An Ontology-Driven Approach for Process-Aware Risk Propagation

Gal Engelberg
Accenture Labs, University of Haifa
Haifa, Israel
gal.engelberg@accenture.com

Mattia Fumagalli
Free University of Bozen-Bolzano
Bolzano, Italy
mattia.fumagalli@unibz.it

Adrian Kuboszek
Avanade
Poland
a.kuboszek@avanade.com

Dan Klein
Accenture Labs
Israel
dan.klein@accenture.com

Pnina Soffer
University of Haifa
Haifa, Israel
spnina@is.haifa.ac.il

Giancarlo Guizzardi
University of Twente
The Netherlands
g.guizzardi@utwente.nl

ABSTRACT

Risk Propagation (RP) is a central technique that allows the calculation of the cascading effect of risk within a system. At the current state, there is a lack of risk propagation solutions that can be used to assess the impact of risk at different levels of abstraction, accounting for actors, processes, physical-digital objects, and their relations. To fill this gap, in this paper, we propose a *process-aware risk propagation approach* that builds on two main components: *i.* an *ontology*, which supports functionalities typical of *Semantic Web technologies (SWT)*, and *ii.* an *ad hoc* method to calculate the propagation of risk within the given system. We implemented our approach in a proof-of-concept tool, which was validated in the cybersecurity domain.

CCS CONCEPTS

• **Information systems** → **Decision support systems**; • **Risk** → *Assessment and propagation*;

KEYWORDS

Risk propagation, risk analytics, ontology-driven risk propagation

ACM Reference Format:
Gal Engelberg, Mattia Fumagalli, Adrian Kuboszek, Dan Klein, Pnina Soffer, and Giancarlo Guizzardi. 2023. An Ontology-Driven Approach, for Process-Aware Risk Propagation. In *The 38th ACM/SIGAPP Symposium on Applied Computing (SAC '23), 2023, Tallinn, EE.* ACM, New York, NY, USA, Article 4, 4 pages. https://doi.org/10.1145/3555776.3577795

1 INTRODUCTION

Risk is a pervasive phenomenon, depending on events that occur in a connected world, where objects interact and cannot be taken in isolation. This structural aspect of risk-affected environments motivates the large application of graph algorithms for analyzing how risk spreads in a given system. The application of these algorithms is commonly known as *Risk Propagation (RP)* [9].

At the current state, RP techniques are applied in different domains for risk analytics, where processes play a central role. For instance, RP is broadly adopted to analyze how occurrences of risk affect the sustainability of producer-consumer networks in supply chains [2]. Similarly, the propagation of risk is used to assess the impact of cyber-attacks on different assets of a given system [10]. In this context, it has been widely recognized that one key open challenge is to devise a solution that can be used to measure the propagation of risk in systems that involve dependencies between processes and physical objects [4, 7]. For instance, *how can cybersecurity risk be propagated from a cyberinfrastructure to the business processes of an organization? How may a machine breakdown affect the productivity of a company? How can we quantify the risk of machinery energy consumption deviation from the allowed thresholds and propagate the risk to the business processes of the host organization?* All these challenges can benefit from a *process-aware*[1] approach to achieve better risk propagation. Such an approach should be able to leverage information about how different processes, objects, and activities connect with each other, in domain-specific contexts (e.g., *customer relationships*, *enterprise planning*, *cyber assets*, and *supply chain*), and also at a domain-agnostic level, by covering concepts that are always present in different application contexts.

This paper advances the state-of-the-art in the research of RP techniques, by proposing a process-aware approach that is aimed at facilitating the assessment of RP between processes and objects with different levels of abstraction. The contribution leverages the combination of *i.* an ontology, which supports functionalities typical of *Semantic Web technologies (SWT)* and semantics-based intelligent systems, encoding a set of rules to be used for representing the risk dependencies within a system composed of objects and processes, and *ii.* a method to calculate the propagation of risk within the represented system. We implemented our approach in a proof-of-concept tool, which was validated in the cybersecurity domain.

