Information Systems 103 (2022) 101818

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/is

Process model abstraction for rapid comprehension of complex business processes

Christina Tsagkani *, Aphrodite Tsalgatidou

Department of Informatics & Telecommunications, National & Kapodistrian University of Athens (NKUA), Panepistimioupolis, Ilisia 157 84, Greece

ARTICLE INFO

Article history:
Received 13 April 2020
Received in revised form 26 May 2021
Accepted 1 June 2021
Available online 19 June 2021
Recommended by Manfred Reichert

Keywords:
Business process modeling
Business process model abstraction
Business process modeling notation
Business process model management

ABSTRACT

Business process management has been widely adopted by many organizations, resulting in the accumulation of large collections of process models. The majority of these models are rather large and complex. Even though such models constitute a great source of knowledge, they cannot be easily understood by all process stakeholders. Process model abstraction techniques have been proven effective in generating easy to comprehend, high-level views on business process models; thus, such techniques change the way that detailed process models may be utilized within an organization. Although much attention has been given to abstract activities of process models, to the best of our knowledge, there are no research works that deliver abstract process model views, by considering as candidates for abstraction not only activities but also other process model elements. In this paper, we present an abstraction approach that simplifies existing process models by focusing not only on the abstraction of activities, but also on the abstraction of data, roles, messages and artifacts. The proposed approach exploits both model structure and element properties, while it is grounded on a set of abstraction rules. A prototype tool has been implemented as a proof of concept; this tool has been used for validating the proposed approach by automatically applying the suggested abstraction rules to different sets of real-world process models. A number of process stakeholders have also been involved in this validation. Thus, it is empirically proved that the presented work is an effective process model abstraction method that increases the usability of complex business process models, as it enables their rapid comprehension by process stakeholders.

1. Introduction

In the last decades, business process management (BPM) has attracted a lot of attention by large organizations that document their operations around business processes [1,2]. In this context, business processes have been proven a great source of knowledge. As a consequence, a lot of efforts have been taking place for building process repositories containing hundreds or even thousands of business process models; see for example the work in [1,3–5]. These process repositories strive to ensure semantic correctness of the processes residing in them, efficient process enactment and execution, exception handling and possibility for business process intelligence [6], i.e. process mining [7,8] and discovery [9]. The exploitation and efficient handling of those process models has become even more vital in today's digitalization era, which poses further challenges and opportunities that need to be addressed [10–12]. Therefore, process stakeholders who are part of an organization's senior leadership team and have an overall view of the organization's process landscape, need to rapidly comprehend complex business processes and exploit all important information available in process model repositories. In this way they are able to take decisions and guide organizational transformation quickly and efficiently. This need requires that process model repositories are accompanied by mechanisms that enable the quick comprehension of existing detailed business process models in order to enable process stakeholders of different levels of expertise to take full advantage of them [13]. For example, process stakeholders who are involved in an organization's daily activities can use the detailed business process models. Also, process stakeholders at higher organization levels (e.g. business directors) can use high level process views, in order to quickly comprehend and exploit them for purposes related to their assignments and responsibilities, such as decision making or streamlining business processes.

Business process model abstraction (BPMA) is a technique that can deliver simplified process model views of detailed process models that contain the whole set of information about the process. This technique enables different kind of process stakeholders to access different aspects of processes with an adapted visualization [14]. Therefore, BPMA has been proven that facilitates the comprehension of large business process models [15,16]

* Corresponding author.
E-mail addresses: tsagkani@di.uoa.gr (C. Tsagkani), atsalga@di.uoa.gr (A. Tsalgatidou).

https://doi.org/10.1016/j.is.2021.101818
0306-4379/

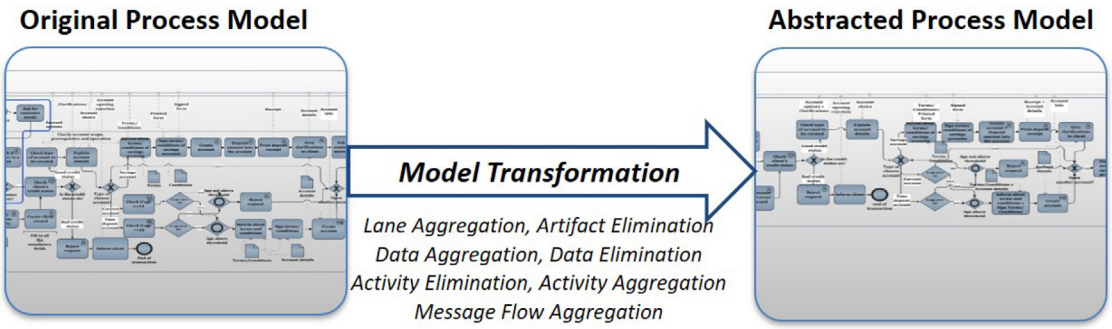


Fig. 1. Process model abstraction approach.

and can serve diverse purposes, such as adapting process models for external partner use or obtaining a process quick view [16]. Specifically, BPMA can transform a large detailed business process model to a coarse-grained one by reducing process elements which are considered insignificant based on the intended use of the process model [16]. This is accomplished by suggesting relative abstraction operations based on the importance of each component [17]. A variety of approaches have been proposed for delivering abstract views of an original process model, and are presented in detail in Section 5. Most of those approaches, see for example the work in [14,18–20], exploit the process model structure in order to deliver abstract process views by mainly performing abstraction on business process model activities. However, as it is also recognized in the work by Smirnov et al. [21], experienced process stakeholders take into account a wider range of properties than just the structure of a process model.

Driven by the abovementioned motivation and situation, the research work presented in this paper, focuses on delivering a process abstraction approach which offers simplified views of business processes. Such simplified process views facilitate quick process comprehension which is essential for certain process stakeholders such as business directors, who are involved in the decision-making process. One of the main contributions of this approach is that it considers as candidates for abstraction not only activities but also other process model elements, namely, data, roles, messages and artifacts. In this way, the suggested approach offers a holistic perspective to process model abstraction, in the sense that it abstracts diverse process model constructs rather than just activities. We would like to note that the suggested approach targets process models defined using the Business Process Model and Notation (BPMN) [22]; the reason for this is that BPMN (i) is the de facto standard for business process modeling, (ii) is characterized by high expressiveness compared to other modeling notations as it can represent graphically any business process [2,23] and, (iii) has great acceptance by industry [24].

Another essential contribution of the suggested approach is that it is based on rules which take into account both the process model structure and the properties of process elements; based on those rules important process model elements are retained, while insignificant process elements are being eliminated and/or aggregated. It is worth mentioning that the suggested approach maintains the overall process model structure by preserving participants' interactions, as they are considered essential for the abstract process and its comprehension [23]. The proposed abstraction rules have been derived by firstly examining BPMN process models. They were subsequently refined, after a few trial applications on a number of process models. We have introduced an initial declarative presentation of these rules in [25], whereas a conceptual description of our abstraction approach that incorporates these rules was introduced in [26]. In this paper, we provide a formal specification of the proposed rules. Also, we introduce a process model abstraction strategy which takes as input an original model (which is usually a fine-grained business process model) and automatically applies on it the proposed set of abstraction rules, in a certain order, so as to produce a simplified process model. A prototype tool has been implemented in order to evaluate the model abstraction approach; the validation considers real-world process models and involves process stakeholders, as it is described in detail in Section 4. An abstract overview of the overall process abstraction approach is depicted in Fig. 1.

In the following section (Section 2), we provide definitions for the concepts that are the foundation for the proposed abstraction rules. The proposed abstraction rules and the way they are applied to a process model in order to perform process model abstraction are analytically presented in Section 3. The prototype implementation of the proposed approach is presented in Section 4, along with the validation performed and the accumulated results. In Section 5, we present related research work and we highlight the contribution of the proposed approach. Finally, Section 6 concludes this paper and highlights future research work.

2. Concepts and definitions

This section introduces the fundamental concepts that are required for the presentation of our approach to process model abstraction. We begin by formally defining a process model in Section 2.1. The properties of process model elements are described in Section 2.2 while the insignificant process model elements are formally defined in Section 2.3. Finally, the basic constructs of a process model fragment are introduced in Section 2.4.

2.1. Process model definition

The proposed approach to process model abstraction applies abstraction operations to multiple process model elements, specifically to activities, data, roles, messages and artifacts. For the purposes of our work, we have extended the definition of process models provided in [27], which is originated from the process model formalism presented in [2], as follows:

Definition 1 (*Process Model*)**.** A process model is a tuple PM = (N, D, ART, P, L, CO, ω, ρ) where

- N = A ∪ G ∪ E is a set of flow objects (which are the actions that may take place inside a business process determining

[1] In all the process models presented in this paper, the manual tasks are depicted with a rounded corner rectangle that does not include any figure marker.

its behavior [23]) that consists of the following mutually disjoint sets: A is a nonempty set of activities; G is a nonempty set of gateways; E is a nonempty set of events; A, E and G are pairwise disjoint;
- D is a nonempty set of data objects;
- ART is a finite set of Artifacts;
- P is a finite set of Pools;
- L is a finite set of Lanes;
- $\omega: L \rightarrow (N \cup CO \cup D \cup ART)$ is a function that maps elements N, CO, D, ART to a lane. This means that a lane consists of the mutually disjoint sets of flow objects, connecting objects, data objects and artifacts;
- $\rho: L \rightarrow P$ is a function that maps a lane to a participant;
- CO = DA ∪ SF ∪ Ass ∪ MF is a finite set of connecting objects that consists of the following mutually disjoint sets:
  - DA is a nonempty set of data associations (which are connections between data objects and activities [23]), that DA ⊆ (A × D) ∪ (D × A);
  - SF is a nonempty set of sequence flows (which are connections between various flow objects [23]), SF ⊆ ((A ∪ G ∪ E) × (A ∪ G ∪ E));
  - Ass is a nonempty set of associations between process elements and artifacts (e.g. text annotations), Ass ⊆ N × ART;
  - MF is a nonempty set of message flows that represent the flow of messages amongst different process participants, or process elements that belong to different participants, or even process elements and other process participants, such that

$$MF \subseteq \begin{cases} N_i \times N_j | N_i \in P_i \wedge N_j \in P_j, & \text{flow between elements of different participants} \\ P_i \times P_j, & \text{flow between different participants} \\ N_i \times P_j, & \text{flow between element and another participant} \end{cases}$$

2.2. Process model element properties

In order to exemplify the notions used to describe process model element properties, we are using a process model that defines the process of opening a new banking account (Fig. 2). In brief, this process depicts that a customer visits "ABC bank" where the first person that gets in contact with is the employee at the service desk. This employee is responsible to interact with the customer, identify his/her needs and guide him/her to the appropriate employee in the customer advisory department. The current customer needs to open a new bank account. Therefore, the employee in the customer advisory will follow certain tasks depending on whether the current customer is already a client of the bank or not. Especially, in case that the customer is a new client, the employee verifies the client's identity and creates a new client record, based on the accompanying documents of the customer. Then, the client's credit status is checked for both new and existing clients. If the credit status is acceptable, the employee informs the client about account opening options and technical/procedural details related to the type of the account that the client desires to open. Otherwise, the request to open an account is rejected and the client is informed that s/he is not eligible for account opening. Depending on the client's decision regarding the type of account s/he desires to open, the employee ensures that the prerequisites are satisfied. For example, in the case of current or deposit account, the age of the client is checked and the client's request is rejected in case that the age limit is not satisfied; otherwise, the client is informed about the terms and conditions, the required documents are signed and the account opening takes place along with the first deposit of the required amount (applies only to the savings account). It should be noted that when the request is rejected, the flow of the process may go back; in this case, the employee explains once again the account options to the client. The process ends when the client does not require further assistance and the related documentation/receipt to the account opening/deposit amount, into the newly created account is handed to him/her.

The approach that we have designed for process model abstraction relies on the properties of process model elements. Based on the notation presented in [22] we suggest that each construct of a process model is associated with a specific type, which can be: Activity, Gateway, Event, Data, Artifact, Pool, Lane and Connecting Objects. In the following definition, we formalize the process model element types.

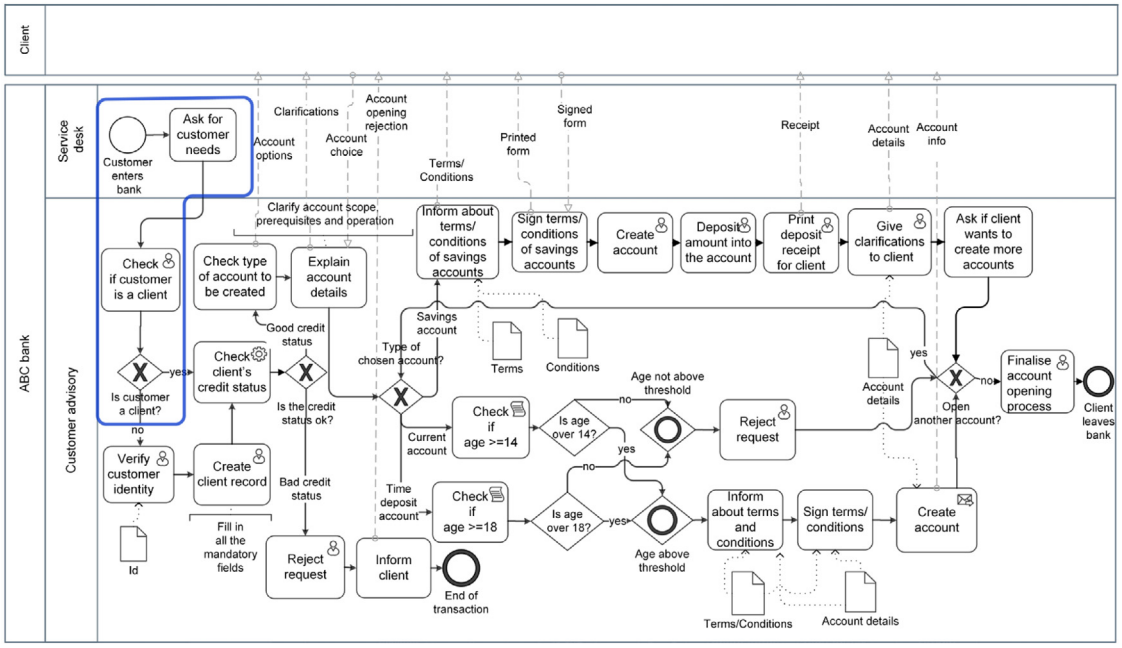


Fig. 2. Account opening process - Original process model.[1]

Table 1
Property values of the connecting objects of Fig. 2.

