Augmenting Processes with Decision Intelligence: Principles for Integrated Modelling

Article *in* Decision Support Systems · January 2018

DOI: 10.1016/j.dss.2017.12.008

3 authors:

Faruk Hasić
KU Leuven
25 PUBLICATIONS **182** CITATIONS
SEE PROFILE

Johannes De Smedt
The University of Edinburgh
37 PUBLICATIONS **295** CITATIONS
SEE PROFILE

Jan Vanthienen
KU Leuven
327 PUBLICATIONS **7,836** CITATIONS
SEE PROFILE

Augmenting Processes with Decision Intelligence: Principles for Integrated Modelling

Faruk Hasić[a,*], Johannes De Smedt[b], Jan Vanthienen[a]

[a] *Leuven Institute for Research on Information Systems (LIRIS), KU Leuven, Naamsestraat 69 B-3000 Leuven, Belgium*

[b] *Management Science and Business Economics Group, University of Edinburgh Business School, 29 Buccleuch Place, Edinburgh, UK*

Abstract

Until recently decisions were mostly modelled within the process. Such an approach was shown to impair the maintainability, scalability, and flexibility of both processes and decisions. Lately, literature is moving towards a separation of concerns between the process and decision model. Most notably, the introduction of the Decision Model and Notation (DMN) standard provides a suitable solution for filling the void of decision representation. This raises the question whether decisions and processes can easily be separated and consistently integrated. We introduce an integrated way of modelling the process, while providing a decision model which encompasses the process in its entirety, rather than focusing on local decision points only. Specifically, this paper contributes formal definitions for decision models and for the integration of processes and decisions. Additionally, inconsistencies between process and decision models are identified and we remedy those inconsistencies by establishing **Five P**rinciples for integrated **P**rocess and **D**ecision **M**odelling (**5PDM**). The principles are subsequently illustrated and validated on a case of a Belgian accounting company.

Keywords: Decision modelling, DMN, Process modelling, BPMN, Integrated modelling, Separation of concerns

1. Introduction

The prevalence of new works on decision modeling and mining, as witnessed by the vast amount of new works on Decision Model and Notation [1–5], shows an increasing interest in documenting, modelling, and analysing the decision dimension of processes. DMN has two levels that are to be used in conjunction. Firstly, there is the decision requirement level, represented by the Decision Requirement Diagram (DRD), which depicts the requirements of decisions and the dependencies between elements involved in the decision model. Secondly, there is the decision logic level, which presents ways to specify the underlying decision logic. Usually, the decision logic is specified in decision table form. An example of a DRD is given in Figure 1. DMN is designed as a declarative decision language. As a result DMN provides no decision resolution mechanism, as this is left to the invoking context (e.g. a process). The same holds for the processing and storage of outputs and intermediate results. Besides DMN, also the Product Data Model (PDM) [6] is a well-known language to capture the dependencies that exist between decisions and their input in workflows.

*Corresponding author

Email address: faruk.hasic@kuleuven.be (Faruk Hasić)

DMN, however, is more driven by the decision and its rationale compared to PDM, which rather focuses on the data and its impact on the workflow.

Organisations use Business Process Management (BPM) and Decision Management (DM) to analyse, and improve their processes. The new DMN standard has the clear intention to be used in conjunction with Business Process Modeling and Notation (BPMN) [5, 7–10]. Since the introduction of DMN, the general consensus is to model decisions outside processes. BPM is moving towards this *separation of concerns* paradigm [11] by externalising the decisions from the process flow.

The contribution of this paper is fourfold: (1) a formal definition of decision models and their relation to process models is established; (2) a list of inconsistencies between process and decision models is provided based on existing literature and on the formal definitions formulated in this paper; (3) a set of modelling guidelines is instituted to remedy the inconsistencies between process and decision models. The guidelines are contributed in the form of **Five P**rinciples for integrated **P**rocess and **D**ecision **M**odelling (**5PDM**), in analogy with [12]; (4) the proposed modelling principles are applied and tested on a real life industry case.

This paper is structured as follows. In Section 2 the design science approach used in this paper is explained, while Section 3 handles the necessities for integrated modelling and decision modelling. In Section 4 a formalisation of the DMN standard and related constructs is provided which will serve as the basis for the approach of integrated modelling. Section 5 outlines challenges of integration by providing scenarios containing inconsistency concerns, followed by Section 6 which extracts principles for integrated process and decision modelling from the previous sections. In Section 7, the modelling principles are illustrated on a case from industry, and in Section 8 a systematic approach to mitigate inconsistencies is provided. Finally, Section 9 discusses the contributions and future work.

2. Methodology

This paper follows a design science approach [13], structured along three different cycles to obtain an artifact, being the **5PDM**. First of all, the *application domain* and *population* was delineated as practitioners who develop models for integrating decisions into processes for process-aware information systems during the relevance cycle. Next, we have identified the *problem* of *inconsistent* use of *decisions* within *processes* and hence the issues that arise regarding maintainability, scalability, flexibility, understandability and reusability of decisions and processes in Sections 1 and 3. We have argued that these are the *relevant* issues tackled when separating concerns in modelling endeavours through the use of the separation of concerns and Service-Oriented Architecture paradigms in Sections 4 and 5. Based on the previous work of the authors, a literature review, and insights from industry (i.e. the case study environment), it was noted that there are *no suitable guidelines*, and that from previously produced models in research *no streamlined approach* was suggested. Next, an initial set of guidelines, i.e. the proposed *solution artefact*, were built in Section 6, according to examples from practice and research. They were *validated by practitioners*, as illustrated in Section 7, and previous work [5], during the design cycle. Finally, this work aims at *formalising the procedure* to adhere to the guidelines in Section 8 and bringing them to the body of literature on decision and process modelling. Note that these cycles work like cogs, and the *relevance cycle* was influenced both by insights from literature, as well as practice and design iterations, while the *rigor cycle* produced initial findings which were reflected in the design.

3. Why Integrated Decision and Process Modelling?

This section provides a motivation and related work for separating and integrating process and decision models. Additionally, we provide a running example that will be used throughout this paper.

3.1. Motivation and Related Work

In the trend towards integration several situations can be identified. Basic solutions see processes represented using only BPMN, or decisions using only DMN. This approach works only in the most straightforward cases, where no decisions are made during the process, or where only the result of a single decision is needed respectively. Slightly more evolved situations see a complete decision model represented by a single activity in a business process. This approach will only be valid for straightforward processes and decisions. Decisions are often emulated using intricate control flows, which can result in cascading gateways. These hidden decisions must be identified in the process. After identifying and modeling these decisions the resulting model must be integrated consistently with the process model. This insufficient separation of concerns results in maintainability issues [5, 14–16]. In more complex processes several decisions might influence both the flow and the result. Representing these decisions and invoking them correctly in the process is crucial for a proper understanding of the process. However, these more convoluted situations have encountered little consideration in literature.

Numerous works have already dealt with data-aware processes and process consistency regarding data management. Extensions regarding data-awareness in process modelling have been proposed as well. In [17] an ontology-based knowledge-intensive approach is suggested, while [18] proposes an enhancement of declarative process models with DMN logic. Furthermore, works concerning data-aware/coloured Petri Nets are available as well, offering a formally sound approach to data and process integration [19]. However, merely focusing on data fragments is not sufficient to incorporate decision-awareness into processes, which DMN aims to achieve. Furthermore, it has been illustrated how the Decision Model [20], a decision representation similar to DMN, can be used within business process models as well [21] to offload the control flow from embedded decisions. The findings of this paper are also compatible with the Decision Model after a straightforward conversion of its decision and input blocks into DRD constructs.

The decision modelling approaches present in literature often breach the separation of concerns between control and data flow, resulting in spaghetti-like processes, thus negatively influencing maintenance, flexibility, scalability and reusability [5, 7, 14, 15, 22–26]. They do this by hard-coding and fixing the decisions in processes. Consequently, splits and joins in processes are misused to represent typical decision artifacts such as decision tables. Recently, more attention was given to the separation of processes and decision logic, as such an approach is supported by the DMN standard [1] that can be used in conjunction with BPMN [5, 15, 27]. Decoupling decisions and processes to stimulate flexibility, maintenance, and reusability, yet integrating decision and process models is therefore of paramount importance [5, 15, 28].

