The CORAS Approach for OPM Based Risk Management

Qian DONG, Meng-qi JI, Yi-fan ZHU and Feng YANG
National University of Defense Technology, Changsha, Hunan, China

Keyword: OPM, CORAS framework, Risk analysis.

Abstract. The CORAS approach supports a model-based risk assessment on security critical systems. OPM paradigm integrates the object-oriented, process-oriented, and state transition approaches into a single frame of reference by employing a combination of graphics and a subset of English statements. This paper presents an overview of the CORAS framework based on OPM paradigm and provides an example of risk analyses to show how the CORAS framework works in risk management process.

Introduction

Risk is the uncertainty that something bad still occur. The function of risk management attempts to identify, evaluate, and mitigate potential risks that could result in a loss.

Risk management is the identification, assessment, and prioritization of risks (defined in ISO 31000 as the effect of uncertainty on objectives) followed by coordinated and economical application of resources to minimize, monitor, and control the probability and/or impact of unfortunate events or to maximize the realization of opportunities. Risk management focus changes during a system’s lifecycle, with objectives to reduce the probability of occurrence of risky processes and their adverse impact on stakeholder objectives and assets.

Project risk management (PRM) is concerned with reducing delays and cost overruns, while satisfying specification (spec) and quality requirements (Chapman and Ward, 2003). Hoffman and Haimes’s studies [3,4] indicated that system objectives in Operational risk management (ORM) such as reliability, safety, security, availability, and business continuity in operational settings subject to risk. Analytical risk-integrated system modelling attempts to define the system’s (multi)objective function, while capturing risk, using mathematical building blocks such as input, output, state variables, decision (control) variables, and random variables.

Several integrated system design and risk management frameworks have been developed as follow: CORAS (an object-oriented UML-based framework), Quantitative risk assessment for component-based systems, BRPIM (business process risk integrated modelling-framework), RiskM (a multi-perspective method). Among these frameworks, CORAS is considered the most thorough and wide-ranging framework, which develops a practical framework for model-based risk management of security critical systems by exploiting the synthesis of risk analysis methods with semiformal specification methods. CORAS diagrams focused on its risk management process phases, therefore UML conventions are not suitable for all diagrams.

Contrasted with aspect decomposition, characteristic of UML/SysML, OPM copes with complexity via detail-level decomposition. Object-Process Methodology (OPM), a holistic, integrated approach to the design and development of systems, which is an ontology- and systems theory-based vehicle for knowledge representation and management that perfectly meets the formality and intuition requirements through a unique combination of graphics and natural language.

The rest of this paper is organized as follows. Section 2 presents the OPM approach, its goals and objectives, and its configuration and design modelling theory and methodology. Section 3 provides related knowledge on risk management especially CORAS project, model-based risk analysis (MBRA). Section 4 demonstrates the application of risk management using OPM development tool OPACT. Finally, Section 5 summarizes this paper and delineates directions of future and complementary research.
The OPM Ontology

More specifically, ontology is the study of the categories of things that exist or may exist in some domain (Sowa, 2001). The product of such a study, called ontology, is a catalog of the types of things that are assumed to exist in a domain of interest from the perspective of a person who uses a specific language, for the purpose of talking about that domain.

Basic Knowledge

The elements of the OPM ontology, shown in Figure 1, are divided into three groups: entities, structural relations, and procedural links.

![Figure 1. The three groups of OPM symbols in the toolset of OPCAT.](image)

Objects are (physical or informatical) things that exist, while processes are things that transform (create, destroy, or change the state of) objects. Table 1 shows things that transform objects in the system.

<table>
<thead>
<tr>
<th>Thing / Attribute</th>
<th>Symbol</th>
<th>Description / OPL sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>![Object Symbol]</td>
<td>A thing (entity) that has the potential of stable, unconditional physical or mental existence. Object Name is an object.</td>
</tr>
<tr>
<td>Process</td>
<td>![Processing Symbol]</td>
<td>A thing representing a pattern of transformation that objects undergo. Processing is a process.</td>
</tr>
<tr>
<td>Essence</td>
<td>![Object Symbol]</td>
<td>An attribute that determines whether the thing (object or process) is physical (shaded) or informational. Processing is physical.</td>
</tr>
<tr>
<td>Affiliation</td>
<td>![Object Symbol]</td>
<td>An attribute that determines whether the thing is environmental (external to the system, dashed contour) or systemic. Processing is environmental.</td>
</tr>
</tbody>
</table>

Besides that, objects can be stateful, to be specific, a state is a situation at which an object can exist at certain points during its lifetime or a value it can assume.

Fundamental structural relations are a set of four structural relations used to denote relations between things in the system. The four fundamental structural relations are described in Table 2 and relation instances are shown in Table 3:

Table 2. The fundamental structural relation names and OPD symbols.