Message Flows (MF)	
• γ(Clarifications) • γ(Account choice) • γ(Account opening rejection) • γ(Terms/Conditions) • γ(Printed form)	• γ(Signed form) • γ(Receipt) • γ(Account details) • γ(Account info) • γ(Account options)

Data Associations (DA)
• γ(Id, Verify customer identity) • γ(Terms/Conditions, Inform about terms and conditions) • γ(Terms/Conditions, Sign terms/conditions) • γ(Account details, Sign terms/conditions) • γ(Account details, Inform about terms and conditions) • γ(Account details, Create account) • γ(Account details, Give clarifications to client) • γ(Terms, Inform about terms and conditions of savings accounts) • γ(Conditions, Inform about terms and conditions of savings accounts)

Associations (Ass)
• γ(Create client record, Fill in all the mandatory fields) • γ(Explain account details, Clarify account scope, prerequisites and operation)

Sequence Flows (SF)	
• γ(Ask for customer needs, Check if customer is a client) • γ(Check if customer is a client, Is customer a client?) • γ(Is customer a client?, Verify customer identity) • γ(Is customer a client?, Check client's credit status) • γ(Check client's credit status, Is the credit status ok?) • γ(Verify customer identity, Create client record) • γ(Create client record, Check client's credit status) • γ(Is the credit status ok?, Reject request) • γ(Reject request, Inform client) • γ(Inform client, End of transaction) • γ(Is the credit status ok?, Check type of account to be created) • γ(Check type of account to be created, Explain account details) • γ(Explain account details, Type of chosen account?) • γ(Type of chosen account?, Check if age>=18) • γ(Type of chosen account?, Check if age>=14) • γ(Type of chosen account?, Inform about terms/conditions of savings accounts) • γ(Inform about terms/conditions of savings accounts, Sign terms/conditions of savings accounts) • γ(Sign terms/conditions of savings accounts, create account)	• γ(Create account, Deposit amount into the account) • γ(Deposit amount into the account, Print deposit receipt for client) • γ(Print deposit receipt for client, Give clarifications to client) • γ(Give clarifications to client, Ask if client wants to create more accounts) • γ(Ask if client wants to create more accounts, Open another account?) • γ(Open another account?, Finalise account opening process) • γ(Finalise account opening process, Client leaves bank) • γ(Open another account?, Type of chosen account?) • γ(Check if age>=18, Is age over 18?) • γ(Is age over 18?, Age above threshold) • γ(Is age over 18?, Age not above threshold) • γ(Check if age>=14, Is age over 14?) • γ(Is age over 14?, Age above threshold) • γ(Is age over 14?, Age not above threshold) • γ(Age not above threshold, Reject request) • γ(Reject request, Open another account?) • γ(Age above threshold, Inform about terms and conditions) • γ(Inform about terms and conditions, Sign terms/conditions) • γ(Sign terms/conditions, Create account) • γ(Create account, Open another account?)

Definition 2 (*Process Element Type*)**.** Let PE be a finite nonempty set of process elements. A process element (PE) may be any element of the sets that belong to the process model (Definition 1). Alongside, Type is a finite nonempty set of process element types, Type = {Activity, Event, Gateway, Data, Artifact, Pool, Lane, Connecting Objects}. The mapping $\boldsymbol{\zeta}$ assigns a value of the set Type to each PE such that $\boldsymbol{\zeta}$: $PE \rightarrow Type$.

The original process model in Fig. 2 is used to illustrate Definition 2. For example, $\boldsymbol{\zeta}$ (Create client record) = Activity and $\boldsymbol{\zeta}$ (Account details) = Data. The process element types are essential since the proposed abstraction algorithm inspects these types, as it is described in Section 3.3.1.

Besides, each process element type is accompanied with a property value that may guide the abstraction operation applied (for example elimination operation). We formalize the concept of property value for the process element types A (Activity), D (Data) and CO (Connecting Objects) of a process model, in Definitions 3–5 which follow next.

Definition 3 (*Activity Property Value*)**.** Let A be a finite nonempty set of activities. Alongside, A_{value} is a finite nonempty set of activity property values A_{value} = {Manual, Service, Send, Receive, User, Business Rule, Script}. The mapping $\boldsymbol{\alpha}$ assigns a value to each element of A such that $\boldsymbol{\alpha}$: $A \rightarrow A_{value}$.

For example, in Fig. 2 when the $\boldsymbol{\alpha}$ function is applied to the activity "Inform about terms/conditions of savings accounts", this activity is mapped to the "Manual" activity value, that is $\boldsymbol{\alpha}$ (Inform about terms/conditions of savings accounts) = Manual.

Definition 4 (*Data Property Value*)**.** Let D be a finite nonempty set of data. Alongside, D_{value} is a finite nonempty set of activity property values D_{value} = {data objects, data inputs, data outputs, data stores}. The mapping $\boldsymbol{\delta}$ assigns a value to each element of D such that $\boldsymbol{\delta}$: $D \rightarrow D_{value}$.

The process model in Fig. 2 illustrates Definition 4. For example, when the $\boldsymbol{\delta}$ function is applied to "Account details" this data is mapped to the "data object" data value that is, $\boldsymbol{\delta}$ (Account details) = data object.

Definition 5 (*Connecting Object Property Value*)**.** Let CO be a finite nonempty set of connecting objects. Alongside, CO_{value} is a finite nonempty set of connecting object property values CO_{value} = {DA, SF, Ass, MF} as explained in Definition 1. The mapping $\boldsymbol{\gamma}$ assigns a value to each element of CO such that $\boldsymbol{\gamma}$: $CO \rightarrow CO_{value}$.

The process model in Fig. 2 illustrates Definition 5. For example, when the $\boldsymbol{\gamma}$ function is applied to the connecting object named "Account options" this connecting object is mapped to the "MF" connecting object value, that is $\boldsymbol{\gamma}$(Account options) = MF.

Similarly, all the connecting objects of Fig. 2 and the connecting object property values that may be mapped to them are depicted next in Table 1, grouped by value.

Generally, the connecting objects, beyond their associated property value, may be connected with either directed or non-directed edges between process elements. In the case of directed edges, the first element of the given set is the source process element, whereas the second element of the set is the target process element. Therefore, given that CO is a finite nonempty set of directed connecting objects, we define two functions, namely $\boldsymbol{\Lambda}$ and $\boldsymbol{\Pi}$, that map respectively each connecting object to its source element and its target element, i.e. $\boldsymbol{\Lambda}$: $CO \rightarrow PE$ and $\boldsymbol{\Pi}$: $CO \rightarrow PE$. For example, in Fig. 2, there is a directed connecting object, specifically, a sequence flow (sf), between the activities "Verify customer identity" and "Create client record" that is sf = {Verify customer identity, Create client record}. By applying the $\boldsymbol{\Lambda}$, $\boldsymbol{\Pi}$ functions to this sequence flow (sf), we are directed to its source and target elements which are, $\boldsymbol{\Lambda}$(sf) = Verify customer identity and $\boldsymbol{\Pi}$(sf) = Create client record.

Moreover, in a data association, we define a function named $\boldsymbol{\kappa}$ in order to identify the activity of a data association that is $\boldsymbol{\kappa}$: $DA \rightarrow A$. For example, given that da = {ID, Verify customer identity}, then $\boldsymbol{\kappa}$ (da) = Verify customer Identity. We would like to note that this function is of great importance. This is because it is used by the defined abstraction rules in order to identify data objects associated to the same activity. These objects can then be aggregated in the abstract process model view (see Section 3.2).

2.3. Insignificant process elements

As we have aforementioned, the proposed abstraction mechanism is mainly based on the assumption that not all model elements within a process model are equally important. Therefore,

the proposed approach aims at applying abstraction operations only to insignificant process elements.

In order to identify the insignificant process elements, we are driven by the requirements related to the intended use of our proposed approach which is to enable process stakeholders' (i.e. directors) rapid comprehension of complex business processes. We also have taken into account the description of each process model element [22]. Therefore, we have made the assumption that the activities, which are not involved in message exchanges between different participants (i.e. the activities that are not part of the process collaboration) and are referring to internal processing of each participant, are not significant; thus, they are candidate for abstraction. Similarly, as long as data is concerned, we assume that data in a process model is insignificant when it is neither an external data input for the entire process nor data output that has resulted from the execution of the entire process. In Definition 6 below, we formalize the insignificant activity elements, while in Definition 7 we formalize the definition of insignificant data elements.

Definition 6 (*Insignificant Activity Elements*)**.** Let A be a finite set of activities and MF a finite set of message flows. Then $\forall\ a_i \in$ A a boolean value is assigned to insign(a_i) as follows:

$$\text{insign}(a_i) = \begin{cases} \text{false}, \exists\ \text{mf} \in \text{MF such that } \Lambda\,(mf)\ OR\ \Pi\,(mf) = a_i \\ \\ \text{true}, \nexists mf\ \in\ MF\ such\ that\ \Lambda(mf)\ OR\ \Pi(mf) = a_i \end{cases}$$

The process model in Fig. 2 is used to illustrate Definition 6. For example, for the activity "Explain account details" there are two message flows ("Account choice" and "Clarifications") whose functions $\boldsymbol{\Pi}$ and $\boldsymbol{\Lambda}$ respectively return the specific activity as a result. Therefore, the insignificance of this activity gets a false value. Similarly, the activities "Check type of account to be created", "Inform client", "Inform about terms/conditions of savings accounts", "Sign terms/conditions of savings accounts", "Print deposit receipt for client", "Give clarifications to client" and "Create account" are associated with message exchanging and are thus considered as significant. The rest of the activities in the process model depicted in Fig. 2 are considered as insignificant since they are not involved in message exchanging.

Definition 7 (*Insignificant Data Elements*)**.** Let D be a finite nonempty set of data elements. Then, $\forall$ $d_i \in$ D a boolean value is assigned to insign (d_i) as follows:

$$\text{insign}(d_i) = \begin{cases} false, \quad \delta\,(d_i) = data\ Input\ OR\ \delta\,(d_i) = data\ Output, \\ \\ true, \quad \delta\,(d_i) = data\ object\ OR\ \delta\,(d_i) = data\ store \end{cases}$$

The process model in Fig. 2 is used to illustrate Definition 7. For example, for the data "Id" taking into consideration that $\boldsymbol{\delta}\left(''\text{Id}''\right)$ = data object then insign("Id") = true. Therefore, "Id" is an insignificant data object that is candidate for abstraction (as it is described later in Section 3.2). Similarly, the data objects "Terms", "Conditions", "Account details" and "Terms/Conditions" are considered as insignificant, too.

Regarding the rest of the process model elements, we make the following assumptions:

- Lanes are considered as non-significant elements, as they are related to the internal organizational structure and describe how work is assigned to different internal roles of a particular process participant [22]. This information is not considered as significant for the purposes of our abstraction approach, which is to offer quick process views for facilitating rapid comprehension, while respecting process participants' interactions.
- Artifacts (text annotations) are also considered as non-significant as they provide additional information without affecting the process flow.
- Pools are considered significant elements as they represent different process participants; this information is very important for comprehending how individual process participants interact and contribute to the whole process.
- Message flows are basically considered as significant elements as they are involved in the interactions between process participants. Nevertheless, they can be abstracted to a certain extent, through the message aggregation process, as it is described later in Section 3.2.

2.4. Process fragments and process fragments properties

The proposed approach groups process elements into smaller structures (called process fragments based on the definition provided in [28]) in order to identify related activity behavior. Therefore, our approach recognizes areas in the process model where the enclosed process elements are semantically relevant. For each of these areas (or process fragments), the proposed model abstraction strategy (outlined in Section 3.3) suggests coarse-grained activities. The suggested activities are semantically equivalent to the inner part of the fragment, so they can replace the latter. Therefore, for the purposes of our research, we have defined that a process fragment (PF) comprises a sequence of activities (PF_{inner}) that constitute the inner elements of the process fragment. Additionally, a process fragment has an entry process element and an exit process element that constitute the outer elements (PF_{outer}) of the process fragment. We formalize process fragment properties next, in Definition 8.

Definition 8 (*Process Fragment Properties*)**.** A process fragment with self-contained semantics satisfies the following:

- It consists of a fragment start node and a fragment end node which are part of the set PF_{outer} such that $PF_{outer} \subseteq (G \cup E)$. For example, a fragment may start with a split and end with a join connector, not necessarily of the same type (e.g. it may start with a parallel gateway and finish with a complex gateway).
- It consists of fragment inner nodes that are part of PF_{inner} such that $PF_{inner} \subseteq$ A.
- Let N_{PFtype} define the types of the elements of a process fragment such that $N_{PFtype} = \{PF_{outer}, PF_{inner}\}$; then function $\boldsymbol{\theta}$ maps each flow object of the set N_{PF} to a N_{PFtype}, that is $\boldsymbol{\theta}$: $N_{pf} \rightarrow N_{PFtype}$.
- $|PF_{inner}| \geq 1$, i.e. the multitude of inner elements of the process fragment should be at least one.
- $|\boldsymbol{\rho}_{pf}| = 1$, i.e. the process fragment is executed by the same process participant.

An example of a process fragment is shown at the top left part of Fig. 2, enclosed in a box. More precisely, the first process fragment has as boundary elements the start event "Customer enters bank" and the exclusive gateway "Is customer a client?" as an exit element; it also has two inner elements, the activities "Ask for customer needs" and "Check if customer is a client". Therefore pf = (PF_{outer} = {Customer enters bank, Is customer a client?}, PF_{inner} = {Ask for customer needs, Check if customer is a client}).

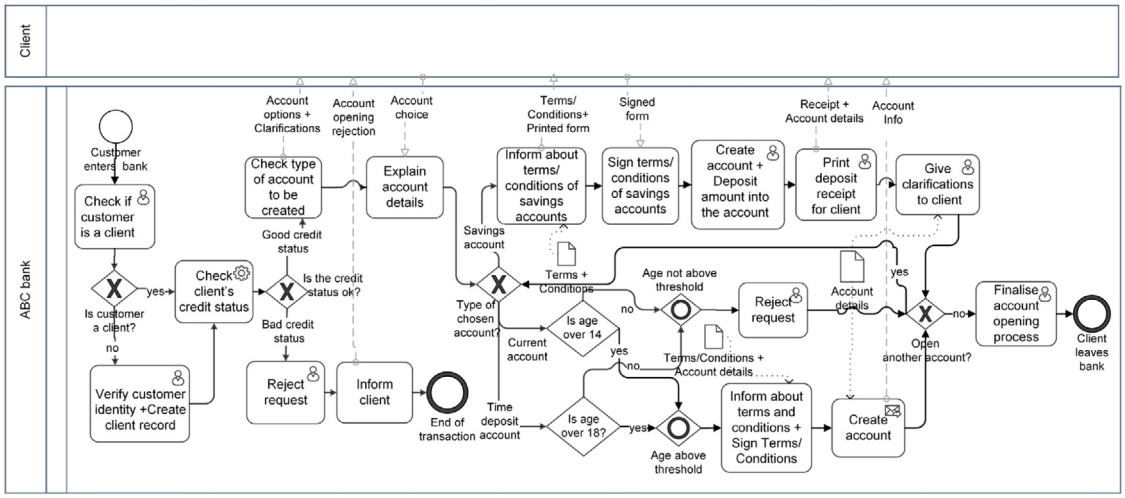


Fig. 3. Account opening process - Automatically abstracted process model.

3. Process model abstraction approach

The proposed business process abstraction approach focuses on process model abstraction for obtaining process quick views, while respecting interaction amongst process participants. As opposed to other proposed approaches, for example the ones in [14] and in [29], which consider only activities as abstraction objects, the proposed approach, apart from activities, also considers message flows, roles, artifacts and data, as abstraction objects. It both supports aggregation and elimination as primitive process abstraction operations, which are described in the following section (Section 3.1). The application of these two abstraction operations is based on a predefined set of abstraction rules, which is described in Section 3.2 and on an abstraction strategy which is described in Section 3.3.

3.1. Primitive process abstraction operations

As we mentioned above, the proposed approach supports two primitive abstraction operations, namely elimination and aggregation which are formally defined in the following.

The elimination operation targets process elements that do not provide valuable information to the process model, meaning that they do not enable quick business process comprehension by interested process stakeholders. Such process elements can be for example, data objects that do not provide input or output to the process model and are associated only to one process model element. In Definition 9 we formalize the model element elimination concept.

Definition 9 (*Elimination Operation*)**.** Let PM be a process model and PM_{pe} be a subset of PM consisting of all elements that need to be eliminated; then, $\boldsymbol{\mu}$ is function that removes elements such as $\boldsymbol{\mu}$: $PM_{pe} \rightarrow \emptyset$. In other words, $\boldsymbol{\mu}$ is a function that maps a set PM_{pe} (that consists of certain insignificant process elements) to an empty set, as its elements are removed from the process model.

For example, the process model depicted in Fig. 3, which is the abstract version of the model that is depicted in Fig. 2, does not include certain process elements of the original process model (Fig. 2), such as the activity "Ask for customer needs", and the data object "Id". This is due to the elimination operation that has been applied. Moreover, other process model elements that have been eliminated are "Ask if client wants to create more accounts", "Check if age $>=$ 14" and "Check if age $>=$ 18".

The other supported abstraction operation is aggregation. Specifically, process elements that are defined as insignificant but still need to be present in some way in the abstract model, are aggregated rather than eliminated. In this way, the information that relates to the abstracted elements is retained in the abstract model but in a simplified way. For example, in case that two activities of a process fragment are connected unconditionally, they are aggregated in the abstract model into one activity. The properties of the resulting activity (e.g. label) consist of the properties of the two activities (before aggregation). In Definition 10 we formalize the model element aggregation concept.

Definition 10 (*Aggregation Operation*)**.** Let PM and PM′ be the original and the abstract process models respectively and PE, PE′ sets of elements; for every process element ***pe*** of a set of elements PE that exist in PM there is an element ***pe***′ in PE′ of PM′ that is mapped to. Function $\boldsymbol{\beta}$: $PE \rightarrow PE'$ indicates the mapping between process elements before and after abstraction. The proposed approach applies the $\boldsymbol{\beta}$ function to map more than one process elements to either:

- a single activity, $a \in A_{PM'}$ or,
- a single data object, $d \in D_{PM'}$ or,
- a single message flow, $mf \in MF_{PM'}$ or,
- a single pool, $p \in P_{PM'}$.

For example, in the abstract process model depicted in Fig. 3, the activities "Verify customer identity" and "Create client record" have been aggregated. The aggregated activity is "Verify customer identity + Create client record". Moreover, other process model elements that have been aggregated are: "Create account" and "Deposit amount into account", "Inform about terms and conditions" and "Sign terms/conditions", "Terms" and "Conditions", "Customer advisory" and "Service desk" and "Terms/Conditions" and "Account details".

As we have already mentioned, the above described abstraction operations are applied based on a predefined set of abstraction rules which is described next in Section 3.2 and on an abstraction strategy, described in Section 3.3.

3.2. Process model abstraction rules

The proposed set of abstraction rules which is described here has been defined based on our experience while working with various real-world business processes and after thoroughly analyzing the elements of the BPMN 2.0 notation. Also, in order

Table 2
Abstraction rules per process model abstraction element.

Abstraction rule	Process model abstraction element	Rule definition
Rule 1	Lane (L)	Lane aggregation rule (Definition 11)
Rule 2	Message Flow (MF)	Message flow aggregation rule (Definition 12)
Rule 3	Data (D)	Data elimination rule (Definition 13)
Rule 4	Data (D)	Data aggregation rule (Definition 14)
Rule 5	Artifact (Art)	Artifact elimination rule (Definition 15)
Rule 6	Activity (A)	Activity elimination rule (Definition 16)
Rule 7	Activity (A)	Activity aggregation rule (Definition 17)

to formally define the rule set, we have examined and analyzed different business processes while considering specific use cases coming from the industry practice. As we have already mentioned in the introduction, the initial set of rules was refined after being applied in a few business models.

The abstraction rules that are described next, identify the conditions that trigger each abstraction operation and are used to abstract existing business process models. These rules are applied either to the whole process model or to process fragments that have self-contained business semantics and eliminate/aggregate process elements that are not of high importance. Thus, these rules aim at reducing the number of process elements, where applicable.

Table 2 summarizes the abstraction rules suggested per abstraction object, along with the names of their associated definitions. Next, the seven (7) suggested abstraction rules are formally defined per abstraction element (i.e. per lane, message flow, data, artifact and activity), while they are elaborated and justified by providing explanatory examples.

Abstraction Rule 1 (*Lane Abstraction by Aggregation*)**.** Lanes are used to distinguish the work performed by the specific roles of a process participant, the related data that are produced or consumed as well as related events and decisions. Lanes are important for the internal organization of a specific process participant; nevertheless, they are not mandatory as long as process choreographies are concerned, where activities are interactions that represent a set (one or more) of message exchanges, which involves two or more process participants. Process stakeholders, such as directors, who are at the higher levels of the organizational hierarchy need to rapidly comprehend complex business processes; hence they need to quickly view the work of each business process participant rather than how the work is further assigned to each specific role of every process participant. Thus, lanes of a pool (i.e. lanes related to a process participant) are considered as candidates for abstraction and may be aggregated. We suggest that this aggregation operation takes place in a way that all information enclosed within lanes is presented in a unified manner under one process participant without any further specialization to role responsibilities (lanes). Rule 11 formalizes this rule.

Definition 11 (*Lane Aggregation Rule*)**.** Let li, lj ∈ L be two different lanes of a nonempty set L, P be a set of pools (process participants) and $\mathbf{p_i}$ be a process participant that belongs to a set P. In the case that function $\boldsymbol{\rho}$ (Definition 1) is applied to lanes such that $\boldsymbol{\rho}(l_i) = \boldsymbol{\rho}(l_j) = p_i$, this indicates that they belong to the same participant. Then, the aggregation function $\boldsymbol{\beta}$ (see Definition 10) is applied to li and lj and their enclosed flow objects, artifacts and data objects (as identified by applying function $\boldsymbol{\omega}$ - Definition 1). The aggregated lanes are presented under the same participant (pool) that is $\boldsymbol{\beta}$ (li, lj) = p_i.

This rule is exemplified using the process models depicted in Figs. 2 and 3. For example, in the original process model (Fig. 2) for the lanes "Customer advisory" and "Service desk", the following apply: $\boldsymbol{\rho}$ (Customer advisory) = ABC bank and $\boldsymbol{\rho}$ (Service desk) = ABC bank. This means that both roles belong to the same participant and thus they are aggregated. In the abstract model (Fig. 3), only the pool name (i.e., ABC bank) is presented enclosing all the process elements, originally coming from the aforementioned lanes.

Abstraction Rule 2 (*Message Flow Abstraction by Aggregation*)**.** Messages are used for the interaction between different process participants. Therefore, they are important for the external communication of a process participant, and thus, they are considered significant process elements; consequently, message elimination is not allowed. However, in some cases, message abstraction can be performed through aggregation, as it is defined in the following definition.

Definition 12 (*Message Flow Aggregation Rule*)**.** Let mf_i, $mf_j \in$ MF be two different message flows of a nonempty set of MF. Also let $\boldsymbol{\Lambda}$ be a function that returns the source element/pool of the message flow and $\boldsymbol{\Pi}$ is a function that returns the target element/pool of the message flow (Section 2.2). The message flow aggregation function $\boldsymbol{\beta}$ (Definition 10) is taking place if at least one of the following is true:

- ($\boldsymbol{\Lambda}(mf_i) = \boldsymbol{\Lambda}(mf_j)) \wedge (\boldsymbol{\Pi}(mf_i) = \boldsymbol{\Pi}(mf_j))$ that is the source and the target elements are the same for both message flows.
- ($\boldsymbol{\Lambda}(mf_i)\bullet = \boldsymbol{\Lambda}(mf_j)) \wedge (\boldsymbol{\Pi}(mf_i)\bullet = \boldsymbol{\Pi}(mf_j))$ that indicates that the consequential element of the source/target element of mf_i is respectively the source element/target element of mf_j. Also, in case message exchange is associated to activities, both elements should have the same property value (Definition 3) A_{value} (e.g. A_{value} = User). For example, in Fig. 2 the messages that satisfy the current condition are "Account options" and "Clarifications". More precisely, the source and target elements of "Account options" are respectively "Check type of account to be created" and "Client", whose succeedding elements are "Explain account details" and "Client". The latter elements constitute the source and target elements of the message "Clarifications". Also, in the specific example, the source activities involved, i.e. "Check type of account to be created" and "Explain account details", acquire the same property value A_{value} which is "Manual".
- ($\bullet\,\boldsymbol{\Lambda}(mf_i) = \boldsymbol{\Lambda}(mf_j) \wedge \bullet\boldsymbol{\Pi}(mf_i) = \boldsymbol{\Pi}(mf_j)$) that indicates that (a) the preceding element of the source element of mf_i is the source element of mf_j and (b) the preceding element of the target element of mf_i is the target element of mf_j. Also, in case that message exchange is associated to activities, both elements should have the same property value (Definition 3) A_{value} (e.g. A_{value} = User). For example, in Fig. 2 the messages that satisfy the current condition are "Receipt" and "Account details". More precisely, the source and target elements of "Account details" are respectively "Give clarifications to client" and "Client", whose preceding elements are "Print deposit receipt for client" and "Client". The latter elements constitute the source and target elements of the message "Receipt". Also, in the specific example, the source activities involved, i.e. "Give clarifications to client" and "Print deposit receipt for client", acquire the same property value A_{value} which is "User".

This rule is exemplified using the process models depicted in Figs. 2 and 3. For example, in Fig. 2, for the messages "Receipt" and "Account details", the following apply: Λ (Receipt) = "Print deposit receipt for client" and Λ (Account details) = "Give clarifications to client", while the process flow object "Print deposit receipt for client" is preceding the process flow object "Give clarifications to client". Also, both messages have as their target element the process "Client". Thus, the message flows "Receipt" and "Account details" can be aggregated; the resulting message flow "Receipt+Account details" can be viewed in the abstracted model of Fig. 3.

Abstraction Rule 3 (*Data Abstraction by Elimination*)**.** In a business process model, data is used to model information required for, or produced by, the execution of process activity and the overall execution of a process. In addition, data may impose constraints on process execution. However, since data are not used in process choreographies [22], we consider that data objects and data stores are insignificant for the overall execution of the process, and thus they are candidates for abstraction (see Definition 7). In this rule we suggest to closely relate the importance of a data object/data store, to its frequency of use (within the process model) that is related to its associations. For example, a data object that is only dependent on one activity in the whole process model has low significance to the overall process model. Therefore, we can safely assume that a data object/data store is candidate for abstraction by elimination if it is connected to only one basic flow object. Definition 13 formalizes Abstraction Rule 3.

Definition 13 (*Data Elimination Rule*)**.** Let $d_i \in$ D be a data object/data store (that means that insign (d_i) = true, Definition 7) and DA′ be a nonempty set of the data associations of the data object/data store d_i. The data elimination function μ applies when the cardinality of its data associations is one, $|DA'| = 1$.

See for example that the data object "Id", in Fig. 2, has only one data association to the activity (flow object) "Verify customer identity"; therefore, we can apply to it the function μ (Definition 9) and, as it can be noticed in the abstracted model of Fig. 3, the specific data object has been eliminated.

Abstraction Rule 4 (*Data Abstraction by Aggregation*)**.** As it is discussed in the previous abstraction rule, data objects and data stores are candidates for abstraction. In this rule, we suggest that data objects and data stores of the same property value D_{value} (Definition 4) need to be retained in the abstracted process model. This concerns data objects or data stores which are used or required by the same or multiple process elements that are directly associated and have the same type A_{value} (Definition 3). However, we suggest to provide a condensed view of such objects. Definition 14 formalizes this rule.

Definition 14 (*Data Aggregation Rule*)**.** Let $d_i, d_j \in$ D be two different insignificant data objects and $da_i, da_j \in$ DA be their data associations to process model activities such that $da_i = \{a_i, d_i\}$ and $da_j = \{a_j, d_j\}$. Also, given that (i) δ is the function that assigns a value (e.g. data input) to each element of D (Definition 4); (ii) α is the function that assigns a value to activity property (Definition 3); and, (iii) κ is the function that returns the activity of a data association (Section 2.2); the data aggregation function β (Definition 10) is applied to d_i and d_j as soon as one of the following conditions takes place:

- $\delta(d_i) = \delta(d_j) \wedge |da_i| = |da_j| \wedge \kappa(da_i) = \kappa(da_j) \wedge \big(\Lambda(da_i) = \Lambda(da_j)\big) \vee \big(\Pi(da_i) = \Pi(da_j)\big)$, meaning that both d_i, d_j (i) are associated to the same activity; (ii) are either both needed or produced by the activity; (iii) do not have any other associations to other flow objects; and, (iv) have the same property value.
- $\delta(d_i) = \delta(d_j) \wedge |da_i| = |da_j| \wedge (\kappa(da_i)\bullet = a_j \vee \bullet\kappa(da_i) = a_j) \wedge \alpha(a_i) = \alpha(a_j) \wedge \big(\big(\Lambda(da_i)\bullet = \Lambda(da_j)\big) \vee (\Pi(da_i)\bullet = \Pi(da_j))\big)$, meaning that both d_i, d_j (i) have the same property value; (ii) do not have any other associations to other flow objects; and, (iii) are both connected and are either needed or produced by a sequence of activities which have the same property value.

See for example, that the data objects "Terms" and "Conditions" in Fig. 2 have the same data property value and are input to the same flow object that is "Inform about terms/conditions of savings accounts"; therefore, they are aggregated, as it can be seen in the abstracted model of Fig. 3.

Abstraction Rule 5 (*Artifact Abstraction by Elimination*)**.** Artifacts in a process model retain additional knowledge regarding the process elements but they do not influence the process model structure [22]. As process stakeholders (i.e. directors) need to rapidly comprehend complex business processes, they need to quickly view a business process; therefore, additional information provided by artifacts quite often impairs the rapid process comprehension. Therefore, an abstract process model can omit this information. Consequently, we suggest that artifacts are eliminated along with their associations (Ass) to the process elements. The following definition formalizes this rule.

Definition 15 (*Artifact Elimination Rule*)**.** Let ART be a non-empty set of all the artifacts that are associated to flow objects (N) of a process model (PM) - see Definition 1. Then, during model abstraction, function μ (Definition 9) maps this set to an empty set, i.e. $\mu: PM_{ART} \rightarrow \varnothing$.

In the original process model in Fig. 2, there are two text annotations associated to process flow objects: "Clarify account scope, prerequisites and operation" and "Fill in all the mandatory fields". Both artifacts are eliminated without checking the existence of any other conditions, resulting in an artifact-free abstract model as depicted in Fig. 3.

Abstraction Rule 6 (*Activity Abstraction by Elimination*)**.** Activities are very important process elements as they are used to analyze what constitutes a business process; hence, describe the work performed by each process participant. Activities involved in message exchanging are considered significant (Definition 6) and are not candidate for abstraction. There are also activities performed internally by a process participant in order to accomplish a specific goal; these activities are candidates for abstraction. Regarding the latter category of activities which, based on Definition 6, are characterized as insignificant activities, we propose in Abstraction Rule 6 that the abstraction of those activities is performed based on the activity property value A_{value} (Definition 3). More precisely, an activity flow object is candidate for abstraction by elimination, if its property value is either Manual, Script, or Business Rule. This is due to the fact that, based on our experience while working with business processes, "Manual" tasks (e.g. telephone call to a customer) do not provide valuable information for process comprehension, to process stakeholders (i.e directors). Also, scripts and business rules run automatically by rule or process engines; therefore, those, too, do not provide valuable information for the comprehension of the whole process model. The following definition formalizes Abstraction Rule 6.

Definition 16 (*Activity Elimination Rule*)**.** Let $a_i \in$ A be an activity that exists in an activity set A and α the function that assigns a value to the activity a_i as defined in Definition 3. Then, the elimination operation μ (Definition 9) is performed to the activity a_i in the following case:

- $\boldsymbol{\alpha}(a_i)$ = Manual ∨ $\boldsymbol{\alpha}(a_i)$ = Business Rule ∨ $\boldsymbol{\alpha}(a_i)$ = Script ∧
- ∄da such that $\boldsymbol{\kappa}$(da) = a_i, meaning that there are no data objects associated with the specific activity ∧
- |SFa_i|= 2, meaning that the cardinality of the activity sequence flow is no more than two ∧
- Insign(a_i) = true, meaning that the activity is characterized based on Definition 6 as insignificant.

For example, the manual activity "Ask for customer needs" in Fig. 2, (i) has the property value "Manual"; (ii) is not connected to any data objects; (iii) has only one incoming and one outgoing sequence flow; and (iv) does not exchange messages with other process participants; therefore, it is eliminated and does not exist in the abstracted model of Fig. 3.

Abstraction Rule 7 (*Activity Abstraction by Aggregation*)**.** As it is discussed in the previous abstraction rule, there are activities candidates for abstraction. In this rule we suggest that activity flow objects are candidate for abstraction by aggregation if (i) they acquire the same activity property value A_{value} (i.e., Service, Send, Receive, User); (ii) they are performed by the same process participant; (iii) they are in a sequence, in the same process fragment; (iv) the cardinality of their sequence flows does not exceed a specific threshold; and, (v) either there are not any data objects associated to them or there is a data object that is connected to all of them. The Abstraction Rule 7 is formalized by the following definition.

Definition 17 (*Activity Aggregation Rule*)**.** Let

- a_i, a_j ∈ A be activities that exist in the activity set A,
- l_i, l_j ∈ L be the lanes of the lane set L of the respective activities,
- da_i, da_j ∈ DA be data associations of the data associations set DA of the respective activities and
- $\boldsymbol{\alpha}$ is the function that assigns to each activity a value as defined in Definition 3.

Then, the aggregation operation $\boldsymbol{\beta}$ (Definition 10) is performed to activities a_i, a_j in the case that the following conditions occur:

- $pfa_i = pfa_j$, meaning that both activities belong to the same process fragment;
- $\boldsymbol{\alpha}(a_i) = \boldsymbol{\alpha}(a_j)$, meaning that both activities have the same activity property value as it is described in Definition 3;
- $a_i \bullet = a_j$, meaning that both activities are in a sequence;
- $\boldsymbol{\rho}(l_i) = \boldsymbol{\rho}(l_j)$, meaning that both activities belong to the same process participant;
- (∄ da such that ($\boldsymbol{\kappa}(da_i) = a_i$ **V** $\boldsymbol{\kappa}(da_j) = a_j$)) **V** (∃ da such that ($\boldsymbol{\Lambda}(da_i) = \boldsymbol{\Lambda}(da_j)$ **V** $\boldsymbol{\Pi}(da_i) = \boldsymbol{\Pi}(da_j)$)), meaning that either there are no data associations to the respective activities, or there are data associations (da_i and da_j) whose source/target is the same data element.
- $|SFa_i| = |Sfa_j| \leq 2$, meaning that there are at most two control flows to both a_i and a_j.
- Insign(a_i) = true **Λ** Insign(a_j) = true, meaning that both a_i and a_j have been defined as insignificant based on Definition 6.

For example, the activities "Verify customer identity" and "Create client record" in Fig. 2, have been aggregated to the activity "Verify customer identity + Create client record" in the abstract model of Fig. 3, as (i) they are performed by the same role "Customer advisory"; (ii) they acquire the same activity property value: $\boldsymbol{\alpha}$ (Verify customer identity) = $\boldsymbol{\alpha}$ (Create client record) = User; (iii) both activities are in a sequence: (Verify customer identity) • = (Create client record); and (iv) both activities are associated with two sequence flows, one incoming and one outgoing.

3.3. Process model abstraction strategy

In this section we describe an abstraction strategy which comprises a sequence of steps that combine multiple abstraction rules. When these steps are applied on a process model in a certain order, they derive an abstract model. Abstraction rules are atomic, i.e., they do not depend on other rules; therefore, various abstraction strategies can be defined. In general, different strategies result to different abstract process models. In our strategy, we restrict rule application in a certain order and we suggest that every rule is applied only once. In the following Section 3.3.1 we present the order of the employment of the abstraction rules, in the form of an algorithm.

Part of the abstraction strategy is also the identification of process fragments which contain semantically relevant process elements. As we have mentioned in Section 2.4, for each process fragment, the proposed strategy suggests coarse-grained activities that are semantically equivalent to the inner part of the fragment, in order to replace the latter. This process is described in detail in Section 3.3.2.

3.3.1. Algorithm for employing the proposed abstraction rules

The proposed abstraction rules are applied in a certain order according to the algorithm which is presented below (Algorithm 1). The algorithm takes as input a process model PM and produces an abstract process model PM′ (that is free from insignificant details and easier to comprehend).

In brief, this algorithm,

- examines process elements of the given process model,
- identifies their Type by applying the function $\boldsymbol{\zeta}$ (see Definition 2) and
- adds them (i.e. the process elements) to the respective set (see lines 1–21).

Precisely, in case a process element is a lane, artifact, or insignificant data (see Definition 7), it is added respectively to the lane set L (see lines 3–4), to the artifacts set ART (lines 5–6) or to the data set D (lines 7–9). In case a process element is a connecting object, its property value is also examined (lines 10–19). In case that the property value of the connecting object (see Definition 5) is "data association" or "message flow" or "association", the connecting object is added respectively to the data association set DA (lines 11–12), the message flow set (MF) (lines 13–14) or the associations set (Ass) (lines 15–16).

Then, process fragments are identified based on Algorithm 2 (line 22) that is analytically presented in the following section. Next, Algorithm 1 creates a copy of the original model PM′ (line 23); where all abstraction rules are applied (lines 24–30). More precisely, the first abstraction rule that is applied is the Lane aggregation rule in line 24 (Definition 11), followed by the Artifact elimination rule in line 25 (Definition 15) and the Data aggregation and Data elimination rules in lines 26–27 (Definitions 13–14). Then, the rules related to activity abstraction (Definitions 16–17) are applied (lines 28–29). More precisely, Activity elimination rule is performed prior to Activity aggregation rule. Next, the rule for message abstraction, i.e. the Message Flow aggregation rule (Definition 12) is applied by analyzing the messages and evaluating the conditions defined in Section 3.2 (line 30). Finally, the abstract model is returned as the output of the abstraction strategy (line 31).

3.3.2. Identification of process fragments

This section demonstrates how our proposed abstraction strategy identifies existing process fragments that contain closely related process elements (as it is described in Section 2.4). The

```
Input: A process model (PM), and a set (PE)
of process elements
Output: An abstract process model (PM') and
a set (PE') of its process elements
1   For each element pe in PE do
2     Case based on function ζ(pe)
3        Case = Lane
4          Add pe to L set
5        Case = Artifact
6          Add pe to ART set
7        Case = Data
8         If insign(data)=true
9         THEN Add pe to D set
10       Case = Connecting Objects
11        IF γ(CO)= DA
12        Then ADD pe to DA set
13        Else IF γ(CO)= MF
14              THEN Add pe to MF set
15              Else IF γ(CO)= Ass
16                    THEN Add pe to Ass set
17                    EndIF
18              EndIF
19        EndIF
20    End Case
21  End For
22  Initialise process fragments sets;          // Algorithm 2 (section 3.3.2)
23  PM'=PM
24  Perform Lane aggregation rule (PM')
25  Perform Artifact elimination rule (PM')
26  Perform Data aggregation rule (PM')
27  Perform Data elimination rule (PM')
28  Perform Activity elimination rule (PM')
29  Perform Activity aggregation rule (PM')
30  Perform Message Flow aggregation rule (PM')
31  Return PM', PE'
```

Algorithm 1: Process model Abstraction Algorithm

identification of process fragments that have self-contained semantics is guided by the definition and the characterization of the process fragment as defined in Definition 8. This algorithm, which is a prerequisite for activity abstraction, takes as input the set N of nodes of a process model and extracts a set of process fragments. Thus, the algorithm traverses control flow elements (N) of the process model. As soon as it reaches a node with the characteristics of the outer node, as it is described in Section 2.4, it creates a new process fragment. In this new fragment, it inserts nodes until the next outer node is identified.

```
Input: set N
Output: a set of process fragments PF
1   Set n the number of elements of N
2   For counter=1 to n do
3      Compute θ(n)
4       IF θ(n)= PF_outer
5      Then Create new pf to PF
6            Insert N[counter] to pf of PF
7            Remove N[counter]
8            Repeat
9              Counter'= counter+1
10             Insert N[counter'] to pf of PF
11           Until  θ (n)= PF_outer
12     Else go to step 2
13  endFor
14  Return PF
```

Algorithm 2: Find Process Fragments

More precisely, Algorithm 2 begins by defining as n, the number of nodes that exist in the set N, line 1. Then each element of the node set N, is mapped to a PF type N_{PFtype} that are {PF_{outer} and PF_{inner}} (lines 2–3). In the case, an outer node is reached, the algorithm continues by creating a new process fragment to the PF set (lines 4–5). Next the algorithm inserts the current node to the newly created process fragment and removes this node from the set N, lines 6–7. After that, it starts iterating by moving to the next element of N, inserting the current element to the newly created pf, until the function $\boldsymbol{\theta}$ maps an element to the value PF_{outer} value (lines 8–11). The algorithm follows this procedure for the rest of the elements of N and finally the set PF is created and it constitutes the output of the algorithm (lines 12–14).

4. Empirical validation

The suggested process model abstraction method calls for validation. To this end, the first objective of the validation is to investigate the percentage (%) of process model abstraction that can be achieved by the proposed approach. Another objective is to estimate the efficiency of the proposed approach, in terms of how well it mimics human behavior during process model abstraction with the aim to facilitate process stakeholder's rapid comprehension.

In the following sections, we firstly provide a brief description of the prototype that we have implemented in order to realize the aforementioned validation objectives (Section 4.1);

Table 3
Validation objectives, validation questions and metrics.

Validation objective	Id	Validation question	Metric
Effectiveness of the proposed method	VQ1	"How much does the proposed process model abstraction strategy simplify complex business process models?"	PMCR (Complexity reduction- c_i)
Efficiency and usability of the proposed method	VQ2	"Does the proposed process model abstraction strategy mimic human behavior in abstracting complex business processes, for rapid comprehension?"	F-score (Precision–Recall)

then, we describe the different data sets that were produced for business or academic purposes and have been utilized in our experiments, along with human participation (Section 4.2) and finally, we discuss the performed validation and the gained results (Section 4.3).

4.1. Prototype implementation

To demonstrate the real-world applicability and effectiveness of our approach, we have developed a prototype tool that supports process model abstraction based on the proposed abstraction strategy and rules. The tool supports the import of process models expressed in BPMN format as well as the creation of new models or the modification of existing ones, by utilizing a BPMN Editor [30]. Both automatic and manual process model abstraction are supported.

More precisely, a user is able to create a new process model or visualize/modify an existing imported one. A user may also activate the automatic application of the process model abstraction strategy that was elaborated in the previous section. In case a user wishes to interfere in the abstraction process, the tool presents a list with all abstraction operations (as suggested by the proposed abstraction strategy). Then the user is able to manually select the abstraction operations to be applied on the selected process model.

When the abstraction process has been completed (either manually or automatically), the tool presents the simplified process model accompanied with statistics which show the percentage reduction of various process model elements. Therefore, in the case that the tool performs abstraction to the process model depicted in Fig. 2, it produces the abstract process model of Fig. 3 accompanied with statistics which show that the Data Object reduction is 50%, the Lane/Artifact reduction is 100% and the Activity/Message Flow reduction is 30%. The resulted abstract model is finally saved by the tool so that it can be visualized later on by the interested users.

4.2. Process model sets and human participation

The proposed abstraction approach has been tested against three different process model sets that consist of ninety (90) process models in total. The first model set is publicly available for research and teaching purposes and comes from the BPM Academic Initiative (BPMAI) [31]. It consists of process models created by students of academic institutes which are using the provided resources for teaching purposes in the area of Business Process Management. This collection was filtered in order to keep only models that comply with the BPMN 2.0. The second model set is taken from the non-normative document that was created by the OMG [22] with the primary goal to assist in interpreting and implementing various aspects of the BPMN 2.0 specification. The third collection consists of real-world process models representing different business processes that are in use in a European organization. This organization, in order to enable the communication between organization stakeholders in a standard manner, attempted to set up a repository with high-quality process models that represent internal organizational procedures in a graphical form. We utilized the first forty-seven (47) process models that were created and made available for our experiments.

The size of the models of the aforementioned three data sets varies; most of the examined models contained from fifteen (15) to one hundred and twenty-nine (129) process model elements. It should be noted that the size of each process model was determined by taking into account only the process elements types that are considered as abstraction objects by the proposed process model abstraction approach; i.e. activities, data, artifacts, lanes and connecting objects. It is also worth mentioning that the existence of other process elements such as gateways and events could increase the process model size; however, this would not affect the validation process as they are not considered as abstraction objects.

Besides, twenty-five (25) professionals with varying expertise participated in our experiments, all coming from the organization that provided us the third process model set. More precisely, we involved fifteen (15) process stakeholders that were part of the senior leadership team and thus had a general knowledge of the process landscape of the organization and ten (10) process stakeholders that had participated in the specification of the process models of this process model set. The reason we invited humans of both categories to participate in our evaluation is that we wanted to observe the selected abstraction operations performed both from a general point of view and specialist's point of view. All participated process stakeholders were interested in utilizing abstract process model views that would enable them to quickly comprehend business processes and perform decision making.

4.3. Validation approach

In this section, we elaborate on the previously defined objectives of validation. In order to perform our validation, we were based on the case study protocol detailed in [32]. To this end we firstly formulate the validation questions to be investigated, which are:

1. "*How much does the proposed process model abstraction strategy simplify complex business process models?*". The determination of the degree of simplification of the resulted models is related to the effectiveness of the abstraction strategy. As a metric to this validation question, we use the process model complexity reduction (PMCR) metric, which is discussed in Section 4.3.1.
2. "*Does the proposed process model abstraction strategy mimic human behavior in abstracting complex business processes for rapid comprehension?*" The determination of how well the proposed process model abstraction method complies with human comprehension of a relevant abstraction, is related to the efficiency and the usability of the proposed strategy. As a metric to this validation question, we use the F-score (Precision and Recall) metric which is discussed in Section 4.3.2.

Table 3 presents the validation objectives and the questions we intend to address. Moreover, the metrics are shown, which are used to realize our objectives.

Table 4
Process model abstraction and complexity reduction.

Process model element	Original process model	Abstracted process model	Abstraction operation	Complexity reduction (c_i)
Activity (A)	23	16	Aggregation (3) Elimination (4)	0.30
Data (D)	6	3	Aggregation (2) Elimination (1)	0.50
Lane (L)	2	0	Aggregation (1)	1.00
Artifact (Art)	2	0	Elimination (2)	1.00
Message Flow (MF)	10	7	Aggregation (3)	0.30

To address the first validation question, we used all three process model sets described in the previous section. We calculated the degree of simplification by comparing the number of process elements appearing in the original process model with the number of process elements appearing in the abstract process model.

To address the second validation question, we empirically explored the third data set that consists of real-world business process models. As we mentioned before, we involved a number of process stakeholders in the experiments who were either familiar with the specific model set or they were members of the senior leadership team of the organization that provided the specific model set for our experiments. During workshops, we discussed with them our objectives for process model abstraction and we encouraged them to use the implemented tool. This way they could select manually the desired abstraction operations out of the ones suggested by the tool and produce abstract models that they could quickly comprehend. In some cases, the process stakeholders suggested more abstraction operations than the ones proposed by the tool. By examining their suggestions, we found out that some of them included abstraction operations that were not based on the process model structure and process element properties but rather on process element label semantics which is out of the scope of the current research work.

In the following, we present the metrics that we have used to achieve the aforementioned objectives and to perform the validation, we explain the experiments and the obtained results, while we conclude with some lessons learned.

4.3.1. Complexity reduction metric discussion and effect on the abstraction procedure

In order to evaluate the complexity reduction of the process models and to determine how much the resulted models were simplified, we decided to use as a model complexity reduction metric, the size (i.e. the total number) of model elements in each process model. This is estimated similarly to the metric of "lines of code" which is based on the program size and used to evaluate the complexity of software [33]. Therefore, the size of a process model is determined by calculating the number of its structural components. It should be noted that gateways and events are not included in the estimation of the model size as those specific process elements are not influenced by the proposed abstraction strategy. Thus, as it is observed in Eq. (1) below, the complexity reduction $\boldsymbol{c_i}$ for a specific process element type (defined in Section 2.2, Definition 2) is the fraction of its cardinality decrease ($|Type| - |Type'|$) to its cardinality in the original process model ($|Type|$).

$$c_i = \frac{|Type| - |Type'|}{|Type|} \tag{1}$$

For example, in the original model of Fig. 2, the number of activities, is twenty-three (23). After abstraction is applied, as it is depicted in Fig. 3, the original number of activities is reduced by seven (7), which means that the complexity reduction for the process activities is 0.30 (7/23). Based on the previously mentioned figures (Figs. 2 and 3), Table 4 presents side-by-side and per abstraction object, the multitude of elements that exist prior and after the abstraction process has been applied. Also, it depicts the multitude of every abstraction operation and the complexity reduction achieved, per examined process model element.

Table 5
Complexity weights for process model elements.

Process model element	Weight
Activity (A)	0.35
Data (D)	0.35
Lane (L)	0.10
Artifact (Art)	0.10
Connecting Object (CO)	0.10

As it can be inferred from Eq. (1) and it is depicted in Table 4, the complexity reduction takes values between 0 to 1. Consequently, if the complexity reduction is near to 0, it means that the process model size is neither highly altered nor influenced by the proposed strategy. On the other hand, if the complexity reduction is close to 1, it means that the process model size is highly reduced by the proposed strategy. Therefore, the higher the complexity reduction of an abstract model, the higher the effectiveness of our strategy.

Furthermore, in order to evaluate the overall complexity reduction of a given process model, we suggested to apply a weighted approach, based on the weights depicted in Table 5. As it is depicted in this table, we have assigned low weights to the process elements which are always associated with a specific abstraction operation in our abstraction strategy. For example, Lanes are always associated with only the aggregation operation, as it was previously described in Section 3.2. Artifacts and connecting objects are also always associated with one abstraction operation (see Section 3.2). As long as data and activity process model elements are concerned (which are the core elements in a business process and are associated with multiple abstraction operations and conditions — see Section 3.2), they have been assigned a higher weight. Besides, these elements, especially when their multitude is increased, they increase the process model complexity, thereupon they weaken process model comprehension; hence their abstraction is of primary importance.

Therefore, as it is defined next in Eq. (2), given the original process model (PM) and the abstracted process model (PM′) produced by our abstraction strategy, the overall Process Model Complexity Reduction (PMCR) is the weighted sum of the complexity reduction of process model constructs (which were presented in Definition 1).

$$PMCR\left(PM, PM'\right) = \sum_i \omega_i \cdot c_i\left(PM, PM'\right)$$
$$i \in \{A, D, ART, LCO\},\ \omega_i \in \mathbb{R}, 0 < \omega_i < 1 \sum_i \omega_i = 1 \tag{2}$$

Table 6
Complexity reduction metric results.

	Original model					Simplified Model					c_i					Weighted c_i					PMCR(PM,PM')
	/A/	/D/	/ART/	/L/	/CO/	/A'/	/D'/	/ART'/	/L'/	/CO'/	/A/	/D/	/ART/	/L/	/CO/	/A/	/D/	/ART/	/L/	/CO/	
m1	16	5	2	0	41	13	0	0	0	32	0,19	1,00	1,00	""	0,22	0,07	0,35	0,10	""	0,02	0,54
m2	15	5	0	7	28	11	1	0	0	21	0,27	0,80	""	1,00	0,25	0,09	0,28	""	0,10	0,03	0,50
m3	42	0	0	2	83	37	0	0	0	78	0,12	""	""	1,00	0,06	0,04	""	""	0,10	0,01	0,08
m4	23	6	2	2	57	17	3	0	0	43	0,26	0,50	1,00	1,00	0,25	0,09	0,18	0,10	0,10	0,02	0,49
m5	23	1	2	2	50	16	0	0	0	39	0,30	1,00	1,00	1,00	0,22	0,11	0,35	0,10	0,10	0,02	0,68
m6	23	6	2	2	55	16	3	0	0	43	0,30	0,50	1,00	1,00	0,22	0,11	0,18	0,10	0,10	0,02	0,50
m11	8	0	1	0	20	7	0	0	0	17	0,13	""	1,00	""	0,15	0,04	""	0,10	""	0,02	0,16
m12	7	0	0	0	13	6	0	0	0	12	0,14	""	""	""	0,08	0,05	""	""	""	0,01	0,06
m13	17	5	3	0	42	16	4	0	0	37	0,06	0,20	1,00	""	0,12	0,02	0,07	0,10	""	0,01	0,20
m14	17	5	3	0	42	15	5	0	0	36	0,12	0,00	1,00	""	0,14	0,04	0,00	0,10	""	0,01	0,15
m15	17	0	1	2	47	11	0	0	0	40	0,35	""	1,00	1,00	0,15	0,12	""	0,10	0,10	0,01	0,24
m16	12	0	1	0	41	7	0	0	0	35	0,42	""	1,00	""	0,15	0,15	""	0,10	""	0,01	0,26
m21	18	0	0	0	31	13	0	0	0	26	0,28	""	""	""	0,16	0,10	""	""	""	0,02	0,11
m22	22	9	5	0	60	14	4	0	0	42	0,36	0,56	1,00	""	0,30	0,13	0,19	0,10	""	0,03	0,46
m23	24	0	2	4	51	20	0	0	0	46	0,17	""	1,00	1,00	0,10	0,06	""	0,10	0,10	0,01	0,27
m24	29	0	2	4	62	23	0	0	0	55	0,21	""	1,00	1,00	0,11	0,07	""	0,10	0,10	0,01	0,28
m25	16	0	1	4	29	14	0	0	0	25	0,13	""	1,00	1,00	0,14	0,04	""	0,10	0,10	0,01	0,21
m26	13	0	0	0	39	11	0	0	0	36	0,15	""	""	""	0,08	0,05	""	""	""	0,01	0,06
AVG	19	2,33	1,50	1,61	43,94	14,83	1,11	0	0	36,83	**0,22**	**0,57**	**1,00**	**1,00**	**0,16**	**0,08**	**0,20**	**0,10**	**0,10**	**0,02**	**0,29**
MAX											**0,42**	**1,00**	**1,00**	**1,00**	**0,30**	**0,15**	**0,35**	**0,10**	**0,10**	**0,03**	**0,68**

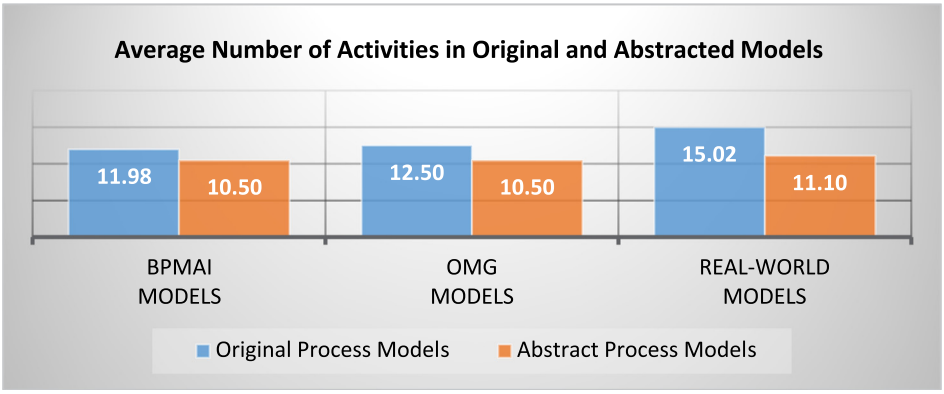


Fig. 4. Activity reduction.

As we mentioned before, in order to evaluate the proposed approach, we used the three process model sets, introduced in Section 4.2. They comprise process models defined by both process stakeholders (in attempt to manage organizational business processes) and students (for academic purposes) that had varying level of expertise as long as process modeling is concerned. The evaluation was performed in a series of experiments on the model sets, using the implemented prototype. For each experiment, we used the imported process models and generated the abstract process models using the algorithm that was presented in Section 3.3.

Next, for reasons of simplicity, we firstly discuss the obtained results on a specific sample of a process model set comprising 18 models (6 from each of the three different model sets), followed by the presentation of the overall obtained results.

Table 6 provides details regarding the influence of all proposed abstraction rules on the selected process model set (comprising of 18 models) and the achieved Process Model Complexity Reduction (PMCR) in each model. Generally, the proposed approach achieves PMCR up to 68%. More precisely, as it is depicted in Table 6, we firstly calculated the complexity reduction in process models in terms of the number of activities |A| , data |D| , artifacts |ART| , lanes |L| and connecting objects |CO| by using Eq. (1). Then, each complexity reduction result was weighted based on the weights presented in Table 5 and finally, for each process model, we calculated the overall process model complexity reduction PMCR by using Eq. (2). The effectiveness of the abstraction algorithm is measured as the percentage of the elements |A|, |D|, |ART|, |L| and |CO| that are removed from the original process model as well as the percentage by which the process model complexity was reduced.

Therefore, as it is presented in Table 6, our algorithm achieves abstraction of the original process model sets by removing up to 42% of the activities, 100% of data objects, artifacts and lanes and 30% of the connecting objects. Especially for the data objects, in cases that they were removed up to 100% during the abstraction process, it was noticed that their original multitude was minimum; hence, their removal did not influence the reliability and comprehension of the abstracted process model. As it is also depicted in this table, all 18 models were significantly simplified. In average, activities were reduced by 22%, data objects by 57%, artifacts and lanes by 100% and connecting objects by 16%. Moreover, the average process model complexity reduction has been calculated to 29% whereas the maximum process model complexity reduction is 68%. Therefore, we observe that the metric PMCR (Table 3) is high; hence the proposed abstraction strategy effectively simplifies complex business process models.

The proposed abstraction strategy, apart from being applied to the selected sample of process models, discussed above, was also applied to all the models of the three process model sets (introduced in Section 4.2) that we had in our disposal. In the following, we present the data related to our overall complexity reduction evaluation. Specifically, the overall model set comprises almost of 90 process models with total number of activities varying from 3 to 42. Fig. 4 shows the average number of activities that exist in the given process models of the acquired model sets

$$recall = \frac{\text{Nbr of abstraction operations automatically applied \& verified by process stakeholders}}{\text{Nbr of abstraction operations suggested by process stakeholders}}$$

Box I.

$$precision = \frac{\text{Nbr of abstraction operations automatically applied \& verified by process stakeholders}}{\text{Nbr of abstraction operations applied by the tool}}$$

Box II.

(i.e. from BPM Academic Initiative (BPMAI), from OMG and from the real-world) before and after the application of our abstraction strategy. For this purpose, in Fig. 4 the x axis depicts the different model sets that were used in our experiments, whereas the y axis depicts the average number of activities in the original and the abstract process models. It is observed that, as the multitude of activities increases in the original models, the same applies to the distance between the total number of the activities before and after abstraction. Therefore, we can conclude that process models which are larger in size, may benefit more from the proposed abstraction strategy.

The results obtained by performing the suggested process model abstraction strategy to all available process model sets, are presented in Table 7. Especially, this table depicts the average percentage of complexity reduction achieved by applying the respective rules, i.e. the Message Abstraction (MA) rule, the Data Abstraction (DA) rules and the Activity Abstraction (AA) rules. It should be noted that, when a process model contains artifacts and lanes, the percentage reduction is always stable to one hundred percent (100%); this is due to the fact that the artifact elimination and lane aggregation abstraction operations are always applied, as it is described in Section 3.2. Hence, this data is not included in Table 7. More precisely, it is observed that:

- The Activity Abstraction (AA) rules achieve similar results to all process model sets. This enhances the validity of the proposed abstraction strategy, taking into consideration the variability of the model sets that were used throughout the experiments.
- The Data Abstraction (DA) rules achieve the best results in the BPMAI and in the real-world process model sets. This is due to the fact that the OMG model set uses a limited number of data objects compared to the other model sets; therefore, a limited set of the defined DA rules is applied. Besides, the best results in OMG model set are gained by the application of the AA rules, as activities are mainly the model elements of the specific model set. Thus, it is inferred that the results of each rule application are influenced by the given process model sets and their modeling constructs.
- The Message Abstraction (MA) rule achieves the lowest percentage scores irrespectively of the process model set. This is due to the fact that it is applied only to exceptional cases, as it is described in Section 3.2. Besides, the OMG model set acquires the minimum number of message flows and hence it achieves the lowest average percentage.

It is worth mentioning that in the real-world process model set, the results were satisfying for all applied rules. This result is very positive, as the real-world model set is considered very important for the validation of the proposed abstraction strategy, in terms of its effectiveness to simplify complex business process models. To sum up, we observe that the complexity reduction (c_i) is high, especially for activities and data that highly influence process stakeholders' quick comprehension; hence the proposed abstraction strategy effectively simplifies complex business process models.

Table 7
Average percentage complexity reduction (c_i).

	Rules for process model abstraction		
	AA	DA	MA
BPMAI models	28,70%	57,20%	3,80%
OMG models	15,20%	10,00%	0,00%
Real-world models	24,60%	60,20%	3,30%

4.3.2. Precision and recall metric discussion and effect on the abstraction procedure

In order to evaluate the usability and efficiency of the proposed abstraction method we compared the abstraction operations suggested and applied automatically by our prototype tool to a given process model, with the abstraction operations specified by process stakeholders that participated in our experiments. To do so, we used as metrics the notions of precision (as a qualitative measure of exactness), recall (as a quantitative measure of completeness) and F-score [34]. These metrics allow the comparison of abstraction operations suggested by the proposed strategy with the abstraction operations suggested by process stakeholders.

More precisely, we defined as recall the fraction of the abstraction operations suggested and applied by the tool for a given process model, which were also verified as relevant by the process stakeholders, over all the abstraction operations suggested by process stakeholders (during manual abstraction) (*recall* is given in Box I).

Also, we defined as precision the fraction of the abstraction operations suggested and applied by the tool for a given process model, which were also verified as relevant by the process stakeholders, over all abstraction operations applied by the tool (during automatic abstraction) (*precision* is given in Box II).

The above two metrics can be combined together into a single measure of accuracy known as F-score which is a harmonic mean of the two measurements .

$$F - score = \left(2 \times \frac{recall \cdot precision}{recall + precision}\right)$$

Based on the above notions and definitions, we calculated the abstraction precision of the abstract process model in Fig. 3, compared to the original process models depicted in Fig. 2. More precisely, in the process model of Fig. 2, our prototype tool identified sixteen abstraction operations to be performed; specifically, two artifact eliminations, four activity eliminations, two data aggregations, one data elimination, three activity aggregations, one lane aggregation and three message flow aggregations. The

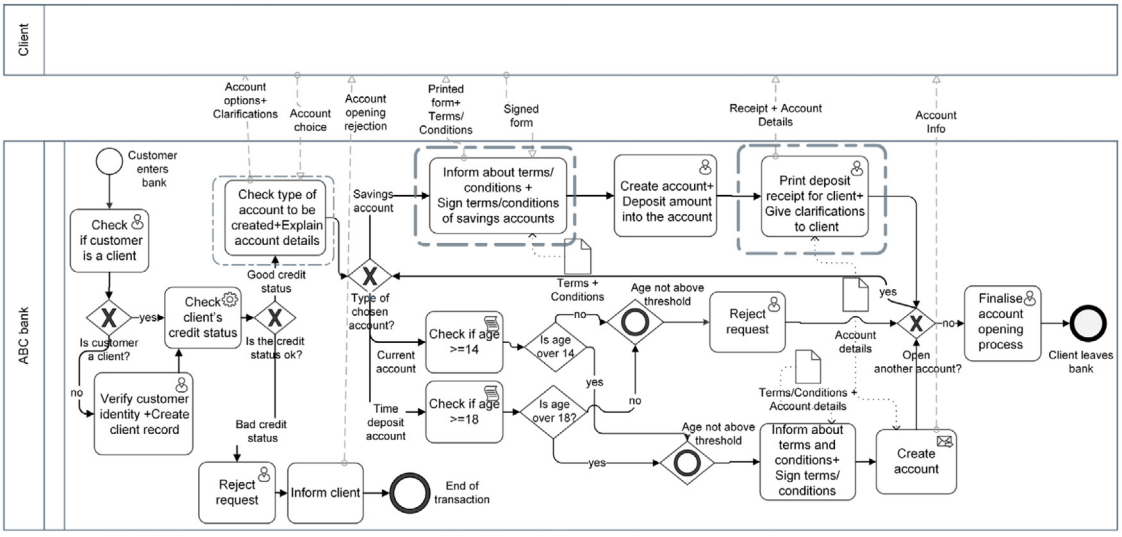
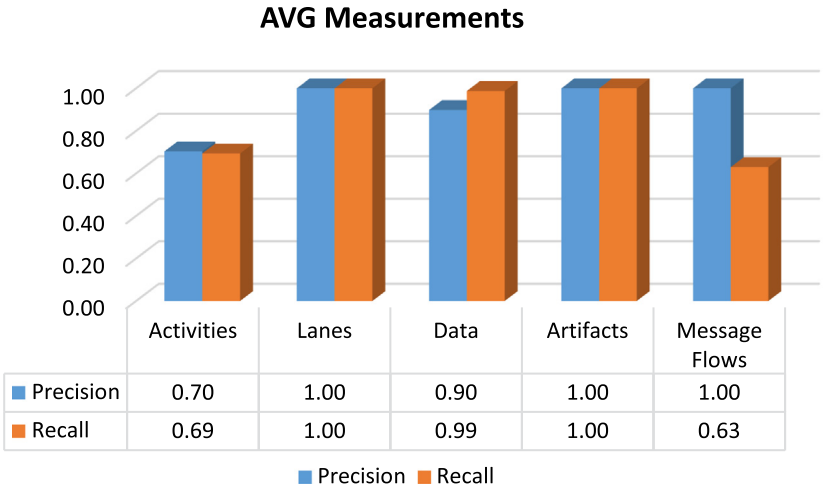


Fig. 5. Account opening process - Abstracted process model by process stakeholders.



	Activities	Lanes	Data	Artifacts	Message Flows
Precision	0.70	1.00	0.90	1.00	1.00
Recall	0.69	1.00	0.99	1.00	0.63

Fig. 6. Average precision and recall per process element.

process stakeholders verified as relevant fourteen (14) out of the sixteen (16) abstraction operations. Therefore, the precision for the abstraction performed in the model of Fig. 2 is 0.88 (i.e. 14/16).

In order to estimate the recall of the abstraction performed automatically by the tool on the model of Fig. 2, we used the process model shown in Fig. 5. It depicts the abstract model derived from the original model of Fig. 2, after the application of the abstraction operations suggested by the involved process stakeholders (during manual abstraction). More precisely, the process stakeholders suggested two artifact eliminations, two activity eliminations, two data aggregations, one data elimination, six activity aggregations, one lane aggregation and three message aggregations. It should be noted that the process stakeholders suggested three new abstraction operations that had not been applied by the tool; the result of the application of these new operations are indicated with a dotted box in Fig. 5. Therefore, the total number of abstraction operations suggested by the process stakeholders were seventeen (17), which means that the recall result is 0.82 (i.e. 14/17).

This difference between the precision and recall is due to the fact that the process stakeholders suggested additional abstraction operations. Especially, aggregation operations, on activities that were associated with message exchanging between participants and had similar label semantics. Finally, given the calculated values of the precision and recall for the given process model, the F-score value was calculated to 0.85. This shows that the proposed approach achieves great efficiency and abstracts an existing process model as humans would do, in an attempt to quickly comprehend a business process.

The experiments in terms of the precision and recall metrics were applied only to the third process model set which, as we mentioned above, comprises real-world models, as it was easier to involve process stakeholders who were already familiar with the described business processes. The average precision and recall results per abstraction object of the acquired process models are depicted in Fig. 6. Moreover, the average F-score gained for each abstraction object is presented in Table 8. Therefore, it is obvious that the proposed abstraction strategy in terms of these metrics:

- Has an excellent performance as long as lanes and artifacts are concerned. This is due to the fact that these model elements, as it was commonly accepted by all process stakeholders, were insignificant and the elimination operation was always performed.
- Performs very well for the data and activity elements of the given process model set. This is due to the fact that

Table 8
Average F-score per process element.

Process model element	F-score
Activity (A)	0,70
Lane (L)	1,00
Data (D)	0,94
Artifact (ART)	1,00
Message Flow (MF)	0,77

Table 9
Average reduction per process element.

Method	Activities	Data	Message flows
Automatic abstraction	24,6%	63,3%	3,8%
Manual abstraction	25,7%	59,3%	7,4%

these model elements are associated with many abstraction operations that are applied upon certain conditions which were in their majority accepted by all process stakeholders participated in the evaluation.

- Achieves the lowest score on the recall metric for message flows. This is due to the fact that these model elements are considered as significant and have undergone limited abstraction by the proposed abstraction strategy. As it was discussed in Section 3.2, these elements can be associated only with the aggregation operation. However, it was observed that process stakeholders often suggested additional abstraction aggregation operations which were mostly triggered by the message flow semantics and not by the structure and the properties of the associated source/target process elements.

Besides, Table 9 depicts the average reduction per process element, particularly for activities, data and message flows. The reduction is depicted both for the automatic process model abstraction performed by our prototype tool and for the manually abstraction performed by the involved process stakeholders.

Based on the results depicted in Fig. 6 and in Tables 8 and 9, we conclude the following per abstraction object:

- Message flows: the proposed strategy exhibits great difference between precision and recall. More precisely, precision is in most cases better than recall which means that, in some cases, as it is shown in Table 9 (for Manual Abstraction), process stakeholders suggested more message abstraction operations that were based on the message flow label semantics.
- Activity: In most process models, the average precision achieved by the proposed strategy is slightly better than recall. Also, as it is depicted in Table 9, the average activity reduction both in automatic and manual abstraction is very similar. Therefore, it is inferred that only in a few cases the involved process stakeholders identified additional abstraction operations. Thus, it is concluded that the proposed approach mimics human behavior at a great extent while producing abstract process models that enable quick comprehension.
- Data: the average recall achieved by the proposed strategy is better than the average precision. Moreover, the average data reduction, in the case that the proposed approach is automatically applied is higher than in the case that the abstraction operations are manually applied. This means that the proposed strategy, as long as data elements are concerned, is better suited in identifying and applying abstraction operations to process models than process stakeholders would do. This is due to the fact that the number of data elements in the process models that were at our disposal and were used in the validation process, was most often low. Therefore, there was no need by business process stakeholders to further reduce their number as it did not influence their comprehensibility of the original model.

4.3.3. Combined metric validation

With combined metric validation we refer to the validation performed, using the Process Model Complexity Reduction (PMCR) metric described in Section 4.3.1, in combination with the F-score, presented in Section 4.3.2. Since the F-score metric evaluation requires the involvement of the process stakeholders, the combined metric evaluation was performed only on the third process model set (comprising 47 real-world process models from industry). The reason for that is that process stakeholders could only be involved in the validation of this model set, as they were familiar only with business processes coming from the specific industry practice.

Thus, for the third model set, as it is described in Table 10, the average F-score which is the metric for our VQ1 is 0.88, whereas the average PMCR which is the metric for our VQ2, is 0.51. It can be observed that the average F-score is quite high meaning that the proposed abstraction approach is able to deliver efficiently and automatically simplified process model views, as humans would manually deliver, for quick comprehension of existing complex business processes. Therefore, it proves that the proposed approach enables process stakeholder's comprehension and increases the usability of complex business process models. On the other hand, the average PMCR is lower. It should be noted that a higher model complexity reduction could be achieved if process models were simplified until process model elements were minimum; however, in most cases, this would weaken the comprehension of process stakeholders. Therefore, this would be against the goal of our approach which is to retain important process model elements and to maintain the overall process model structure in order to not harden the overall business process comprehension. Thus, the acquired model complexity reduction is considered as a satisfactory result and proves that the proposed approach effectively simplifies detailed business process models.

4.3.4. Observations and lessons learnt

We would like to highlight that, in addition to the evaluation results presented in the previous sections, we concluded the following observations and lessons learnt, in cooperation with the process stakeholders who were involved in the evaluation of the proposed strategy:

Table 10
Validation objectives, validation questions, metrics and results.

Validation objective	Id	Validation question	Metric	Result
Effectiveness of proposed method	VQ1	"How much does the proposed process model abstraction strategy simplify complex business process models?"	PMCR- (Complexity reduction- c_i)	0.51
Efficiency and usability of proposed method	VQ2	"Does the proposed process model abstraction strategy mimic human behavior in abstracting complex business processes, for rapid comprehension?"	F-score (Precision–Recall)	0.88

1. The process model complexity reduction increases in process models with artifacts and/or with multiple roles which have been modeled as "Lanes" in BPMN 2.0. This is due to the fact that in both cases there is always one abstraction operation which is performed (that is Artifact elimination or Lane aggregation respectively), without examining further conditions. Complexity reduction is also increased when there is:
   - a large multitude of data objects that acquire one data association to process activities (based on Definition 13);
   - a large multitude of process fragments with a large multitude of inner model elements; this is due to the fact that such process fragments contain a large multitude of activities that are in a sequence; hence, the amount of abstraction operations is increased;
   - a small amount of process interactions amongst different participants; in such cases, a greater number of process elements are characterized as insignificant by the proposed strategy (as it has been defined in Section 2.3) and participate in the abstraction process.

   Therefore, process model complexity reduction is influenced by the process model set at hand; hence the modeling constructs of the acquired process models.
2. Complexity reduction is rarely influenced by the message aggregation abstraction rule. Indeed, as it was noticed during our experiments, the message flow abstraction operation was applied only in a few models. This stems from the fact that such abstraction operations are associated with limited conditions as it was elaborated in Section 3.2.
3. The precision and recall metrics are increased for larger process models, meaning that process models which are larger in size, may benefit more from the proposed abstraction strategy.
4. The abstraction operations suggested by the process stakeholders highlighted amongst others that they additionally consider process element label semantics. In these cases, it is noticed that the recall is less than precision as the process stakeholders have suggested more abstraction operations than the ones suggested by the implemented tool. This influences the f-score value, which in such cases is reduced; hence, it affects the validation of the proposed approach.
5. The abstraction operations suggested by the process stakeholders were varying depending on their experience and their expertise in process modeling. More precisely, there was a tendency for detailed business processes by process stakeholders who were involved in the specification of the process models. On the other hand, process stakeholders that were part of the senior leadership team had the tendency to propose additional abstraction operations based on process element label semantics. Therefore, the process stakeholders' experience and expertise in process modeling affects the validation of the proposed approach.
6. The resulting abstracted process models are highly dependent on the order that the abstraction rules are applied by the abstraction strategy as it has also been discussed in Section 3.3.1.
7. The proposed approach can be well applied to detailed business process models, as the ones that we used during our evaluation, while we have no evidence yet on how the suggested approach could be applied to process models containing mostly abstract elements.

5. Related work

Process model comprehension is a research area that attracts a lot of attention by the research community. A vast body of research focuses on studying the factors that influence process model comprehension [35–38] and methods to get new insights on process model comprehension [39–41]. Also, techniques have been proposed that enrich process models to foster their comprehension [42–44]. The work that is presented in this paper, is focused on techniques that simplify detailed process models and produce quick process model views that facilitate and enhance the process model comprehension by interested process stakeholders. Hence, it assists particularly the process stakeholders that are part of the leadership team of an organization and quite often use process models for decision making purposes, as it was elaborated in Section 1. The main contribution of this paper in this area is the derivation of simplified process models by abstracting process model elements of various kinds (i.e., activities, data, artifacts, lanes and connecting objects) based on both process model structure and process element properties.

There are a lot of research efforts aiming at providing simplified models, focusing on various aspects of a process model. For example, in the work by Smirnov [45] business process model abstraction is performed by analyzing process behavioral profiles, while in the works by Fahland and Van Der Aalst [46], Günther and Van Der Aalst [47], and by Senderovich et al. [48,49] abstraction is applied only to operational data. Also, in the work by Weber et al. [50] the authors suggest simplified process models by identifying certain process smells and applying refactoring techniques using for instance process footprint similarities. In this section, we present research efforts that provide generic or domain specific process model abstraction methods, based on the process model structure, i.e. approaches which are similar to our work; thus, the non-structural process model abstraction efforts that do not take into account the control-flow aspects during model abstraction (like the work by Chapela-Campa et al. [51], de San Pedro et al. [4] and Tax et al. [52]) are not included in our review.

The techniques described by Bobrik et al. [53], Cardoso et al. [54], Polyvyanyy [55], Polyvyanyy et al. [15,27], Reichert [56] and Reichert et al. [57] view processes as graph structures consisting of nodes that represent tasks and edges representing transitions; therefore, they view process abstraction as graph abstraction. The approaches described by Chiu et al. [29] and Eshuis and Grefen [18] apply abstraction on UML process models. Also, the approach of Sadiq and Orlowska [58] apply abstraction to workflow graphs, whereas Pankratius and Stucky [59] apply abstraction on Petri-Net based process models. Moreover, the most recent approaches to process abstraction that are described by Ramos-Merino et al. [14], Köpke et al. [60], Wurm et al. [61], Fatima and Shahzad [19], Wang et al. [20], Yongchareon et al. [62], Khlif and Ben-Abdallah [63], Kchaou et al. [64] as well as older ones, such as the works by Liu and Shen [65], Mafazi et al. [66], Meyer and Weske [67], Harzmann et al. [68] and Smirnov et al. [21] have as an input process models. Most of these approaches consider activities as abstraction objects. Exceptions to this, are the works by Meyer and Weske [67] and Harzmann et al. [68] which, besides activities, also consider data as abstraction objects and provide an abstraction strategy that bases its abstraction decisions on the occurrences of data objects in the process models and their relations in a given data model. Another exception is the work by Wurm et al. [61] that suggests exploiting activities, data and roles of a business process model in order to achieve process individualization. However, this work presents a conceptual model for business process individualization, it is in its genesis and it calls for further research. Lastly, Khlif and Ben-Abdallah [63] and Kchaou et al. [64] suggest a process model

transformation approach to enable process model understandability, by focusing on activities and roles while performing abstraction operations. The process model abstraction technique presented in this paper, takes into account, not only activities but an extended list of process model elements as well, as abstraction objects. Specifically, since the proposed technique targets process models expressed in BPMN 2.0, we exploited the expressiveness of BPMN. Then, we suggested to simplify process models not only based on control-flows, but also based on the nature (i.e. the properties) of all process model elements, while at the same time, we provided the means for abstracting activities, data, roles, message flows and artifacts.

Moreover, most of the techniques, e.g. Cardoso et al. [54], Chiu et al. [29], Liu and Shen [65], Pankratius and Stucky [59], Polyvyanyy [55], Polyvyanyy et al. [27], Köpke et al. [60], Pérez-Castillo et al. [69], Fichtner [70], Ramos-Merino et al. [14] and Wang et al. [20], apply to a specific use case, such as adapting process views to be used by an external partner or by a specific role in an inter-organizational environment, etc. Our approach has the same goal as Ramos-Merino et al. [14], Cardoso et al. [54] and Wang et al. [20], which is to provide process model quick views. More precisely, Ramos-Merino et al. [14] suggest the simplification of process models in order to provide an easier understanding of graphic representation. To this end, they suggest the discovery of small reducible patterns in the whole model and their substitution with optimized versions, utilizing a pattern repository. However, this work, in contrary to our work, is focused on limited process model elements types (i.e., activities and gateways). Moreover, Cardoso et al. [54] suggest the creation of quick views by respecting ordering constraints and roles while preserving relevant activities. The approach that we propose considers not only ordering constraints but also properties and associations of process elements, while preserving activities, relevant data and message flows of the given process model. In addition, we do not preserve specific roles in the abstract models. This is due to the fact that this knowledge, as we stated in Section 3.2, is only related to the internal organizational structure and may be omitted in a simplified process model view which is intended to be used mostly by process stakeholders who are involved in the decision-making process of an organization. Furthermore, Wang et al. [20] suggest forming clusters based on the process model structure and activity semantics, while they apply only activity aggregation. On the contrary, we take into account additional process model perspectives, for example process element properties, data objects, associations and roles, while abstracting process models and we apply a greater number of abstraction operations.

Furthermore, most of the existing abstraction techniques (see for example the work by Bobrik et al. [53], Mokaddem et al. [71], Cardoso et al. [54], Liu and Shen [65], Mafazi et al. [66], Meyer and Weske [67], Polyvyanyy [55], Polyvyanyy et al. [15,27], Reichert [56], Reichert et al. [57] and Sadiq and Orlowska [58]) use reduction rules for structural transformations. Specifically, in [54], the authors analytically describe a workflow reduction method that applies a set of reduction rules until only one atomic task exists. These reduction rules are triggered when specific patterns (sequence, AND-block, XOR-block and loops) are identified within a workflow. Moreover, in this work the reduction rules concern both structural transformations and non-functional properties evaluation method in an attempt to achieve process order and non-functional properties preservation abstraction. Also, in an older work by Sadiq and Orlowska [58], reduction rules (terminal, sequential, adjacent, closed and overlapped) are applied iteratively, in order to overcome process structural conflicts of workflow graphs and verify its correctness. These techniques either derive an atomic task or a very condensed view of the original workflow graph. Thus, they may deteriorate human comprehension related to business process. The abstraction strategy that we are proposing is based on a single application of each presented abstraction rule in an attempt not to have great information loss. Also, it respects the overall structure, especially by retaining the interaction points amongst process participants.

Besides, the work by Bobrik et al. [53], Reichert [56] and Reichert et al. [57] aim at flexible configuration and personalization of large process models. The approach of these authors is based on user preferences and formalizes processes as graphs consisting of nodes of activities/gateways and edges that describe the control flow. The authors propose a strategy that takes as parameter a set of activities to be reduced or aggregated and then determines the appropriate elementary operations to be applied using a set of reduction and aggregation rules. Finally, a customized process view is created. On the other hand, in the work by Polyvyanyy [55] and Polyvyanyy et al. [15,27], the authors suggest that the user specifies manually the tasks or the collection of tasks that need to be generalized and then the abstraction rules are triggered (Q-type, S-type, P-type and R-type abstraction). Also, Yongchareon et al. [62] suggest a unified framework that enables users to define their process views requirements, using the View Definition Language, disregarding the types of process models (BPEL and BPMN). Similarly, Kammerer et al. [72] suggest a descriptive language that enables the definition of abstractions on process models in order to obtain process model views. Finally, Mafazi et al. [66] suggest a questionnaire method in order to recommend a configuration of model abstractions tailored to a specific abstraction goal expressed as constraints on the abstraction relation and process transformation operators. The approach aims at controlling the operators that are applied (based on user's answers) in the process models in order to derive a simplified one. Our work, as the abovementioned research works, exploits process model structure, while defining the process rule-based abstraction strategy. However, in contrast to the previously mentioned research works that manually select, either process model components to be abstracted or abstraction operations to be applied, our proposed abstraction strategy can be applied automatically to the whole process model without any user interference.

Last but not least, Liu and Shen in [65] use a rule-based strategy that incorporates rules for atomicity, membership and order preservation, to derive virtual activities and provide process views that are driven from the process roles. Moreover, Mokaddem et al. [71] suggest the exploitation of past rules and recommend activity refactoring rules that best apply to the current example. Also, Meyer and Weske [67] consider data and activities as abstraction objects and provide an abstraction strategy that bases its abstraction decisions on the occurrences of data objects in the process models and their relations in a given data model. For this reason, they formalize a rule set that aims at providing abstraction consistency, preserving granularity alignment, etc. In these techniques, the rule set that is used is either focused only on the activities of the process or on the occurrences of data objects within the process model. Our rule based-strategy goes beyond considering only activity or data object dependencies, as it exploits the properties and associations of all process model elements (i.e. activities, data, roles, message flows and artifacts), while applying the rule-based strategy which embeds two different abstraction operations (i.e. aggregation and elimination).

6. Conclusion

In this paper, we address the need of a novel approach that enables full exploitation of existing detailed process models and extends their usability. Such an approach can be of great value

to process stakeholders in every organization that need to be able to quickly comprehend large business processes in order to make strategic decisions. In order to facilitate such cases, we presented an approach which provides a novel solution to process model abstraction. Compared to other research works that use reduction rules and consider control-flows in order to abstract process models, we propose a holistic process abstraction strategy which goes beyond existing works. Specifically, our approach is applied to multiple process models elements, i.e. to activities, data, message flows, roles and artifacts. Also, it takes into account process model element properties, such as types and type values, as well as associations to other process elements, in order to determine appropriate abstraction operations. To the best of our knowledge, there are no research works that exploit a wider range of process elements and their properties as well as the process model structure, in order to deliver abstract process models.

The presented approach takes as input a fine-grained BPMN complex process model. Then, it applies transformation rules which omit information, such as artifacts and lanes that are not considered important for quick model comprehension by process stakeholders involved in the decision-making process. It produces as output an abstract process model view. The specified rules have been derived based on our experience in business process modeling and by examining the Business Process Model and Notation (BPMN 2.0) which is a de-facto standard for modeling business processes and has a great acceptance by industry. A prototype tool has been implemented as a proof of concept and in order to validate the proposed approach.

The experimental validation, which involved a number of process stakeholders, provided a strong support for the usability and efficiency of the presented approach. To be more precise, the comparison of the results obtained by the automatic application of the suggested abstraction strategy, with the way that humans would manually perform the abstraction aiming at quick model comprehension, exhibited great similarity and proved its usability and efficiency. As an example, the average precision for activity and data abstraction was 0.70 and 0.90, whereas the overall recall was 0.69 and 0.99 respectively. Therefore, the proposed technique increases the usability of existing process models as it enables the simplification of existing detailed process models with great efficiency. Moreover, the experiments proved that the proposed technique is effective as it reduces process models up to 68%; they also demonstrated that larger in size process models could benefit more from our suggested abstraction strategy.

We would like to note that there are some limitations in our approach which constitute the basis of our future work. Firstly, the experiments revealed that the complexity reduction results are influenced by the given process model set and its modeling constructs. Nevertheless, the fact that the models were coming from variable sources, specifically from industry and academic institutes, provide a good indication that the suggested approach can be effectively applied to a broad range of business domains. Secondly, the abstraction operations suggested by process stakeholders who had been involved in the process specification of the models, highlighted that, in some cases, they additionally consider process element label semantics. In these cases, the recall was less than precision as the manual abstraction identified more abstraction operations compared to the ones suggested by the implemented tool. Therefore, as we have already mentioned, the F-score value was also influenced. Last but not least, it was noticed through the validation procedure and especially during manual abstraction that the identification of the relevant abstraction operations was influenced by the experience and the expertise of those involved in the validation process.

As we aforementioned, these limitations and the overall validation of our approach guide our future research plans. More precisely, we plan to enhance the abstraction strategy so as to consider process element label semantics while suggesting respective abstraction operations. We also plan to enhance the validity of our approach through our cooperation with other large organizations, acquiring larger and diverse process model sets. Besides, we intend to extend the validation by involving more humans with diverse experience and expertise in process modeling and then extract inferential statistics. Last but not least, we intend to experiment with more abstraction strategies beyond the proposed one, while using the same rules, in order to study the differences in the obtained results.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

References

[1] M. Dumas, M. La Rosa, J. Mendling, H.A. Reijers, Fundamentals of Business Process Management, second ed., Springer, Heidelberg, 2018.
[2] M. Weske, Business Process Management: Concepts, Languages, Architectures, Springer Verlag, 2007.
[3] R. Akkiraju, A. Ivan, Discovering business process similarities: An empirical study with SAP best practice business processes, in: International Conference on Service-Oriented Computing, Springer, Berlin, Heidelberg, 2010, pp. 515–526.
[4] J. de San Pedro, J. Carmona, J. Cortadella, Log-based simplification of process models, in: H.R. Motahari-Nezhad, J. Recker, M. Weidlich (Eds.), BPM 2015, in: LNCS, vol. 9253, Springer, Cham, 2015, pp. 457–474.
[5] M. Rosemann, Potential pitfalls of process modeling: partB, Bus. Process Manage. J. 12 (3) (2006) 377–384.
[6] K. Lyytinen, Digitalization and routines-another look at business process management, in: International Conference on Business Process Management, Springer, Cham, 2019.
[7] S.J. van Zelst, J.C. Buijs, B. Vázquez-Barreiros, M. Lama, M. Mucientes, Repairing alignments of process models, Bus. Inform. Syst. Eng. (2020) 1–16.
[8] J. vom Brocke, M. Jans, J. Mendling, H.A. Reijers, Call for papers, issue 5/2021, Bus. Inform. Syst. Eng. (2020) 1–3, http://dx.doi.org/10.1007/s12599-020-00630-7.
[9] M.L. van Eck, N. Sidorova, W.M. van der Aalst, Guided interaction exploration and performance analysis in artifact-centric process models, Bus. Inform. Syst. Eng. 61 (6) (2019) 649–663.
[10] M. Fischer, F. Imgrund, C. Janiesch, A. Winkelmann, Strategy archetypes for digital transformation: Defining meta objectives using business process management, Inform. Manage. (2020) 103262.
[11] E. Niemi, S. Pekkola, The benefits of enterprise architecture in organizational transformation, Bus. Inform. Syst. Eng. (2019) 1–13.
[12] J. Vom Brocke, W. Maaz, P. Buxmann, A. Maedche, J.M. Leimeister, G. Pecht, Future work and enterprise systems, Bus. Inform. Syst. Eng. 60 (4) (2018) 357–366.
[13] O. Turetken, A. Dikici, I. Vanderfeesten, T. Rompen, O. Demirors, The influence of using collapsed sub-processes and groups on the understandability of business process models, Bus. Inform. Syst. Eng. (2019) 1–21.
[14] M. Ramos-Merino, L.M. Álvarez Sabucedo, J.M. Santos-Gago, F. de Arriba-Pérez, A pattern based method for simplifying a BPMN process model, Appl. Sci. 9 (11) (2019) 2322.
[15] A. Polyvyanyy, S. Smirnov, M. Weske, Business process model abstraction, in: J. vom Brocke, M. Rosemann (Eds.), Handbook on Business Process Management 1. International Handbooks on Information Systems, Springer, Berlin, Heidelberg, 2015.
[16] S. Smirnov, H.A. Reijers, M. Weske, T. Nugteren, Business process model abstraction: a definition, catalog, and survey, Distrib. Parall. Databases 30 (1) (2012) 63–99.
[17] J.F. Maier, C.M. Eckert, P.J. Clarkson, Model granularity in engineering design – concepts and framework, Des. Sci. (3) (2017).
[18] R. Eshuis, P. Grefen, Constructing customized process views, Data Knowl. Eng. 64.2 (2008) 419–438.

[19] B. Fatima, K. Shahzad, Process model abstraction: Identifying business significant activities, in: International Conference on Intelligent Technologies and Applications, Springer, Singapore, 2018, pp. 277–288.
[20] N. Wang, S. Sun, D. OuYang, Business process modeling abstraction based on semi-supervised clustering analysis, Bus. Inform. Syst. Eng. 60 (6) (2018) 525–542.
[21] S. Smirnov, H.A. Reijers, M. Weske, From fine-grained to abstract process models: A semantic approach, Inf. Syst. 37 (8) (2012) 784–797.
[22] Business process modeling notation (BPMN), 2014, http://www.omg.org/spec/BPMN/.
[23] M. Chinosi, A. Trombetta, BPMN: An introduction to the standard, Computer StandArds & Interfaces 34 (1) (2012) 124–134.
[24] P. Harmon, C. Wolf, The state of business process management 2016, in: BPTtrends, 2016.
[25] C. Tsagkani, A. Tsalgatidou, Abstracting BPMN models, in: Proceedings of the 19th Panhellenic Conference on Informatics, ACM, 2015, pp. 243–244.
[26] C. Tsagkani, A. Tsalgatidou, Rule-based business process abstraction framework, in: Proceedings of the 6th International Symposium on Business Modeling and Software Design, ScitTePress, 2016, pp. 173–178.
[27] A. Polyvyanyy, S. Smirnov, M. Weske, The triconnected abstraction of process models, in: U. Dayal, J. Eder, J. Koehler, H.A. Reijers (Eds.), Business Process Management. BPM 2009, in: Lecture Notes in Computer Science, vol. 5701, Springer, Berlin, Heidelberg, 2009.
[28] H. Eberle, T. Unger, F. Leymann, Process fragments, in: OTM Confederated International Conferences on the Move to Meaningful Internet Systems, Springer, Berlin, Heidelberg, 2009, pp. 398–405.
[29] D.K. Chiu, S.C. Cheung, S. Till, K. Karlapalem, Q. Li, E. Kafeza, Workflow view driven cross-organizational interoperability in a web service environment, Inform. Technol. Manage. 5.3-4 (2004) 221–250.
[30] Camunda BPMN modeler, 2019, https://demo.bpmn.io/ (accessed 09 2019).
[31] Business process management academic initiative (BPMAI), 2015, https://bpmai.org/BPMAcademicInitiative (accessed 12 2019).
[32] P. Brereton, B. Kitchenham, D. Budgen, Z. Li, Using a protocol template for case study planning, in: 12th International Conference on Evaluation and Assessment in Software Engineering (EASE), Vol. 12, 2008, pp. 1–8.
[33] S.H. Kan, Metrics and Models in Software Quality Engineering, Addison-Wesley Longman Publishing Co. Inc., Boston, MA, USA, 2002.
[34] R.A. Baeza-Yates, B.A. Ribeiro-Neto, Modern Information Retrieval (Vol. 46), ACM press, New York, 1999.
[35] J. Mendling, J. Recker, H.A. Reijers, H. Leopold, An empirical review of the connection between model viewer characteristics and the comprehension of conceptual process models, Inform. Syst. Front. 21 (5) (2019) 1111–1135.
[36] J. Mendling, M. Strembeck, J. Recker, Factors of process model comprehension—findings from a series of experiments, Decis. Support Syst. 53 (1) (2012) 195–206.
[37] R. Petrusel, J. Mendling, H.A. Reijers, How visual cognition influences process model comprehension, Decis. Support Syst. 96 (2017) 1–16.
[38] M. Zimoch, R. Pryss, T. Probst, W. Schlee, M. Reichert, Cognitive insights into business process model comprehension: preliminary results for experienced and inexperienced individuals, in: Enterprise, Business-Process and Information Systems Modeling, Springer, Cham, 2017, pp. 137–152.
[39] M. Winter, R. Pryss, T. Probst, M. Reichert, Towards the applicability of measuring the electrodermal activity in the context of process model comprehension: Feasibility study, Sensors 20 (16) (2020) 4561.
[40] M. Winter, R. Pryss, T. Probst, M. Reichert, Learning to read by learning to write: Evaluation of a serious game to foster business process model comprehension, JMIR Serious Games 8 (1) (2020) e15374.
[41] M. Zimoch, R. Pryss, J. Schobel, M. Reichert, Eye tracking experiments on process model comprehension: lessons learned, in: Enterprise, Business-Process and Information Systems Modeling, Springer, Cham, 2017, pp. 153–168.
[42] A.A. Andaloussi, A. Burattin, T. Slaats, A.C.M. Petersen, T.T. Hildebrandt, B. Weber, Exploring the understandability of a hybrid process design artifact based on DCR graphs, in: Enterprise, Business-Process and Information Systems Modeling, Springer, Cham, 2019, pp. 69–84.
[43] H.A. Reijers, T. Freytag, J. Mendling, A. Eckleder, Syntax highlighting in business process models, Decis. Support Syst. 51 (3) (2011) 339–349.
[44] W. Wang, M. Indulska, S. Sadiq, B. Weber, Effect of linked rules on business process model understanding, in: International Conference on Business Process Management, Springer, Cham, 2017, pp. 200–215.
[45] S. Smirnov, Business Process Model Abstraction (Ph.D. thesis), Business Process Technology, HPI, Potsdam, 2012.
[46] D. Fahland, W.M. Van Der Aalst, Simplifying mined process models: An approach based on unfoldings, in: Business Process Management, Springer, Berlin, Heidelberg, 2011, pp. 362–378.
[47] C.W. Günther, W.M. Van Der Aalst, Fuzzy mining–adaptive process simplification based on multi-perspective metrics, in: Business Process Management, Springer Berlin Heidelberg, 2007, pp. 328–343.
[48] A. Senderovich, A. Shleyfman, M. Weidlich, A. Gal, A. Mandelbaum, P^ 3-folder: Optimal model simplification for improving accuracy in process performance prediction, in: International Conference on Business Process Management 2016, Springer International Publishing, 2016, pp. 418–436.
[49] A. Senderovich, A. Shleyfman, M. Weidlich, A. Gal, A. Mandelbaum, To aggregate or to eliminate? Optimal model simplification for improved process performance prediction, Inf. Syst. 78 (2018) 96–111.
[50] B. Weber, M. Reichert, J. Mendling, H.A. Reijers, Refactoring large process model repositories, Comput. Ind. 62 (5) (2011) 467–486.
[51] D. Chapela-Campa, M. Mucientes, M. Lama, Simplification of complex process models by abstracting infrequent behaviour, in: International Conference on Service-Oriented Computing, Springer, Cham, 2019, pp. 415–430.
[52] N. Tax, N. Sidorova, W.M.P. van der Aalst, Discovering more precise process models from event logs by filtering out chaotic activities, J. Intell. Inf. Syst. 52 (1) (2019) 107–139.
[53] R. Bobrik, M.U. Reichert, T. Bauer, Parameterizable Views for Process Visualization, Tech. Report TR-CTIT-07-37, CTIT, University of Twente, 2007.
[54] J. Cardoso, A. Sheth, J. Miller, J. Arnold, K. Kochut, Quality of service for workflows and web service processes, in: Web Semantics: Science, Services and Agents on the World Wide Web 1.3, 2004, pp. 281–308.
[55] A. Polyvyanyy, Abstraction of Process Specifications, in: Proceedings 4th Ph.D. retreat of the HPI research school on Service-Oriented Systems Engineering Service-Oriented Systems Engineering, 4, 2010.
[56] M. Reichert, Visualizing large business process models: challenges, techniques, applications, in: La Rosa Marcello, P. Soffer (Eds.), Business Process Management Workshops. BPM 2012 International Workshops, Tallinn, Estonia, September 3, 2012. Revised Papers, in: LNBIP, vol. 132, Springer, Berlin Heidelberg, 2012, pp. 725–736.
[57] M. Reichert, J. Kolb, R. Bobrik, T. Bauer, Enabling personalized visualization of large business processes through parameterizable views, in: 27th ACM Symposium on Applied Computing, 9th Enterprise Engineering Track, ACM Press, 2012.
[58] W. Sadiq, M.E. Orlowska, Analyzing process models using graph reduction techniques, Inf. Syst. 25 (2) (2000) 117–134.
[59] V. Pankratius, W. Stucky, A formal foundation for workflow composition, workflow view definition, and workflow normalization based on petri nets, in: Proceedings of the 2nd Asia-Pacific Conference on Conceptual Modelling-Volume 43, Australian Computer Society, Inc., 2005, pp. 79–88.
[60] J. Köpke, J. Eder, M. Künstner, Projections of abstract interorganizational business processes, in: International Conference on Database and Expert Systems Applications, Springer, Cham, 2014, pp. 472–479.
[61] B. Wurm, K. Goel, W. Bandara, M. Rosemann, Design patterns for business process individualization, in: International Conference on Business Process Management, Springer, Cham, 2019, pp. 370–385.
[62] S. Yongchareon, C. Liu, X. Zhao, Uniflexview: A unified framework for consistent construction of BPMN and BPEL process views, Concurr. Comput.: Pract. Exper. 32 (11) (2020) e5646.
[63] W. Khlif, H. Ben-Abdallah, Integrating semantics and structural information for BPMN model refactoring, in: 2015 IEEE/ACIS 14th International Conference on Computer and Information Science (ICIS), IEEE, 2015, pp. 656–660.
[64] M. Kchaou, W. Khlif, F. Gargouri, Temporal, semantic and structural aspects-based transformation rules for refactoring BPMN model, in: ICETE (1), 2019, pp. 133–144.
[65] D.R. Liu, M. Shen, Workflow modeling for virtual processes: an order-preserving process-view approach, Inf. Syst. 28.6 (2003) (2003) 505–532.
[66] S. Mafazi, W. Mayer, G. Grossmann, M. Stumptner, A knowledge-based approach to the configuration of business process model abstractions, in: Int'l Workshop on Knowledge-Intensive Business Processes, 2012, pp. 60–74.
[67] A. Meyer, M. Weske, Data support in process model abstraction, in: Conceptual Modeling, 2012, Springer, 2012, pp. 292–306.
[68] J. Harzmann, A. Meyer, M. Weske, Deciding data object relevance for business process model abstraction, in: International Conference on Conceptual Modeling, Springer, Berlin, Heidelberg, 2013, pp. 121–129.
[69] R. Pérez-Castillo, M. Fernández-Ropero, M. Piattini, Business process model refactoring applying IBUPROFEN. An industrial evaluation, J. Syst. Softw. 147 (2019) 86–103.
[70] M. Fichtner, Parameterizable Process Views in Imperative Process Models (Master Thesis), 2019.
[71] C.E. Mokaddem, H. Sahraoui, E. Syriani, Recommending model refactoring rules from refactoring examples, in: Proceedings of the 21th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, ACM, 2018, pp. 257–266.
[72] K. Kammerer, J. Kolb, M. Reichert, PQL-a descriptive language for querying, abstracting and changing process models, in: Enterprise, Business-Process and Information Systems Modeling, Springer, Cham, 2015, pp. 135–150.