The separation of concerns has enjoyed plenty of attention, mainly in the domain of software modelling and design [11], but recently it has become an evident trend in BPM as well. This has moved decision management towards the paradigm of Service-Oriented Architecture (SOA), by representing decisions as externalised services. In research several conceptual decision service platforms [23, 29] and ontologies [30] have been proposed. Separation of concerns and SOA offer firm motivation for keeping multi-perspective modelling tasks isolated and founded on a basis which can be used to ensure consistency. The integrated modelling and externalisation was already considered in terms of business rules [31, 32]. With DMN, externalisation of decisions has become

a possibility, since decisions can be encapsulated in separate decision models. These decisions are modelled separately from other concerns, such as processes, and they are implemented as a service which we call *Decision as a Service (DaaS)*. Other information systems, e.g. process-aware information systems, can invoke the decision services from the separate decision layer on demand, i.e. *Decision on Demand (DoD)*. Consequently, a decision model can be invoked and used as a service, adhering to the SOA paradigm and benefiting maintainability, scalability, flexibility, and reusability [5, 7, 8, 11, 15, 23, 24, 26, 28, 31, 33]. This emphasises the necessity for a separate, yet integrated modelling of decisions and processes.

3.2. Running Example

In this paper the integration of decision and process modelling will be elucidated through a case study in a Belgian accounting firm. By law, Belgian accounting firms are obligated to provide a decision model to the public authorities on which they base their decision of accepting or rejecting customers. Figure 1 depicts the decision model for customer acceptance at the firm. `Customer Acceptance` is decided based on the customer's `Risk Level`, which on its turn depends on a `Financial Position Check` and a `Background Check` of the customer. The decision logic is externalised to this model and a complementary process model is provided in Figure 2. This

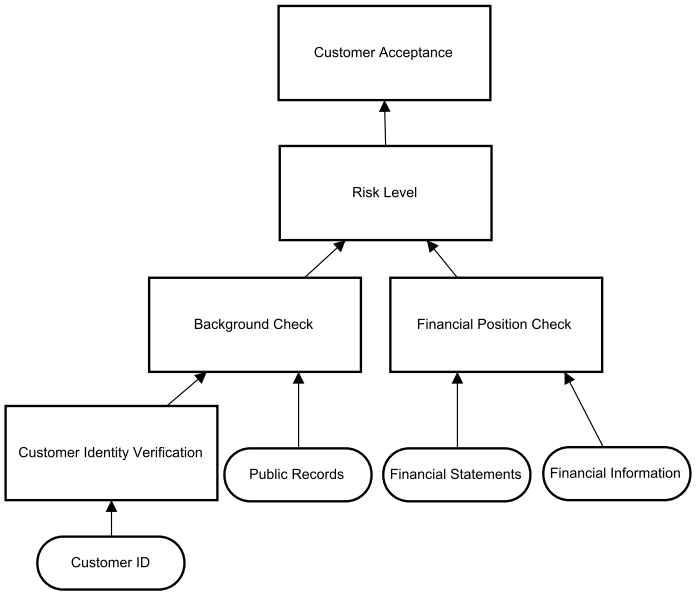

Figure 1: Decision model for customer acceptance at a Belgian accounting firm

process model will have to comply with the decision model in order to correctly fulfill the customer acceptance process, i.e. the process model must be modelled consistently with the decision model. However, Figure 2 contains plenty inconsistencies, as will be discussed in the following sections.

4. Formal Definitions

In this section, a formal basis is given for DMN constructs and for the connection between decisions and processes, which is key for the integration.

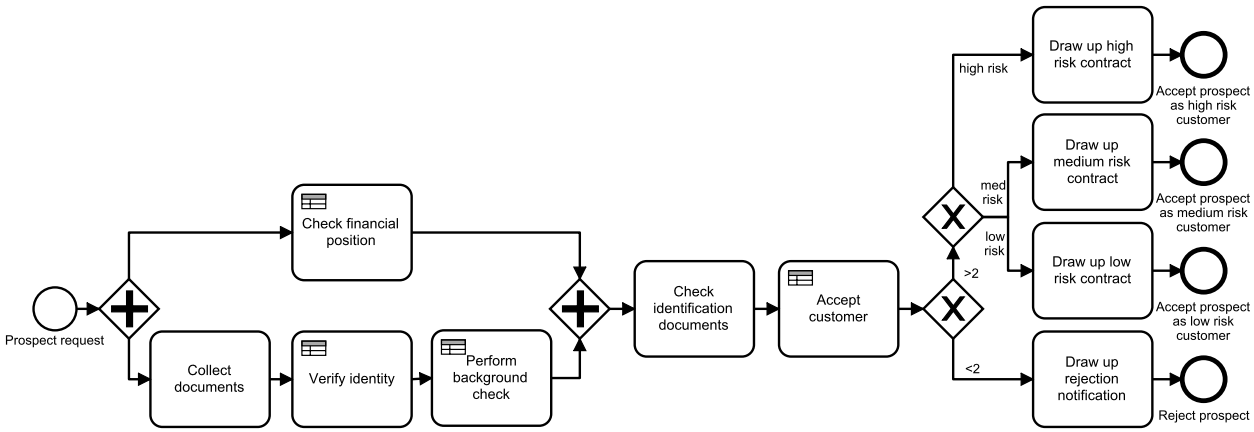

Figure 2: Process model for customer acceptance at a Belgian accounting firm

4.1. Basic DMN Constructs

We formalise DMN to aid us in the consistent integration of processes and decisions. We adopt the definition of *decisions* and *decision requirement diagrams* from [28, 34] and expand them to include subdecisions, interfaces, and invocability.

Definition 1. *A decision requirement diagram DRD is a tuple (D_{dm}, ID, IR) consisting of a finite non-empty set of decision nodes D_{dm}, a finite non-empty set of input data nodes ID, and a finite non-empty set of directed edges IR representing the information requirements such that $IR \subseteq (D_{dm} \cup ID) \times D_{dm}$, and ($D_{dm} \cup ID$, IR) is a directed acyclic graph (DAG).*

The DMN specification allows a DRD to be an incomplete or partial representation of the decision requirements in a decision model. The complete set of requirements R_{DM} is derived from the set of all DRDs. The information contained in this set can be combined into a single DRD representing the entire decision requirements level, i.e. the decision requirement graph (DRG). We extend the notion of a DRG, in such a way that a DRG is a DRD which is self-contained as explained in Definition 2.

Definition 2. *A decision requirement diagram $DRD \in R_{DM}$ is a decision requirement graph DRG if and only if for every decision in the diagram all its modeled requirements, present in at least one diagram in R_{DM}, are also represented in the diagram.*

According to DMN a decision is the logic used to determine an output from a given input. In BPMN a decision is an activity, i.e. the act of using the decision logic. Another common meaning is that a decision is the actual result, which we call the output of a decision. We define a decision using its essential elements.

Definition 3. *A decision $d \in D_{dm}$ is a tuple (I_d, O_d, L), where $I \subseteq ID$ is a set of input symbols, O a set of output symbols and L the decision logic defining the relation between symbols in I_d and symbols in O_d.*

In case of decision tables, a commonly used reasoning construct in decision models, I_d and O_d contain the names of the input, and output elements, respectively, and L is the table itself, i.e. the set of decision rules present in the table. Note that, since a DRD is a DAG, $I_d \cap O_d = \emptyset$. In DRDs these decisions d_i are represented by the decision nodes $D_i \in D_{dm}$. We will use D to refer

to both a decision and its representing node in a DRD. From the definition of DRGs we can derive an important property of decisions. From Definition 2 we know that a DRG contains exactly all information requirements of its decisions. Thus there can only exist one DRG with D as its single top-level decision. We use DRG_D to denote this DRG.

To identify incorrect uses of decisions in process models it is important to know their structure. Decisions are often structured hierarchically.

Definition 4. *A decision D' is a subdecision of decision D if and only if it is part of DRG_D, but not D itself.*

This order of decisions and subdecisions can be defined by using the property that DRDs are directed acyclic graphs, from Definition 1. From this property we know that each DRD has a topological order. The concept of topological orders is closely related to partial orders resulting in Property 1.

Property 1. *The topological order of a DRD induces a partial order $\leq$ on the decisions contained in the DRD.*

When integrating decisions with processes, reducing the coupling between both is important for flexibility and maintainability, as stated by the Service-Oriented paradigm. In this paradigm, decisions are viewed as an external service which exists as a unit decoupled from the process that can be invoked by the process. In [15, 33], the advantages of decision services are elaborated on, showing that the Service-Oriented paradigm enables flexibility, maintainability and scalability. This is achieved by making abstraction of decisions in the process and only connecting the process to the decisions through an interface of the decision service. In order to define the interface, Definition 5 first defines the input requirement set.

Definition 5. *The decision input requirement set $dirs_D$ of a decision D is the set of all sets of input data which are sufficient to invoke D. $dirs_D$ contains sets of input data directly or indirectly required by D. The largest set in $dirs_D$ is the set of all input data nodes for which there exists a path to D in DRG_D. The smallest set in $dirs_D$ is D's input set I_D. $dirs_D$ is constructed inductively by the following rule:*

$I_D \in dirs_D$ and for all $s \in dirs_D$ if there is an $i \in s$ such that $i \in O'_D$ for some D' in DRG_D, then $s \setminus \{i\} \cup I'_D \in dirs_D$.

A decision's interface is the combination of its input requirement set and its output set. Thus, decision interfaces can be defined as in Definition 6.

Definition 6. *The interface IF_D of a decision D is a tuple $(dirs_D, O_D)$, where $dirs_D$ is the input requirement set and O_D the output set of D.*

In DMN, decisions are constrained to have no side-effects so they comply with the principles of a service. As such each decision, with its associated interface, can be seen as a decision service. Consequently only the information available in a decision's interface should be used in the process. Each decision in a DRD has its own output set and these sets should be disjoint. The outputs of subdecisions are identified as *intermediate results*. An output O is an intermediate result of decision D if and only if $O \notin O_D$ and there exists a subdecision D' of D for which $O \in O_{D'}$.

Executing a decision in DMN is referred to as *invoking* the decision. Using the definition of a decision's interface it becomes possible to define when a decision can be invoked. Generally a decision can only be made if all required inputs are available. This is especially important when decisions are invoked in a process. Definition 7 determines the invocability of a decision.

Definition 7. *A decision D is invocable from a set of data elements S if there exists an $s \in dirs_D$ such that $s \subseteq S$. Given at least the values of all data elements in one of the sets in $dirs_D$ the output of D can be determined.*

If a decision is invocable, so are its subdecisions, as stated in Theorem 1:

Theorem 1. *If a decision D is invocable from a set of data objects S, then so are all of its subdecisions.*

Proof for Theorem 1. *Assume D' is a subdecision of D. Since there is a path from D' to D in DRG_D, there is also a path from each of the input requirements of D' to D. Thus, $dirs_{D'} \subseteq dirs_D \subseteq S$.*

We have formalised relevant constructs in the decision model. However, in order to discuss model integration, the decision model must be correlated with the process model. We formalise that connection in what follows.

4.2. The Key to Integration: Decision Activities and Intermediate Results

In this subsection, we propose a typology for different activities used for making decisions in processes. By doing so, we will link the decision model to the process model requiring the decisions. Decisions do not surface solely as the driver of control flow. Rather, they both encompass the routing of cases, i.e., because of decision outcomes that steer toward a certain activity tailored towards supporting its output, and the changes in the data layer of the process as well. For a consistent integration, distinguishing between decision making activities and intermediate results is of paramount importance, especially in the case of separation of concerns, where the decision model is externalised and holistically integrated with the process model. This categorisation is imperative for the identification of types of activities that are representatives of the *decision* model in the *process* model:

Definition 8. *The input and output data variables of process activities in A are defined as follows:*
$I : A \to V$, function assigning activities that deliver input for a variable,
$O : A \to V$, function assigning activities that deliver output for a variable.

This enables the construction of the following activity types:

1. **Operational activities ((no) inputs, no outputs):** do not have any influence on the process' decision dimension and only act as a performer of an action that is tied to that specific place in the control flow. They might serve as the end of a decision. They are provided with the decision inputs needed, which are not used further in the process, $A_o = \{a \in A \mid O(a) = \emptyset, \}$.

2. **Administrative activities (no inputs, outputs):** they introduce decision inputs into the process, $A_a = \{a \in A \mid I(a) = \emptyset \wedge O(a) \neq \emptyset\}$.

3. **Decision activities (inputs, outputs):** serve a decision purpose by transforming inputs into an outcome, $A_d = \{a \in A \mid I(v) \neq \emptyset \wedge O(v) \neq \emptyset\}$.

It holds that $A_a \cup A_o \cup A_d = A$. Typically, the decision points that are used for decision mining in processes are of the decision activity type, but tailored towards deciding which activity should be performed next based on the event labels, instead of encompassing the process in its entirety.

We can now make the connection with decisions and process models:

Definition 9. *A decision in a business process can be defined as follows:*

A decision in a process model, $d^a \in D_{dm}$ is a tuple $(I_{d_a}, O_{d_a}, L_{d_a})$, where $a \subseteq A_d$, $O_{d_a} \subseteq O(a)$, $I_{d_a} \subseteq I(a)$ and $L_{d_a} \subseteq L$.

Now that we have defined the connection between decision activities in the process and the decision in the decision model, we can also define process-decision model consistency. Given a decision model, the process model should ensure that the decisions it invokes are invocable at that point in time, and that the decision results can only be used by the process if they have been invoked explicitly by said process. Hence, keeping in mind the previous definitions, we can define consistency for a process-decision model:

Definition 10. *A process model is consistent with a decision model if and only if the following two conditions hold:*

1. No intermediate results of non-invoked subdecisions are used.

2. Each (sub)decision invoked in the process, must be guaranteed to be invocable at that stage of the process.

5. Integration Scenarios and Inconsistencies

In this section we shortly describe possible integration scenarios and subsequently extrapolate inconsistencies that might occur in those scenarios.

5.1. Integration Scenarios

Outlines of possible process-decision integration scenarios are provided by [5, 28]. We refer to those papers for a full description of possible integration scenarios. Two extreme scenarios occur when there is only one model and hence no need for integration: a scenario describing a simple process without decisions, and a scenario with decisions where no actual process is needed. A third scenario with only one model is possible: both decisions and processes are present, but they are intertwined within the same model, as decisions are hard-coded within the process. This scenario clearly breaches the separation of concerns paradigm. A fourth scenario treats decisions as local concerns, as part of the decision logic pertaining to XOR-gates within the process is separately encapsulated in a decision model. A more challenging scenario exists when, instead of dealing with local decisions, interrelated decisions span over multiple activities of the process. These decisions will influence the process in multiple ways, not only in terms of control flow at gateways. They will shape the flow of the process, the outcome of the process and the process modelling itself, as will be illustrated in the coming sections. The current scenario establishes long-distance dependencies between activities, data, control flow, and decisions in the process model, enabling a decision model to span over multiple activities instead of being contained to a single decision point in the process. Data management to liaise the data generated by activities that feed into decisions will be paramount for the integration of such process and decision models.

5.2. List of Inconsistencies

In this subsection, possible inconsistencies that might arise between the process model and the decision model are described, as the goal is to identify potential inconsistencies and subsequently to alter the process to restore consistency. Furthermore, we formally define the (in)consistencies based on the formal definitions from the previous sections. All the inconsistencies are also directly linked

with the relevant decisions and properties, which all heavily rely on the integration definition, i.e., Definition 10.