<table>
<thead>
<tr>
<th>Structural Relation Name</th>
<th>Forward</th>
<th>Backward</th>
<th>Description</th>
<th>Symbol Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation</td>
<td>Participation</td>
<td></td>
<td>denotes the relation between a whole thing and its parts,</td>
<td>![Aggregation Symbol]</td>
</tr>
<tr>
<td>Exhibition</td>
<td>Characterization</td>
<td></td>
<td>denotes the relation between a general thing and its specializations, giving rise to inheritance</td>
<td>![Exhibition Symbol]</td>
</tr>
<tr>
<td>Generalization</td>
<td>Specialization</td>
<td></td>
<td>denotes the relation between an exhibitor (attributes and/or operations) and the things that characterize the exhibitor</td>
<td>![Generalization Symbol]</td>
</tr>
<tr>
<td>Classification</td>
<td>Instantiation</td>
<td></td>
<td>denotes the relation between a class of things and an instance of that class.</td>
<td>![Classification Symbol]</td>
</tr>
</tbody>
</table>
Table 3. Structural relation and OPL representation.

<table>
<thead>
<tr>
<th>OPD with 3 Refineables</th>
<th>OPL Sentences with 1, 2, and 3 refineables</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Diagram" /></td>
<td>A consists of B. A consists of B and C. A consists of B, C, and D.</td>
</tr>
<tr>
<td><img src="image2.png" alt="Diagram" /></td>
<td>A exhibits B. A exhibits B and C. A exhibits B, C, and D.</td>
</tr>
<tr>
<td><img src="image3.png" alt="Diagram" /></td>
<td>B is an A. B and C are As. B, C, and D are As.</td>
</tr>
<tr>
<td><img src="image4.png" alt="Diagram" /></td>
<td>B is an instance of A. B and C are instances of A. B, C, and D are instances of A.</td>
</tr>
</tbody>
</table>

Complexity Management

As noted, the complexity of an OPM model is controlled through refinement/abstraction mechanisms to address the main requirements: completeness and clarity, which respectively means, the system must be specified to the last relevant detail and the resultant model must be legible and comprehensible. The *in-zooming/out-zooming* mechanism, in which the entity (primarily a process or a state) being detailed is shown enclosing its constituent elements; the *unfolding/folding* mechanism, in which the entity (primarily an object) being detailed is shown as the root of a structural graph; and state expressing/suppressing, which allows for showing or hiding the possible states of an object. These mechanisms enable OPM to recursively specify a system to any desired level of detail without losing legibility and comprehension of the complete system.

For example, in Figure 2, the process F1 is unfolded to expose its parts, processes F1.1 and F1.2, and its feature, object S1.1. The in-zooming/out-zooming mechanism exposes/hides the inner details of a thing within its frame. The process F1 is in-zoomed, showing the process flow – first F1.1 is executed and creates object S1.1, and then process F1.2 is activated, consuming object S1.1.

![Figure 2. Applying OPM scaling mechanisms to process F1.](image.png)

Reasons to Choose OPM

OPM paradigm integrates the object-oriented, process-oriented, and state transition approaches into a single frame of reference by employing a combination of graphics and a subset of English. Instead of highlighting one aspect at the expense of suppressing the other to enhance the comprehension of the system as a whole, structure and behavior coexist in the same OPM model.

Though OPM methods, various system aspects can be inspected in tandem for better comprehension. Complexity is managed via the ability to create and navigate via possibly multiple detail levels, which are generated and traversed through by several abstraction/refinement mechanisms.

Contrary to UML and its nine diagram types, OPM illustrates system’s structure, behavior and architecture in the same diagram type, enabling the expression of mutual relations and effects between them and reinforcing the understanding of a system as a whole, thus, providing a solid basis for representing and managing knowledge about complex systems, regardless of their domain. Several reasons to choose OPM as the basis for this framework due to its following features:
1. Using a single diagram type at varying levels of detail to unify the static-structure and dynamic-procedural aspects, which reduces clutter and incompatibilities, even in highly complex systems.

2. Decomposing system specification into self-similar OPDs in great detail and managing system complexity through recursive seamless refinement-abstraction mechanisms.

3. Combining semantically equivalent graphical and textual views, which makes OPM catering to systems architects, domain practitioners.

4. Enabling extending the core system model to additional aspects while maintaining full coordination with the core model, as well as the capability to generate meta-models.

5. A freely available OPM-based CASE tool – OPCAT, which implements almost all the OPM concepts, is allowed for implementation.

The CORAS Framework

CORAS is a research and technological development project. One of the main objectives of CORAS is to develop a practical risk management framework, exploiting methods for risk analysis, semiformal methods for object-oriented modelling, and computerised tools, in other words, unambiguous, and efficient risk assessment of security critical systems.

Objectives

The intention of the CORAS project is to develop an integrated framework for model-based risk analysis of security critical systems. Providing an integrated methodology to aid the design of secure systems is the overall objective for the CORAS project as following:

1) Aimed at improving the risk analysis of security critical systems. By exploiting the synthesis of risk analysis methods with semi-formal specification oriented modelling methods and computerised tools, to put a practical framework for a precise, unambiguous and efficient risk analysis into use. The emphasis of this synthesis is on appropriately combine of risk analysis methods to the security critical systems and using (semi-)formal modelling techniques to obtain a suitable overview of complex security critical systems;

2) Through extensive experimentation in real applications such as the fields of e-commerce and telemedicine, Assessing the applicability, usability and efficiency of this framework which places particular emphasis on information security defined broadly by:
   - Confidentiality: only appropriate access is allowed to data;
   - Integrity: no unauthorised changes (either in storage or transmission) are made to data;
   - Availability: ensuring that data is accessible as required;
   - Accountability: ensuring that users are accountable for their security.

OPM Based Representation of CORAS Framework

The CORAS framework for a risk management process based on model includes four parts: a system documentation framework, a model-based risk management process, a system development and maintenance process and a platform for tool-integration. The relationships between them are illustrated by OPM modelling method, as shown in the following Figure 3:

Figure 3. The CORAS framework for a model-based risk management process.
Through the zoom and in-zoom mechanism of OPM approach, the detail descriptions of these parts are provided in the following sections.

**The CORAS System Documentation Framework.** As indicated by Figure 4, the CORAS system documentation framework is based on RM-ODP which defines the standard reference model for distributed systems architecture, based on object-oriented techniques, accepted at the international level. The CORAS documentation framework contains:

1. Viewpoints that divides the system documentation (Figure 5).
2. ODP foundations that provides modelling, specification, structuring, risk management and security concepts and terminology (Figure 6).
3. Distribution module that defining transparencies and functions required to realize these transparencies (Figure 7).
4. Conformance module that addressing implementation, consistency and conformance checking requirements (Figure 8).
5. Risk management module containing risk assessment methods, risk management processes, and a specification of the international standards on which CORAS is based (Figure 9).

![Figure 4. The CORAS System Documentation Framework.](image)

![Figure 5. Viewpoints.](image)

![Figure 6. ODP foundation.](image)

![Figure 7. Distribution module.](image)

![Figure 8. Conformance module.](image)

![Figure 9. Risk management module.](image)

**The CORAS Platform for Tool Integration.** The CORAS platform is based on data integration implemented in terms of XML technology. Figure 10 outlines the overall structure.

![Figure 10. Platform for tool integration.](image)

The platform is built up around an internal data representation formalized in XML/XMI. Standard XML tools allow experimentation with the CORAS platform and can be used by the CORAS crew during the trials. Relevant aspects of internal data representation can be mapped to the internal data representations of other tools based on the eXtensible Stylesheet Language (XSL). This allows the integration of sophisticated case-tools targeting system development as well as risk analysis tools and tools for vulnerability and treat management.

**The CORAS Model-Based Risk Management Process.** As indicated by Figure 11, The CORAS risk management process provides a sequencing of the risk management process into the following five sub-processes:

1. Context Identification: Identify the context of the analysis that will follow. The approach proposed here is to select usage scenarios of the system under examination (Figure 12).
2. Risk Identification: Identify the threats to assets and the vulnerabilities of these assets (Figure 13).

3. Risk Analysis: Assign values to the consequence and the likelihood of occurrence of each threat identified in sub-process 2 (Figure 14).

4. Risk Evaluation: Identify the level of risk associated with the threats already identified and assessed in the previous sub-processes (Figure 15).

5. Risk Treatment: Address the treatment of the identified risks (Figure 16).

The CORAS methodology for a model-based risk management process builds on:
– HAZard and OPerability study (HAZOP);
– Fault Tree Analysis (FTA);
– Failure Mode and Effect Criticality Analysis (FMECA);
– Markov analysis methods (Markov);
– Goals Means Task Analysis (GMTA);
– CCTA Risk Analysis and Management Methodology (CRAMM).

Their relationships can be provided as Figure 11. The sub-process are shown from Figure 12 to Figure 15:
**The CORAS System Development and Maintenance Process.** The CORAS system development and maintenance process is based on an integration of the AS/NZS 4360 standard for risk management and an adaptation of the RUP for system development. RUP is adapted to support RM-ODP inspired viewpoint oriented modelling. Emphasis is placed on describing the evolution of the correlation between risk management and viewpoint oriented modelling throughout the system’s development and maintenance lifecycle. The process could be described as Figure 17:

![Figure 17. The CORAS system development and maintenance process.](image)

**OPM-based Use Case**

In this section, an OPM-based Use Case would be proposed to experiment with all aspects of the overall assessment. The trial was based on the authentication mechanism. The description of the trial could be divided into three parts: dynamic behavior scenario, risk scenario description and the risk management process.

**Dynamic Behavior Scenario**

Using the modelling approach presented above, Figure 18 provides a high-level description of the E-Commerce platform behavior with respect to the user authentication and identification.

![Figure 18. e-Commerce user authentication behavior.](image)

According to the above chart, when a user accesses the Login page, the server creates a unique session ID to identify the unique client. The session ID associates each user’s client with the user’s data stored on the server and it is denoted as the parameter “(sn)”. Similarly, the username and password also seen as parameters. Users can also, access the platform as visitors without authentication but cannot use all functions.

The server stores the session numbers of all users, both registered and visitors, and associates the registered ones with their corresponding data. When a user logs out, A session terminates in which case the session number used to identify the session is removed from the database and the user is presented with the “Logout” page.

Invalid requests are those where the parameters were modified with invalid values. In these cases, the server responds with the Login page. Following some time without interaction with the server, a session times-out and the corresponding user is logged off. The use of time-outs prevents the use of bookmarks for accessing specific pages in the platform.
The use of session numbers for client identification has some repercussions in the behavior of the application. For example, a session does not terminate immediately when a client disconnects. Users can bookmark a specific application page, exit their browser, then restart the browser, and go to the specific page via the bookmark before the timeout period expires.

**Risk Scenario Description**

The use of session numbers for client identification can have undesirable consequences if a malicious actor captures a client’s session ID. This actor can use the session ID to login in the platform using a second account with an offensive profile (e.g., vulgar names) using the legitimate user’s session ID. From that point on, all interactions of the legitimate user with the platform will have the profile of the second account. This behavior can be expressed in Figure 19:

![Figure 19. Risk scenario description.](image)

A legitimate user, Client-A, logs in the platform with the username/password un-A/pw-A, using session ID sn-A, and receives back a page personalized to Client-A user-A. After a malicious user, Client-B, logs in with the session ID of Client-A, the profile that Client-A accesses is the profile of Client-B (e.g., the Client-B’s name and shopping lists).

This example is not the result of implementation defects, but a consequence of the design decision to use session IDs for client identification. It should be noted that the use of cookies is subject to similar deficiencies.

**Risk Management Process**

In this section, the OPM-based CORAS framework is used to demonstrate the specific process of risk management. The overall process is shown below:

![Figure 20. The OPM-based CORAS framework of the scenario.](image)

Then the sub-processes are presented through the refinement mechanism in OPM. First step is context identification with the help of CRAMM and HAZOP. Initially CRAMM was applied in order to provide an identification of assets, which in turn provide a basis and justification for the security requirements that the mechanism need to meet as the Figure 21:
The second step is to identify risks which contains three activities as identify threats to assets, identify vulnerabilities of assets and document unwanted incidents. In this case, the threats to assets are browser or proxy misconfiguration and the openness of internet as shown in Figure 22:

The third step is risk analysis includes consequence evaluation and frequency evaluation, for consequence evaluation is used to specify consequences which can describe all possible behavior in the system as shown below:

The forth step is risk evaluation which is used to determine risk management priorities by comparing the level of risk against predetermined standards, target risk levels or other criteria which contains five activities as presented in Figure 24:
The last step is risk treatment which is the selection and implementation of appropriate options for dealing with risks which can be divided into two activities as shown in Figure 25:

![Figure 25. The risk treatment of the scenario.](image)

**Summary**

Built on the foundational ontology represented by most minimal set of stateful objects and processes that transform them, the domain-independent nature of OPM makes it suitable as a general, comprehensive, and multidisciplinary framework for knowledge representation and reasoning. OPM owns the ability to provide comprehensive lifecycle support of all complexity levels such as conceptual modeling, design, implementation, analysis and lifecycle management. Besides that, the dual knowledge representation in graphics and text and the capability to automatically switch between these two modalities, making OPM cater to represent and manage knowledge. Consequently, OPM offers an improved methodology for risk management framework.

The study attempted to propose a CORAS framework based on OPM methodology that is used in the risk management. Through literature review, the study analyzed the relevant approaches proposed by previous studies and the characteristics of CORAS framework. After considering the requirements of the framework, the study adopted the OPM approach to establish the ontology.

By case study, this study proved that the CORAS framework based on OPM methodology could decrease risk threats to the project. Simultaneously, the outcome also confirmed the effectiveness. Through the framework, the organization could not only manage project risk more efficiently, but also disseminate the organization risk knowledge effectively.

**References**


