and security specifications, the systems design will consist in amalgamating the results of these two branches of the process.

2) In order to satisfy the companies’ needs, several technical technologies and architectures are available to ensure the development of quality software products. The logical organization, which describes the static’s aspects and dynamics of the system in terms of classes and objects, is indeed independent of technologies used. The second axion founder of the X process consists in treating in parallel the logical and technical views system following approach MDA\(^4\) starting from the functional and security needs previously specified. Lastly, the realization of the system consists in integrating the logical model in technical architecture.

4.2. Re-Use and Maintenance

The phase of specification of the functional needs produces models for the means and long term in order to capitalize the information system function. In addition, the phase of specification of the security constraints produced models for the short term in order to define the requirements of security measures imposed on the information systems after the capture of the threats comes from the environment. If the environment requires news give in term of security, it is enough to integrate the specification of the new constraints of security in the logical model and technical architecture to update the system without last by the specification functional. The phase of the analysis and design produces models for the means and long term in order to represent the logical view of system in term of classes and objects. In the other dimensioned, the phase of technical architecture makes it possible to develop models for the court and medium term in order to specify the technical design of the system. The logical model is indeed independent of technical architecture. We can thus carry out the same logical model under various architectures and technical technologies dependant on same functional and security requirements.

4.3. Structure and Description of the Phases

The following figure presents the general outline of the X process. The parallel specification, on a side of the functional needs and security constraints, and on another side of the logical design and technical architecture of the system gives the shape of X to the process.

\[\text{Fig 7. The General Outline of the X Process}\]

4.3.1. Initial Collection of the Needs

The initial collection of the needs is the first phase of the X process. This stage plays a very important part because it constitutes the starting point of the system modeling process. It consists in carrying out a first location of the functional needs and security requirements by using the text to define the conditions (functional and security), and simple diagrams (functional and secure context model) to visualize

---

\(^4\) MDA (Model Driven Architecture): approach which proposes a systematic and clear separation of business logic and the implementation logic. OMG: Initiate MDA in 2000.
the context of the system. The principal objective of this phase is thus to prepare the two phases of specification.

4.3.2. Specification of the Functional Needs

This phase aims to supplement the initial collection of the functional needs carried out during the preceding phase, and to define the functional border between the system and its environment by specifying the awaited activities of the various system users. This stage produces a model which makes it possible to control the good adequacy of the functional needs with those for the users. The technique of the use cases is the angular stone at this stage. It makes it possible to specify the whole of the interactions sequences between the system and its actors. This phase also consists in describing the dynamics of use cases by using the text and the various dynamic models.

4.3.3. Specification of the Security Constraints

This phase consists in supplementing the specification of the various security constraints listed in the phase of initial collection of the needs, and which dimension the logical and technical design system. Into this phase, we introduce the concepts of security cases and security cases model. A security cases is used to represent the security services provided by the system for the various actors. This phase also consists in describing the critical scenarios using the various dynamic models of UML, as well as the identification of the possible attacks.

4.3.4. Collaboration and Validation

This phase consists in coordinating between the models of the two branches of specification, then to validate the functional needs and the security constraints with the customer. In the context of an iterative and incremental process, controlled by the risks, this stage also makes it possible to share the project in iterations by assigning risk levels to each use case and security case, while starting with the most critical cases in terms of project management in order to cancel the risks of failure.

4.3.5. Analyze and Design

This phase describes the static and dynamic aspects system in terms of classes and objects, as well as the collaborations and the interactions between these objects. It produces a design model of system which defines the structure and the behavior of the objects of the system users within the application of their functional and security needs. The interactions between the objects can be described by means of two UML diagrams: the sequence diagram and the collaboration diagram. This stage also organizes the classes and the objects of the system in categories according to logical criteria.

4.3.6. Technical Structures

This phase consists in counting all the technical constraints and choices. It produces models that make it possible to express the implementation and physical constraints (nodes and connections physical of the system), and to define an architecture based on preventive provisions, taking in consideration the functional and security constraints to ensure the security system against potential threats.