The remainder of this paper is structured as follows. Section 2 describes the method embedded in our approach. In Section 3 we discuss some implementation aspects and we report on a demonstration to validate our solution. Finally, in Section 4 we reflect on our results and elaborate on future work.

SAC '23, March 27-March 31, 2023, Tallinn, Estonia
© 2023 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9517-5/23/03.
https://doi.org/10.1145/3555776.3577795

[1] Here, by adapting the definition provided in [3], we take "process-aware" as *"regarding systems that involve processes"*.

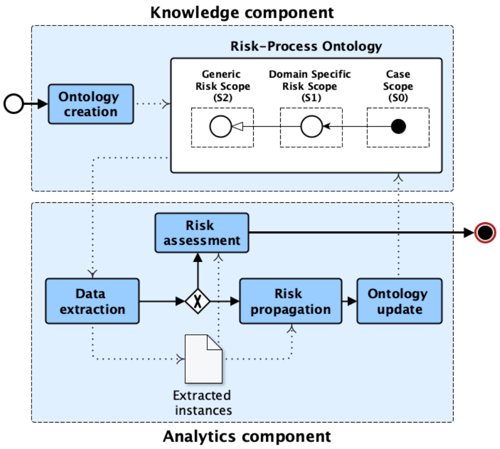

Figure 1: An overall view of the approach.

2 APPROACH DESCRIPTION

Our approach is grounded on the definition of "risk" provided in [8]. Then, we use risk to "*quantify the possibility of reaching some given objectives*", where such a quantity value is derived from the combination of the probability that a certain *risk event* occurs and a set of "*severity values*". For example, suppose that an attacker has read/write access to a database, namely, he can damage the database *integrity* and *confidentiality*. The read/write access represents the risk event and the severity values will be associated with the database integrity and confidentiality features. We employ here a simplified definition of risk as "an effect of uncertainty on objectives". An in-depth analysis of risk with regards to the proposed approach (inspired by [11, 12]), is part of the future work.

In the scope of this paper, the main observation is that we calculate the risk as $R = P * (S_1, ..., S_n)$, where P provides the probability that a risk event occurs, and each S_j encodes a severity value. Accordingly, the propagation task will start from a risk value, associated to a "*risk event*" (e.g., "*device damaging*"). The whole approach is aimed at capturing how the risk associated with this risk event can spread through all the elements (objects and processes) involved (directly or indirectly) in the event itself.

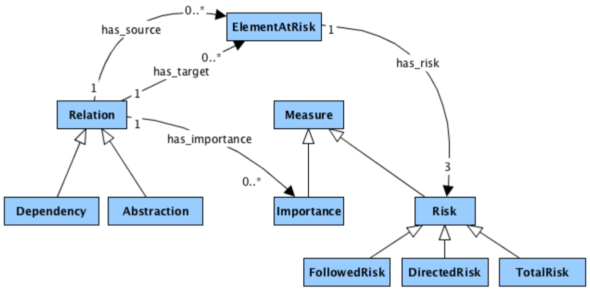

Figure 2: S_2 concepts and relations.

Figure 1 provides an overall view of our solution, which is composed by two main components.

2.1 Knowledge Component

The knowledge component holds an ontology (see Figure 1, *Risk-Process Ontology*), which is divided into three scopes. The *first scope* S_2, is composed of a set of generic concepts and relations related to risk, which are always required independently of any specific business domain. The *second scope*, S_1, extends S_2 with a set of domain-specific concepts and relations. S_1, is then mapped into the *third scope* S_0, composed of a use-case-specific types and instances.

Figure 2 provides a lightweight representation of S_2, composed of the minimal set of constructs required for the process-aware RP task. The main concept in this scope is `ElementAtRisk` which stands for both process types or objects at risk. For example, an `ElementAtRisk` could be specialized in S_1 by a concept representing a physical component of a system, such as a *"machine"*, or a business abstract concept such as a *"business activity"*. We keep implicit the different types of `ElementAtRisk` (as objects, process types) including their relations, and that the ultimate scope is with regards to business objectives, and values. However, since our main goal is to describe the overall approach, we take this lightweight model, which will be extended in future work.

