A survey on automation of security requirements in service-based business processes

Fernando A.A. Lins* and Erica T.G. Sousa
Department of Statistics and Informatics,
Federal Rural University of Pernambuco,
Recife, PE, Brazil
Email: fernandoaires@ufrpe.br
Email: erica.sousa@ufrpe.br
*Corresponding author

Nelson S. Rosa
Center of Informatics,
Federal University of Pernambuco,
Recife, PE, Brazil
Email: nsr@cin.ufpe.br

Abstract: Service-oriented computing (SOC) and business process management (BPM) are essential topics in computer science. Companies are widely adopting business process standards, such as BPMN, to model their business processes, while automatising them by using services from the SOC world. The automation leads to a gain of efficiency in executing business processes and facilitates the execution of tests. However, the automation also raises fundamental security concerns, especially considering that it can use external services available on the internet, e.g., web services. For example, companies are not comfortable to use systems that communicate over the internet without guarantees that secure actions have been adequately used. The main objective of this work is to provide a holistic view of current initiatives and tools to model and enforce security requirements in service-based business processes along with open research and practical challenges on this subject. The intention is that this work can serve as a source of theoretical and pragmatic ideas for those who want to execute their business processes having in mind security concerns.

Keywords: business process management; BPM; service-oriented computing; SOC; security; web services.


Biographical notes: Fernando A.A. Lins graduated in Computer Engineering at the Pernambuco University, Brazil. He also received his MSc and PhD degrees from the Federal University of Pernambuco, Brazil. Currently, he is an Associate Processor in the Federal Rural University of Pernambuco, Brazil. He has published more than 40 papers in relevant conferences and journals related to computer science, and his research interests include distributed systems, security, service oriented systems, cloud computing and internet of things.
1 Introduction

There is an increasing need for using and enforcing security requirements in business processes. Companies are widely adopting business process modelling standards to express and design the functional requirements of their businesses. Besides, security requirements have been considered a significant concern among system developers and users. Based on these facts, the association between these two topics is inevitable. In fact, several people from academia and industry are viewing security as the most relevant and needed non-functional requirement, especially considering the dissemination of web-based systems and the cloud. Companies would not adopt a solution that sends data over the internet until they can be certain that preventive actions were taken.

Nowadays, one of the most adopted notations to model business activities is the business process model and notation (BPMN) standard (OMG, 2013). In BPMN, the business process is essentially a collection of activities that takes one or more inputs and generates the associated output. This type of modelling standard is an attractive resource that facilitates the development of high-level business models. However, the business process modelling may not be enough. In many cases, the user wants to execute the modelled system in a computational platform. To perform this task, a considerable number of users have adopted service-oriented solutions (Papazoglou et al., 2007), which involve the utilisation of external web services to implement the needed functionalities. As web services are usually available at distinct locations, the execution of the business process may involve communication over the internet. In this case, sensitive data may be sent over the internet and be accessed by unauthorised persons. Furthermore, a user that has an in-depth knowledge of the business (business expert) may want to define specific security requirements, such as to encrypt data or to use authentication techniques, which need to be enforced in essential activities such as ‘credit card payment’. This fact leads to the necessity of using security concepts in business processes.

It is possible to state that one of the key activities of the business process management (BPM) (Van Der Aalst, 2013) is the modelling, i.e., the use of notations, like BPMN, to describe the business process formally. For several years, the focus of the
A survey on automation of security requirements in service-based business

BPM was on the system design level only, i.e., on the modelling activities. Nowadays, however, high-level BPM models are either executed by execution engines or mapped into executable models and configurations that can be directly executed. In essence, abstract models should be specified in such way that the business expert can describe requirements without requiring an in-depth understanding of BPM tools and standards.

To incorporate security requirements into the business process automation is not a trivial task, especially those executed by web service compositions. The explicit modelling and enforcement of security requirements are still a challenge for many reasons. Firstly, diversity at the security background of users involved in the business process definition. Secondly, lack of precise definition of security requirements at the business process level. Thirdly, the complexity of notations to express security requirements. Fourthly, difficulty in integrating security requirements into the business process definition, e.g., users may face problems to put together functional and security requirements in the same model. Fifthly, the complexity of mapping security requirements into security mechanisms, especially considering a large number of security enforcement resources that can be used. Finally, the complexity of enforcing security requirements at execution time. Based on that, the focus of this paper is to present current methodologies, tools and techniques to address all of these relevant issues.

In this context, the primary objective of this paper is to provide a holistic view of current initiatives and tools to model and enforce security requirements in service-based business processes along with open research and practical challenges on this subject. The intention is that this work can serve as a source of theoretical and pragmatic ideas for those who want to execute their business processes having in mind security concerns.

Before presenting the selected research initiatives, it is worth mentioning how the literature review was conducted. Firstly, the research question of this survey is centred on “how to automate service-based business processes considering security requirements.” This central research question was divided, for pedagogical reasons, into three distinct topics. These issues are related to modelling, translation and enforcement of security requirements in business processes. The keywords used in the search were ‘security business processes’, ‘secure business processes’ and ‘BPM security’. Finally, priority was given to initiatives published in the last ten years. The main idea is to be up-to-date, while not forgetting pioneer studies. The databases used in the search process were: IEEE, ACM, Elsevier, Springer, Google Scholar and the Google search engine itself.

This paper is organised as follows. Section 2 describes initiatives and open challenges on modelling and designing security-aware business processes. Next, Section 3 concentrates on presenting works and open issues related to the translation of business processes security requirements. Section 4 focuses on the enforcement of the security actions at the execution level. Finally, Section 5 presents the conclusions of this work.

2 Security requirements at business process level

A large number of works focuses on how to model security aspects of business processes. In recent years, users specify business processes using the widely known notation BPMN (OMG, 2013). BPMN has been adopted as the ‘de facto’ technology for modelling business processes. However, BPMN does not provide support for specifying non-functional requirements like security. This fact leads to the adoption of strategies to
incorporate security requirements into the business process automation as shown in the following subsections.

2.1 Graphical extensions of BPMN

Turki et al. (2012) presents a model-driven approach for modelling security requirements of service-based business processes. Figure 1, inspired on Turki et al. (2012), provides an overview of this solution.

Figure 1  Overview of Turki’s approach for security requirements modelling (see online version for colours)

Turki’s approach consists of three phases. In the first phase, namely BPMN business process modelling, the business expert models the business process using the standard BPMN elements. Next, in phase service and security information design, more information related to the service and security design are inserted. Finally, the improved business process can be used as input of phase business process code generation, which is responsible for generating execution-level resources for the security enforcement.

Turki also proposes intuitive graphical notations (generally based on the visual of locks) to enrich BPMN diagrams with security properties. This point facilitates the visualisation of the security requirements by users not familiarised with security terms and notations. Furthermore, these graphical notations are mapped into specific security requirements, such as integrity, confidentiality, authentication and privacy, i.e., each specific security requirement has a particular graphical notation. Hence, there are several graphical notations and not only one representing security. This fact is an advantage if we consider that the graphical elements can express security specific requirements. Meanwhile, this can also be considered a disadvantage, because the business experts have to learn the meaning of each graphical notation to understand the enriched BPMN model.

In another related initiative, Rodriguez (2011) extended BPMN to allow the incorporation of security requirements into business process diagrams. For each security requirement, a dedicated BPMN graphical notation is introduced and can be associated with various BPMN standard elements. This association is done through a BPMN security extension, which relates the BPMN standard notation and the proposed security
A survey on automation of security requirements in service-based business

elements. Such as Turki et al.’s (2012) approach, Rodriguez (2011) introduces an intuitive graphical notation to represent security. In comparison to Turki’s work, the graphical notations are very similar, but Rodriguez’s (2011) approach proposes specific internal codes used to differentiate each security requirement. Some examples of internal codes are: CO relates to confidentiality, IN means integrity and AU refers to authentication.

Figure 2  Example of security requirement modelling based on Rodriguez’s approach (see online version for colours)

Figure 2 presents a business process enriched with ‘confidentiality’ and specified using the Rodriguez’s proposal. The business expert expresses that confidentiality actions must be executed just before task credit card payment.

Comparing Turki and Rodriguez approaches, it is possible to highlight that Rodriguez presents graphical elements that may be more expressive than Turki because the internal codes are easier to remember than different graphical notations.

Menzel et al. (2010) present a domain-independent security model and security patterns. The primary objective of this initiative is to facilitate the generation of security configurations based on the modelled requirements. For this purpose, Menzel adopts a model-driven approach in such way that information at the modelling layer is gathered and translated into a domain-independent security model. Menzel’s approach uses BPMN to express business processes. Meanwhile, the author also proposed an UML-based configuration to represent the relationships between security and business process elements.

Regarding modelling, Menzel introduced the notion of ‘profile’. A profile is directly connected to a security goal to identify the security level desired by the user. For example, the business expert may determine that a specific security goal should have a ‘high’ or ‘low’ profile. If the business expert selects the ‘high’ profile, a stronger encryption algorithm and key length should be chosen. Otherwise, if the ‘low’ profile is selected, a simpler encryption algorithm can be chosen. Based on that, users can model a set of security configurations that better fit what they need. This strategy is not exactly a new idea because some web browsers adopt similar strategies to configure the security parameters of the user. However, this is an interesting idea, because business experts can express the desired security level without having to be familiarised with specific security concepts and technologies.

Similarly to Turki and Rodriguez, Menzel also proposed specific graphical notations to express security needs in BPMN business process diagrams. More specifically, Menzel proposed two artefacts named organisational trust and security group. The first artefact represents the ‘trust relationship’, which can be used to state the confidence between two or more participants better. Menzel considers that this is an important security
information that should be represented in the BPMN diagram. An essential property in this context is the ‘rating’, which can vary from negligible (no need to take any security action or to be concerned with security issues) to extreme (security issues can cause several damages for people and enterprise). Another graphical element was proposed to represent the ‘rating’ property that represents each grade level.

The second artefact proposed by Menzel, security group is used to group tasks, artefacts and pools that share similar security intentions. This artefact has a specific graphical notation, which is a dashed rectangle. Figure 3 presents an example of the utilisation of this proposed artefact. The security group artefact is grouping the BPMN tasks insert flight reservation parameters and credit card payment. The same ‘rating’ is valid for this artefact, the business expert can choose the desired level of each security property, which can range from negligible to extreme.

Figure 3  Example of security group artefact (see online version for colours)

Lins et al. (2016) proposed another relevant work. In this work, the author introduces a strategy named BPA-Sec to represent security requirements and graphical notations to insert them in a BPMN business process model. According to the author, the first step towards the explicit treatment of security requirements in business processes definition is the ability to express them. Rosa et al. (2004) introduces a set of abstractions, namely NF-attribute, NF-statement and NF-action, to model and reason about non-functional requirements. To use the abstractions to precisely specify how the security requirements should be fulfilled, BPA-Sec extended the original semantics of these abstractions and added other ones, adopting some principles described in the following.

Firstly, the notion of a NF-action ‘affecting’ a NF-attribute (as initially defined by Rosa) was extended, so that it may also ‘implement’ a NF-attribute. For example, when a NF-action like ‘to encrypt data’ is implemented, it does not merely affect the NF-attribute security. It implements it as well. Secondly, an additional abstraction named NF-group was introduced, that lets users group NF-actions in such a way that facilitates their reuse. Thirdly, the NF-property abstraction was proposed to enrich the definition of the NF-actions. For example, the security NF-action ‘to encrypt data’ has the NF-properties ‘encryption algorithm’ and ‘key length’, which serve to include more details about it. With this abstraction, more information about the enforcement of security mechanisms is provided, making this implementation closer to the business expert needs. Finally, graphical elements were proposed to be used at modelling level, enabling business experts to enrich their models with security requirements.
Figure 4 shows a business process annotated with security requirements. The business process includes activities for a ‘car reservation’, which has four tasks and two of them were annotated with security requirements. It is possible to note that both NF-attributes confidentiality (represented by a cloud) and NF-actions UseAuthentication and UseCryptography (modelled by rectangles) appear in this figure and not only the security attribute.

In comparison to the Menzel’s approach, BPA-Sec proposes a shorter core set of generic abstractions and notations that could be used to express various security requirements rather than providing a fixed set of dedicated elements. Another relevant point to be highlighted is the presence of security actions in the BPMN graphical diagram. BPA-Sec proposed a graphical notation specifically for this purpose. By using graphical elements to represent security details may explicitly show the business expert some relevant information. However, some users may feel more comfortable in analysing a cleaner diagram and setting the specific actions or configurations in other location.

Another relevant initiative in the security modelling context is Ahmed and Matulevicius (2014). In this initiative, the authors propose a method to introduce security requirements to business processes that considers collaboration between business and security analysts. This method includes the proposition and use of specific graphical notations to express security objectives in business processes. The modelling process is developed through the use of security risk-oriented patterns. The method itself is composed of activities such as security objectives identification, security risk analysis, the annotation of security requirements in the business process model and so on.

It is possible to state that Ahmed and Matulevicius (2014) is another interesting initiative that proposes the insertion of graphical notations in business process models. However, one contribution that differs from other mentioned initiatives, such as Menzel et al. (2010) and Lins et al. (2016), is the presentation of a collaborative strategy to integrate business and security analysts in the modelling process. For example, both
analysts perform security objectives identification. This collaborative strategy can provide a richer security configuration because all involved analysts contribute in the modelling process.

In another interesting work, Altuhhova et al. (2013) proposes a structured approach that extended BPMN to express security risks and risks treatment in BPMN business models. This approach focuses on how BPMN activities could be annotated with security concerns. One of the main contributions is the development of a ‘security risk-aware BPMN’, which can be used to express security assets, security risks and countermeasures (Altuhhova et al., 2013).

To model security requirements in BPMN business processes, Altuhhova initially defines that different colours can be used to represent different resources in the BPMN diagram. More specifically, black, red and blue colours are used to express asset-related constructs, risk-related constructs and risk treatment-related constructs, respectively. Altuhhova also proposed a lock to represent the security objective which is similar to other abstractions presented previously. This lock also has a specific code inside that determines the security purpose: confidentiality (‘C’), integrity (‘I’) and availability (‘A’).

Also, BPMN traditional elements (especially tasks) are used to model not only functional activities but also security activities, such as to authenticate users, scan incoming messages and so on. By doing that, security and business experts can adequately model security activities using a BPMN-based process approach. This strategy substantially differs this approach from others and has some implications. To better illustrate it, a simple example is presented in Figure 5.

**Figure 5**  Example of security modelling using Altuhhova approach (see online version for colours)

![Example of security modelling using Altuhhova approach](image)

In functional terms, this example has only two events (*user request* and *user request was processed*) and one task (*process user request*). Regarding security, the business expert wants to express that the user must be authenticated before processing the request. To accomplish this, the modeller added the tasks *authenticate user* and *deny request*, the gateway *user was authenticated?* and the event *user request was denied*. It is possible to observe that the security modelling of this figure has a different colour, i.e., the functional elements are in black, while security-related elements are in blue.
It is worth observing that the expressiveness of Altuhhova’s approach is evident. By marking the security modelling in different colours, the diagram highlights the security elements in the diagram. Also, modelling security requirements using a process-oriented strategy may facilitate the understanding of the security process by the business expert. The business expert is familiarised with process-oriented modelling, especially because this expert uses a process-oriented standard to model the business model. However, someone can argue that this approach overloads the BPMN model not only with additional notations but also in the process flow, by merging the functional and security flows in the same diagram. Depending on the number of security requirements to be modelled, the resulting security flow can be significantly higher in comparison to the functional flow.

Saleem et al. (2012) proposed another interesting initiative in the area of security requirements modelling. The author introduces a domain specific language (DSL) to support the specification of security requirements in business processes. According to Saleem et al. (2012), DSLs can be used to formalise a modelling language capable of representing both different business domains and system aspects like security. Saleem’s approach deals with confidentiality, integrity, availability and traceability/auditing. Each of them was associated with a security symbol, which is a graphical element that expresses the security requirement in a BPMN diagram. To illustrate Saleem’s approach, Figure 6 presents a simple example in which the business expert wants to model that the data used in the process credit card payment must support confidentiality-related mechanisms.

Figure 6 Modelling security risks example using Saleem approach (see online version for colours)

It is possible to observe that this graphical notation differs somehow from ones presented previously. It is not only a lock with code inside but a symbol that intuitively represents what has to be done. For example, in Figure 6, the idea is to secure the data that should be transmitted. It is possible to note that these symbols differ, in conceptual terms, from other security graphical notations and this fact can impose barriers to the adoption of this approach. However, the proposed symbols are more intuitive, as they better represent, in a visual way, what the business expert is trying to express.

Finally, Maines et al. (2016) proposes an innovative approach for representing security in BPMN diagrams. The authors advocate the use of three dimensions (3D) for modelling and representing BPMN elements and security requirements. The author evaluates some existing approaches, such as Rodriguez et al. (2011), Saleem et al. (2012) and Salnitri et al. (2015) and then concludes that the use of 3D can mitigate existing issues such as complexity management and cognitive overload. With the 3D usage, it is possible to model and control a large number of symbols, without incurring in cognitive overload. However, not many tools exist to enable the 3D usage, both for functional or security modelling.
2.2 Textual extensions of BPMN

Mulle et al. (2011) proposes a security language embedded into BPMN to represent security requirements and constraints. This is carried out by using BPMN artefacts with the desired requirements. According to the authors, the reasons for that are because this approach both conforms to the BPMN 2.0 at a syntax level and reduce the implementation efforts as BPMN artefacts are available in most available BPMN modelling tools (Mulle et al., 2011). In Mulle’s approach, security constraints are inserted in BPMN as structured text annotations, in a declarative form. In this approach, annotations have a generic structure, which is presented as follows:

\[
<< \text{Annotation term : list(parameter - name) = "value" >>}
\]

This security annotation can have none, one or more parameters. Each parameter is particular to some BPMN modelling elements, such as activities and data objects. It is possible to observe that Mülle proposes an in-depth security language in which a substantial number of constraints can be expressed. All of these annotations are specified and appear in the modelled BPMN business process. However, depending on the number of security constraints, the business model can become overloaded, because a considerable number of text annotations may be required to express the user security needs.

Other relevant work related to textual annotation in BPMN is Salnitri (2016) and Salnitri et al. (2014). The authors propose SecBPMN, a security extension for the BPMN standard. This security extension introduces security requirements in business processes using both graphical notations and textual annotation. Most initiatives focus on only one of these possibilities. Salnitri (2016) and Salnitri et al. (2014) proposes both a considerable set of graphical notation, which can be used directly in the BPMN diagram and a query language for representing security policies named SecBPMN-Q.

It is possible to highlight that Salnitri contributed in both graphical notation and textual annotation areas. The interesting point of this initiative is the possibility to enrich the BPMN diagram with both graphical and textual information, increasing the security enrichment options. However, as mentioned in the previous subsection, users may not feel comfortable in analysing a BPMN diagram overloaded with security information. And specifically, in this case, both with graphical and textual notations.

In another related initiative, Varela-Vaca et al. (2013) proposes a standard representation to express security countermeasures to allow their automation. This initiative adopts security patterns for dealing with security risks. In short, the author suggests an approach that uses security patterns for the selection of specific countermeasures for specific risks. This selection is carried out based on information provided in the requirements artefacts and primary business goals. To accomplish that, a formal ontology was proposed to represent concepts such as security intentions, security goals, risk attribute, risk type and security level (Varela-Vaca et al., 2013).

The author also proposes that risks can be inserted into the business process high-level model. To illustrate this proposition, Figure 7 shows an example in which a user request arrives and should be:

1. received and analysed by the system
2. processed.
The business expert annotates the security risks (related to authentication and availability) in the BPMN business process using BPMN elements.

One of the main contributions of Varela-Vaca’s approach is to define how to incorporate security risks into BPMN and associate possible security patterns that can be used to treat these risks. Despite that, considering that the risks are inserted in the BPMN diagram as simple texts, the BPMN diagram may become too overloaded if there are a high number of security risks.

2.3 Comparative overview

Table 1 summarises the approaches presented previously. The most relevant initiatives were categorised considering three aspects:

1. adoption of graphical notation
2. adoption of textual annotations
3. if the initiative supports the security properties enrichment, e.g., by specifying with cryptography algorithm should be used in the confidentiality enforcement.

Table 1: Summary of approaches on incorporating security into BPMN models

<table>
<thead>
<tr>
<th>Approach</th>
<th>Kind of extension</th>
<th>Enrichment of security properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ahmed and Matulevicius (2014)</td>
<td>Graphical</td>
<td>No</td>
</tr>
<tr>
<td>Altuhhova et al. (2013)</td>
<td>Graphical</td>
<td>No</td>
</tr>
<tr>
<td>Lins et al. (2016)</td>
<td>Graphical</td>
<td>Yes</td>
</tr>
<tr>
<td>Maines et al. (2016)</td>
<td>Graphical</td>
<td>No</td>
</tr>
<tr>
<td>Menzel et al. (2010)</td>
<td>Graphical</td>
<td>Yes</td>
</tr>
<tr>
<td>Mulle et al. (2011)</td>
<td>Textual</td>
<td>Yes</td>
</tr>
<tr>
<td>Rodriguez et al. (2011)</td>
<td>Graphical</td>
<td>No</td>
</tr>
<tr>
<td>Saleem et al. (2012)</td>
<td>Graphical</td>
<td>No</td>
</tr>
<tr>
<td>Salnitri (2016), Salnitri et al. (2014)</td>
<td>Graphical and textual</td>
<td>No</td>
</tr>
<tr>
<td>Turki et al. (2012)</td>
<td>Graphical</td>
<td>Yes</td>
</tr>
<tr>
<td>Varela-Vaca et al. (2013)</td>
<td>Textual</td>
<td>No</td>
</tr>
</tbody>
</table>
The main reason for including the security properties enrichment in this comparison is based on the fact that, depending on the context, the business expert may require a more detailed modelling approach or not.

It is worth observing that two different kinds of approaches are somehow conflicting. The first kind proposes graphical notations, i.e., the insertion of graphical symbols to represent security in BPMN diagrams. In turn, the second type advocates the use of textual annotations, i.e., the insertion of text in the mentioned diagram. One particular approach advocates the use of both approaches in the same diagram, but as previously mentioned, the diagram may become overloaded with different types of notations and semantics.

It is possible to affirm that graphical notations are being more adopted in comparison to textual ones, as can be noted in Table 1. Someone can argue that graphical notations may have a better impact on the modelling process that textual ones. However, more research is needed not only to show which one is the best option but also to define which type of graphical notation is more adequate.

2.4 Open challenges

Next, we present a set of open challenges on modelling of business processes regarding security issues:

- **Implementation of web-based and collaborative versions of BPMN business modelling tools with security extensions.** Currently, a considerable number of BPMN modelling tools and security extensions works on a desktop setup. An interesting capability should be the collaborative modelling. For example, Ahmed and Matulevicius (2014) proposes a collaborative framework in which business and security analysts contribute to the security enrichment. The proposition of a collaborative BPMN tool can make viable the online collaboration, e.g., in a virtual meeting.

- **Standard BPMN security extension.** Existing BPMN extensions (Lins et al., 2016; Rodriguez et al., 2011; Varela-Vaca et al., 2013; Turki et al., 2012) address security concerns in BPMN diagrams. However, each initiative proposes its notation, which creates a barrier to their adoption as they make inviable the collaboration between security analysts from different companies using different modelling strategies. In addition, by analysing the initiatives described in this section, it is possible to observe that some of them propose graphical notations that model specific security requirements, e.g., non-repudiation Salnitrini et al. (2015). Other initiatives introduced a small number of graphical elements, in which they can represent a significant number of security requirements. It is important to investigate a generic approach that can offer an attractive level of detail without compromising the notation overload.

3 Translation of security requirements into executable configurations

This section focuses on initiatives related to the translation of security requirements into executable artefacts. Regarding automation, high-level security requirements have to be
mapped to a set of executable resources, e.g., configuration files. How should this task be carried out? This issue is the central problem addressed in this section.

The first work to be presented was proposed by Menzel et al. (2010). The author introduces a model driven architecture (MDA)-based transformation process to translate high-level system models enriched with security requirements into executable level security artefacts. Menzel et al. (2010) aim to incorporate MDA concepts and design pattern ideas in the same approach, which can be used to guide the generation of the required security configuration files.

**Figure 8** Model-driven security translation in SOA Security Lab (see online version for colours)

To illustrate the strategy proposed by Menzel, Figure 8 shows a general overview of the proposed approach. This approach consists of three layers. Initially, *security intentions* are modelled in layer *modelling*. Next, the security intentions are transformed into a platform independent model (PIM) in layer *PIM*. This translation process also handles the mapping of security intentions into *security constraints* and the mapping of security profiles into security patterns. The resulting model is a PIM model. Finally, in layer *web service layer*, WS-security assertions are generated based on the security constraints and will be used as a basis to enforce the security requirements at the execution level.

Two important points should be highlighted in the context of Menzel’s work:

- A policy meta-model is used to support the specification of security requirements. The author proposes a meta-model that serves as an abstraction layer for security process languages and standards. This strategy is essential to enable the generation of security configuration files in different policy standards and security enforcement resources;

- Utilisation of security patterns. Design pattern is a well-known concept in several areas of computer science and Menzel proposed its usage because it provides reusable expert knowledge. The author recommends a security pattern approach based on a formalised meta-model. For example, security intentions are related to the security pattern ‘problem’ and the security constraints are related to the security pattern ‘solution’. In this way, MDA and design patterns are used together to provide the desired solution.
In summary, Menzel proposed an approach to define security requirements at modelling level and facilitate the transformation of these requirements into WS-SecurityPolicy security configurations. The business expert cannot only model the security intention at high-level but also translate it using resources like design patterns. The SOA Security Lab also adopts a model-driven approach, which helps to better present the general structure of the solution and the mapping strategy. However, the security actions are already pre-defined with related security properties and do not allow the addition of other properties or even the modification of an already defined one.

Wolter et al. (2009) also proposed an interesting work on the translation of security requirements. The author introduced an approach for inserting business requirements into business processes with more MDA essential elements and more details in the translation architecture in comparison to Menzel et al. (2010). Wolter et al.’s (2009) framework starts with the creation of the computation independent models (CIM), which are developed by business and security experts with the support of a particular modelling tool. The business process modelling step generates a computation-independent model. The creation of the CIM model is carried out by a process modelling tool (Wolter et al., 2009), which is a prototype that has a security and a business process editor. Using this tool, the business expert can generate a model enriched with security requirements used, in the CIM to PIM transformation, as input to produce platform independent security configurations. More precisely, platform specific security policies. These configurations are based on the MDA concept of PIMs. Next, specific configurations files for the enforcement of the security policies (more precisely, ready-to-deploy security policies) are generated and can be used at a low level (in conjunction with the executable process definition). Finally, the generated files are used as input by a SOA-based environment, which is responsible for executing the business process and enforcing the security actions using an execution engine and security enforcement modules.

Wolter’s work is close to SOA Security Lab (Menzel et al., 2010). The author detailed the collaborative environment used as a basis for the construction of the transformation framework. Next, some MDA concepts were used to guide the translation process, i.e., CIM to PIM and PIM to PSM transformations. A key point of this work is the utilisation of a method to check the correctness of the transformation. However, Wolter’s approach does not use a service information in the translation process.

In another relevant research initiative, Argyropoulos et al. (2017) proposes the use of process-level security patterns to support the operationalisation of security requirements in business processes. These patterns, which can be considered reusable process fragments (Argyropoulos et al., 2017) can be integrated into business process models to enforce security requirements. The main idea is that the patterns should be generic enough to be instantiated by different security approaches and technologies.

To support the proposed approach, Argyropoulos introduced a framework and this framework is composed of three components. One of these components is the model transformation component, which has transformation rules to enable the generation of a security-annotated process skeleton and a secure business process model. An interesting contribution is the proposition of a hybrid reference process model, which aims to capture the variability related to the possible security technologies. This idea is very interesting, however, the hybrid reference process model itself was not fully described, which prevents a more accurate analysis of its capabilities.

Schmeling et al. (2011) proposed another work related to the security automation. This work has three outstanding contributions:
A survey on automation of security requirements in service-based business

In short, Schmeling et al. (2011) proposed a theoretical automation approach along with a toolset. The proposed execution environment is shown in Figure 9.

Figure 9 Overview of Schmeling translation/execution environment (see online version for colours)

The first relevant point to be observed in this figure is the presence of two modelling artefacts: BPMN business process and non-functional model. Some experts produce both artefacts with the aid of a modelling editor also proposed by Schmeling et al. (2011). After the modelling of the business process and the non-functional concerns, the translation takes place. The translator has been implemented as a model-to-model transformation based on the XTend language (Schmeling et al., 2011).

Despite being a promising approach, the translation process itself was not fully described. Also, there is not a tool to perform the translation process. These facts prevent an accurate analysis of the translation mechanisms.

In another interesting research initiative, Salnitri (2016) and Salnitri et al. (2015) introduce a novel approach to support automated generation of secure artefact-centric implementations. More specifically, meta-models are proposed to specify the mapping relationships between the modelling and execution levels. This initiative is the result of
an academic and industrial collaboration, in which SecBPMN2 (Salnitri, 2016) was integrated into SAP River (Salnitri et al., 2014), an industrial artefact-centric language.

The translation process proposed by Salnitri is only focused on available SecBPMN2 security annotations and each one of these security annotations has a corresponding enforcement rule. Considering this context, there are specific rules for the following security requirements: integrity, authenticity, accountability, non-repudiation, auditability, confidentiality, privacy, binding of duties, separation of duties, availability and non-delegation (Salnitri et al., 2015).

It is possible to observe that Salnitri focused on a large number of security requirements, including availability and non-delegation, which are not considered in related research initiatives. Besides, the security translation rules were implemented in a prototype, which allows its actual use by practitioners. However, this prototype only focuses on the translation from SecBPMN2 and SAP River, which impose barriers to its adoption. Besides, as described by Salnitri et al. (2015), the code must be manually inserted for the execution of the security enforcement. Skeletons are generated, which must be completed with more accurate information like the needed cryptography algorithm.

Finally, Lins (2012) and Lins et al. (2016) introduce an integrated methodology and tooling to automate security-aware service-based business processes. One of the core elements of this work is the BPA-Sec translator, which is responsible for carrying out the necessary translations to generate the platform-independent and platform-specific artefacts. Some relevant characteristics of this initiative include the BPMN to WS-BPEL translation, decoupling from specific BPMN tools, decoupling from a particular WS-BPEL orchestration engine and decoupling from a specific security enforcement module.

Figure 10   BPA-Sec translator overview (see online version for colours)
Inspired by the MDA approach, the BPA-Sec translator has two internal components: platform independent translator and platform specific translator. Figure 10 presents a general overview of this tool.

The first component of the BPA-Sec translator, namely platform independent translator, is responsible for generating the needed platform-independent artefacts (related to the executable business process, service information and security information) that can be used in two situations:

- The platform-independent artefacts are used directly as input to the platform specific translator to generate all required executable artefacts. This is the default flow of the BPA-Sec approach (Lins, 2012). If the BPA-Sec environment natively supports the chosen execution-level tools (e.g., orchestration engine), the translations allows both the automation process and the enforcement of the business process and security requirements at low-level.

- The platform-independent artefacts can serve as the basis for both implementing security-aware business processes and documentation purposes. Instead of directly generating executable configurations, the involved experts can work on the manual development of the business process and security actions. In this way, BPA-Sec Translator still useful when the BPA-Sec environment does not natively support the execution-level tools.

Considering the standards adopted in the BPA-Sec translator, namely BPMN and WS-BPEL, the platform independent translator maps security annotated BPMN specifications into a platform-independent WS-BPEL along with service and security descriptions. The annotated business process specification (BPMN business process enriched with security and service information) goes through the parser module to generate the annotated BPMN intermediate artefact. Next, this artefact is used in the generation of three intermediate artefacts: service, security and WS-BPEL artefacts. These artefacts use Java classes to represent internal information. In fact, a particular package contains all classes utilised by the translation process to describe entities in the various standards.

The second internal element of the BPA-Sec translator is the platform specific translator. Unlike the platform independent translator, its core translator has only two internal translators, namely specific BPMN to WS-BPEL translator and specific security translator. Meanwhile, there are only two serialisers: specific WS-BPEL serialiser and specific security serialiser. The service translator and the service serialiser are not needed because the service information is used by the specific BPMN to WS-BPEL translator to generate the platform specific WS-BPEL. The output of this translator is the platform specific artefacts, which are composed of a platform specific executable WS-BPEL artefact and platform specific security artefacts.

At this point, it is pertinent to highlight that BPA-Sec proposes particular translation rules for the security actions properties. To have an automatic automation process, it is also necessary to consider the properties of security actions in the translation process. Otherwise, to generate specific code for the security action still possible, but this code has to be updated to set specific parameters that are fundamental not only to the security actions enforcement, but also to better define the desired level of these actions. For example, in a security action like cryptography, the key length has a relevant role, because of a short key length, e.g., 56 bits, is more insecure in comparison to a 256-bits
key. If security actions properties are available in the translation process, it is reasonable to expect that richer configurations could be achieved. This fact is considered one of the main contributions of the BPA-Sec initiative, i.e., to consider security actions properties not only at modelling level but also in the translation process. Besides, BPA-Sec proposes both functional and security translation in the same approach, which is an interesting contribution.

3.1 Comparative overview

Table 2 presents a comparative overview of the initiatives introduced in this section. To compare them, three aspects have been considered: if the solution is based or inspired by the MDA, if they have contributions on the functional translation area, i.e., translation of high-level into executable business processes and if they are supported by tools (or a set of tools).

<table>
<thead>
<tr>
<th>Approach</th>
<th>MDA-inspired approach</th>
<th>Functional translation</th>
<th>Tooling support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argyropoulos et al. (2017)</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Lins et al. (2016)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Menzel et al. (2010)</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Salnitri (2016), Salnitri et al. (2015)</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Schmeling et al. (2011)</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Wolter et al. (2009)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

As can be observed in the table, most initiatives tried to fully adopt or just use MDA concepts. This can be considered a positive point, because MDA helps to structure the translation process and the existence of platform-independent and platform-specific artefacts contributes to the dissemination of the solution, as the platform-independent artefacts are not related to one particular orchestration engine or a security module. Other relevant aspect described in Table 2 focuses on the proposal of functional translation mechanisms in conjunction with the security translation. In a considerable number of cases, business users may be interested in both functional and security translation and having these possibilities in the same solution is interesting. Finally, the last aspect is related to the existence of tools to execute or evaluate the proposed solution. This point is important because the proposal of theoretical solutions only can impose barriers to their adoption.

3.2 Open challenges

Despite the recent advances in the translation of security requirements, some research opportunities are still open:

- **Formalisation of translation rules.** Works on the translation of security requirements into security mechanisms describe the translation rules in an informal way, e.g., (Lins et al., 2016; Salnitri et al., 2015). In some cases, the rules are not even clearly
defined. Hence, the formal specification of these rules should allow their systematic use, facilitate the reuse, validation and analysis of conflicts.

- **Causal relationship.** Runtime information can subside users in deciding on web service, business process or security requirements changes. Meanwhile, these changes should happen without completely stopping the whole system. A causal relationship between high-level requirements and their respective security mechanisms can guarantee that changes to the security requirements reflect directly on their runtime enforcements.

- **Intermediate platform-independent notation standardisation.** As in the case of BPMN security requirements modelling, in which many possibilities can be chosen, there is not a standard manner to specify security requirements in a platform-independent context. The existence of a standard platform-independent notation can help not only the collaborative modelling between distinct BPMN security extensions but also can serve as a basis for the enforcement of security requirements for external tools.

### 4 Enforcement of security requirements at execution level

This section focuses on how to enforce security requirements of business processes at the execution level. It is important to remember that business processes, in the context of this paper are implemented using service compositions. Based on this fact, the works shown in the following centre on the enforcement of security requirements considering an environment based on web service compositions.

#### 4.1 Enforcement of security requirements using pre-built security tools

Menzel et al. (2010) proposed the generation of security configuration files that can be executed by Apache Rampart (Apache Software Foundation, 2016). In other words, Menzel generates WS-SecurityPolicy compliant configurations, which are then just deployed in Apache Rampart.

Apache Rampart implements an extensive set of security protocols, such as WS-security (versions 1.0 and 1.1), WS-secure conversation, WS-trust and WS-SecurityPolicy (versions 1.1 and 1.2). Apache Rampart is deployed as a module in Axis2. This module introduces some security handlers responsible for carrying out the necessary security actions, e.g., encryption and decryption of messages and authentication. The SOAP header and body compose the SOAP message and Rampart performs actions on the SOAP header to enforce the desired security. Figure 11 presents a simple example that shows the Apache Rampart execution.

It is worth observing that Apache Rampart is just a security module and not a business process orchestration engine. In fact, this module works as an auxiliary resource for a considerable number of orchestration engines. Generating WS-SecurityPolicy enforcement code is an attractive strategy because business experts can use their preferred WS-BPEL orchestration engine along with Apache Rampart to enforce the security actions.
4.2 Security enforcement using aspect oriented programming concepts

A seminal initiative in the area of business process security enforcement with aspect-oriented programming (AOP) is the AO4BPEL/AO4BPMN (Charfi et al., 2010). This work presents a middleware-based solution to support security requirements in WS-BPEL compositions.

As WS-BPEL has not support for security, Charfi proposes a framework to integrate security requirements and web service composition specifications. To enforce this integrated specification, he proposed a framework composed of two main elements: a web service-based middleware, which provides support for reliability and security and a WS-BPEL process container, which also has an associated deployment descriptor. A security web service allows secure interactions between the WS-BPEL process and the participant services. This service provides essential security functions like cryptography. The process container wraps WS-BPEL processes and provides security support and this process container is implemented as a set of aspects. Finally, the deployment descriptor is a XML-based file responsible for specifying the security requirements of the business process activities and the needed security artefacts. The deployment descriptor is an essential component of Charfi’s architecture. The descriptor contains all security requirements that have to be fulfilled at execution time. In a design point of view, a programmer uses the WS-BPEL definitions to specify the functional requirements of the system and use the deployment descriptor to define the security requirements. To better illustrate the strategy proposed by Charfi, Figure 12 presents a general overview of the AO4BPEL security enforcement framework architecture.
Figure 12  Overview of the AO4BPEL initiative architecture (see online version for colours)

Some important points should be highlighted. The deployment descriptor is used only for specific reasons, i.e., it is not responsible for the enforcement of the security requirements. In fact, in the proposed framework, the security middleware services are responsible for the security enforcement. Each WS-BPEL process is executed inside a process container and this container intercepts the execution of selected activities in the context of the business process, e.g., the invoke activity. These activities are directly related to the enforcement of security requirements in the system. Figure 12 shows the deployment descriptor connected to the WS-BPEL business process and this descriptor specifies that particular business activities should use reliability actions. To accomplish that, the process container intercepts invocations in the business process execution and an external middleware service is invoked to enforce the required action.

Another important point in Charfi’s approach is the adoption of the aspect-oriented paradigm to build a container that can be easily extended. The main idea is to add new security middleware services by creating relevant aspects, which can be done by an AO4BPEL or AO4BPMN designer. Based on this strategy, it is possible to add a new security middleware service without changing the implementation of the WS-BPEL engine.

The main point of the AO4BPEL is the enforcement of the business process security requirements at execution level using AOP concepts. In practical terms, AO4BPEL proposes a security enforcement environment in which new security capabilities can be added, without requiring the adoption of other WS-BPEL orchestration engines or even updating the source code of the engine.
In another related work, Yahya et al. (2013) shows how users can adopt the AOP in a modular way. This modularity is achieved with the use of the AspectJ and some transformations between the BPMN model to AspectJ code are proposed. The functional automation produces Java code that represents an executable version of the business process and the non-functional automation generates the AspectJ executable code.

Additionally, Yahya et al. (2013) presents generated AspectJ codes to enforce common security requirements, such as confidentiality, integrity, binding of duties and authentication. The use of AspectJ to enforce security requirements in business processes is an interesting contribution of this work.

4.3 Other approaches

Another important initiative in the area of security enforcement is the BPA-Sec (Lins et al., 2016). This work proposes the BPA-Sec deployer and enforcer whose responsibilities include the generation of ready-to-deploy artefacts and the enforcement of security requirements. These responsibilities are described as follows.

- The first task, generation of ready-to-deploy code, is performed based on the target orchestration engine [e.g., Apache ODE (Apache Software Foundation, 2013)] and security enforcement module [e.g., Apache Rampart (Apache Software Foundation, 2016)]. This generation addresses the way that the target execution resources will receive the generated artefacts for execution. For example, a particular orchestration engine may require receiving input files in a zip format. This point is necessary to the automation process. If the business expert needs to configure the generated artefacts to deploy the business process and security actions, the automation cannot take place automatically;

- The second task is related to the security enforcement. The orchestration engine is responsible for executing the business process and providing security mechanisms to enforce some security requirements. Meanwhile, BPA-Sec deployer and enforcer enforce security requirements not provided by the standard built-in security module of the orchestration engine. BPA-Sec deployer and enforcer defines and deploys service interceptors into the message flow of the orchestration engine’s integrated security module. These interceptors call handlers for enforcing security requirements when the built-in security module of the orchestration engine cannot enforce such requirements.

Figure 13 presents an overview of the BPA-Sec deployer and enforcer architecture. This module consists of three internal components: business process deployable code generator, security configuration generator and auxiliary security enforcement module. The generators create ready-to-deploy code and the module is related to security enforcement. Both generators are composed of three similar internal elements: loader, configurator and deployer. The loader is responsible for receiving previously generated platform specific artefacts and forwarding it to the next element in a proper format. The configurators perform operations with the received files to create ready-to-deploy artefacts. Finally, the deployers are invoked to deploy the generated files in the execution-level resources, e.g., orchestration engines.

The last internal component of the BPA-Sec deployer and enforcer is the auxiliary security enforcement module. This component enforces the security requirements or
manages this enforcement when needed. The auxiliary security enforcement module interacts with the orchestration engine to implement the security actions in the business process execution. If the orchestration built-in security module implements all required security actions, no action is required. Otherwise, the auxiliary security enforcement module is invoked. The auxiliary security enforcement module has two internal elements: security handlers and security interceptors. Security handlers provide the security support not offered by the orchestration engine’s built-in security enforcer. Meanwhile, security interceptors define points in which service requests and responses are intercepted.

Figure 13  Architecture overview of the Bpa-Sec deployer and enforcer (see online version for colours)

It is worth observing that BPA-Sec deployer and enforcer is not a new orchestration engine. Its responsibility is to aid orchestration engines in enforcing security actions. Besides, only WS-BPEL orchestration engines are supported. If the business expert wants to adopt other engines, it is necessary to use the platform-independent files generated by the BPA-Sec translator, presented in the previous section and manually, enrich them with specific security configurations. Hence, business experts face difficulties if they choose an engine not natively supported by BPA-Sec deployer and enforcer.
4.4 Comparative overview

Table 3 shows a comparative overview of the existing initiatives on security enforcement of service-based business processes. Three aspects are used as a basis for this comparison: if the initiative uses pre-built security modules and tools (like Apache Rampart), if the work is based on aspect-oriented approaches and if the initiative adopts the WS-Security standard (OASIS, 2006).

<table>
<thead>
<tr>
<th>Approach</th>
<th>Pre-built security modules/tools</th>
<th>AOP-based approaches</th>
<th>Approaches based on WS-security</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charfi et al. (2010)</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Lins et al. (2016)</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Menzel et al. (2010)</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Yahya et al. (2013)</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

It is possible to observe that several works use pre-existing security modules to enforce security requirements. The most adopted security enforcement module for this purpose is Apache Rampart, which provides security enforcement following the WS-security stack. These solutions are highly recommended in the cases that security requirements such as confidentiality and authentication need to be enforced. The use of aspect-oriented approaches can also be considered as an interesting possibility to enforce security. These approaches are becoming more relevant when considering the use of non-standard security requirements (e.g., security requirements that are not supported by traditional security standards such as WS-Security) and especially when the goal is to execute new security actions. By using interceptors, AOP makes possible the introduction of new actions, in such way that security experts can implement and use them in the corresponding service composition. Finally, the use of widely accepted and well-known standards is helpful in business processes executed through web service compositions.

4.5 Open challenges

By observing the current state of the art in the enforcement of security requirements, some open challenges can be identified:

- **Support to enforcement of multiple security requirements.** Existing works focus either on a set or a single security requirement (e.g., Menzel et al., 2010; Lins et al., 2016). However, complex systems may require a significant number of security enforcement resources, which are not provided by existing solutions. For example, Salnitri et al. (2015) focuses on more than ten security requirements, but the enforcement is only to a particular environment, the SAP River. Hence, it is important to provide a holistic framework, which can both provide support to a considerable number of security requirements enforcements and can also be used in other environments.

- **Monitor security requirements.** The monitoring of security requirements and service partners may generate valuable information to redesign business processes. Based on
that, one interesting emerging trend is the development of accurate monitoring approaches, which can act in conjunction with orchestration engines. In this case, when a particular security requirement is violated, the execution environment can provide warning messages and possibly perform recommendations to the business expert.

- **Enforcement of security requirements in the cloud and containers.** Considering the significant interest on cloud computing and containers by companies is viable to have versions of business process automation tools taking into consideration such environments. For example, Lins et al. (2016) proposes an environment to automate security-aware business processes in the cloud. However, specific points must be addressed to accomplish this task. For example, to move the solution to the cloud can bring other security issues to the environment, which must be analysed by the business and security analysts. Another relevant point is the compatibility of current translation mechanisms with the security resources available in these environments.

5 Conclusions

The dissemination of BPM standards and service-oriented computing (SOC) is currently a fact. The BPMN, for example, is being widely adopted by companies to model their business processes. The need for executing business processes, in this context, is also relevant. Business experts want to execute the modelled business process due to diverse reasons, e.g., to provide a system that fulfils the business needs or for testing purposes. Meanwhile, service-based systems are being used in the BPM area to support this task as it facilitates the reuse of pre-existing and already tested services, pay per usage business model and so on. Considering that these services may be available at various locations over the internet and security is always a concern in these cases, it is essential to provide security mechanisms to ensure that preventive actions were taken to make this process secure.

The focus of this survey was to present current initiatives on the automation of business processes considering security requirements. This study classified these initiatives into three main groups focusing on modelling, translation and enforcement mechanisms. Some of these works have contributions in one or more groups, while they can be considered viable options to addresses some of the most relevant security-aware business process automation issues.

In each one of the presented groups, a comparative overview was performed to evaluate the research initiatives and the open research challenges were described. The main idea is to provide theoretical and practical support for industrial readers and software designers to model, translate and enforce security requirements for business processes. For example, by the analysis of Table 1, business analysts can choose a modelling strategy that better fits their needs. The number and quality of the research initiatives presented on this survey allow affirming that the automation of security requirements in business processes is a relevant area. However, improvements are still necessary to achieve a broad dissemination, especially related to the standardisation of the security requirements modelling and the translation process.
References


A survey on automation of security requirements in service-based business