4.3.7. Integration, Coding and Tests

This phase consists in integrating the logical model in the technical architecture, as well as the coding and the realization of the system. This phase also makes it possible to test the code units carried out and to validate the functions of the developed system.

4.4. Policy of Iteration and Project Management

After the initial collection of the needs, and the definition of security constraints and functional needs with the customer, the system is dividing into strongly coherent functional packages (in term business) and slightly coupled (independent). For this purpose, it is advisable to specify the security packages allowing the structuring of the security constraints related to each functional package. A functional package together with its security correspondent are called Business package. It will be then requested from the customer to define priority levels for each business package in order to deliver the most demanded functions. The phase of specification is done by iteration: each iteration corresponds to a business package. After the identification of the use cases and security cases of system, the project leader validates these cases with the customer or the actors concerned. If the whole of the requirements ensured by the use cases and security cases does not meet the needs of customer, the team of specification must go back to the specification phase and correct the errors.

After the validation of the use cases and the security cases of the business package corresponding to iteration, we will have to associate risk levels to each case. For this purpose, it will be necessary to start with the most critical use and security cases in terms of project management in order to eliminate the major risks. Once the assignment of the risk levels is established, one shares the project in iterations. Each iteration includes a whole of use and security cases. The progression of the logical and technical design is also of an iterative type. The analysis and the logical design start with the identification of the classes that are candidates, starting from the use cases of the same iteration. These classes will then be detailed, supplemented and optimized. The work of refinement consists in adding, modifying or removing classes, associations or attributes. One benefits from the security branch to define the security constraints on the data, and from the dynamic analysis for the addition of the operations. After the development of the logical model of iteration, one specifies the technical architecture allowing the exploitation of
this model. If the logical and technical models of iteration achieve the laid down goals, one then goes through the implementation. The stain of software industrialization relate to the installation of the means and the tools which will allow the release of iteration.

4.5. The X, UP and XP Processes

The X process complies well with the characteristics of an Unified Process (UP) and it is based on a disciplined approach focused on Extreme Programming (XP) in order to control, throughout the cycle of process development, the assignment of the tasks and the roles of the various actors.

4.5.1. The X Process Verifies the Characteristics of Unified Process

The Unified Process is a development process of software built on UML; it is iterative, incremental, centered on architecture and is controlled by the requirements of the users. The X process verifies these various characteristics.

The X process build starting from UML

It is difficult to consider the X process without covering it with UML like support; the concepts used are specifically related to UML.

The X process is controlled by the requirements of users:

The requirements of the users are first approached by considering two types of needs:
1. The functional needs which correspond to the business functions of the system.
2. The security needs (requirements) which correspond to the security services which must be fulfilled by the system. For each use case, one carries out the description of the whole of interactions between the actors and the system. The concepts used in this description highlight the various classes and objects of the system. On the other side, security cases and critical scenarios allow the definition of critical objects and classes as well as security levels on the data.

For the architecture design, the technical choices must be controlled by the functional needs and the security constraints, the hardware configuration must integrate the preventive devices, and the exploitation components of the system must meet the functional and security requirements.

The X process is iterative and incremental

The choice of an iteration rests on two factors:
- An iteration takes into account a certain number of use cases and security cases.
- An iteration treats the major risks in priority. With each iteration, the designers identify and specify the use and security cases, create a logical and technical design, integrate and implement these designs in components and check that those conform to the use cases and the security cases. As soon as iteration answers the above objectives, the development goes through the following iteration.

Within the delivery of the firstly required parts of system, our policy of project management consists in defining priority levels for each business package in order to deliver the most required parts. Within the reduction of the failure risks, it is advisable to associate risk levels for each use and security case, in order to start with the most critical cases in terms of project management.

The X process is centered on architecture

The architecture of a software system can be described like the various views of the system. The architecture “4+1” views proposed by PH.Kruchten [16] present five overlapping views: the user's needs view, the logical or design view, the processes view, the components or realization view and the deployment view. Each view is a projection in the organization and the structure of the system which is interested in a particular aspect of this system. The various views are approached by the X process.