Within a system, the risk is propagated from one `ElementAtRisk` to another according to some given relations. In the current approach, in order to model risk propagation, we identified two main types of relations. First, `Dependency` relations, which are used to model phenomena where the risk is propagated through a workflow composed of processes. For instance, two business activities can be connected by `Dependency` relations like *"triggers"* or *"causes"*. Second, `Abstraction` relations, represent cases where the risk is propagated from a lower to a higher level of abstraction. For example, the risk of a physical machine can be propagated to related business activities. Given a network of elements at risk and their connections, we identify three types of `Risk`. We call `FollowedRisk` the risk propagated through `Dependency` relations and `DirectedRisk` the risk propagated through `Abstraction` relations. `TotalRisk`, in turn, stands for the overall risk of an object, considering both its `DirectedRisk` and `FollowedRisk`.

Note that the knowledge component is aimed at supporting risk calculation from different perspectives, which can be represented within S_2 through the `Measure` concept and some *ad hoc* attributes. For example, in a cybersecurity use case, the risk could be quantified as from the CIA-triad standard [6], namely according to its potential impact on *availability*, *confidentiality*, and *integrity* of the related business activities. Finally, the knowledge component allows the user to control the *amount of risk propagated from one element to another* via the `Importance` concept, which is used to weight any given relation. For example, a confidentiality risk that was measured over a device and propagated to its correlated business activity should not necessarily be propagated to the following activity. In that case, the system supports omitting the propagation of a confidentiality risk from an activity to the following by setting an `Importance` of zero.

2.2 Analytics Component

The analytics component is used for a data extraction step that consists of querying the ontology of the knowledge component

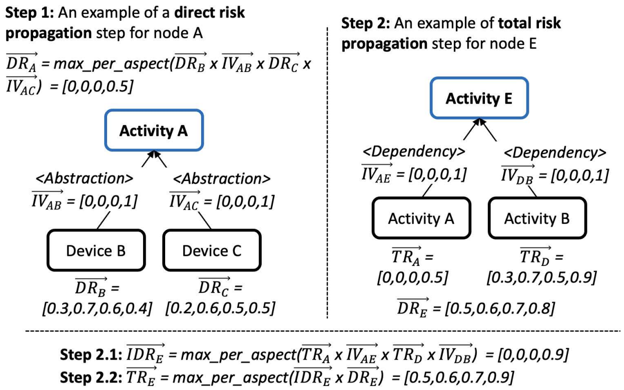
Step 1: An example of a **direct risk propagation** step for node A
$\overrightarrow{DR_A}$ = max_per_aspect($\overrightarrow{DR_B}$ x $\overrightarrow{IV_{AB}}$ x $\overrightarrow{DR_C}$ x $\overrightarrow{IV_{AC}}$) = [0,0,0,0.5]
Activity A
<Abstraction> $\overrightarrow{IV_{AB}}$ = [0,0,0,1]
<Abstraction> $\overrightarrow{IV_{AC}}$ = [0,0,0,1]
Device B
Device C
$\overrightarrow{DR_B}$ = [0.3,0.7,0.6,0.4]
$\overrightarrow{DR_C}$ = [0.2,0.6,0.5,0.5]
Step 2: An example of **total risk propagation** step for node E
Activity E
<Dependency> $\overrightarrow{IV_{AE}}$ = [0,0,0,1]
<Dependency> $\overrightarrow{IV_{DB}}$ = [0,0,0,1]
Activity A
Activity B
$\overrightarrow{TR_A}$ = [0,0,0,0.5]
$\overrightarrow{TR_D}$ = [0.3,0.7,0.5,0.9]
$\overrightarrow{DR_E}$ = [0.5,0.6,0.7,0.8]
Step 2.1: $\overrightarrow{IDR_E}$ = max_per_aspect($\overrightarrow{TR_A}$ x $\overrightarrow{IV_{AE}}$ x $\overrightarrow{TR_D}$ x $\overrightarrow{IV_{DB}}$) = [0,0,0,0.9]
Step 2.2: $\overrightarrow{TR_E}$ = max_per_aspect($\overrightarrow{IDR_E}$ x $\overrightarrow{DR_E}$) = [0.5,0.6,0.7,0.9]

Figure 3: Risk propagation steps, a running example.

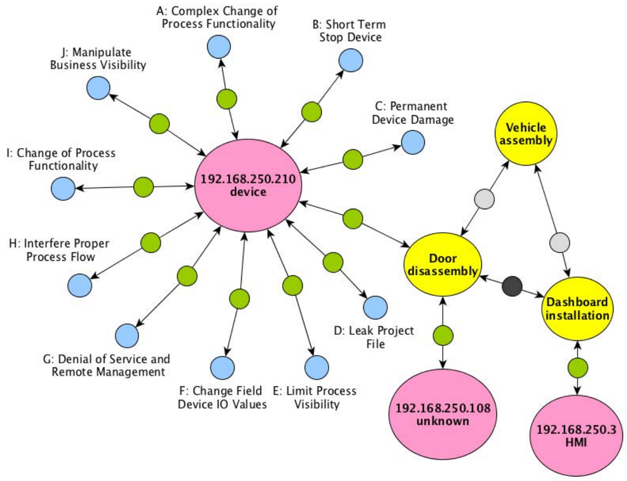
A: Complex Change of Process Functionality
B: Short Term Stop Device
J: Manipulate Business Visibility
C: Permanent Device Damage
I: Change of Process Functionality
192.168.250.210 device
Vehicle assembly
H: Interfere Proper Process Flow
Door disassembly
Dashboard installation
D: Leak Project File
G: Denial of Service and Remote Management
F: Change Field Device IO Values
E: Limit Process Visibility
192.168.250.108 unknown
192.168.250.3 HMI

Figure 4: A subgraph of the case scope (S_0).

through the S_2 constructs. Data extraction returns a *labeled property graph* structure where nodes and edges represent instances of ElementAtRisk and Relation respectively. Risk and Importance values are represented, instead, as vectorized properties of nodes and relationships respectively. Notice that we assume the risk over the leaf nodes (elements with a lower level of abstraction) as being given prior to the RP task.

Once the labeled property graph is generated, RP can be performed. This task occurs in two steps: *.i* the graph is traversed via a *Depth-first Search (DFS)* algorithm [1]; *.ii* the RP for each node is defined according to a risk function denoted as *max_per_aspect*. In the proposed method we take a worst-case scenario approach by quantifying the risk according to the *maximal risk per perspective*. For example, in a case where a business activity depends on two devices, and each has a different availability risk, a worst-case scenario approach assumes that both devices could be compromised by an attacker, and a shutdown of at least one device will disable the correlated activity. Thus, the propagated risk towards the business activity is set according to the maximal availability risk of both devices. The risk function gets a bag of vectors ordered by the different risk perspectives and returns the maximal value for each perspective.

Figure 3 provides an example of an RP task and the two steps composing it. The risk in the running example is a vector made of four values, each one representing a risk quantification from a different perspective. For example, in a cyberattack the following device perspectives could be affected: *confidentiality*, *integrity*, *safety*, and *availability*.

1 In the first step of the risk propagation task, the DirectedRisk (see $\overrightarrow{DR}$ in Fig. 3, Step 1) is propagated from the leaf nodes to the nodes with a higher level of abstraction using the Abstraction relation. The bag of vectors for each node is composed of the DirectedRisk vectors of the incoming nodes ($\overrightarrow{DR}_B$ and $\overrightarrow{DR}_C$), multiplied by the corresponding Importance vectors ($\overrightarrow{IV}_{AB}$ and $\overrightarrow{IV}_{AC}$) over the incoming edges. The multiplication is an *element wise*, namely, each element in the DirectedRisk vector is multiplied with the corresponding element in the Importance vector.
2 Once the DirectedRisk is propagated, the second step occurs according to two main sub-steps:
   2.1 The FollowedRisk vector of a node (denoted as $\overrightarrow{IDR}$) is calculated. In this case, the bag of vectors for each node is composed of the TotalRisk vectors (denoted as $\overrightarrow{TR}$) of its incoming nodes multiplied by the corresponding Importance vectors over the incoming edges. Notice that, the FollowedRisk vector over leaf nodes is set to zero.
   2.2 The TotalRisk of a node is calculated. In that case, the bag of vectors is composed of its FollowedRisk and DirectedRisk vectors. After step 2.2., the Risk-Process ontology can be updated with new risk values.

Finally, the analytics component also accounts for another step, what we call here "risk assessment". Here, the ontology can be queried to assess and analyze the risk state of the whole system in multiple ways. For instance, elements can be queried according to their level of abstraction and risk. Furthermore, alerts can be generated considering the deviation of the quantified risk from a pre-defined threshold (denoted as a *cardinal risk*). Similarly, analysts could analyze what is the element at cardinal risk, identify the risk's root causes, and prioritize mitigation steps accordingly.

3 IMPLEMENTATION AND DEMONSTRATION

This section discusses implementation details, and reports on a demonstration to validate the approach.

Implementation. The *knowledge component* is deployed on *Neo4J graph database platform*, the *analytics component* and the whole pipeline orchestration are implemented as a *Python* application which interacts with Neo4J via an *ad hoc* Neo4J python library[2]. The program and the database interact as described in Figure 1.

To represent the ontology scopes, we adopted the *Ontology Web Language (OWL)*[3]. The model's *concepts*, *relations*, and *attributes*

[2] https://neo4j.com/, https://www.python.org/
[3] https://www.w3.org/OWL/

are expressed as *classes*, *object properties*, and *data properties*, respectively. S_0 is expressed as classes' *individuals* and their *properties* assertions. Once the OWL file of the three scopes is constructed, we import it to the database using the Neo4J *NeoSemantics* plugin[4]. This plugin transforms the OWL file into a *Labeled Property Graph (LPG)* structure. In this structure, the constructs of the model and the data are represented as nodes and edges within a graph database.

Demonstration. We demonstrate the approach through a cybersecurity risk assessment use case of a vehicle assembly manufacturing process. This example serves for showing how the proposed approach can be used for quantifying the risk of devices being compromised by a cyberattack, and then measuring the impact over the domain-specific risk scope.

In this scenario, the main concepts captured by the ontology (see S_2 and S_1) can be grouped into *a)* a *physical layer* composed of devices (denoted as `CyberAsset`) that could be compromised by an attacker; *b)* potential *intervention actions* (denoted as `CyberImpact`), which an attacker could perform over each device; *c)* processes (each one grouped as as `ProcessElement`). Cyber assets and process/activity elements are connected via relations of type `CorrelatedTo`, process/activity elements are connected via relations of type `ComponentOf` and `FollowedBy`. According to the S_2 distinctions, `CorrelatedTo` and `ComponentOf` are classified as `Abstraction` relations, `FollowedBy` is classified as a `Dependency` relation. Considering the given conceptualization, the risk is then measured over the different `CyberImpact` instances and propagated to `CyberAsset` and `ProcessElement` instances. Notice that, in this demonstration we measured risk according to the commonly used *CIA-triad* for a cybersecurity risk assessment, where the risk vector is composed of the perspectives of *confidentiality*, *integrity*, and *availability*. For example, a denial-of-service `CyberImpact` holds a substantial risk of availability, while a data manipulation `CyberImpact` holds a substantial risk of integrity and confidentiality. Since the risk is measured within an industrial facility, we extend the standard approach with a safety perspective.

Figure 4 shows a snapshot of the case scope (S_0) instantiating the ontology concepts. The `FollowedBy` (denoted as light-grey nodes) and the `ComponentOf` (dark-grey nodes) represent relations between process elements (yellow nodes), which are represented by three processes, namely: `VehicleAssembly`, `DoorDisassembly` and `DashboardInstallation`. The graph provides then the common cyber assets for each *ProcessElement* as well. This is represented by the *CorrelatedTo* relation (green nodes) between process elements and cyber assets instances (pink nodes). As from Figure 4, `DoorDisassembly` is connected with two `CyberAsset` instances, and `DashboardInstallation` relates to just one instance. Finally, the graph encodes the potential vulnerabilities of the selected cyber assets, by connecting them to a set of threat instances, categorized as `CyberImpact` (blue nodes), and each one is associated with a given risk vector. The subgraph in Figure 4 shows also that one `CyberAsset` instance (left) is connected to 10 `CyberImpact` instances, while the rest of the `CyberAsset` instances in the subgraph are not connected, i.e., they can be considered as "secure".

Once the ontology is set and imported, we run the *data extraction* step. Here we use Neo4J to query and extract elements at risk and the relations that are relevant to the RP task. The query uses S_2 constructs to support various domain-specific entities and relations and returns a set of records encoding a *i)* relation between a source to a destination element, *ii)* the risk vector over the source object, and the *iii)* importance vector of the relation.

After the data extraction step, the RP algorithm can be applied, generating values as from the example provided in Fig. 3. For instance, given Fig. 4, for each `CyberImpact` a risk vector is provided (e.g., in our example "C" returns $\overrightarrow{DR_C} = [0, 0.83, 0, 0]$) and this is propagated to `CyberAsset` objects, then to the `ProcessElement` objects given the S_0 data extracted from the designed ontology (e.g, given all the related `CyberImpact` nodes, "192.168.250.210" `CyberAsset` will be associated to $\overrightarrow{DR} = [0.53, 0.83, 0.33, 0.84]$. More details about the demonstration outputs can be found in [5] and at https://youtu.be/x4pxBp16vOQ.

4 CONCLUSION AND PERSPECTIVES

This paper presents an application that leverages the combination of *i)* a risk-process ontology and *ii)* a new method to calculate the propagation of risk between processes and objects with different levels of abstraction. Given the current promising results, we envision three main future perspectives. Firstly, we plan to leverage previous work on risk and value modeling [12] and provide a well-founded ontology for process-aware RP. Secondly, we are going to implement and compare different algorithms for the calculation and propagation of risk. Finally, we are going to test the approach over multiple risk-sensitive domains (e.g., *finance* and *healthcare*).

ACKNOWLEDGMENTS

This work was done in collaboration with Accenture Labs, Israel.

REFERENCES

[1] Baruch Awerbuch. 1985. A new distributed depth-first-search algorithm. *Inform. Process. Lett.* 20, 3 (1985), 147–150.

[2] Nishat Alam Choudhary et al. 2022. Risk assessment in supply chains: a state-of-the-art review of methodologies and their applications. *Annals of Operations Research* (2022), 1–43.

[3] Marlon Dumas et al. 2013. *Fundamentals of business process management.* Vol. 1. Springer.

[4] Gal Engelberg. 2022. Process-Aware Attack-Graphs for Risk Quantification and Mitigation in Industrial Infrastructures. (2022).

[5] Gal Engelberg et al. 2022. Towards an Ontology-Driven Approach for Process-Aware Risk Propagation. (2022). https://doi.org/10.1145/3555776.3577795 arXiv:arXiv:2212.11763

[6] Kim Fenrich. 2008. Securing your control system: the" CIA triad" is a widely used benchmark for evaluating information system security effectiveness. *Power Engineering* 112, 2 (2008), 44–49.

[7] Oscar González-Rojas et al. 2021. Quantifying Risk Propagation Within a Network of Business Processes and IT Services. *Business & Information Systems Engineering* 63, 2 (2021), 129–143.

[8] ISO 31000 2018. *Risk Management - Guidelines.* Standard.

[9] Jiaojiao Jiang et al. 2016. Identifying propagation sources in networks: State-of-the-art and comparative studies. *IEEE Communications Surveys & Tutorials* 19, 1 (2016), 465–481.

[10] Georgios Kavallieratos et al. 2021. Cyber risk propagation and optimal selection of cybersecurity controls for complex cyberphysical systems. *Sensors* 21, 5 (2021), 1691.

[11] Ítalo Oliveira et al. 2022. An Ontology of Security from a Risk Treatment Perspective. In *International conference on conceptual modeling*. Springer.

[12] Tiago Prince Sales et al. 2018. The common ontology of value and risk. In *International conference on conceptual modeling.* Springer, 121–135.

[4] https://neo4j.com/labs/neosemantics/