***Inconsistency 1 (I1)* - exclusion of decision outcomes:** Not all outcomes from the decisions are included in the process model. Decisions can (re)direct the flow of the process. In an integrated process-decision model, all outcomes of the decision should be represented in the control flow if that decision redirects the process. Modelling all possible decision outcomes in the process is vital for a correct conclusion of the process.

Formally, if a decision D with an output set O_D in the decision model DM leads to a change in control flow in the process, then all elements of O_D should be present in the control flow resulting from the decision activity A_D which links D from DM to the process, i.e., in the state space of the process, the occurrences of $o \in O_D$ lie between the occurrences of A_D and the accepting states. This is an outcome of Definitions 9 and 10.

***Inconsistency 2 (I2)* - inclusion of decision logic in the process:** An inappropriate way to model parts of the decision logic is to embed decision logic in gateways. In cases where a process contains decision logic, the process is incapable of accommodating to changes in the underlying decision model. When changes occur, the process itself needs to be adapted. This occurs when the separation of concerns is not adopted strictly and thus the decision logic is not separated and encapsulated in an independent decision model. Hence, **I2** does not allow for evolution of both models disjointedly.

More precisely, the decision logic L of a decision D should not be part of the process. Rather, L belongs to a decision $D \in D_{dm}$, where D_{dm} is the finite non-empty set of decision nodes belonging to the DRD. Hence, the L is encapsulated in the decision model DM. The process can invoke D through its interface IF_D, which was defined as a tuple $(dirs_D, O_D)$. Through IF_D, the process provides the input requirement set $dirs_D$ needed for the enactment of D, and again through the interface, the decision model returns the output set of D, i.e. O_D. Hence, the process accesses the decision model through an interface and is agnostic of the underlying decision logic. This is an outcome of Definitions 5 and 6.

***Inconsistency 3 (I3)* - exclusion of intermediate results:** Inconsistencies arise when subdecisions are not modelled in the process, despite the fact that the process uses the outcome of those subdecisions. Therefore, certain parts of the flow could be disturbed and render the process inconsistent. Hence, a process model that is consistent with the decision model should ensure that all the subdecisions that contain an intermediate result which is relevant for the process execution are explicitly invoked.

Explicitly, if the process uses the intermediate result $O_{D'}$ of a subdecision D' of higher level decision D, then D' must be represented by a decision activity $A_{D'}$ in the process. As such the process can invoke D' through the subdecision's interface $IF_{D'}$ by providing the necessary input requirements from $dirs_{D'}$, after which $IF_{D'}$ will provide $O_{D'}$ to the process. This is an outcome of Definitions 6 and 10.

***Inconsistency 4 (I4)* - inclusion of process-unrelated subdecisions:** Opposite to **I3**, more decisions than necessary can be included in the process. This occurs when decisions which do not contain relevant intermediate results for the process are modelled within the process. In this case the process becomes unclear and overly complex. Along with that, by modelling every subdecision in the process, the decision enactment or execution steps become fixed. This contradicts the declarative nature of decisions and reduces the flexibility provided by the decision model.

Specifically, if a subdecision D' with an output set $O_{D'}$ in the decision model DM does not lead to a change in control flow in the process, and if the process does not use intermediate result $O_{D'}$ of D', then no decision activity $A_{D'}$ representing subdecision D' should be modelled in the process. This is an outcome of Definitions 7 and 9.

***Inconsistency 5 (I5)* - unsound ordering of decision hierarchy:** This occurs when the order of decision activities in the process model is contradictory to the hierarchy of decisions in the decision model. Consequently, the process cannot function correctly, as decisions are forced to enact without the prerequisite enactment of necessary subdecisions. This order of decisions and subdecisions introduces a partial order as shown in Property 1.

Hence, for two decisions D_1 and D_2 we say $D_2 \leq D_1$ if and only if there is a directed path from D_2 to D_1, i.e. D_2 is a subdecision of D_1. Since decisions are declarative, this partial order does not dictate an execution order, but rather a requirement order. Using this order induced by Property 1 and the result from Theorem 1 we know that if a decision D is invoked in the process any decision D' for which $D' \leq D$ will be invocable when placed directly in front of D.

***Inconsistency 6 (I6)* - exclusion of subdecisions affecting control flow:** Depending on the outcome of certain subdecisions the control flow of the process may be diverted to include additional activities, to generate exceptions or even to lead to process termination. Excluding these subdecisions that have an influence on the control flow of the process leads to process-decision inconsistency. This inconsistency is closely related to **I3**: while **I3** focuses on the exclusion of generated data by certain subdecisions, this inconsistency focuses on the change of control flow.

Explicitly, if a subdecision D' with a decision output set $O_{D'}$ in the decision model DM leads to a change in control flow in the process, then D' must be represented by a decision activity $A_{D'}$ in the process. Additionally, all elements of $O_{D'}$ should be reflected in the control flow following $A_{D'}$ which links the subdecision D' from the decision model DM to the process. This is an outcome of Definitions 7 and 9.

***Inconsistency 7 (I7)* - absence of input data:** Decision activities require prerequisites to function correctly. These prerequisites can be the outcome of certain subdecisions, as illustrated in **I3**, but also take the form of for instance user-generated input data. The inconsistency in this case occurs when the required input data is not available in a process when a certain decision task needs to be executed.

Formally, if the process contains a decision activity A_D referring to a decision D in the decisiom model DM, then the process must make sure that $dirs_D$, i.e. the required input set for the decision D, is available within the process at the time decision activity A_D is executed. Only then can the process invoke decision D through its interface IF_D. This is an outcome of Definitions 5 and 10.

6. Principles for Consistent Integration

In this section we provide a set of principles for integrated process and decision modelling. The principles are derived based on the integration scenarios and the formalisation from the previous sections. The principles state what should be included in a process model and what should be excluded from a process model which is linked to a corresponding decision model. **Five P**rinciples for integrated **P**rocess and **D**ecision **M**odelling (**5PDM**) are derived to support consistency between the two models:

P1. Model all necessary decision output flows. If after enacting the decision, no output flow is dedicated to the decision outcome, the process will prove to be inconsistent. Namely, if a decision outcome that is not modelled in the control flow of the process occurs after the decision enactment, then the process cannot proceed properly.

P2. Do not include decision logic in the process model. Otherwise, maintainability, flexibility and scalability of the process might be impaired. Rather, the underlying decision logic should be externalised, encapsulated in the decision model and invoked as a service by the process.

P3. Model all subdecisions whose intermediate results are used by or are relevant for the process as decision activities in the process (**P3.1**). Subdecisions can only be invoked explicitly if they are rep-

resented in the process model. Using intermediate results of subdecisions that are not represented in the process leads to data inconsistency between the process and decision model. If a certain subdecision directly impacts the process control flow, the decision should be explicitly represented in the process model by a corresponding decision activity (**P3.2**). The decision might steer the control flow of the process towards additional activities, exception handling or even process termination. Excluding such decisions from the process leads to inconsistency. However, do not include more decision activities than necessary. Only the top-level decision and subdecisions relevant for the process enactment in terms of control flow, intermediate results and data management should be represented in the process itself. All other process-irrelevant subdecisions should not be modelled explicitly in the process (**P3.3**). Furthermore, modelling all subdecisions violates the declarative nature of decision modelling and reduces the flexibility provided by the decision model.
P4. Place all relevant decision activities in the correct order within the process. This is paramount for the correct enactment of the process and the underlying decisions, since intermediate results of subdecisions are often needed later in the process to enact higher level decisions. Modelling the subdecision before the higher level decision in the process is therefore vital for a correct management of intermediate results and data and hence for a proper decision and process enactment. Disregarding the decision hierarchy results in an inconsistent process-decision model.

Principles for integrated process-decision modelling (5PDM)
P1: `Include` *all* necessary *decision outcomes* in the process control flow
P2: `Exclude` *decision logic* and cascading XOR-splits from the process
P3: `Include` only *subdecisions* that directly *influence* the process
P3.1: `Include` *subdecisions* whose *results* are used in the process
P3.2: `Include` *subdecisions* that *affect* the process *control flow*
P3.3: `Exclude` *subdecisions* that are or *irrelevant* to the process
P4: `Include` *decision hierarchy* in decision activity modelling
P5: `Include` *input data* and *intermediate results* for decision enactment

Table 1: 5PDM

P5. Model all data objects and intermediate results necessary for a correct process and decision enactment. The decision model depicts the hierarchy of decisions and hence the inputs and intermediate results that are necessary for enacting the decisions that are relevant for the process. If not all required data are represented in the process model, the decision activities requiring that data will not be executed properly and ensuring a sound process enactment becomes a difficulty. Thus, the process must facilitate a correct data management to be able to invoke a decision through its interface.

A short overview of the **5PDM** principles is provided in Table 1.

7. How to integrate decision and process models

In this section, we will address the inconsistencies in the running example of Figure 2 and rework it according to the 5PDM. We have also developed a second example, however, due to page constraints that example is not incorporated in this paper. Rather, the example is available online in Section 3 of a *technical report* of our home institution [35].

7.1. Inclusion of All Decision Outcomes in the Control Flow

A first concern with the process model provided in Figure 2 is that not all possible outcomes of the `Accept Customer` decision activity are represented in the control flow of the process model.

The XOR-split that follows the `Accept Customer` decision activity offers flows for cases where the score is greater than 2 or smaller than 2. However, for a score equal to 2 there is no corresponding path in the process model. This situation corresponds to **I1**.

To remedy this inconsistency, **P1** will be used. the process model should include a flow that supports the decision outcome of a score equalling 2. A solution is presented in Figure 3, where an additional flow exits the XOR-gate and guides the indecisive cases with score equal to 2 towards the executive meeting where the customer acceptance will be resolved.

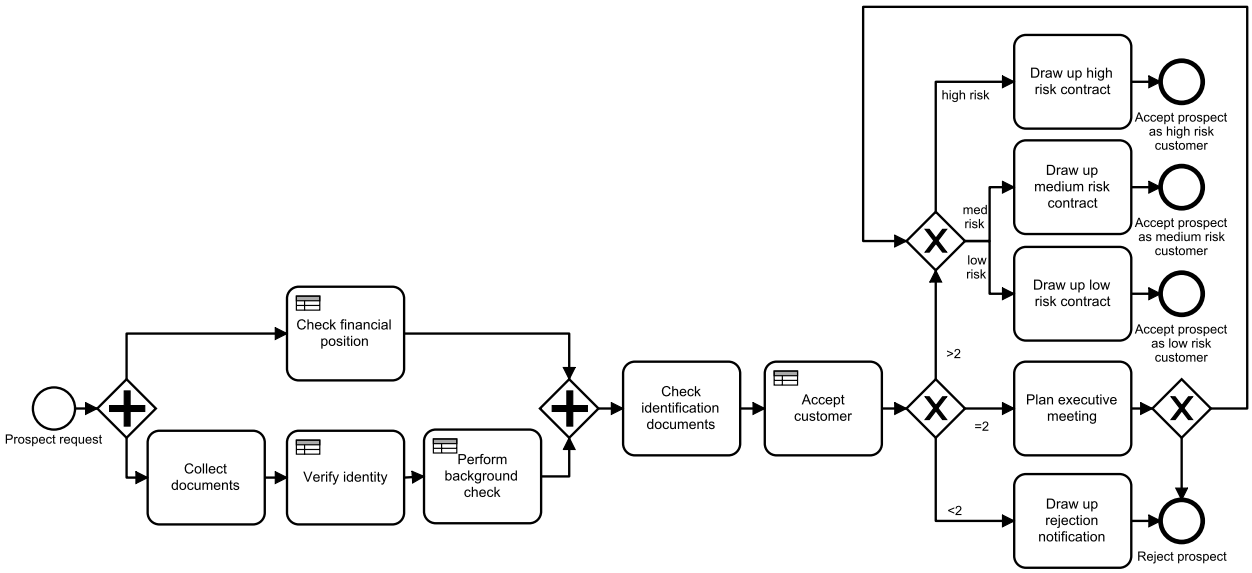

Figure 3: Iteration 1

7.2. Exclusion of Decision Logic from the Process Model

Figure 3 includes all possible decision outcomes at the XOR decision point after `Accept Customer`. However, part of the decision logic is included in the process model, as the business rule that determines the outcome of decision activity `Accept Customer` has been hard-coded in the control flow of the XOR decision point. If the resulting score is smaller than 2, the customer will be rejected; if the score is greater than 2, the customer will eventually be accepted; and if the score equals 2, the executive meeting will address the customer acceptance issue. Suppose that along the way, this business rule for customer acceptance changes and that a score higher than 4 leads to customer acceptance; lower than 4 to customer rejection; and equal to 4 to the executive meeting. This business rule can be easily changed in the decision model. However, once the rule is changed, the process model does not longer consistently comply with the decision model. Hence, in this case, changes in the process model are necessary as well to achieve consistency. This corresponds with **I2** and should be remedied by **P2**. Better is not to include decision logic in the process model and to simply model the outcome of the decision, as done in Figure 4. Instead of hard-coding the scores in the control flow, only the decision outcome of *accept, pending, reject* is modelled in the control flow. The actual decision logic is encapsulated in the decision model, improving the agility and maintainability of the process model.

Figure 3 contains different constructs containing decision logic. After a customer gets accepted, the XOR-gateway following the acceptance resembles a decision tree, where a distinction is made between low risk, medium risk and high risk customers. These are outcomes of the subdecision `Risk Level` in the decision model in Figure 1. They all lead to similar activities of drawing up a

contract. Hence, there is no need to model outcomes of a subdecision in the control flow, as this is another instance of including decision or data flows in the process model. The outcome of the `Risk Level` subdecision is determined in the decision model. Since the top-level decision `Customer Acceptance` is invoked by decision activity `Accept Customer`, the subdecision `Risk Level` is also implicitly invoked and there is no need to additionally model that part of the decision in the control flow. This is again a typical instance of **I2**, as control flows are often misused to represent decision logic. This issue is treated according to **P2** in Figure 4, where the redundant gateway is excluded from the process model.

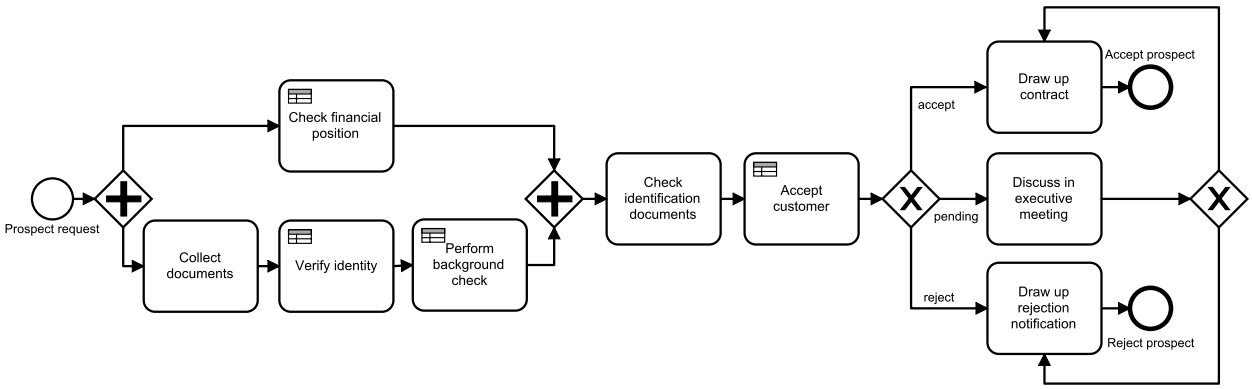

Figure 4: Iteration 2

7.3. Inclusion of Subdecisions Directly Influencing the Process

I3 focuses on the use of intermediate decision results in the process model. In Figure 4 the `Draw up contract` activity prepares a contract for an accepted customer based on the customer's *Risk assessment file*, an intermediate result of the `Risk Level` subdecision in the decision model in Figure 1. However, the process model in Figure 4 does not recognise this intermediate result. Hence, drawing up the contract will not be possible since the *Risk assessment file* is missing. In order to include this file in the process model, the subdecision `Risk Level` that produces it should be modelled as a decision activity in the process. Figure 5 includes the necessary subdecision and intermediate result in accordance with **P3.1**.

Additionally, in Figure 4 the process remains the same regardless the validity of the identity verification executed by activity `Check Identification Documents`. In order to make the decision more reliable, the process could decide to only proceed when the identity is valid. The decision model in Figure 1 provides a subdecision `Customer Identity Verification`. Instead of using the generic activity `Check Identification Documents` as in Figure 4, it is preferential to use a decision activity referring to the subdecision `Customer Identity Verification` in the decision model. Depending on the outcome of this subdecision, i.e. whether the identity documents are valid or not, the control flow of the process can be diverted to include additional activities to ensure that a valid identity is provided before the process can continue. This is achieved in Figure 5 by replacing the generic activity `Check Identification Documents` by the decision activity `Verify Identity`. If the documents are deemed valid, the process can continue; if deemed invalid, an additional activity is introduced that requests valid documentation. In order to remedy **I6**, subdecisions that have an influence on the control flow of the process, must be modelled as decision activities in the process according to **P3.2**.

Figure 4 includes decision activities that do not produce relevant intermediate results for the process at hand and that do not influence the control flow of the process. More precisely, `Check`

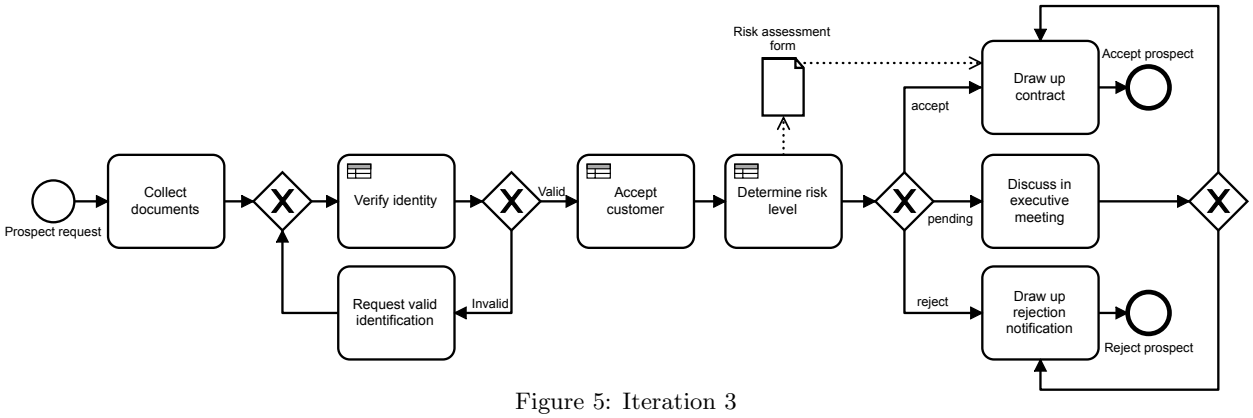

Figure 5: Iteration 3

`Financial Position` and `Perform Background Check` decision activities are represented in the process model. If the decision activity does not refer to the top-level decision or does not provide an intermediate result that is used in the process, or does not influence the control flow of the process, it is not necessary to model that decision activity. Thus, Figure 4 contains **I4**. The obsolete decision activities can be excluded from the process model, as they can eventually be invoked by another higher level decision that is represented in the process model. Figure 5 provides a process model adhering to **P3.3**. Furthermore, representing all the decisions from the decision model in the process model opposes the declarative nature of the decision model, since by modelling all possible decision activities in the process, the decision execution is hard-coded in the process as well. This may lead to unnecessary delays in the process, e.g. the parallel gateway in Figure 4 forces the process to stop until both branches are joined before the process can continue further, while in reality this may not be ideal. In Figure 5, no such issues are present. Thus, Figure 5 conforms to **P3.1**, **P3.2** and **P3.3**, hence conforming to **P3** and only containing relevant subdecisions that influence the process in terms of data, intermediate results, and control flow.

7.4. Inclusion of Decision Requirement Hierarchy

A consistent process model should respect the decision requirement hierarchy provided by the decision model. Figure 5 violates this condition, as the `Determine Risk Level` subdecision activity is located after the top-level decision activity `Accept Customer`. The decision `Customer Acceptance` *requires* the outcome of the subdecision `Risk Level`, and this hierarchy should be respected by the order of the decision activities in the process model. Decision activity `Accept Customer` will require an outcome of decision activity `Determine Risk Level` before the former can be executed successfully. This corresponds to **I5**. To restore the decision requirement order in the process model one incorporates **P4** and simply switches the two decision activities. Figure 6 provides a model that solves this inconsistency.

7.5. Inclusion of Relevant Data and Advanced Data Management

In Section 4.2 we defined decision activities as activities that have an input and an output, with a logical connection between the two. Figure 6 does not respect these definitions as it violates **I7**. In Figure 6 three decision activities are present, yet none of them has the required inputs. Only decision activity `Determine Risk Level` shows an output in the form of a *Risk Assessment File*, while other decision activities don't exhibit any output data object. Hence, Figure 6 shows poor data management and decisions cannot be enacted without the proper data input. Decision activity `Verify Identity` is linked to the subdecision `Customer Identity Verification` in the decision model in Figure 1. The decision model reveals that the *Customer ID* is needed as input

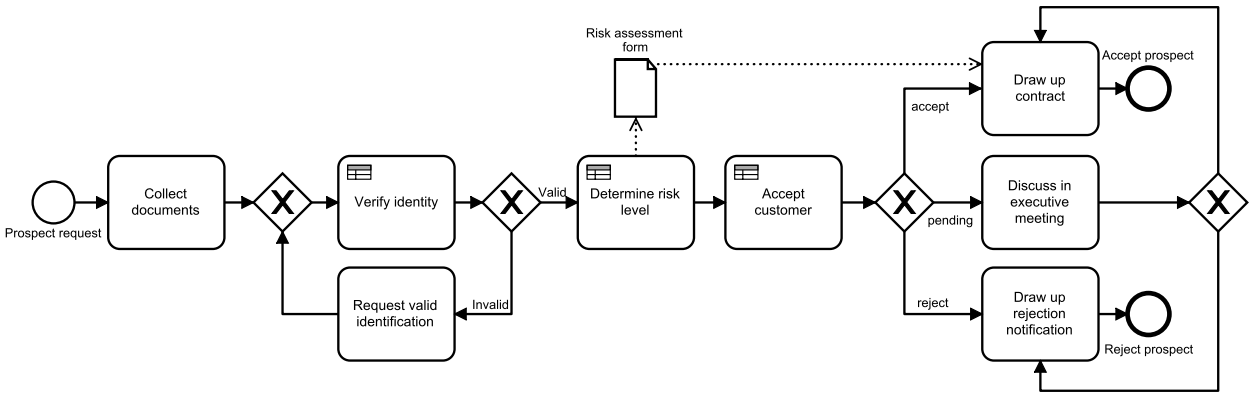

Figure 6: Iteration 4

for this subdecision and the process model in Figure 6 does not provide this indispensable data management.

Figure 7 remedies the data management issues for all decision activities in the process model and links them to the relevant constructs in the decision model, hence conforming to **P5**. The activity classification is key to solving this problem. The decision model in Figure 1 requires input data, namely *Customer ID*, *Financial Statements*, *Financial Information* and *Public Records*. As explained in Section 4.2, input data is produced by administrative activities. Those activities have no input, but do produce output. In the process in Figure 7, the input data needed for a sound enactment are produced by administrative activities `Collect Documents`, `Request Valid Identification` and `Look Up Information`. The relevant data objects are then linked to the (sub)decision activities that exploit them as input data, e.g. decision activity `Verify Identity` uses *Customer ID*, produced by `Collect Documents` or `Request Valid Identification`. Each decision activity also produces an output data object, as previously specified in the definition for decision activities.

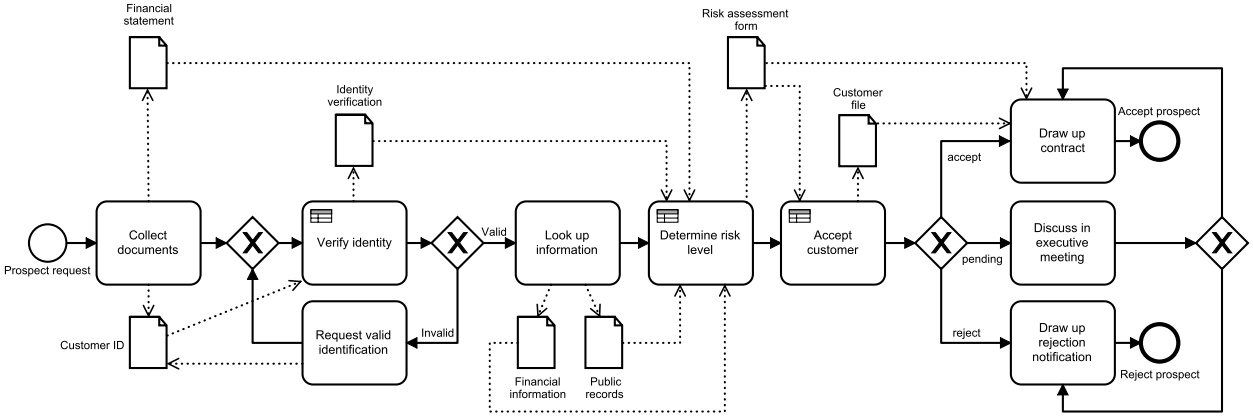

Figure 7: Iteration 5

The intermediate results of subdecision activities such as `Verify Identity` and `Determine Risk Level` are used in higher level decision activities as input, in accordance to the decision model. The intermediate result of decision activity `Verify Identity`, *Identity Verification*, is used in decision activity `Determine Risk Level`, together with the other input data required to enact

the subdecision `Risk Level` in the decision model. Likewise, the intermediate result of decision activity `Determine Risk Level`, *Risk Assessment File*, is adopted as input for the decision activity `Accept Customer`, which corresponds with the top level decision `Customer Acceptance` in the decision model. The outcome of the top level decision is then used to determine whether or not to accept the customer and to draw up a contract if the customer gets accepted. A final activity classification from Section 4.2 refers to operational activities, i.e. activities that might have an input, but that produce no decision output. In Figure 7 `Draw Up Rejection Notification`, `Draw Up Contract` and `Discuss In Executive Meeting` are representatives of the operational activity classification.

8. Resolving Inconsistencies

This section deals with resolving inconsistencies and adhering to the principles of integrated modelling in a systematic way. In [36] soundness is defined to achieve **P1** for isolated decision points. **P1** and **P2** are rather straightforward: make sure that every decision outcome can be handled by the process flow and avoid hard-coding decision logic in processes. Principle **P3.2** is the complement of **P1**, as **P1** suggests that when a decision activity that impacts the control flow is modelled, all its outcomes should be taken into account by the control flow. On the other hand, **P3.2** determines that if a decision impacts the control flow of the process, it should be explicitly modelled as a decision activity in the process.

However, **P3.1**, **P3.3**, **P4** and **P5** require additional attention. Using the formal basis from the previous sections, we defined process-decision model consistency by two conditions in Definition 10:

1. No intermediate results of non-invoked subdecisions are used.

2. Each (sub)decision invoked in the process, must be guaranteed to be invocable at that stage of the process.

The first condition in Definition 10 refers mainly to **P3.1** and **P3.3**, while the second condition acknowledges **P4** and **P5**. In the following subsections we will address each of these necessary conditions.

8.1. Resolving the Use of Intermediate Results

Violations against the first condition for consistency of Definition 10 can be resolved in the following steps:

1. Identify the subdecisions producing intermediate results that are used in the process, both in terms of data objects and control flow.

2. Add these subdecisions to the process model in the form of decision activities in the correct hierarchical order.

3. Do not include the remaining subdecisions into the process.

Note that this solution incorporates **P3.1**, **P3.3** and even **P4**. In the running example of the Belgian Accounting firm case in Section 7, we identified that the `Risk Level` decision produces a *Risk Assessment form* as intermediate result and that said result is needed later on in the process. After identifying a subdecision with its relevant intermediate result, we incorporated the subdecision in the process under the `Determine Risk Level` decision activity and consequently we also took the decision hierarchy into account by placing the `Determine Risk Level` decision activity before

the decision activity `Accept Customer` that represents the top level decision and that requires the outcome of the subdecision `Risk Level`.

The topological order derived from Figure 1 induces that `Customer Identity Verification` $\leq$ `Risk Level` and that `Risk Level` $\leq$ `Customer Acceptance` according to Property 1. Thus, these decisions can be represented in the process model by their respective decision activities as long as they respect the topological order provided in the DRD. The first model adhering to all three steps with regard to resolving intermediate results in Section 7 is the one in Figure 6. Here, the intermediate result of the relevant subdecision is identified and the corresponding decision activity is incorporated in the process, while respecting the topological order of the DRD and while excluding process-irrelevant subdecision activities.

8.2. Resolving Invocability Inconsistencies

The second condition provided in Definition 10 revolves around invocability of (sub)decisions. In order to invoke a certain decision in the process through a decision activity, all relevant data needed for invoking that decision and its subdecisions must be available. Additionally, if subdecisions of the decision that is invoked are modelled within the process by means of decision activities, the intermediate results of the subdecisions must be readily available as well. Note that this condition mainly refers to **P4** and **P5**, i.e. the decision requirement hierarchy will ensure the availability of intermediate results (**P4**) and **P5** additionally assures the presence of other indispensable input data. Hence, violations against the second condition for consistency of Definition 10 can be resolved in three simple steps:

1. Apply the topological hierarchy of decisions from the decision model to the order of modelled decision activities in the process.

2. Ensure that all input data present in the decision model is present in the process model as well, either as external data or as internal process data.

3. For every decision activity in the process make sure that all input data and intermediate results of it's subdecision activities in the process are linked to the decision activity as input data objects.

Consider Figure 7 and decision activity `Determine Risk Level` which represents the decision `Risk Level` from the decision model in Figure 1. According to the decision model, the `Risk Level` decision requires intermediate results from its subdecisions . In order to enact `Financial Position Check` the *Financial Statements* and *Financial Information* input data is needed. In Figure 7, this data is linked to the decision activity as it was generated in the process earlier on by two administrative activities, `Collect Documents` and `Look Up Information`. To enact the second subdecision, `Background Check`, the *Public Records* input data is necessary as well. This too is provided by an administrative activity of the process and linked to the decision activity `Risk Level`. Finally, the intermediate result of the `Customer Identity Verification` subdecision of `Background Check` is required as well. This intermediate result was produced earlier by the process, since the `Verify Identity` decision activity was invoked with the necessary *Customer ID* input provided by an administrative activity. This intermediate result, i.e. *Identity Verification*, is linked as an input data object to decision activity `Determine Risk Level`. Hence, `Determine Risk Level` has all input data and/or intermediate results necessary to invoke the `Risk Level` decision and the process can proceed. Thus, ensuring that all necessary input data and intermediate results for a decision are available before the decision is invoked, resolves the invocability inconsistency.

9. Conclusion and Future Work

This work provides insights and principles for integrated process and decision modelling. While most previous works approach the problem in a straightforward way, i.e. only considering decision points and containing decisions to one specific place in the process, we analyse decisions holistically as they can span over multiple activities and even over the entire process. A DMN formalisation and classification of process activities is provided to connect decisions to processes. Next, based on the formalisation, inconsistencies are revealed. To remedy these inconsistencies, **Five P**rinciples for Integrated **P**rocess and **D**ecision **M**odelling (**5PDM**) were derived. The usefulness of **5PDM** is illustrated through a case from a Belgian Accounting firm. Additionally, a systematic stepwise approach towards consistent integration of processes and decisions was contributed. This approach relies on a sound management of intermediate results of decisions and on correctly matching the information requirements of decisions to process data.

In future endeavours we will investigate how the decision model can further aid in refactoring the process model. Additionally, decision making across distributed processes [37] in cooperative information systems is of particular interest for Internet of Things (IoT) application areas [2].

References

[1] OMG, Decision Model and Notation 1.1 (2016).

[2] F. E. Horita, J. P. de Albuquerque, V. Marchezini, E. M. Mendiondo, Bridging the gap between decision-making and emerging big data sources: an application of a model-based framework to disaster management in brazil, Decision Support Systems 97 (2017) 12–22.

[3] J. M. Perez-Alvarez, M. T. Gomez-Lopez, L. Parody, R. M. Gasca, Process instance query language to include process performance indicators in dmn, in: 20th International Enterprise Distributed Object Computing Workshop (EDOCW), IEEE, 2016, pp. 1–8.

[4] F. Hasić, J. De Smedt, J. Vanthienen, Towards assessing the theoretical complexity of the decision model and notation (dmn), in: Enterprise, Business-Process and Information Systems Modeling, CEUR, 2017, pp. 64–71.

[5] F. Hasić, L. Devadder, M. Dochez, J. Hanot, J. De Smedt, J. Vanthienen, Challenges in refactoring processes to include decision modelling, in: Business Process Management Workshops, Lecture Notes in Computer Science, Springer, 2017.

[6] I. T. P. Vanderfeesten, H. A. Reijers, W. M. P. van der Aalst, Product based workflow support: Dynamic workflow execution, in: CAiSE, Vol. 5074 of Lecture Notes in Computer Science, Springer, 2008, pp. 571–574.

[7] J. Vanthienen, F. Caron, J. De Smedt, Business rules, decisions and processes: five reflections upon living apart together, in: SIGBPS Workshop on Business Processes and Services, 2013, pp. 76–81.

[8] T. Biard, A. Le Mauff, M. Bigand, J.-P. Bourey, Separation of decision modeling from business process modeling using new decision model and notation(dmn) for automating operational decision-making, in: Working Conference on Virtual Enterprises, Springer, 2015, pp. 489–496.

[9] C. Combi, B. Oliboni, A. Zardiniy, F. Zerbato, Seamless design of decision-intensive care pathways, in: IEEE International Conference on Healthcare Informatics (ICHI), IEEE, 2016, pp. 35–45.

[10] R. Ghlala, Z. K. Aouina, L. B. Said, Bpmn decision footprint: Towards decision harmony along bi process, in: International Conference on Information and Software Technologies, Springer, 2016, pp. 269–284.

[11] J. Gordijn, H. Akkermans, H. Van Vliet, Business modelling is not process modelling, in: Conceptual modeling for e-business and the web, Springer, 2000, pp. 40–51.

[12] J. Mendling, H. A. Reijers, W. M. van der Aalst, Seven process modeling guidelines (7pmg), Information and Software Technology 52 (2) (2010) 127–136.

[13] A. R. Hevner, S. T. March, J. Park, S. Ram, Design science in information systems research, MIS Quarterly 28 (1) (2004) 75–105.

[14] W. D. Roover, J. Vanthienen, On the relation between decision structures, tables and processes., in: OTM Workshops, Vol. 7046 of Lecture Notes in Computer Science, Springer, 2011, pp. 591–598.

[15] F. Hasić, J. De Smedt, J. Vanthienen, A service-oriented architecture design of decision-aware information systems: Decision as a service, in: OTM Confederated International Conferences "On the Move to Meaningful Internet Systems", Vol. 10573 of Lecture Notes in Computer Science, Springer, 2017, pp. 353–361.

[16] F. Hasić, J. De Smedt, J. Vanthienen, Developing a modelling and mining framework for integrated processes and decisions, in: "On the Move to Meaningful Internet Systems". OTM 2017 Workshops, Vol. 10697 of Lecture Notes in Computer Science, Springer, 2017.

[17] L. Rao, G. Mansingh, K.-M. Osei-Bryson, Building ontology based knowledge maps to assist business process re-engineering, Decision Support Systems 52 (3) (2012) 577 – 589.

[18] S. Mertens, F. Gailly, G. Poels, Enhancing declarative process models with dmn decision logic, in: Enterprise, Business-Process and Information Systems Modeling, Springer, 2015, pp. 151–165.

[19] E. Serral, J. De Smedt, M. Snoeck, J. Vanthienen, Context-adaptive petri nets: Supporting adaptation for the execution context, Expert Systems with Applications 42 (23) (2015) 9307–9317.

[20] B. Von Halle, L. Goldberg, The decision model: a business logic framework linking business and technology, CRC Press, 2009.

[21] E. Abbasi, K. Abbasi, Business process modeling and decision model integration, in: Information & Communication Technologies, 2013 5th International Conference on, IEEE, 2013, pp. 1–7.

[22] B. Weber, M. Reichert, J. Mendling, H. A. Reijers, Refactoring large process model repositories, Computers in Industry (2011) 467–486.

[23] A. Zarghami, B. Sapkota, M. Z. Eslami, M. van Sinderen, Decision as a service: Separating decision-making from application process logic, in: EDOC, IEEE Computer Society, 2012, pp. 103–112.

[24] H. van der Aa, H. Leopold, K. Batoulis, M. Weske, H. A. Reijers, Integrated process and decision modeling for data-driven processes, in: International Conference on Business Process Management, Springer, 2015, pp. 405–417.

[25] F. Hasić, L. Vanwijck, J. Vanthienen, Integrating processes, cases, and decisions for knowledge-intensive process modelling, in: International Workshop on Practicing Open Enterprise Modeling, CEUR, 2017.

[26] J. Hu, G. Aghakhani, F. Hasić, E. Serral, An evaluation framework for design-time context-adaptation of process modelling languages, in: Practice of Enterprise Modelling (PoEM), Lecture Notes in Computer Science, Springer, 2017.

[27] OMG, Business process model and notation (BPMN) 2.0 (2011).

[28] L. Janssens, E. Bazhenova, J. De Smedt, J. Vanthienen, M. Denecker, Consistent integration of decision (DMN) and process (BPMN) models, in: CAiSE Forum, Vol. 1612 of CEUR Workshop Proceedings, CEUR-WS.org, 2016, pp. 121–128.

[29] A. Bock, H. Kattenstroth, S. Overbeek, Towards a modeling method for supporting the management of organizational decision processes, in: Modellierung, Vol. 225 of LNI, GI, 2014, pp. 49–64.

[30] E. Kornyshova, R. Deneckère, Decision-making ontology for information system engineering, in: ER, Vol. 6412 of Lecture Notes in Computer Science, Springer, 2010, pp. 104–117.

[31] S. Goedertier, J. Vanthienen, Compliant and flexible business processes with business rules, in: Proceedings of the CAISE Workshop on Business Process Modelling, Development, and Support BPMDS, 2006, pp. 94–103.

[32] W. Wei, M. Indulska, S. Sadiq, Guidelines for business rule modeling decisions, Journal of Computer Information Systems (2017) 1–11.

[33] M. Mircea, B. Ghilic-Micu, M. Stoica, An agile architecture framework that leverages the strengths of business intelligence, decision management and service orientation, in: Business Intelligence-Solution for Business Development, InTech, 2012, pp. 15–32.

[34] J. De Smedt, F. Hasić, J. Vanthienen, Towards a holistic discovery of decisions in process-aware information systems, in: Business Process Management, Vol. 10445 of Lecture Notes in Computer Science, Springer, 2017, pp. 183–199.

[35] F. Hasić, J. De Smedt, J. Vanthienen, An Illustration of Five Principles for Integrated Process and Decision Modelling (5PDM), Tech. rep., KU Leuven (2017).

[36] K. Batoulis, M. Weske, Soundness of decision-aware business processes, in: Business Process Management Forum, Springer, 2017, pp. 106–124.

[37] J. Becker, D. Pfeiffer, Solving the conflicts of distributed process modelling: Towards an integrated approach., in: ECIS, 2008, pp. 1555–1568.