4.5.2. The X Process is Based on XP

Extreme Programming is an agile method based on practices intended to organize the work of a development team. More generally, practices XP are underlain by the four following principles [22]:
* The Communication: XP supports human contact, direct communication, rather than the bulk-heading of the activities and the exchanges of formal documents. The developers directly work with the customer and the testers are integrated into the team of development.
Feedback: XP Practices are conceived to give a maximum of feedback on the course of the project in order to correct the trajectory as soon as possible. In particular, the beginning points of any iteration offer to the team the means of feedback on its operation and of constantly improving it in the iterations.

Simplicity: XP takes up the following challenge: “what can we stop making while effectively creating a software which meets the real needs for the customer?”. This search for simplification touches the process itself, but also the tools manufactured and the design of the application.

Courage: it is mainly a question of courage to maintain a frank and open communication, and to accept and take action if news is bad, etc.

To check these four principles, the X process implements the following practices:

**Team work and the implication of the customer:**

The X process shares development in several teams (team of functional specification, team of security specification, logical design teams, technical teams, coding, realization and testers team) and supports the communication between the various teams as well as the communication with the customer or the users.

![Fig10. Actors of the X Process](image)

**Programming Controlled by The Tests:**

For each planned scenario, a whole of receipt tests is elaborated. These tests consist in checking each functionality required by the customer. The customer must thus take part in these tests. In complement of the receipt tests, which are used to prove to the customer that the software fills its objectives, the X process carries out unit tests. These tests make it possible to specify and validate the behavior of each portion of code added.

**Iterative Cycles Controlled by The Customer:**

The team regularly delivers versions of the software; the releases of new versions are connected at an intensive pace to obtain a maximum feedback on the advance of the developments. At the beginning of each iteration, the customer and the various teams of development meet for the planning. This meeting is presented in the form of a real working session, which gives to the various actors the opportunity to align their comprehension on what must be carried out.

5. Conclusion

In this paper, we have tackled the highly vast subject of computer security, while concentrating on security at the model level. It is a transverse approach, where the concept of security is being included in the modeling of critical systems, and where the UML extensions are able to help master the control of security. The security model is a representation of security derived from a “vision of the world”. The model defines what must be defended (type and flow of information), against what (threats) and why (sensitivity of information). It can be more or less complete, but in all cases it just emphasizes the risks from where we can deduct a security policy. [6]

UML is not a closed notation: it is generic, extensible and configurable by the user. Where necessary, we can use extension mechanisms. This paper presents new profiles of UML for the modeling of security aspects. The secure context model and the security cases model for the specification of the security needs, the critical scenarios model consists in describing the interactions or the actions which involve a risk and the secure interactions of objects model for the specification of the security constraints on the messages exchanged by objects. In the analysis model, we define security properties for the data. At last, for the modeling architecture, the protected hardware configuration model allows to express the implementation constraints at the physical level with the integration of the prevention tools in order to fulfill the security requirements.

To face the difficulties of the designers for apprehending the security concepts, this paper proposes a step of security control within a development process. The X process checks the characteristics of an unified process and allows the implementation of the various practices of the XP agile method (Extreme Programming). It constitutes a screen to integrate the best practices in modeling. This process allows a parallel specification of the functional needs and security constraints, as well as a parallel design of the logical and technical view of a system. This process also supports the team work and meets the needs for systems evolution. The X process lays down a good policy of iteration controlled by the risks and the priorities of the customer, as well as good tactics of progression while defining the objectives to be reached for each phase. Lastly, the door remains open for other extensions in the various development phases.

The important points which remain to be developed are:

- The realization of attacks simulations on UML protected models in order to validate these models and to correct the security faults if required.
- The translation of the secure models in code.
- The integration of the various approaches of risk analysis in the development process.

381
The realization of automatic tools for the generation of protected models

References: