Architecture-based Qualitative Risk Analysis for Availability of IT Infrastructures

Emmanuele Zambon[1], Sandro Etalle[1,2], Roel J. Wieringa[1], Pieter Hartel[1]
[1] *University of Twente,*
{*emmanuele.zambon, sandro.etalle, r.j.wieringa, pieter.hartel*}*@utwente.nl*
[2] *Technical University of Eindhoven, s.etalle@tue.nl*

June 20, 2009

Abstract

An IT risk assessment must deliver the best possible quality of results in a time-effective way. Organisations are used to customise the general-purpose standard risk assessment methods in a way that can satisfy their requirements. In this paper we present the QualTD Model and method, which is meant to be employed together with standard risk assessment methods for the qualitative assessment of availability risks of IT architectures, or parts of them. The QualTD Model is based on our previous quantitative model, but geared to industrial practice since it does not require quantitative data which is often too costly to acquire. We validate the model and method in a real-world case by performing a risk assessment on the authentication and authorisation system of a large multinational company and by evaluating the results w.r.t. the goals of the stakeholders of the system. We also perform a review of the most popular standard risk assessment methods and an analysis of which one can be actually integrated with our QualTD Model.

1 Introduction

An IT Risk Assessment (RA) must deliver the best possible quality of results in a time-effective way. Standard methods for RA provide useful general guidelines for organisations to build a robust and complete Information Security Management System (ISMS) [16], but they often lack implementation details, due to the fact that they need to be as generally applicable as possible. For this reason organisations usually develop their own, customised, RA method by merging and implementing a set of standard methods in a way that fits their requirements.

The problem addressed in this paper was presented to us by a large multinational company and it regards the method the company uses to assess availability risks. While it is satisfied with the fact that using the present RA method they can perform RA in time, the company aims at improving the result quality of their RAs by: (a) assessing availability risks more precisely, and (b) reducing the level of subjectivity and uncertainty involved in the evaluation of availability risks (i.e., when determining the impact level of a threat). At the same time, the company wants to keep the method feasible in terms of both the amount and the detail level of the information required, and of the time and resources needed to carry out an RA. In other words, any improvement of their current RA method, resulting from our activity, should not require information that the team carrying out the RA can not obtain, and should ensure that the results of the RA can still be delivered on time to the requester.

The natural choice to achieve the company's goals is to decompose the risk in elementary data such that: (a) the decomposition is objective, i.e., has a true relationship with the complex risk to be assessed and (b) the elementary data can be collected cost-effectively and are objective. Architecture-based RA methods usually fit requirement (a), but there is no proof that they will fit requirement (b) in this particular case.

In this paper we introduce the Qualitative Time Dependency (QualTD) Model and we show how we applied it to the industrial case. The QualTD Model is based on our previous quantitative Time Dependency (TD) Model [27], but geared to industrial practice since it does not require quantitative data which is often too costly to acquire in this context. The QualTD Model allows one to qualitatively assess availability risks by taking into account the dependencies among the constituents of an IT infrastructure.

Te evaluate the performance of the QualTD Model in a real-world case, and its applicability to other cases:

1. We carry out an assessment of the availability risks on the global identity and authentication management system of the company using the QualTD Model to assess the impact of the threats and vulnerabilities present in the system.
2. We compare the results obtained with our method with the ones coming from a previous assessment carried out by the company using their internal RA method.
3. Based on the results of point (2) we identify some general factors that justify the adoption of our model also in other cases.
4. We indicate how to generalise the approach we followed in the present case to other assessments, carried out following other popular (standard) RA methods.
5. We provide a brief review of the RM techniques based on dependency graphs which we found in the literature, and we discuss the results they deliver and their applicability to the present RA case.

Our results indicate that:

1. The QualTD Model satisfies requirement b), i.e., it is feasible to embed the QualTD Model in the company's RA methodology without requiring too much time or unavailable information. In Section 5 we describe the process of building the model, the steps needed and by estimating the required time.
2. The QualTD Model satisfies requirement a), i.e., the QualTD method delivers better results in terms of accuracy and reduces the number of subjective decisions. In Section 6 we analyse the differences in the results delivered by the two approaches. We also show that the QualTD method reduces the number of subjective decisions the risk assessor has to take, thus making the RA more inter-subjective.
3. Other methods do not satisfy requirement b), i.e., they could not be applied to the present case, due to the fact that they require information that is unavailable or that requires too much time to be extracted (see Section 9).
4. The QualTD Model can be used in combination with other existing standard methods, under some conditions, which include (a) the information available is enough to apply the QualTD Model and (b) the compatibility of the target method with some key features of the QualTD Model. We show this in Section 8.

This work can be seen as a general validation of the feasibility and the usefulness of the QualTD Model.

2 The industrial context

The organisation We carried out the case study at a large multinational company with a global presence in over 50 countries (from now on we call it the Company) counting between 100.000 and 200.000 employees. The Company IT unit supports the business of hundreds of internal departments by offering thousands of applications accessed by approximately 100.000 employee workstations and by many hundreds of business partners. The IT facilities for the European branch are located at one site: our RA was conducted at that site. IT services are planned, designed, developed and managed at the Company's headquarters; those services, such as e-mail or ERP systems, are part of the IT infrastructure which is used by all the different Company's branches all over Europe.

The stakeholders of the IT service are: (1) the Company's Global IT Infrastructure (GIT) management department, (2) the Risk Management and Compliance (RMC) department, (3) users: the Company's units using IT services (including GIT and RMC) and (4) an outsourcing company managing parts of the IT infrastructure on behalf of GIT.

GIT provides basic IT infrastructure services such as desktop management, e-mail and identity management. IT services are designed internally by GIT and then partly outsourced for implementation and management to another company. The outsourced tasks include specialised coding, server management, help-desk and problem solving services.

RMC supports the compliance to internal policies and best practices of the Company IT services; part of the tasks of RMC is to perform on-demand security RAs for the IT services of GIT. An RA is usually requested by the owner of the IT service each time a new service is developed or a new release of an existing one is about to be deployed.

The other business units of the Company rely on IT services for the continuity of their business. Some of these IT services are developed and managed by the business unit itself (e.g., if they are specific to the competence area of the unit), while global company services (e.g., authentication, e-mail system) are provided by GIT. For efficiency reasons, like in most other large organisations, business units exchange services by means of a "enterprise internal market": one business unit pays another one for the use of a given service and the service provider unit finances its activities by means of these funds. This mechanism increases the efficiency of internal service management.

The implementation and the management of some IT services are outsourced to another company, which we call the Service Provider. Although the servers running the IT services are owned by the Company and physically kept within its data centers, the Service Provider manages the OS and the software running on them. Moreover, for some services, the Company outsources also the development (e.g., coding, deployment) of the custom applications to the Service Provider. The Service Provider has signed contracts with the Company which include Service Level Agreements (SLAs) regarding both the security of the information managed by the outsourcing company and the availability of the outsourced services.

The target of assessment The system on which we focus our case study is called Oxygen. Oxygen is the global Identity Management for employees and sub-contractors of the Company. The goals of the system are:

1. *Identity Management*: to provide enterprise-wide standard identities for all employees and contractors of the Company, integrate identities with the different identity authoritative sources (e.g., the Human Resources information system) and manage them through a governed process and ensure regulatory and privacy compliance.
2. *Identity/Account Linking and data synchronisation*: to provide a holistic view of the many accounts possessed by a person, enforce account termination when a person leaves the Company, enable data synchronization among identity provider and identity consuming systems for data accuracy and provide credential mapping, a foundation for Single Sign-On.
3. *Identity Service for authentication and authorisation*: to provide operational directory services for general applications to be used for authentication and authorisation, provide the unique, standard, organisation-wide identifiers for employees and contractors and provide a foundation for advanced authentication and authorization in the future.

Oxygen is designed and implemented by the GIT department, while the management of the servers running it is outsourced to the Service Provider.

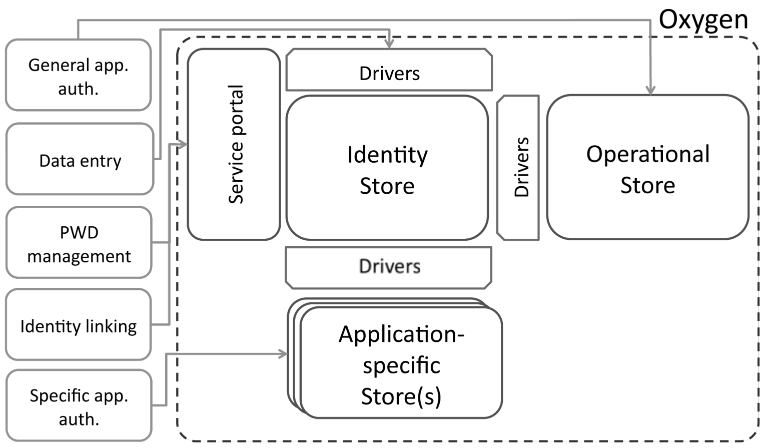

Figure 1: An overview of the Oxygen architecture

Figure 1 depicts the design of Oxygen: the system is composed of a number of *identity stores*, which are identity databases implemented by means of directory services. The main Identity Store keeps information about all of the identities and their attributes. The Operational and the Application-specific stores contain a (partial) replica of this information and are accessed by the different applications which require identities for authentication and identification. Replication of the identities stores is required for performance reasons.

Oxygen collects identity data from different authoritative sources, such as the information system of the Human Resources department. Data acquisition is performed by means of drivers, which also take care of synchronising data between the different identity stores.

In addition to the identity stores, Oxygen exports also a service portal, which allows employees of the Company to manage part of their identity record (e.g., updating their home address, changing password).

The existing RA methodology In 2008, the RMC department carried out an RA on the Oxygen system following its internal RA process, which is mainly based on the guidelines provided by BS7799-3 [5], while the official security control policy is compliant with the ISO 27002 [17] standard.

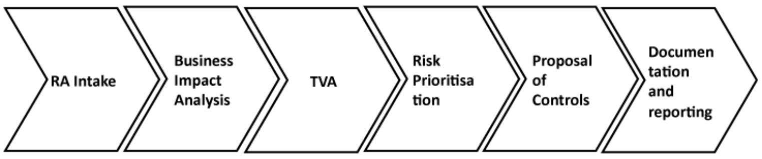

Figure 2: The internal RA process

Figure 2 depicts the process followed by RMC, which is composed of 6 steps:

1. *RA intake*: During this step the RA team (composed of people people from the RMC department) and the requester project responsible agree on the scope of the RA and the Target of Assessment (ToA). The requester also submits proper documentation about the IT service to the RA team.
2. *Business Impact Analysis (BIA)*: during this step the RA team, together with the owner of the ToA, determines the desired levels of Confidentiality, Integrity and Availability for the ToA (e.g., HIGH integrity and availability and LOW confidentiality). They do this by analyzing the impact that a breach of one of the three security properties on the information managed by the ToA would have on the business unit in a realistic worst-case scenario. They also determine which requirements the ToA has with respect to different legislations and regulations concerning information security (e.g., SOX compliance).
3. *Threat/Vulnerability Assessment (TVA)*: during this step the RA team analyses the ToA and determines which threats/vulnerabilities the ToA is exposed to. Risk identification is based on a fixed list of threats/vulnerabilities which has been derived from a number of existing RA standards (e.g., BS7799-3, ISO 17799, BSI IT-Grundshuts [5, 15, 36]) and customised to fit the needs of the Company. Based on the IT security expertise of the RA team each threat is given a qualitative estimate of likelihood and impact. The BIA influences the TVA in the sense that the threat list is customised according to the required levels of confidentiality, integrity and availability of the ToA: the higher the security level, the more detailed the list.
4. *Risk prioritisation*: risk prioritisation is based on the evaluation of likelihood and impact of the threats/vulnerabilities. Risk is evaluated as a combination of likelihood (which also includes vulnerability likelihood) and impact. Threats and vulnerabilities are then prioritised based on their risk level: the higher the risk, the higher the priority for controls.
5. *Proposal of Controls*: in this step the RA team proposes a plan to cope with the identified risks, and identifies controls to mitigate the likelihood of the threats or to protect the ToA from the identified vulnerabilities. Examples of proposed controls include password policies, authentication mechanisms or Intrusion Detection/Prevention Systems.
6. *Documentation and reporting*: during this step the RA team presents the results of the RA to the requester. It is not mandatory for the requester to communicate with the RA team about follow-up actions taken as a consequence of the RA.

The average time for an RA is approximately three weeks, depending on the size of the ToA (usually, RMC carries out RAs on ToAs which are comparable in size with Oxygen.) Roughly, the first week is spent on steps 1 and 2 and for reading all the relevant documentation, another week is spent in steps 3 and 4, and the remaining week is spent in step 5 and to prepare the final report to be exposed during step 6. The RA team consists of two people performing the same task independently and then peer-rewiewing each other's findings to come to a more objective final result.

The RA team uses three main sources of information: (a) documentation provided by the requester, (b) interviews with the requester and (c) vulnerability scans and other forms of direct investigation of security weaknesses.

Documentation includes results from previous assessments (i.e., RAs and security auditing activities), all the design and develop documents (i.e., functional specifications, security design, technical infrastructure design and software design) and SLAs and outsourcing contracts.

Interviews with the requester are performed after reading the documentation to clarify doubts and to set the boundaries of the RA. Another interview is performed for the BIA and, after step 4 for a preliminary discussion about the main risks identified.

Optionally, the RA includes active forms of investigation of security weakness. The general principle RMC follows is *trust but verify*, which means that documentation about security measures implemented is trusted, but verified in its main aspects by means of, for example, vulnerability scanners.

3 Case study design

Recall that our goal is to apply the QualTD Model to the assessment of availability risks on the Oxygen system and to compare the results we obtain with those obtained from the previous RA.

To design our case-study we follow the paradigm proposed by Wieringa et al. in [25, 26] for technical research. The paradigm says that to evaluate a solution we check the following two claims:

1. `solution` *&* `context` *produces* `effects`
2. `effects` *satisfy (to an acceptable extent)* `stakeholder-motivated criteria`

Wieringa et al. observe that each technological solution which is applied in a context produces some effects on it. The effects may (or may not) contribute to satisfy some goals defined by the stakeholders of the research context. The evaluation criteria set by the stakeholders must be in a measurable or comparable form, so that if two different solutions are applied to the same context, they can be evaluated and compared w.r.t. these criteria. The reasoning scheme can be applied when a solution is specified but not yet implemented [10] or after a solution is implemented [22].

In our case, the technical solutions to be evaluated are the RAs performed on the Oxygen system: the first done following the RA methodology of the Company and the second made by integrating that with the QualTD Model. The context in which we apply these solutions is described in Section 2.

Stakeholders, goals and criteria First, we derive the evaluation criteria according to the stakeholders, which are the ones we introduced in Section 2. To do this we first list their goals w.r.t. the Oxygen system, then we extract measurable criteria from the goals of each stakeholder. We derive the goals by analysing the description of the activities GIT provided us during the interviews; subsequently we define some criteria to measure those goals. Finally, we validate goals and criteria by means of interviews with the stakeholders. For the sake of presentation we only report the results of this activity in the list below. For more details please refer to the work of Wieringa et al. [25].

- *RMC*
  1 The goal *Ensure good quality of the RA Service* is measured by the quality criterion *# of relevant risks identified during an RA vs. # of non-relevant risks identified during an RA.*
  2 The goal *Make the RA process more efficient* is measured by the quality criterion *# of hours employed for an RA by the members of the team.*
  3 The goal *Make the RA process less subjective* is measured by the quality criterion *# of subjective decisions/estimates done during an RA.*
- *GIT*
  4 The goal *Ensure cost/effective mitigation controls and timely mitigation plans* is measured by the quality criterion *Cost for managing High/Medium/Low risks.*
  5 The goal *Use global (shared) solutions to solve the same problem in different systems* is measured by the quality criteria *# of months to implement controls* and *# of different solutions employed to solve the same problem in different systems.*
  6 The goal *Implement controls with the least possible contractual and financial impact* is measured by the quality criterion *# of controls with contractual and financial impact.*
- *Services depending on Oxygen*

7 The goal *Have the authentication/identity service for their application available when needed* is measured by the quality criteria *# of times authentication was not available in one month* and *# of times identity management was not available in one month.*

- *The Service Provider*

8 The goal *Manage systems with the least possible effort and by remaining compliant with SLAs* is measured by the quality criteria *Euro/resources employed for managing HW/SW and to guarantee SLAs (including consequences for not fulfilling contractual obligations).*

Validation process The criteria in the above list will be used in Section 6 to compare the quality of the model-based RA method we are about to introduce with the RA method of the Company.

4 The Qualitative Time Dependency (QualTD) Model

We now introduce the model supporting our RA method, which we use in the assessment of the availability risks for Oxygen. To illustrate the ideas we provide a running example showing how the QualTD Model can be employed.

The QualTD Model represents the ToA, the incidents that can affect it and the effects of their propagation on the ToA itself. A QualTD Model also includes the availability threats and the vulnerabilities which are present on the ToA. The model delivers as output the global impact and the risk levels of the availability incidents hitting the ToA.

We split the presentation of the model according to the three phases of an RA the model supports: (1) definition of the ToA, (2) risk identification and (3) risk evaluation. To simplify the exposition we use the following sets to indicate domains: $\mathbb{T}$ is the set of all the time intervals (expressed in minutes), $\mathbb{E}$ is the set of all the possible dependency (edge) types and it is defined as $\mathbb{E} = \{AND, OR\}$, $\mathbb{D}$ is the set of all the qualitative values expressing duration (e.g., `Short`, `Long`), $\mathbb{L}$ is the set of all the qualitative values expressing likelihood (e.g., `Likely`, `Unlikely`), $\mathbb{C}$ is the set of all the qualitative values expressing business value/criticality of an asset (e.g., `Critical`, `Unimportant`), $\mathbb{H}$ is the set of all the qualitative values expressing business harm (e.g., `Severe`, `Neglectable`) and $\mathbb{R}$ is the set of all the qualitative values expressing the risk (e.g., `High`, `Low`).

Figure 3 summarises the components of the QualTD Model and the relations among them.

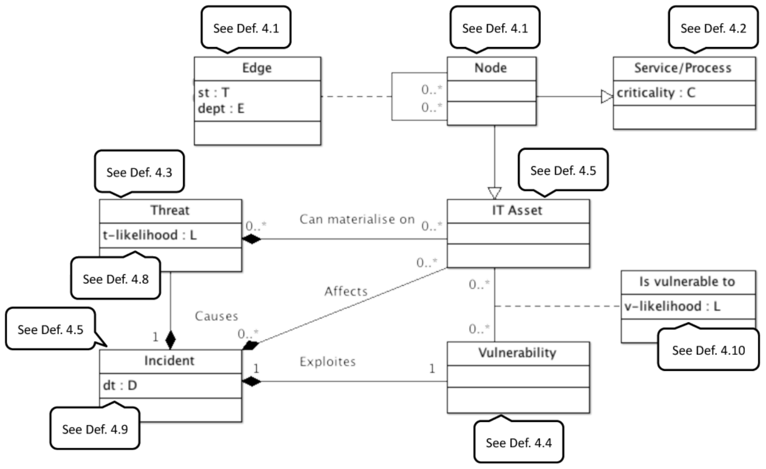

Figure 3: UML Class Diagram of the QualTD Model. Nodes of the dependency graphs are both assets and services/processes. An incident is a composition of a threat, a vulnerability on a set of assets.

4.1 Definition of the ToA

We model the ToA by means of an AND/OR graph which represents the components of the ToA and their functional/technical and organisational dependencies.

Definition 4.1 (Dependency graph) *A dependency graph is a pair* $\langle N, E\rangle$ *where* N *is a set of nodes representing the constituents of the ToA, and* E *is a set of edges between nodes* $E \subseteq \{\langle u, v, \mathrm{dept}, \mathrm{st}\rangle \mid u, v \in N, \mathrm{dept} \in \mathbb{E}$ *and* $\mathrm{st} \in \mathbb{T}\ \}$.

The nodes N of the graph are the constituents of an IT infrastructure, e.g., IT services, applications, servers, network components and locations, together with the business processes the infrastructure supports. Different IT components can be represented by means of a single node in the graph, according to the abstraction level required by the RA. For example, in a company-wide assessment we could represent an IT service (i.e., a set of servers and all the applications running on them) by means of a single node, while for the assessment of a specific IT system we model each component as an individual node.

An edge from node b to node a indicates that a depends on b. The graph supports both AND and OR dependencies. In the former case this means that a becomes unavailable when any node it depends on is disrupted. In the latter case a becomes unavailable when all nodes it depends on are disrupted. Each edge is also annotated with the survival time (*st*), which indicates the amount of time v can continue to operate after u is disrupted.

If a node a has an AND dependency on nodes b and c and an OR dependency with nodes d and e at the same time, we read this as a having an AND dependency on nodes b, c and x, with d and e having an OR dependency on node x. At the same way, the survival time of node a w.r.t. nodes d and e becomes the survival time of node x w.r.t. d and e, and the survival time of node a w.r.t. x is set to zero. This concept is shown in Figure 4.

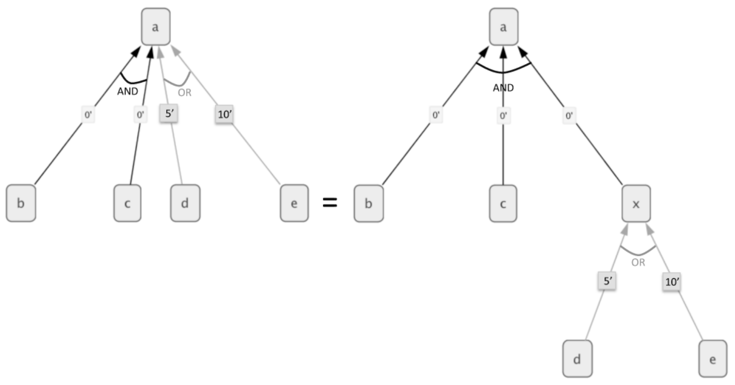

Figure 4: Equivalence of a graph with mixed AND and OR dependencies.

Running example - Part 1 *The ToA in this example is an IT system providing two IT services (`Service1` and `Service2`), and implemented by means of three applications (`App1`, `App2` and `App3`) running on two different servers (`Server1` and `Server2`). `Service1` is implemented by `App1` and `App2` in such a way that if only one of them is off-line, the service results to be off-line as well. `Service2` is implemented by `App2` and `App3` in such a way that both applications must be off-line to put off-line the service. `App1` and `App2` run on `Server1`, while `App3` runs on `Server2`. According to this description, we build the dependency graph $g = \langle N, E\rangle$ as follows:*
$N = \{\mathtt{Service1}, \mathtt{Service2}, \mathtt{App1}, \mathtt{App2}, \mathtt{App3}, \mathtt{Server1}, \mathtt{Server2}\}$ *, and*
$E = \{ \ \langle \mathtt{Server1}, \mathtt{App1}, AND, 0\rangle, \ \langle \mathtt{Server1}, \mathtt{App2}, AND, 0\rangle, \ \langle \mathtt{Server2}, \mathtt{App3}, AND, 0\rangle,$
$\langle \mathtt{App1}, \mathtt{Service1}, AND, 0\rangle, \ \langle \mathtt{App2}, \mathtt{Service1}, AND, 0\rangle, \ \langle \mathtt{App2}, \mathtt{Service2}, OR, 0\rangle,$
$\langle \mathtt{App3}, \mathtt{Service2}, OR, 0\rangle \ \}$.
Figure 5 shows the dependency graph we will use for our running example.

To complete the description of the ToA we include in the model an estimate of the criticality of the business processes and of the IT services in the perspective of the RA requester.

Definition 4.2 (Process/Service criticality) *Given a dependency graph $g = \langle N, E\rangle$, the criticality of a process/service is a mapping* $\text{criticality} : N \to \mathbb{C}$.

criticality is defined only for those nodes which represent IT services or business processes. It expresses the damage the company suffers if the node becomes unavailable. For example, in a production company, an IT service supporting a production line, which is a core business function, has a higher criticality than, e.g., personal e-mail for employees.

Running example - Part 2 *According to the business unit which uses the IT system, the criticality level of `Service1` and `Service2` is respectively `Low` and `High`.*

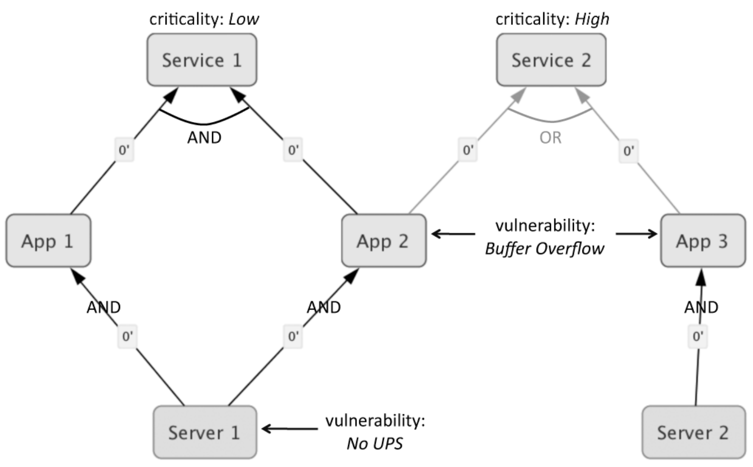

Figure 5: The dependency graph representing the ToA in our running example.

4.2 Risk identification

After modelling the ToA, we identify the vulnerabilities which are present on it, as well as the threats which could materialise on it, and compromise its availability.

Definition 4.3 (Threat) *Given a dependency graph $g = \langle N, E\rangle$, a threat is a potential cause of an availability incident, that may result in the disruption of one or more nodes of g. We call T the set of all the threats to the ToA.*

This is a common definition of threat, similar to that given in BS7799-3 [5] and ISO 27001 [16]; moreover, it is fully compatible with the concept of threat the Company has adopted in its internal RA method.

Running example - Part 3 *We identify two threats which can materialise on the ToA: a `Power outage` can bring off-line the servers and a Denial of Service (`DoS`) attack can cause the unavailability of the applications. Our set of threats is therefore $T = \{$`Power outage`$,$ `DoS`$\}$.*

Definition 4.4 (Vulnerability) *Given a dependency graph $g = \langle N, E\rangle$, and the set of threats T, a vulnerability is a weakness of a node (or group of nodes) in N that can be exploited by one or more threats in T. We call V the set of vulnerabilities on the ToA.*

Also in this case, our definition of vulnerability is consistent with both the definition given in RA standards, and with the concept of vulnerability the Company has adopted in its internal RA method.

Running example - Part 4 *We identify two vulnerabilities which can be present on the nodes of the ToA: `Server1` does not have an UPS for power continuity in case of outage; moreover, `App2` and `App3` may crash after a buffer overflow attack. Our set of vulnerabilities is therefore $V = \{$`No UPS`$,$ `Buffer overflow`$\}$.*

We model an availability incident as a specific threat materialising on a particular component of the IT architecture by exploiting a specific vulnerability.

Definition 4.5 (Incident) *Given a dependency graph $g = \langle N, E\rangle$, a set of threats T and a set of vulnerabilities V, an availability incident i is a 3-uple $\langle M, t, v\rangle$ with $M \subseteq N$, $t \in T$ and $v \in V$, describing the combination of three events:*

1. *v is a weakness of each node $n \in M$*
2. *t materialises (simultaneously) on each node $n \in M$*
3. *t exploits v to materialise*

We call I the set of all incidents generated from g, T and V. Moreover, we say a node n is directly affected by an incident $i = \langle M, t, v\rangle$ if $n \in M$.

Running example - Part 5 *By combining g, T and V we identify four incidents that can hit the ToA: (i_1) A power outage causes `Server1` to stop because there is no UPS, (i_2) a DoS attack is performed on `App2` by exploiting the buffer overflow vulnerability, (i_3) a DoS attack is performed on `App3` by exploiting the buffer overflow vulnerability, and (i_4) a DoS attack is performed both on `App1` and `App2` by exploiting the buffer overflow vulnerability. Our set of incidents is therefore* $I = \{i_1, i_2, i_3, i_4\}$ *where:*
$i_1 = \langle\{\texttt{Server1}\}, \texttt{Power outage}, \texttt{No UPS}\rangle$, $i_2 = \langle\{\texttt{App2}\}, \texttt{DoS}, \texttt{Buffer overflow}\rangle$,
$i_3 = \langle\{\texttt{App3}\}, \texttt{DoS}, \texttt{Buffer overflow}\rangle$, $i_4 = \langle\{\texttt{App2}, \texttt{App3}\}, \texttt{DoS}, \texttt{Buffer overflow}\rangle$.

The last concept we introduce for risk identification is incident propagation.

Definition 4.6 (Incident propagation) *Given a dependency graph* $g = \langle N, E\rangle$ *and an incident* $i = \langle M, t, v\rangle$, *we say that* i *can propagate to a node* $n \in N$ *if:*

1. $n \in M$, *or*
2. $\exists e \in E \mid e = \langle m, n, AND, st\rangle$ *and* i *propagates to* m, *or*
3. $\forall e \in E \mid e = \langle m, n, OR, st\rangle$, i *propagates to* m.

Running example - Part 6 *We want to know if the incident* $i_1 = \langle\{\texttt{Server1}\}, \texttt{Power outage}, \texttt{No UPS}\rangle$ *propagates to* `Service1`. *Although* `Service1` *is not directly affected by the incident, it depends on* `App1` *and* `App2`, *which in turn depend on* `Server1`. `Server1` *is directly affected by the incident, therefore we know that* i_1 *will propagate to* `Service1`.

Definition 4.7 (Nodes affected by the propagation of an incident) *Given a dependency graph* $g = \langle N, E\rangle$ *and an incident* $i = \langle M, t, v\rangle$, $\text{Prop}_i = \{n \in N \mid i \text{ propagates to } n\}$.

Running example - Part 7 *According to Definition 4.7, the set of nodes affected by the incident* $i_1 = \langle\{\texttt{Server1}\}, \texttt{Power outage}, \texttt{No UPS}\rangle$ *is* $Prop_i = \{\ \texttt{Server1}, \texttt{App1}, \texttt{App2}, \texttt{Service1}\ \}$.

An availability incident propagates on the IT infrastructure because of the technical/functional and organisational dependencies that connect the constituents of the infrastructure. For example, a power outage on a datacenter will result in some servers being unavailable, as well as the applications running on these servers. This disruption causes the IT services depending on the disrupted applications to become unavailable in turn, and propagates from servers to the (key) business processes supported by the IT services.

4.3 Risk evaluation

The last piece of information we include in the model regards likelihood and duration of incidents. In more detail, an availability threat is characterised by two indicators: (1) the threat likelihood and (2) the time needed to solve the disruption caused by the threat when it materialises, e.g., a `Short` or `Long` disruption.

Definition 4.8 (Threat likelihood) *Given the set of threats* T, *the threat likelihood is a mapping* $\text{t-likelihood} : T \to \mathbb{L}$.

Running example - Part 8 *Security analysts have assigned a likelihood to the threats in* T *using the following scale:* `Very Likely`, `Likely` *and* `Unlikely`. *The likelihood of* `Power outage` *is* `Unlikely` *and the likelihood of* `DoS` *is* `Likely`.

The likelihood of a threat is an estimate of the probability of the threat materialising on the ToA. Here we have made the (simplifying) assumption that the likelihood of a threat is a property of the threat itself and it is independent from the asset the threat occurs on. The assumption holds for most of the threats, but not for targeted attacks (i.e., attacks crafted for and directed to a specific IT asset), since the likelihood of the attack is influenced by the value of the targeted asset. In this case we split the threat into a number of new threats, each of them representing a specific asset being targeted.
It is common practice in qualitative RAs to assess the likelihood of threats by means of so-called likelihood models. Each model combines different parameters, e.g., difficulty of the attack, resources needed, etc. to determine the final likelihood of a threat. However, it is out of the scope of this work to specify such a model; in the literature there exist works proposing models for specific contexts (e.g., eTVRA [23] for telco networks).

Definition 4.9 (Incident duration) *Given a dependency graph* $g = \langle N, E\rangle$ *and a set of incidents* I, *the incident duration is a mapping* $\text{dt} : I \times N \to \mathbb{D}$.

dt(i,n) is an estimate of the (average) time a node n is out of service when incident i occurs. If we consider, for example, a buffer overflow attack freezing an application, the disruption time is the time needed to detect that the application is no longer running and to restart it. We do not take into account the time needed to fix the vulnerability exploited by the threat (e.g., the time to patch the system), unless this activity is needed to restore the functionalities of the system. To keep the model qualitative, and to match the Company methodology, we apply a discretisation of the disruption time in terms of `short` disruption (i.e., shorter than a given threshold) and `long` disruption (i.e., longer than a given threshold), which constitute our $\mathbb{D}$ set.

Running example - Part 9 *According to the stakeholders of the IT system, an incident is classified as a* `Long` *disruption if it takes more than 3 hours to be repaired, as a* `Short` *one otherwise. The contract signed with the power company guarantees that a power disruption is repaired on average in 6 hours. Therefore, i_1 is classified as a* `Long` *disruption. Since restoring* `App2` *or* `App3` *after they crashed only requires a restart, incidents i_2, i_3 and i_4 are classified as* `Short` *disruptions.*

We now associate vulnerabilities with their likelihood.

Definition 4.10 (Vulnerability likelihood) *Given a dependency graph $g = \langle N, E\rangle$, and the set of vulnerabilities V, the vulnerability likelihood is a mapping* $\text{v-likelihood} : V \times P(N) \to \mathbb{L}$ *, where $P(N)$ is the power set of N.*

The *v-likelihood(v, N_v)* is an estimate of the probability that the vulnerability v is present in the set of homogeneous (i.e., nodes which can suffer from the same vulnerability with the same likelihood) nodes N_v. The simplest and most frequent case is when we determine the likelihood of a vulnerability being present on a single node of g. However, we might also need to consider the likelihood of a vulnerability being present on a set of homogeneous nodes which are involved in a specific incident. For example, consider the case in which some malware causes a number of servers to stop working by exploiting a vulnerability which is present in an application deployed on all of these servers: in this case we need to estimate the likelihood of the vulnerability being present on all of the servers running the application with the vulnerability, since the resulting incident would affect all of them at once.

In case of an accurate RA (e.g., when it is possible to do technical vulnerability verification such as penetration testing), the fact that an application is present on an IT component can be determined without uncertainty; for example by making sure a buffer overflow affects a web server by trying to exploit it. However, in most cases, due to lack of time, the RA team has to rely on indirect (and therefore uncertain) information, for example, by consulting the NIST National Vulnerability Database [38] to check if the web server may suffer from a specific buffer overflow vulnerability. *v-likelihood* is the expression of this uncertainty.

Running example - Part 10 *Security analysts have assigned a likelihood to the vulnerabilities in V using the following scale:* `Very Likely`*,* `Likely` *and* `Unlikely`*. The likelihood of* `No UPS` *and* `Buffer overflow` *is* `Very Likely`*.*

4.4 Output of the QualTD Model

We use the information contained in the model to calculate the risk associated with an incident, which is influenced by the likelihood that the threat occurs in the ToA (which is a property of the ToA), the likelihood that a vulnerability is present in a node or a set of nodes (which expresses the uncertainty about whether or not the vulnerability is present in the nodes) and the estimated disruption severity. In more detail, an incident causes (by propagation) a disruption with a certain duration on some nodes of the dependency graph which have a certain criticality. We call this combination the *global impact* of the incident.

Intuitively, the more critical the processes/services affected and the longer the disruption, the greater the impact of the incident will be, i.e., the global impact of an incident is monotone.

Definition 4.11 (Global impact) *Given a dependency graph $g = \langle N, E\rangle$, an incident $i = \langle M, v, t\rangle$, a monotone composition function* $\text{harm} : \mathbb{C} \times \mathbb{D} \to \mathbb{H}$ *mapping criticality and duration to business harm, and a monotone aggregation function* $\text{impact-agg} : \mathbb{H} \times ... \times \mathbb{H} \to \mathbb{H}$ *; the global impact of i is defined by the mapping* $\text{golbal-impact} : I \to \mathbb{H}$ *, such that:*

$$\text{global-impact}(i) = \text{impact-agg}_{n \in \text{Prop}_i}(\mathit{harm}(\mathit{criticality}(n), \text{dt}(i,t))) \tag{1}$$

Running example - Part 11 *The risk assessment team has decided that the global impact of an incident is calculated using the following rules:*
a) the global impact is `Critical` *if the incident causes the disruption of at least one service with* `High` *criticality;*
b) the global impact is `Moderate` *if the incidents causes a* `Long` *disruption on any service, or a* `Short` *disruption of at least a service with* `Medium` *criticality;*
c) the impact is `Insignificant` *otherwise.*
For example, if we take the above definition a), the impact-agg *function is implemented by the "at least one service" statement, and the* harm *function is implemented by associating any disruption of a service with* `High` *criticality to the* `Critical` *impact. According to these rules the criticality of i_1, i_2, i_3 and i_4 is respectively:* `Moderate`, `Insignificant`, `Insignificant`, `Critical`.

Now that we have defined the incident global impact we can evaluate the incident risk, which is a composition of the likelihood of the threat, the likelihood of the vulnerability and the global impact of the disruption caused by the threat materialising.

Intuitively, this means that the more likely it is that a threat materialises on an asset (or a set of assets), or the more likely it is that the asset is vulnerable to that threat, and the more harmful the threat is, the more reasons there will be to protect it against this incident. As for the global impact, also the incident risk is therefore monotone.

Definition 4.12 (Incident risk) *Given an incident $i = \langle M, t, v\rangle$, the incident risk is a monotone composition function* i-risk : $\mathbb{L} \times \mathbb{L} \times \mathbb{H} \rightarrow \mathbb{R}$ *mapping* t-likelihood(t), v-likelihood(v) *and* global-impact(i) *to the risk level of i.*

Running example - Part 12 *As for the global impact, the risk assessment team has decided that the risk level of an incident is calculated using the following rules:*
a) the risk level is `High` *if either the incident has a* `Critical` *global impact and at least* `Likely` *threat and vulnerability likelihood, or if the global impact is* `Moderate` *and threat and vulnerability likelihood are both* `Very Likely`;
b) the risk level is `Medium` *if either the incident has a* `Critical` *global impact and the vulnerability likelihood is at least* `Likely`, *or if the global impact is* `Moderate` *and the vulnerability likelihood is at least* `Likely`;
c) the risk level is `Low` *otherwise.*
In this case, i-risk *is implemented by means of these three rules, which associate the combination of global impact, threat likelihood and vulnerability likelihood to the corrispondent risk level. According to these rules, the risk level of i_1, i_2, i_3 and i_4 is respectively:* `Medium`, `Low`, `Low` *and* `High`.

An additional operation one would like to do is to aggregate the incident risk in terms of threats and vulnerabilities. Evaluating risk in terms of threats and vulnerabilities is important to determine both the risk profile of the ToA, i.e., which threat sources are the most harmful, and to prioritise vulnerabilities to be addressed (i.e., patched) first.

Definition 4.13 (Threat/Vulnerability risk) *Given a dependency graph $g = \langle N, E\rangle$, a threat t and the set of incidents $I_t = \{i \mid i = \langle M_t, t, v_t\rangle\}$, a vulnerability v and the set of incidents $I_v = \{i \mid i = \langle M_v, t_v, v\rangle\}$ and a monotone aggregation function* risk-agg : $\mathbb{R} \times ... \times \mathbb{R} \rightarrow \mathbb{R}$;
the risk of a threat t is an aggregation of the risk level of all the possible incidents which can originate from that threat (I_t), i.e., the mapping t-risk : $\mathbb{R} \times ... \times \mathbb{R} \rightarrow \mathbb{R}$ *such that:*

$$\text{t-risk}(t) = \text{risk-agg}_{i \in I_t}(\text{i-risk}(i)) \tag{2}$$

Similarly, the risk of a vulnerability v is the aggregation of the risk level of all the possible incidents in which that vulnerability has been exploited (I_v), i.e., the mapping v-risk : $\mathbb{R} \times ... \times \mathbb{R} \rightarrow \mathbb{R}$ *such that:*

$$\text{v-risk}(v) = \text{risk-agg}_{i \in I_v}(\text{i-risk}(i)) \tag{3}$$

Running example - Part 13 *If we use* Max *as the aggregation function* risk-agg *for calculating threat and vulnerability risk level, we assign each threat/vulnerability the maximum risk level of the incidents they are involved in. In this way, the risk level of* `Power outage` *and* `DoS` *is respectively* `Medium` *and* `High`. *Accordingly, the risk level of* `No UPS` *and* `Buffer overflow` *is respectively* `Medium` *and* `High`.

The QualTD Model supports the traceability of the RA results. For instance, suppose the RA has been carried out, and after some time we want to recall why a DoS is a `High` *risk for our system; we can go through the records of the model and discover that:*

1. *it is* `Likely` *that a* `DoS` *is carried out by exploiting a* `Buffer overflow` *on both* `App2` *and* `App3`,

2. *both `App2` and `App3` are `Very Likely` to be prone to a `Buffer overflow`*
3. *the resulting incident causes a `Short` disruption of the `High` critical service `Service2`,*
4. *according to points 1–3 and to the impact and risk level definitions, the risk of a `DoS` in the system is `High`.*

When doing impact and risk evaluation we use the composition and aggregation functions *harm*, *impact-agg*, *i-risk* and *risk-agg*, which operate with qualitative values (e.g., `High` likelihood and `Low` impact): the definition of the composition and aggregation functions is outside the scope of our model and it is left to the choice of the RA team. However, these functions must be monotone and semantically sound w.r.t. the meaning that the qualitative values involved have for the stakeholders of the RA. For example, the definition of `Critical` impact we give in the running example part 11 is semantically sound; whereas it would not have been sound if we defined as `Critical` an incident causing a `Short` disruption on a service with `Low` criticality. In the running example and in Section 5 we describe two possible implementations of *harm*, *impact-agg*, *i-risk* and *risk-agg*, based on descriptive tables which define all the possible combinations of input and output values.

5 Availability RA using the QualTD Model

In this section we describe how we employed the QualTD Model in the new RA of Oxygen. Four steps of the Company internal RA process were affected by the use of the QualTD Model, as we show in Figure 6. First, we included in the *RA Intake* the activity of building the dependency graph. We spent 10 days to perform this task. We also re-performed part of the BIA: instead of only defining the security requirements for Confidentiality, Integrity and Availability, we also assessed the criticality level of the main IT services of the ToA. We spent on this one hour. Finally, we carried out the *Threat Vulnerability Analysis* and *Risk prioritisation* by using the QualTD Model as we explained in Section 4. We employed 9 days to perform this task.

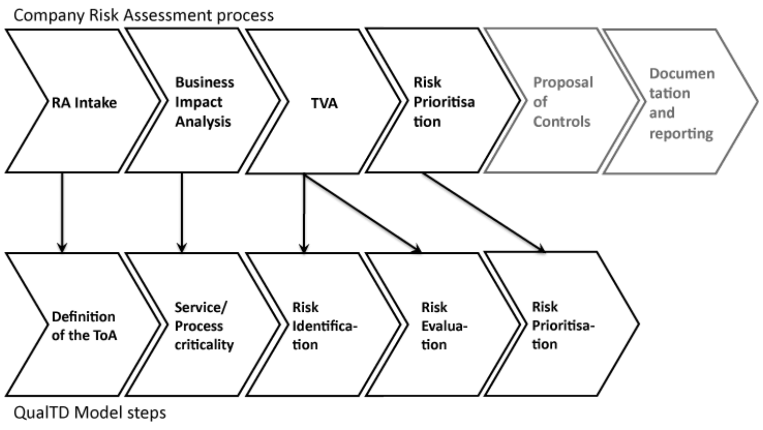

Figure 6: The internal RA process combined with the QualTD Model. Supported steps are drawn in black.

To build and run the QualTD Model for Oxygen we relied on two sources of information: technical documentation and interview sessions. In practice we had at our disposal the same documentation the RA requester provided for the previous RA, as we describe in Section 2. In more detail, four documents were made available for the RA:

1. *The functional specification document* This document describes which are the functionalities provided by Oxygen and how the functional architecture is designed, i.e., which software components are implemented, what is their task and how they relate to each other.
2. *The security architecture and design document* This document describes which security measures are implemented, e.g., server redundancy, and how they are implemented, e.g., which services are redundant and where they are located.
3. *The internal SLA document* This document describes the quality of service parameters which are guaranteed to the users of Oxygen. In the context of availability, this document describes the availability figures for the different services provided by Oxygen, e.g., the authentication service is guaranteed to be available 99% of the times.

4. *The network diagram* This document describes which are the actual servers running the different components of the Oxygen system, which software they are running and in which datacenter they are being managed.

We now describe in detail the activities we performed. For the sake of exposition we split the description according to the tasks that compose the Company RA process. Each task is further split according to the specific step of the QualTD Model of Section 4.

5.1 RA Intake

Defining the ToA The first step towards the QualTD Model-based RA is building the dependency graph for Oxygen. We modelled five types of nodes in the dependency graph, according to the indications of the quantitative Time Dependency Model [27], extracting the information about them from the available documentation. According to the level of abstraction required for this RA, we modelled the following node types:

1. *Datacenters*: from the security architecture document and the network diagram we extracted the two buildings hosting the datacenters in which the servers are split for redundancy purposes.
2. *Network components*: from the security architecture document and the network diagram we extracted the firewalls protecting the different servers and enabling access to the Oxygen services from the internal network.
3. *Servers*: from the security architecture and the network diagram we extracted which servers are used.
4. *Applications*: from the security architecture, the network diagram and the functional specification documents we extracted the applications running on each server.
5. *IT Services*: from the functional specification and the internal SLA document we extracted the services exported by Oxygen, linking them to the applications implementing them.

The most challenging task in building the dependency graph was to determine the dependencies among the nodes. The dependencies among buildings, network components, servers and applications could be inferred from the network diagram and the security architecture. Unfortunately, the functional specification document, which should link software to IT services, only referred to "logical" software components, which are not directly linked to the servers and the applications running on them. For instance, the functional component which acquires identity information from the different authoritative identity sources is actually implemented by three different applications: a Java-based web service, a Directory service and a DBMS; in turn, the DBMS also supports other functional components. To determine these dependencies we proceeded by refinement: whenever in the documentation we found that a certain application runs on a certain server, or that the application implements a certain service, we drawn a new dependency among these nodes. Then, we cross checked the information from the functional specification and the network diagram documents to check if the dependencies we found were consistent throughout all the documents. When we found an inconsistency, we updated the model and iterated the process. We reached a "stable" version of the model after the third iteration.

The resulting graph is made of 64 nodes and 112 edges. Among the nodes we count 12 IT services, 32 applications, 14 servers equally distributed between 2 datacenters and connected simultaneously to 2 different network segments by means of 2 different firewalls. Building the first prototype version of the graph took us approximately one week, using only the four documents we described as a source.

After building this prototype version of the dependency graph we checked it with the RMC personnel during an interview session: we showed the graph and explained the reasons motivating each dependency drawn; we then asked for possible missing ones. For example, we showed that a failure in the DBMS would lead to the unavailability of the identity data acquisition service and we asked if this conclusion was consistent with their knowledge of the system. The answer was positive; in effects no inconsistencies were found during this session. Finally, we performed another interview session with the developers of the system to further check for consistency and completeness of the dependency graph. During this session we focused our explanation of the graph on the reasons motivating the choice of modelling a dependency between two nodes. For example, we motivated the choice of drawing a dependency from the DBMS to the application server since the Web Service uses the DBMS to store configuration parameters, and the unavailability of the DBMS would cause the Web Service to be unable to operate in turn. We found some discrepancies between our model and the behaviour of the system which is currently implemented. These discrepancies were due to inaccurate or outdated information in the functional specification document: we decided to keep the graph coherent with the actual implementation of Oxygen, instead of the one present in the documentation. The previous RA did not spot these discrepancies, as the analysis of the ToA required to build the dependency graph is much more detailed than the analysis required for an assessment which does not require to build any formal model.

5.2 Business Impact Analysis

After we built the dependency graph, we proceeded with the Business Impact Analysis (BIA) step, which consists of determining the required level of availability for the whole Oxygen system and the criticality level of all the IT services exported by Oxygen. We did this by means of an interview session with the GIT department board, together with a member of the RMC department.

Since the required level of availability for Oxygen had already been assessed during the previous RA, we only made sure that that part of the BIA was still valid. The GIT personnel confirmed that Oxygen requires a `High` level of availability. We then used this parameter during the risk identification phase for the selection of the threats and vulnerabilities to be used, as we describe in Section 5.3.

The next step of the BIA, which is not part of the RA method of the Company consists of assessing the criticality of the IT services. For each IT service in the dependency graph we asked the GIT personnel if it had a `High`, `Medium` or `Low` criticality. In this way we defined the *criticality* function (see Definition 4.2).

After this last interview we had a final (approved) version of the dependency graph representing the ToA.

5.3 Threat/Vulnerability Analysis

Risk identification Because of the design of our case study, we did not need to search for new threats or vulnerabilities: recall that the RMC department adopted a threat/vulnerability list for their RAs, which was extracted from a number of standard RA methods and customised to fit the needs of the Company. To be able to compare the results of our RA with the previous one we used the same threats and vulnerabilities.

The list comprises a total of 121 threats and vulnerabilities. Since we were only interested in assessing availability risks, we selected a subset of this list: the set T was composed of 22 threats and the set V of 39 vulnerabilities. The RMC personnel had previously determined for each entry if it has an impact on confidentiality, integrity or availability: we relied on this labelling and we only select threats and vulnerabilities with an impact on availability. Moreover, according to the Company RA method, threats and vulnerabilities are selected based on the required level of Confidentiality, Integrity or Availability for the ToA: since the level of Availability of Oxygen has not changed in the two RAs, we are allowed to use the same availability threats and vulnerabilities.

The next step we performed was to link threats with vulnerabilities. During the previous RA, threats and vulnerabilities were assessed separately, and the RMC personnel had assigned them a likelihood and impact estimate based on their professional judgement. On the other hand, our QualTD Model requires us to link threats with vulnerabilities, therefore making explicit the reasoning that was implicitly done during the previous assessment. We did this by selecting, for each of the 22 threats, which one of the 39 vulnerabilities the threat can exploit to materialise. We proceeded as usual for validating our threat-vulnerability mapping, i.e., we explained our choices to the RMC personnel during an interview session and we integrated our mapping based on their opinion. Although no major inconsistency was found, we had to change a small number of mappings, because of a misinterpretation of the meaning of some threats.

Subsequently, we determined which nodes of the dependency graph were targeted by threats and in which nodes a certain vulnerability was present. To do this we evaluated which kind of node the threat/vulnerability applies to; for example, a power disruption can only affect a datacenter, a DoS attack can only affect software nodes.

Finally, we enumerated the availability incidents following Definition 4.5. This task was performed automatically by intersecting threats with the nodes they target, vulnerabilities with the nodes they are present in and threats with the vulnerabilities they can exploit. We inserted all these information in a database. Therefore, listing incidents was nothing more than building a view on the existing table schema. We checked our results with the RMC personnel, to detect inconsistencies in our decisions. We found no discrepancy, as mapping threats and vulnerabilities to asset types was quite an unambiguous task.

Risk evaluation We borrowed from the previous RA the estimate of the likelihood of each threat and vulnerability, i.e., the definition of the *t-likelihood* and *v-likelihood* functions of Definition 4.8 and Definition 4.10. The estimate was done in terms of `High`, `Medium` and `Low` likelihood level, according to an evaluation model defined by the RMC personnel and based on eight different parameters (e.g., time needed for the attacker, technical skills needed, etc.). The reason why we did not do our own estimate of the likelihood is twofold: first, we needed to ensure that the results of the two RAs could be comparable and, since our model only implies a different way in estimating the impact, likelihood had to be kept fixed. Second, since the results of this second RA are meant to be used by GIT, we wanted the likelihood estimates to be based on the professional judgement of the RMC personnel, instead of ours.

To assess incident duration (i.e., the *dt* function of Definition 4.9) we first set the threshold between a `Short` and `Long` duration by means of the internal SLA document. This document gives an availability figure for the IT services provided by Oxygen. For example, the identity data acquisition service will be available with a certain frequency in a month. We set the threshold as the longest amount of time (in hours) the service can be out of service while remaining compliant with its SLA. For example, if the availability figure is 99.5% in a month (i.e., 30 days), we set 4 hours as our threshold. In this way we distinguished between `Short` incidents (i.e., those shorter than the maximum tolerated disruption time in a month) and `Long` ones (i.e., those which last longer than the maximum tolerated disruption time in a month). Subsequently, we analysed the time needed to solve each of the incident. We considered both the time needed to detect the disruption and the time needed to fix the problem. The resulting total disruption time, which we compared with the threshold, is the sum of these two parameters. We performed this analysis based on both the information we gained from the SLA document the Company has signed with the outsourcer, and the opinion of the developers of the Oxygen system. The SLA document contains the maximum response time for incidents happening in the portions of the system for which management has been outsourced. For all the remaining parts of the system we relied on the judgement of the GIT developers.

At this stage we acquired all the information needed to run the model and obtained the *global impact* of the incidents and their risk. For each incident i we used the dependency graph to determine the set $Prop_i$ of the processes and services which were affected by the incident given the asset the incident directly targets as we described in Definition 4.7. Subsequently, we used Table 1 to determine the global impact level. The definitions we used reflect the requirements for availability the GIT has set on Oxygen during the meeting in which we assessed the criticality of services/processes, and are an implementation of the combination of the composition function *harm* and the aggregation function *impact-agg* of Definition 4.11.

Table 1: Global impact level determination.

Impact level	*Definition*
`Critical`	At least one service/process with `High` criticality is disrupted for a `Long` period of time.
`Serious`	At least one service/process with `High` criticality is disrupted for a `Short` period of time.
`Significant`	At least one service/process with `Medium` criticality is disrupted for a `Long` period of time.
`Moderate`	At least one service/process with `Medium` criticality is disrupted for a `Short` period of time.
`Marginal`	At least one service/process with `Low` criticality is disrupted for a `Long` period of time.
`Insignificant`	No service/process is disrupted or at least one service/process with `Low` criticality is disrupted for a `Short` period of time.

We then used the definitions of Table 2 to determine the risk level associated with every incident. The definition of the risk level we give is based on the indications of the RMC personnel and it is an implementation of the function *i-risk* of Definition 4.12.

The choice of using these two tables to evaluate the global impact and the risk level was driven by two main motivations: first, the functions defined by the tables are surjective and monotone, therefore they are compliant with the requirements of Definition 4.11 and Definition 4.12, and allow one to trace back the reasons causing the assignment of a certain risk level to a certain incident (see Running example 13). Secondly, the alternative choice of assigning a numerical value to each qualitative one (e.g., `High` = 3, `Med` = 2 and `Low` = 1) and then perform mathematical operations on them (e.g., sum, multiplication or average) would not work in our case. In fact, although this is a very popular and widely adopted technique in RAs (e.g., see Cunningham et al. [6]), it only provides meaningful results if we know the exact *ratio* among the qualitative values (e.g., if we knew that `High` is exactly three times `Medium` we could assign 9 to `High` and 3 to `Medium`). Since our RA was

Table 2: Incident risk level determination.

Risk level	*Definition*
`High`	Impact is `Critical`, threat and vulnerability likelihood are at least `Medium`. Impact is `Serious`, threat and vulnerability likelihood are `High`.
`Med-High`	Impact is `Serious`, threat and vulnerability likelihood are at least `Medium`. Impact is `Significant`, threat and vulnerability likelihood are `High`.
`Med`	Impact is `Critical`, threat or vulnerability likelihood are at least `Medium`. Impact is `Serious`, threat or vulnerability likelihood are at least `Medium`. Impact is `Significant`, threat and vulnerability likelihoods are at least`Medium`.
`Med-Low`	Impact is `Significant`, threat or vulnerability likelihood are at least `Medium`. Impact is `Moderate`, threat and vulnerability likelihood are at least `Medium`. Impact is `Marginal`, threat and vulnerability likelihoods are`High`.
`Low`	Impact is `Moderate`, `Marginal` or `Insignificant`.

carried out in a complete qualitative way, we only know that `High` is bigger than `Medium`, but we do not have any indication on how big the ratio is between them, therefore, we cannot perform any mathematical operation on these values. In other words, we work with values in an *Ordinal scale*, while to carry on the other approach we would need at least values in an *Interval scale*, as shown by Herrmann [11].

5.4 Risk prioritisation

Having determined the risk level, we ranked availability incidents according to their risk. However, to complete the outcome of the threat/vulnerability assessment step, we also needed to rank the most dangerous threats and vulnerabilities for Oxygen. We did this by assigning each threat/vulnerability the risk of the incident they cause, which has the highest level associated. In other words, we used *max* as the aggregation function *risk-agg* of Definition 4.13.

6 Evaluation

In this section we evaluate the QualTD Model in two ways. First, we compare the results of the RA carried out following the Company's method (from now on RA_1) and the one we performed using the QualTD Model (from now on RA_2). To do that we use four evaluation criteria from the list of Section 3. These parameters are: (1) the number of relevant risks vs. the number of non-relevant risks identified by the assessment, (2) the number of hours employed to carry out the RA, (3) the number of subjective decisions that the RMC personnel has to take and (4) the cost of managing availability risks. The other criteria of the list are not decidable by a risk analyst but would be observable after the system has been in use for a while. A risk analysis will have an impact on how the system scores on these criteria but based on our evaluation alone we cannot tell what the impact of our method will be. Additionally, we benchmark the QualTD Model using the maturity model provided in the ISACA RiskIT framework for risk management.

In the sequel of this section we will conduct the detailed analysis, but, for the sake of presentation, we anticipate here the results in a nutshell: (1) the QualTD Model has improved the (perceived) accuracy of RA_2 by increasing the number of relevant risks identied, (2) it introduced an overhead in the number of hours employed, (3) it helped reducing the subjectivity of RA_2 and (4) thanks to the effects of points (1) and (3), the QualTD Model supports a better risk prioritisation, which is one of the requirements for optimising the cost of risk mitigation.

Preliminaries In this analysis we assume that, given a method to calculate the risk in an RA, the quality of an RA is only determined by the knowledge of the risk assessor about: (a) the ToA, (b) threats and their likelihood, (c) vulnerabilities and their likelihood and (d) how threats, vulnerabilities relate to each other and impact the ToA. We choose not to include all the social/organisational factors, e.g., the relationships among the stakeholders and their commitment to IT security, the alignment of all the stakeholders w.r.t. the organisation business goals, etc. These factors are indeed very important for the success of a RA but, for the sake of this evaluation, we assume them to have remained steady in the Company throughout the two RAs, and therefore to have no impact. For more examples of other IT RA social/organisational success factors, please refer to [31, 8].

The experiment we carried out compares the results of two RAs, performed sequentially by different people on the same IT system. For these reasons, to keep the experiment under control, we needed to make sure that: (1) the order in which the RAs were carried out does not influence their results, and (2) the quality of the results does not depend on the security skills of the people carrying out the RAs.

To accomplish these conditions we conducted RA_2 before having access to the results of RA_1, but using the same sources of information; we used the same list of threats and vulnerabilities, as well as the same likelihood estimation, in both the RAs and we made sure the method we employed to build the dependency graph and to relate threats, vulnerabilities and nodes did not depend on the particular security skills of the risk assessor.

Table 3 summarises the conditions that we respected to ensure the two RAs are comparable.

Table 3: RA comparison control variables

	(1) RA Order	*(2) Security skills*
(a) ToA	Used the same documentation in the two RAs. RA_2 is blind to the results of RA_1 (see Section 5.1).	The technique to build the dependency graph does not require particular security skills (see Section 5.1).
(b) Threats & likelihood	The same threat list and likelihood estimation was used for RA_1 and RA_2 without any change (see Section 5.3).	Only the security skills of the RMC team have been employed in the two RAs for threat identification and likelihood estimation (see Section 5.3).
(c) Vulnerabilities & likelihood	The same vulnerability list and likelihood estimation was used for RA_1 and RA_2 without any change (see Section 5.3).	Only the security skills of the RMC team have been employed in the two RAs for vulnerability identification and likelihood estimation (see Section 5.3).
(d) Combining threats, vulnerabilities and nodes	RA_2 does not use any information of RA_1 about this (see Section 5.3).	The technique to combine threats, vulnerabilities and nodes does not depend on particular security skills (see Section 5.3).

Criterion 1: # of relevant risks vs. # of non-relevant risks identified The first evaluation criterion is given by the number of identified relevant risks w.r.t. the non-relevant ones and it expresses the `result quality` of an RA method. To determine the performance in identifying relevant risks of RA_2 w.r.t. RA_1, we compared and analysed the results of the two RAs together with the RMC personnel.

First, we made sure that risks were evaluated following the same criteria in both RAs, i.e., given the same threat and vulnerability likelihood and impact levels, the resulting risk level is the same. Secondly, we analysed the cases in which the two RAs gave different results, by analysng the reasons of the difference. Table 4 summarises our findings.

Table 4: Summary of the number of differences between the two RAs.

	Threats	*Vulnerabilities*	*Total*
Related to Availability	22	39	61
Original RMC RA overestimates risk level	1	2	3
Original RMC RA underestimates risk level	5	13	18
Differences caused by factors not related to the QualTD Model	1	6	7
Differences caused by using the QualTD Model	5	9	14

The first important finding is that the RMC personnel acknowledges that risk estimation in RA_2 is more accurate than in RA_1 for all the cases. In seven cases, the reason of the difference was due to external causes that do not involve the use of the QualTD Model. For example, in RA_1 the vulnerabilities regarding the configuration of the Company network were usually underestimated on purpose. This because the final report of the RA carried out without the model was directed to the GIT board, who is not directly managing the Company network. Consequently, the judgement of the RMC team was that it was not useful to point out the obvious in the report, since the RA requester had no way of managing that kind of risk. The remaining 14 differences are due to a better quality of RA_2.

According to our analysis, the success of RA_2 is due to the fact that the QualTD Model enables the risk assessor to estimate with more precision the consequences of a threat materialising, and also to determine the impact of the vulnerabilities, by explicitly linking them to the incidents they can cause: all these operations are hard to perform without an architecture model that allows one to reason about the availability impact.

Criterion 2: # of hours employed for an RA We split the analysis on time consumption of the two RAs according to the four steps of the Company RA process supported by the QualTD

Model.

1. *RA intake*: the time needed to accomplish this step with the Company method is on average one week. Building the dependency graph certainly constitutes an overhead, since it requires to formalise the knowledge acquired from the documentation and it also requires at least one additional meeting with the developers of the system. In our case, we spent approximately two weeks to finish the RA intake step using the QualTD Model. About half a week was needed to gain knowledge of the Company, which would not have been necessary by an experienced RA team in the Company itself. So we think that an RA team of the RMC department, experienced as we are, would have needed about 1.5 weeks to build the dependency graph. Whether this is worth the investment depends on the benefits to be gained from this in terms of a more accurate RA and in terms of the reusability of this graph for future RAs of this or other (related) systems.
2. *BIA*: including the estimation of the service/process criticality into the Business Impact Analysis is an inexpensive task, since it is already included in the procedure followed by the RMC personnel, only in an informal way. Moreover, we experienced that it was easy for the GIT to rank the services by criticality, since this knowledge is part of their everyday business. Formalising service/process criticality took less than a person-hour.
3. *TVA*: differently to the Company method, the QualTD Model explicitly requires to link threats and vulnerabilities to the nodes of the dependency graph to evaluate the risk. This task took us approximately four days more than the time normally employed by the RMC personnel. However, this is partly due to the fact that we had to "learn" and get used to the definitions of the threats and vulnerabilities of the list provided by the Company. We estimate that, should we have known them better we would have done the same job in half the time. Moreover, another good part of the work was that of manually linking threats and vulnerabilities to nodes; we did this step by hand and it was very time consuming: a proper GUI would have saved us other time.
4. *Risk prioritisation*: using the QualTD Model does facilitate this step. In fact, following the Company RA process, the RA team has to perform a (time-expensive) peer review of the risk evaluation performed by each member of the team, i.e., the team members have to go through their personal estimation of likelihood and impact for each threat/vulnerability and, in case they find any discrepancy, determine the reasons motivating each decision and reach a final agreement on the proper likelihood/impact levels. The QualTD Model is able to automatically prioritise threats and vulnerabilities, and therefore it makes it easier the prioritisation task. Moreover, as risks are evaluated in a more detailed level (i.e., incidents instead of threats/vulnerabilities), the QualTD Model facilitates the discussion on the final impact level of threats/vulnerabilities. For example, during the discussion with the RMC personnel on the outcomes of RA_2, we used the model to explain why a certain threat or vulnerability had a certain risk level by going into detail on the incidents that these threats and vulnerability are involved in. This technique was judged very useful and practical by the RMC personnel. It is also possible to re-use most of the work of linking threats, nodes and vulnerabilities for future RAs on the same ToA, this would reduce to zero the difference with the original method in the time consumption on the TVA step.

Criterion 3: # of subjective decisions taken in an RA Reducing the subjectivity of the RA approach is one of the original goals of the RMC department, which aims at (a) delivering better quality results by identifying as many potential and relevant risks as possible, and (b) being able to justify the reasons why a certain threat or vulnerability was given a certain risk level.

The QualTD Model supports the first objective by "forcing" the risk assessor to systematically explore all the possible combinations of threats and vulnerabilities, thus reducing the risk of mis-estimating the importance of a certain threat or vulnerability.

Regarding the second objective, since the QualTD Model requires to explicitly enumerate availability incidents, it is easier for the risk assessor to trace back the reasons why a threat/vulnerability was assigned a certain risk level. Moreover, a member of the RMC department has to give (explicitly or implicitly) four subjective estimates to evaluate a single incident: the likelihood of the threat, the likelihood that the vulnerability is present in some nodes, the duration of the incident and the criticality of the services/processes it hits. By applying the QualTD Model, the global impact of an incident is based on the criticality of the nodes involved, which is given by the RA requester. In this way we are reducing by one fourth the number of estimates to be done by the RMC personnel for each incident. As we mentioned before, the Company aknowledges that applying the QualTD Model resulted in an increase on the accuracy of risk level evaluation of threats and vulnerabilities. We think that this is also due to the fact that the QualTD Model reduces the subjectivity of the impact estimates.

Criterion 4: Cost of risk management The budget for managing risks, as for everything else in the real world, is always limited. Even with an unbounded budget, it is not possible to achieve 100% security, and this is even more true when the budget is finite. In this perspective, optimising the costs of risk management means achieving at least the same security level at a lower price. To achieve this goal it is important to adequately prioritise the risks one wants to manage in terms of: (a) the risk level and, (b) the cost to mitigate that risk. By providing a more precise risk prioritisation based on (a), the QualTD Model supports part of the decision process of prioritising risks for mitigation purposes. At present time however, the model does not include any means of prioritising risks w.r.t. (b). Actually, our model bears similarities with the quantitative TD Model [27] which, on the other hand, does include countermeasures and enables one to run an optimisation algorithm which select the best risk mitigation strategy taking into account (a) and (b). We believe that the same approach is applicable also to the QualTD Model with few modifications. This is however beyond the scope of this paper, and left as future work.

6.1 Evaluation of the QualTD Model under an IT-Management perspective

Organisations are used to run their IT processes following guidelines provided in specific IT governance standards, among which CobiT [31] is a de facto standard.

To benchmark the implementation of the IT governance control processes it suggests, CobiT provides the so-called maturity models (derived from the Software Engineering Institutes Capability Maturity Model [21]), by which one can measure the present state of the organisation, decide a target goal and evaluate progress towards that goal. The maturity model CobiT provides for the risk management process is mainly focused on organisational aspects, such as embedding risk management in the organisation decision process. Under this perspective, the adoption of the QualTD Model would provide little or no benefit.

However, recently ISACA released a new framework, RiskIT [39], which is based on CobiT but is specific for risk management. RiskIT provides a maturity model to evaluate the risk evaluation process adopted by organisations, which consists of five levels, from non-existent (i.e., when the organisation cannot understand how IT risks affect its performance), to optimised (i.e., when risk evaluation is completely embedded at all levels of the organisation, sources of information are fully available and formal methods and supporting tools are systematically used).

A crucial difference from a level 2 (Repeatable) to a level 3 (Defined) risk evaluation, is to use systematic dependency analysis to determine the business impact of the possible risk factors. Since systematical dependency analysis is one of the main features of the QualTD model, we believe that an organisation which aims at establishing a risk evaluation process with maturity level greater than 2 may adopt our model to satisfy this requirement.

Also, one of the requirements to establish a level 4 (Managed) risk evaluation process is to use risk analysis tools, possibly integrated with other tools from which they can gather information or deliver results. Due to its formalised and systematical nature, the QualTD Model is designed to be implementable in an RA supporting tool, which is what we partly did to support the Oxygen RA. Our tool can be integrated with IT enterprise architecture tools (e.g., TOGAF [40] or ArchiMate [29]), as well as configuration management tools to efficiently build the dependency graph, as we show in our previous work [27].

7 Generalising to other scenarios

Based on the experience of the case-study, we observe that there are two main factors which determined the success of the QualTD Model.

First, the model forces the RA team to follow a systematic approach, this means that there is less space for human errors and that the model provides an affordable way to deal with the complexity of the ToA. The QualTD Model shares this characteristic with many other model-based approaches, as for example model checking techniques. This also means that, as other model-based approaches, it requires a preliminary investment in terms of time and resource to build the model. With this case-study we showed that the time investment does not exceed 50% of the time spent in an RA carried out without the model, and the resources commonly available for an RA are sufficient to build the model. In general, this investment can be very worthwhile (because e.g. it allows one to reuse the information gathered or it allows one to identify problems that would remain undetected with other techniques), or just a waste of resources. In our case, as confirmed by the RMC team, a QualTD Model built for an RA can be widely re-used in the following RAs of the same ToA; the resource investment can be compensated by reusing the model in successive RAs. This makes it particularly suitable for the cases in which the ToA is periodically subject to RAs. More in general, our model-based risk assessment approach, should only be used when it either allows one to save resources in the long run (as explained above) or when the need for accurate results is worth the effort of using it. In the case analyzed here, Oxygen is an availability-critical system for the Company, and therefore the need for accuracy in the assessment justified the time overhead it introduced. Also, the need to optimise the budget for risk mitigation could be a leading factor for choosing the QualTD Model and afford its initial time overhead. Another scenario in which using the QualTD Model could be convenient is when the dependency graph can be built automatically (e.g., when a configuration management database is already present and can be used to build the graph), since in this case there is almost no time overhead.

A second success factor of the QualTD Model is that it links the knowledge about security with the components of an IT architecture, their technical and functional dependencies and their importance. With this case-study we showed that the QualTD Model structures information in a way that is simple enough to be used and complete enough to cover all the aspects that are important for a security RA. In fact, we did not find any uncovered risk area in RA_1 which was not covered in RA_2. For this reason, we think that the QualTD Model is particularly suitable to be used to assess the availability risks of an IT infrastructure or of parts of it.

8 Generalising to other methods

In this section we determine if the QualTD Model can be coupled with other (standard) methods other than the Company's one. To do this, we first make a taxonomy of standard RA methods, then we determine which are the characteristics of those methods which are compatible with our model.

A taxonomy of RA methods To provide a state-of-the-art picture on RA methodologies we follow the survey from ENISA (European Network and Information Security Agency) [35]. The survey consists of a list of sixteen RA methods currently in use. Among these methods we only consider international standards, i.e., those which are available in English and which are actually in use in more than one country. According to these criteria, we reduce the initial list of sixteen methods to ten: CRAMM [32], EBIOS [34], ISAMM [33], ISO 13335-2 [13], ISO 17799 (now ISO 27002) [17], ISO 27001 [16], IT-Grundshuts [36], MEHARI [37], OCTAVE [28], NIST SP 800-30 [24]. The remaining six methods are dropped because of two reasons: Austrian IT Security Handbook, Dutch A&K Analysis and MARION because only available in a single language (German or Dutch), while ISF, MAGERIT and MIGRA because of lack of relevant documentation. Finally, since the list on the ENISA survey is not admittedly complete, we integrate it with another popular method, the Australian/New Zeland standard for risk management AS/NZS4360 [30], and with CORAS [7], the method resulting from the EU-funded project IST-2000-25031. We explicitly choose to exclude Common Criteria [14] from this list as it is not properly an RA method, even if it requires some risk analysis to be performed.

Table 5: Classification of the RA methods.

Methodology	*Evaluation scale*	*Impact evaluation*	*Risk evaluation*
CRAMM	Qualitative	Based on open damage scenarios	Type 1
EBIOS	Qualitative	Based on security needs	Type 2
ISAMM	Quantitative	Based on monetary loss	Type 3
ISO 13335-2	Both	Based on the business harm	N/A
ISO 17799	Qualitative	Based on the business harm	N/A
ISO 27001	Qualitative	N/A	N/A
IT-Grundschutz	Qualitative	Based on open damage scenarios	Type 5
MEHARI	Qualitative	Based on fixed damage scenarios	Type 1
OCTAVE	Qualitative	Based on critical assets	Type 4
NIST SP 800-30	Qualitative	Based on open damage scenarios	Type 1
AS/NZS 4360	Both	Based on a balance between business harm and business advantages	Type 5
CORAS	Both	Based on open damage scenarios	Type 5

For the sake of presentation, we classify the twelve methods by means of three parameters: (1) the scale used to evaluate risk and risk factors (quantitative or qualitative), (2) which factors the method proposes to evaluate the impact level and (3) the underlying view on how risk is evaluated.

Parameter (1) determines if the risk level measures a something that can be (meaningfully) expressed in numbers (e.g., money), or something which can only be expressed with labels (e.g., high, medium, low). In other words, a qualitative method measures the level of a risk factor in an ordinal scale (i.e., only ordering among values are known), while a quantitative method uses measures in interval or ratio scales (i.e., it is known the magnitude of the difference between two values, ratio scales also define an absolute and non arbitrary zero point).

Parameter (2) indicates which factors the method proposes to influence the impact of a security event (i.e., a threat, a vulnerability or an incident), and to which extent the method is constrained by these factors. Some methodologies only give general guidelines (e.g., the damage to the organisation), while others strictly define a particular set of parameters (e.g., the money loss, or the affected business processes).

Parameter (3) investigates what determines the risk level of a security event and how different properties are combined. To this end we elaborated five different profiles (`Type 1` to `Type 5`):

1. `Type 1`:
   $Risk(Threat, Asset) = Likelihood(Threat) \otimes Vulnerability(Threat, Asset) \otimes Impact(Threat, Asset)$
   Risk is analysed w.r.t. a threat and an asset, or a group of assets and it is evaluated as the combination of the likelihood of the threat, the vulnerability level of the asset(s) to the threat and the impact of the threat on the asset(s). This approach can be applied both to fine-grained assessments (i.e., taking into account single assets and asset-specific threats) and to more high-level assessments (i.e., taking into account only classes of assets and high level threats).

2. `Type 2`:
   $Risk(Threat, Asset, Needs) = Impact(Threat, Needs) \otimes Vulnerability(Threat, Asset)$
   Risk is analysed w.r.t. a threat, an asset and some security needs on the system and it is evaluated as the combination of the vulnerability of the asset and the impact of the threat on the security needs. This approach is suitable where security requirements are clearly specified, for example for software products developed by following a rigourous software engineering process.

3. `Type 3`:
   $Risk(Threat, Asset) = AnnualLossExpectancy(Threat, Asset) =$
   $Probability(Threat, Asset) \otimes AverageLoss(Threat, Asset)$
   Risk is analysed w.r.t a threat and an asset, is intended as the annual loss expectancy (in monetary terms) and it is evaluated as the combination of the probability of the threat affecting the asset and the average loss of the resulting incident. This approach is suitable in all the situations in which decisions are taken based on a financial cost/benefit analysis (e.g., insurance companies), and in which quantitative data is available (e.g., for critical infrastructures).

4. `Type 4`:
   $Risk(Threat, CriticalAsset) = Impact(Threat, CriticalAsset) \otimes Vulnerability(CriticalAsset)$
   Risk is analysed w.r.t. a threat and an asset that has previously been identified as critical, and it is assessed as the combination of the impact of the threat on the critical asset and the vulnerability of the asset. This approach is suitable where there are critical assets to be protected (e.g., for utility network infrastructures).

5. `Type 5`:
   $Risk(Incident, Asset) = Likelihood(Incident) \otimes Consequences(Incident, Asset)$
   Risk is analysed w.r.t. and incident (i.e., a combination of a threat and some vulnerabilities) and an asset, and it is evaluated as the combination of the likelihood of the incident and the consequences of the incident itself. Unlike for the `Type 1` approach, this approach attributes risk levels only to security incidents (i.e., a threat exploiting a vulnerability) to assess their risk. This means that it is more suitable to be applied to fine-grained RAs and it is harder to apply to the high-level ones.

Table 5 depicts the results of the classification. Most of the methodologies are meant to be used with qualitative measurements, and this confirms the fact that most RAs today are carried out in a qualitative way, mainly due to lack of reliable quantitative data or to time constraints [3].

Regarding impact level evaluation, we observe that ISO 13335-2 and ISO 17799 are completely open to the impact level evaluation process, they only specify that the impact of a security event is tied to the business harm suffered from the organisation. Furthermore, AS/NZS 4360 also specify the possibility of a business advantage of undertaking a certain risk, e.g., leaving servers un-patched may lead to a quicker time to market for the organisation. CRAMM, IT-Grundshuts, NIST SP 800-30 and CORAS specify more precisely how the impact level should be assessed, since they introduce the concept of damage scenarios: the RA team should identify different impact scenarios (e.g., from `Catastrophic` to `Marginal`) which describe the negative consequences of a risk event on the organisation. We say that these scenarios are “open” as these methods do not specify a particular set of scenarios or they do not require to use the ones they propose. On the other hand, MEHARI is based on a “fixed” impact scenario, i.e., the description of the consequences is fixed, and the risk assessor can only rank them. EBIOS imposes that the impact level of a security event is assessed in terms of which security needs (i.e., a security requirement on the IT assets) the event is violating. Similarly, in OCTAVE the impact level is measured in terms of how “hard” the security event is hitting a mission-critical asset (e.g., a server which has been pre-determied to be critical for the organisation). Finally, ISAMM measures impact by means of the money the organisation can loose because of a security event.

Regarding risk level evaluation, we observe that CRAMM (which mostly implements the principles given in BS7799-3 [5]), MEHARI and NIST SP 800-30 share the same common view on risk, i.e., they all consider risk as a combination of the likelihood and the impact of a threat to hit a (group of) assets and the vulnerability level of this (group of) assets. Similarly, IT-Grundshuts, AS/NZS 4360 and CORAS consider risk as the combination of the likelihood of an incident (i.e., a threat exploiting some vulnerabilities) and the consequences (positive or negative) of this incident happening. On the other hand, `Type 2`, `Type 3` and `Type 4` profiles are intrinsically tied to a particular approach to RA, since `Type 2` and `Type 4` rely on qualitative concepts for defining risk (e.g., critical assets, security needs) and `Type 3` relies on the quantitative concepts of probability and average monetary loss. Finally, we observe that the methods of the ISO family do not adopt any risk analysis profile. This is due to the fact that, according to ENISA [35], ISO 13335-2 is a very general guideline to set up a risk management framework, while ISO 17799 and ISO 27001 are not properly methods for risk management, but rather compliance standards, reporting a list of controls for good security practices and the requisites that an existing method should have to be standard-compliant respectively.

Generalisation According to our method classification scheme, the original RA method followed by the Company is qualitative, based on the business harm and `Type 1` concerning risk level evaluation. The new RA method which integrates the QualTD Model remains qualitative, but it is based on open damage scenarios and has a basic risk level evaluation of `Type 5`. We also define a procedure to translate the evaluation of incident risks to threat and vulnerability risks, making the evaluation scheme compliant to the original `Type 1`.

From the perspective of the risk level scale, the QualTD Model can only be used together to a qualitative RA method; we proposed in [27] the TD Model which can only be used with quantitative ones.

From the perspective of impact level determination, we showed in the present case how the QualTD Model can be used together with methods evaluating the impact in terms of business harm. On the other hand, for methods adopting damage scenarios, the integration with our model is only possible if the scenario descriptions used by the organisation undertaking the RA can be associated with the unavailability of a node in the dependency graph. For example, if the scenario description refers to the e-mail service being unavailable, then we can associate this scenario to the unavailability of all the IT components on which the e-mail service relies upon, by means of the dependency graph. For methods in which the impact level is based on critical assets, e.g., OCTAVE, the QualTD Model cannot be applied as it is, since in the current specification we do not give a definition of critical assets. However, one possible way of adapting the model to this purpose consists in first determining the most critical processes/services and then using the dependency graph to find the nodes supporting those processes/services.
Furthermore, we observe that it is hard to integrate the QualTD Model with methods based on security needs, such as EBIOS. Innerhofer-Oberperfler and Breu [12] introduced an approach, which shares some similarities with ours, and is suitable to be used in combination with these methods: we will discuss it in more detail in Section 9.
Finally, in the present specification of the model, we do not consider the business advantage of a certain risky factor, as required by AS/NZS 4360: this is the only obstacle we see for the integration of the QualTD Model with this standard.

Regarding risk level evaluation, the QualTD Model can be integrated with any method adopting the `Type 5` approach. For example, our model could be used in combination with CORAS as an additional, availability-specific, technique to determine the consequences of threats, in substitution of the traditional HazOp, FTA and FMECA techniques.
We showed in the present case how we integrated the QualTD Model with a `Type 1` method by means

of a threat and vulnerability risk level definition table. We believe that this approach is applicable in general if it is time and information-wise feasible for the risk assessor to explicitly enumerate the vulnerabilities are present in the ToA.
Integration with a `Type 4` is instead more challenging, as it would require an approach similar to the one we described previously for OCTAVE.
Finally, RAs following `Type 2` and `Type 3` methods cannot be integrated with our model, due to the fact that `Type 2` methods already (implicitly) take into consideration the consequences of incident propagation in the definition of the security needs for each asset in the ToA, while `Type 3` methods are quantitative.

9 Related work

Academic researchers have started lately to use dependencies in security RAs to improve their quality. They have addressed this topic from multiple perspectives, such as information security, business administration and software engineering. The background of researchers, together with the context in which the technique is applied leads them to use different type of dependencies. In the literature we find three kinds of dependencies: security dependencies, software dependencies and organisational and technical/functional dependencies. In this section we will examine previous literature on these three fields which matches our work. Moreover, since our method considers the third kind of dependencies, in the final part of this section we also enumerate some techniques to build technical/functional dependency graphs.

Security dependencies Baiardi et al. [2] propose a framework for RA of information infrastructures by building a hyper-graph of security dependencies, i.e., dependencies on the security properties of the system: confidentiality, integrity and availability. The dependency graph is a form of attack graph in which nodes are the components of the infrastructure, and edges between nodes represent the dependency of a component on some security properties of the component it is linked to. Threats are represented as users of the infrastructures possessing some security properties on some contents, while vulnerabilities are conditions allowing the extension of security rights from one component to another. The framework allows one to rank countermeasures and create risk mitigation plans. A countermeasure can reduce the vulnerability level of a component, update dependencies, update the initial properties of a threat or increase the resources needed for an attack.

Attack graph-based approaches are known to have scalability problems (e.g., see Lippmann et al. [20]) in terms of the number of hosts under assessment. This is due to the fact that building such graphs requires a large amount of work which can be only partially automated. Moreover, they require extensive and difficult to obtain attack details: we doubt that such an information would have been readily available in the Oxygen RA.

Software dependencies Goseva-Popstojanova et al. [9] present a semi-quantitative approach for assessing reliability and availability related risks at early phases of a software life cycle by using the UML representation of the ToA. In this work, the authors use dependencies between software components to assess the likelihood of a fault propagating from a component to the other. In more detail, they use the following UML constructs: software architecture diagrams, use case diagrams, sequence diagrams and state charts of software components. By means of this information, they estimate the probability of failure of a software component, and the probability of failure of two software components interacting with each others. They consider the complexity of a software component in order to calculate its probability of failure, and the number of messages exchanged by components to determine the probability of an interaction failure. They give the impact of a failure w.r.t. a qualitative scale from *Minor* to *Catastrophic*. Then, they calculate the risk level distribution of each UML use case scenario by building a Markov model from the scenario sequence diagram. Finally, they average all the single use case risk distribution to determine the overall system risk. This approach could not be adapted to our case, since we do not have an UML representation of the ToA. Moreover, we are interested in assessing risks caused by external sources (e.g., DoS attacks) and no quantitative figures about the likelihood of such events were available.

Organisational and technical/functional dependencies Innerhofer-Oberperfler and Breu [12] propose a model-driven approach for assessing IT-related risks using an enterprise architecture as the basis of the model. They group entities of the enterprise architecture in four hierarchical layers: business, application, technical and physical layer. They derive – by refinement – business security objectives and requirements from this enterprise architecture and from the dependencies among its constituents. The refinement process follows a top-down approach starting from high-level business units to technical and physical devices. Then, they identify and analyse risks to the security requirements by selecting threats and vulnerabilities from standard security methodologies, e.g., BSI IT-Grundshuts [36]. Once risks are identified, they do a bottom-up aggregation of risk scenarios to make sure risks become clearly understandable at each level of the organisation (i.e., from technical to business levels). The approach is qualitative and not linked to a specific threat-list, with a risk analysis technique very similar to the one presented in the EBIOS [34] method. The strong point of this approach is that RA is fully embedded on the organisation at all levels, from the technical level to the business management level. Our case focuses on a single IT system, involving a few departments of the Company: we have little information regarding the high-level goals of the organisation and the main difficulty in applying this method would be that of deriving a full list of security requirements from high level goals.

Kim et al. [19] propose a model to assess and prioritise security risks and their treatment in the context of a communication infrastructure. They do this by determining the magnitude of damages produced by a threat to the assets of the ToA, also taking into account incident propagation. To model incident propagation they use technical and functional dependencies among the assets of the communication infrastructure: for each threat they create a workflow (graph) of the incident propagation, with the assets as nodes and the relevant dependencies as edges. They annotate each edge with the probability that the destination node is affected by the damage on the source node. Finally, they model the vulnerability level of an asset by considering the "age" of the asset. Starting from the assumption that systems ages over time, and because of the increased level of knowledge attackers gain on the weaknesses of the asset, attacks are supposed to have a greater probability of success over time if the system is not timely patched. Using incident propagation graphs and likelihood distribution functions, the authors are able to calculate the risk of an infrastructure over time, and to prioritise the actions to be taken to control those risks. The data available in our case is insufficient to estimate the level of weakness of the system over time; moreover, according to the SLAs with the outsourcing company, patching is performed quite regularly on Oxygen: therefore the weakness (vulnerability) level of the assets is almost constant.

Building a dependency graph Every method using dependencies for RA is based on the possibility of constructing a dependency graph describing the ToA. Building the dependency graph is an"extra" step which is not required by traditional RA methodologies, which probably rely on the same knowledge of the ToA, but in an implicit form. For this reason, building the dependency graph in a time-effective way is essential for the applicability of dependency-based RAs.

A technical/functional dependency graph can be built either manually or automatically. Manual methods involve acquiring information by functional and technical documentation and from interviews, like we did in the Oxygen RA. On the other hand, *Static Dependency Analysis* [18] and *Active Dependency Discovery* [4] are two automatic techniques to automatically create the graph.

The former method is based on using application configuration files to derive dependencies, e.g., the `web.xml` file for Java web applications. The main drawback of this method is that it does not generate a full, cross-domain dependency graph. This is due to the fact that some dependencies are never derivable from a configuration file, and to the high number of different formats configuration files can take.

The latter method consists of measuring the variation of certain QoS parameters (e.g., availability or response time) of the ToA after some of its components are deliberately perturbed. For example, by simulating a network traffic overload it is possible to measure the dependency of the response time of a software component w.r.t. the network service it relies on. The limitation of this approach is twofold: first it is invasive for the system, since it requires to modify the system itself or its context, secondly it is labour intensive, since it requires to test all the possible perturbations to be sure that all the dependencies are captured.

An example of an Active Dependency Discovery technique was proposed by Bagchi et al. [1] for the availability of e-commerce environments. The authors propose to inject faults on the test/benchmark environment of the ToA and detect availability dependencies; the same dependencies are then assumed to hold also on the production system. This technique allows one to quickly build a dependency graph without the need to know perfectly the implementation details of the ToA. However, to build a reliable dependency graph, the test/benchmark system must be identical to the production system, which is not the case for Oxygen.

10 Conclusions

In this paper we introduce the new QualTD Model and method for the qualitative assessment of availability risks based on the propagation of availability incidents in an IT architecture. We apply the model and method to a real-world case by performing an RA on the authentication and authorisation system of a large multinational company. We compare the results of this RA with the ones obtained from a previous RA carried out internally by the company on the same system. We then evaluate the results w.r.t. the goals of the stakeholders of the system.

Our results show the feasibility of the QualTD Model and method, and indicate that the model provides better results in terms of accuracy and reduces the number of subjective decisions taken by the risk assessor. The reasons of success are mainly due to the systematic nature of the approach and to the completeness of the information the model includes. These factors help the risk assessor to deal with the complexity of the ToA in such a way that no relevant risk factor is neglected. Our analysis also shows that the QualTD Model is particularly suitable to assess the availability risks of IT infrastructures or parts of them, when RAs are carried out regularly on the same target and when the outcomes of the RA are used to prioritise the risk mitigation strategies.

In addition, we analyse 12 risk assessment standard methods, and we discuss which characteristics of the standard methods are compatible with the QualTD model. Our analysis shows that the QualTD Model can be used in combination with many of the most popular RA standard methods, which indicates a wide range of applicability of the method, also in organisations not using the same RA method we used in this case.

Finally, we make a review of academic works we found in the literature which apply dependency analysis to RA. We show the type of risk analysis these methods allow to do and we discuss their applicability to our real-world case. Out analysis shows that none of the methods examined are directly applicable to our case either because they require information that was not readily available, or because they cannot satisfy the requirements of the stakeholders.

General References

[1] S. Bagchi, G. Kar, and J. Hellerstein. Dependency Analysis in Distributed Systems using Fault Injection: Application to Problem Determination in an e-commerce Environment. In *DSOM '01: Proc. 2001 International Workshop on Distributed Systems: Operations & Management* , 2001. http://www.research.ibm.com/PM/DSOM2001_dependency_final.pdf.

[2] F. Baiardi, S. Suin, C. Telmon, and M. Pioli. Assessing the Risk of an Information Infrastructure Through Security Dependencies. *Critical Information Infrastructures Security*, 4347/2006:42–54, 2006.

[3] S.P. Bennet and M.P. Kailay. An Application of Qualitative Risk Analysis to Computer Security for the Commercial Sector. In *Eighth Annual Computer Security Applications Conference*, pages 64–73. IEEE Computer Society Press, April 1992. http://ieeexplore.ieee.org/xpls/abs_all.jsp?isnumber=5913&arnumber=228232&count=25&index=15.

[4] A. Brown, G. Kar, and A. Keller. An Active Approach to Characterizing Dynamic Dependencies for Problem Determination in a Distributed Application Environment. In *IM '01: IEEE/IFIP International Symposium on Integrated Network Management*, pages 377–390, 2001.

[5] BS 7799-3 - Information security management systems - Part 3: Guidelines for information security risk management, 2006.

[6] B. Cunningham, T. Dykstra, E. Fuller, C. Gatford, A. Gold, M.P. Hoagberg, A. Hubbard, C. Little, S. Manzuik, G. Miles, C.F. Morgan, K. Pfeil, R. Rogers, T. Schack, and S. Snedaker. *The Best Damn IT Security Management Book Period.* Syngress Publishing, November 2007.

[7] F. den Braber, I. Hogganvik, M. S. Lund, K. Stolen, and F. Vraalsen. Model-based security analysis in seven steps — a guided tour to the CORAS method. *BT Technology Journal*, 25(1):101–117, 2007.

[8] A. Evangelidis, J. Akomode, A. Taleb-Bendiab, and M. Taylor. Risk Assessment & Success Factors for e-Government in a UK Establishment. In *Electronic Government*, volume 2456/2002, pages 93–99. Springer Berlin / Heidelberg, 2002.

[9] K. Goseva-Popstojanova, A. Hassan, A. Guedem, W. Abdelmoez, D.E.M. Nassar, H. Ammar, and A. Mili. Architectural-level risk analysis using UML. *IEEE Transactions on Software Engineering*, 29:946 – 960, October 2003.

[10] C.A. Gunter, E.L. Gunter, M.A. Jackson, and P. Zave. A reference model for requirements and specifications. *IEEE Software*, 17(3):37–43, May/June 2000.

[11] D.S. Herrmann. *Complete Guide to Security and Privacy Metrics.* Auerbach Publications, Boston, MA, USA, 2007.

[12] F. Innerhofer-Oberperfler and R. Breu. Using an Enterprise Architecture for IT Risk Management. In *ISSA '06: Proc. Information Security South Africa Conference*, 2006. URL: http://icsa.cs.up.ac.za/issa/2006/Proceedings/Full/115_Paper.pdf.

[13] ISO/IEC 13335:2001 - Information Technology - Security techniques - Guidelines for the management of IT security, 2001.

[14] ISO/IEC 15408:2006 - Common Criteria for Information Technology Security Evaluation. http://www.commoncriteriaportal.org/thecc.html, September 2006.

[15] ISO/IEC 17799:2000 - Information Security - Code of Practice for Information Security Management, 2000.

[16] ISO/IEC 27001:2005 - Information technology – Security techniques – Information security management systems – Requirements, 2005.

[17] ISO/IEC 27002:2005 - Information technology – Security techniques – Code of practice for information security management, 2005.

[18] G. Kar, A. Keller, and S. Calo. Managing Application Services over Service Provider Networks: Architecture and Dependency Analysis. In *NOMS '00: Proc. of the 7th IEEE/IFIP Network Operations and Management Symposium*, pages 61–75. IEEE Press, 2000.

[19] I-J. Kim, Y-J. Jung, JG. Park, and D. Won. A Study on Security Risk Modeling over Information and Communication Infrastructure. In *SAM '04: Proc. of the International Conference on Security and Management*, pages 249–253. CSREA Press, June 2004.

[20] R.P. Lippmann and K.W. Ingols. An Annotated Review of Past Papers on Attack Graphs. Technical report, Defense Technical Information Center OAI-PMH Repository [http://stinet.dtic.mil/oai/oai] (United States), 1998. http://en.scientificcommons.org/18618950.

[21] M. C. Paulk, C. V. Weber, B. Curtis, and M. B. Chrissis. *The capability maturity model: guidelines for improving the software process.* Addison-Wesley Longman Publishing Co., Inc., 1995.

[22] R. Pawson and N. Tilley. *Realistic Evaluation.* Sage Publications, 1997.

[23] J.E.Y. Rossebo, S. Cadzow, and P. Sijben. eTVRA, a Threat, Vulnerability and Risk Assessment Method and Tool for eEurope. In *ARES '07: Second International Conference on Availability, Reliability and Security*, pages 925–933. IEEE Computer Society Press, April 2007. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4159893.

[24] G. Stoneburner, A. Goguen, and A. Feringa. NIST SP 800-30 - Risk Management Guide for Information Technology Systems. Technical report, NIST National Institute of Standards and Technology, 2002.

[25] R.J. Wieringa and J.M.G. Heerkens. Designing requirements engineering research. In *CERE '07: Workshop on Comparative Evaluation in Requirements Engineering*, pages 36–48. IEEE Computer Society Press, October 2007. http://eprints.eemcs.utwente.nl/13002/.

[26] R.J. Wieringa, N. Maiden, N. Mead, and C. Rolland. Requirements engineering paper classification and evaluation criteria: a proposal and a discussion. *Requirements Engineering Journal*, 11:102–107, November 2006.

[27] E. Zambon, D. Bolzoni, S. Etalle, and M. Salvato. Model-Based Mitigation of Availability Risks. In *BDIM '07: Second IEEE/IFIP International Workshop on Business-Driven IT Management*, pages 75–83, Munich, May 2007. IEEE Computer Society Press.

Web References (Last Accessed: June 2009)

[28] C. J. Alberts and A. J. Dorofee. OCTAVE Criteria. Technical Report ESC-TR-2001-016, Carnegie Mellon - Software Engineering Institute, December 2001. http://www.cert.org/octave/.

[29] The ArchiMate project, 2009. http://www.archimate.org.

[30] Risk management - AS/NZS 4360:2004. http://www.riskmanagement.com.au/, October 2004.

[31] CobiT 4.1 - Control Objectives for Information and related Technology. http://www.isaca.org, 2007.

[32] CRAMM v5.1 Information Security Toolkit. http://www.cramm.com, 2009.

[33] A. Deladrière and M. Morrison. The risk management challenge. http://www.bankingfinance.be/40915/default.aspx, March 2008.

[34] EBIOS - Expression des Besoins et Identification des Objectifs de Sécurité - Section 2: Approach. http://www.ssi.gouv.fr/en/, 2004.

[35] ENISA. Risk Management: Implementation principles and Inventories for Risk Management/Risk Assessment methods and tools. Technical report, European Network and Information Security Agency (ENISA), June 2006. http://www.enisa.europa.eu/rmra/rm_home.html.

[36] BSI Standard 100-1 - Information Security Management Systems (ISMS). http://www.bsi.de/english/gshb/, 2005.

[37] MEHARI 2007 - Risk Analysis Guide. https://www.clusif.asso.fr/en/clusif/present/, April 2007.

[38] Nist national vulnerability database. http://nvd.nist.gov/, 2009.

[39] D. Steuperaert, S. De Haes, R. Massa, B. Peeters, and S. Reznik. Enterprise Risk: Identify, Govern and Manage IT Risk - The Risk IT Framework. ISACA Exposure Draft: http://www.isaca.org/Template.cfm?Section=Risk_IT&Template=/TaggedPage/TaggedPageDisplay.cfm&TPLID=79&ContentID=48749, February 2009.

[40] The Open Group. TOGAF (The Open Group Architecture Framework), 2003. http://www.opengroup.org/architecture/togaf8-doc/arch/.