Adding Decision Management to Robotic Process Automation

Maximilian Völker(✉), Simon Siegert, and Mathias Weske

Hasso Plattner Institute, University of Potsdam, Potsdam, Germany
simon.siegert@student.hpi.de, {maximilian.voelker,mathias.weske}@hpi.de

Abstract. Robotic Process Automation promises to release employees from repetitive and monotonous work, providing space for creative and innovative tasks. RPA tools provide a wide range of techniques to automate user interactions, including filling forms and copying values between applications. While it is accepted that decisions play an important role in business processes, they are not a first-class citizen in RPA. This paper proposes a framework and a software architecture that integrates decision management into RPA. The work is evaluated by a prototype that introduces Decision Model and Notation (DMN) capabilities to the RPA software tool UiPath by utilizing Camunda's decision engine.

Keywords: Robotic Process Automation · Decision management · RPA design

1 Introduction

Over the last few years, Robotic Process Automation (RPA) gained momentum in research, fueled by its adoption in industry [1]. With the promise of taking digital, repetitive work off the hands of employees, companies are increasingly using RPA to improve the efficiency of their processes while freeing up human labor for more creative and innovative tasks [3,23]. Using software robots, RPA tools imitate the behavior of human users, such as mouse and keyboard inputs, but are also able to, for example, query web services and control applications [14,23]. As artificial intelligence techniques have improved, the scope of RPA applications has continued to expand. Nowadays, RPA can not only imitate interactions but also gains more and more human capabilities, such as text recognition and learning from past executions [7,14,19].

Despite the progress in terms of functionality, the bot development process did not change much. Mostly targeting low-code or no-code developers, many RPA tools offer graphical user interfaces to create new bot workflows using predefined building blocks [14].

In [22] we have observed that RPA vendors often only support *if/else* or *switch* constructs to steer the process. Thus, processes with complex decision logic are either very cumbersome to implement or the potential RPA process may even be discarded for automation, which motivates the need for better decision management in RPA.

Similar issues were encountered for the business process modeling standard BPMN, such that decision-intensive processes led to complex process models with many nested branches [4,24]. In traditional business process management, the Decision Model and Notation (DMN) standard [21] was introduced to resolve the problem by separating the decision logic from the process flow [4]. In this work, we examine whether and how elements of DMN, as a proven modeling standard for decisions, could be integrated into RPA. For this purpose, we analyze the RPA lifecycle and transfer elements from DMN to RPA to benefit from its powerful but still visual representation of decision logic.

After an introduction to DMN and RPA presented in Sect. 2, a motivating example is introduced in Sect. 3. For the integration, the RPA lifecycle is analyzed and related to DMN in Sect. 4, outlining potential synergies, and a generic software architecture is described. In Sect. 5, a prototype is presented that demonstrates the integration of DMN in the RPA vendor UiPath, enabling new use cases that were previously difficult to realize. Additionally, limitations of the approach are discussed. Section 6 summarizes the contribution and provides hints for future research.

2 Preliminaries

In this section, preliminary knowledge about the decision model and notation standard as well as robotic process automation is provided.

2.1 Decision Model and Notation

Modeling complex decisions using control flow often results in large, spaghetti-like models as it, for example, has been reported for the Business Process Model and Notation (BPMN) standard [24]. While such models are capable of correctly representing the decision logic, they are not only difficult to maintain, e.g., when a particular aspect in the decision logic changes, but their complexity also impedes communication using the model [4].

To solve this issue, the Decision Model and Notation (DMN) standard [21] was introduced. It allows to separate the decision logic from the control flow logic in process models represented in BPMN [20,21]. DMN enables the automated evaluation of decisions and can therefore be used in automated business processes. Nevertheless, the standard also focuses on comprehensibility for non-technical users [10].

DMN provides various notation elements for representing highly complex decisions [21]. The central element is the *Decision Table*, which specifies the rules on how to derive the correct decision result from given input values. More specifically, a decision table consists of (i) a set of input parameters required for making the decision, (ii) a set of output variables whose values have to be determined based on the input parameters, as well as (iii) a list of rules that match values or value-ranges of the input parameters and assign the appropriate values to the output variables. So-called hit policies are used to specify how these rules are evaluated, e.g., whether only the first matching rule should be applied or the outputs of all matching rules should be returned.

An example for a decision table is given in the subsequent section in Fig. 3.

The DMN standard comprises additional elements, such as Decision Requirements Graphs, which are not further considered here.

2.2 Robotic Process Automation

While BPMN focuses on larger, often interdepartmental or even cross-organizational processes, the comparably new technology of Robotic Process Automation (RPA) concentrates on local workflows mainly on a single workstation [11]. The main goal of RPA is to automate frequent and rule-based workflows performed by a user on a computer [3,15,23], such as transferring data between different systems, like from an e-mail to a customer-relationship-management program. On the one hand, this is intended to relieve the user of such repetitive, monotonous tasks; on the other hand, it is expected to reduce the error rate and thus increase the overall quality [23]. To perform such automation, so-called RPA bots are utilized, small software clients that imitate the behavior of the user, e.g., by simulating mouse and keyboard interactions or more advanced operations [3,23].

In this context, many RPA software vendors target business users, i.e., automation with RPA should ideally be possible quickly and preferably without any programming knowledge [9,17]. Thus, many providers offer a graphical user interface to create RPA bots by combining predefined "building blocks" and thereby specifying a flow of individual automation operations [14].

The steps to introduce robotic process automation are reflected in the RPA lifecycle introduced by Jimenez-Ramirez et al. [16], which is given in Fig. 1.

The lifecycle enables the governance of entire RPA projects and ensures that the RPA software's performance is increased iteratively.

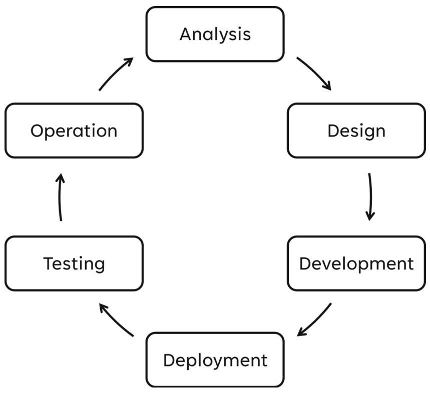


Fig. 1. RPA lifecycle (cf. [16])

The lifecycle starts with a context *analysis* phase to identify suitable processes for automation. Subsequently, the previously selected processes are further specified and modeled for automation in the *design* phase. In the *development* phase, these models are converted into executable programs, which are eventually run in the *deployment* phase. After the deployment, the bots are checked for errors in the *testing* phase, whereas in the subsequent *monitoring* phase, the robots are further operated and maintained. Gained performance metrics and insights into errors are then included in the next iteration of the lifecycle.

2.3 Decisions in RPA

For designing and developing RPA bots there is, unlike for business processes, no standard notation. Consequently, various vendor-specific syntaxes have emerged and therefore the support for different types of decision points in RPA bots differs from vendor to vendor. Table 1 shows the different possibilities for modeling decisions in selected, leading RPA tools, that are supported natively, i.e., are offered as building blocks for bots by default. All investigated providers offer basic *if/else* nodes with two outgoing control flow branches. Also, most of them provide the possibility to prompt the user a dialog with an input field or selection, which could be used to defer decisions to a human worker. More advanced constructs, such as *if/elseif* or *case* statements, i.e., elements with more than two outputs, are already less common and implemented with varying complexity.

Table 1. Native decision capabilities of some RPA providers

Decision Type	UiPath	Blue Prism	Automation Anywhere	Robot Framework
Human dialog	Yes	No	Yes	Yes
If/Else (2 outputs)	Yes	Yes	Yes	Yes
If/ElseIf/Else	No	No	Yes	Yes
Switch/Case	Yes	Yes	No	No

As a result, RPA encounters the same problem as BPMN. Workflows with more complex decisions lead to bloated models that are hard to understand and maintain [22], which is a problem because RPA is supposed to take over rule-based processes and be easy to use. Of course, similarly to script activities in BPMN, many RPA vendors offer blocks for executing custom code in which decision logic could be realized. While this might be feasible for certain use cases, in general, it contradicts the philosophy of RPA to be accessible even without programming knowledge. In addition, the decision logic itself is hidden from business users, who thus cannot gain a holistic view of the robot's function.

3 Motivating Example

In the following, a motivating example is presented to illustrate the challenges of decision-making in RPA models. Suppose a company frequently sends out advertising material in various forms, such as simple postcards, letters, and parcels. Depending on the form and scope of the campaign, different shipping costs accrue, and different departments of the company must be involved. For example, postcards for a national campaign can be directly issued by the applicant for 50 ct per piece, international postcards and domestic letters have to be commissioned via the sales department, and international letters, as well as any parcels, must be arranged with the logistics.

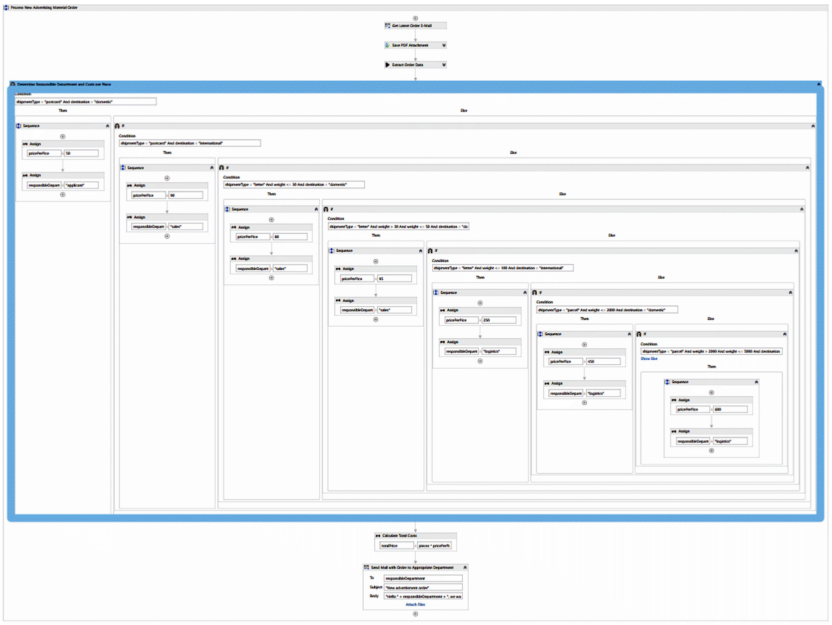


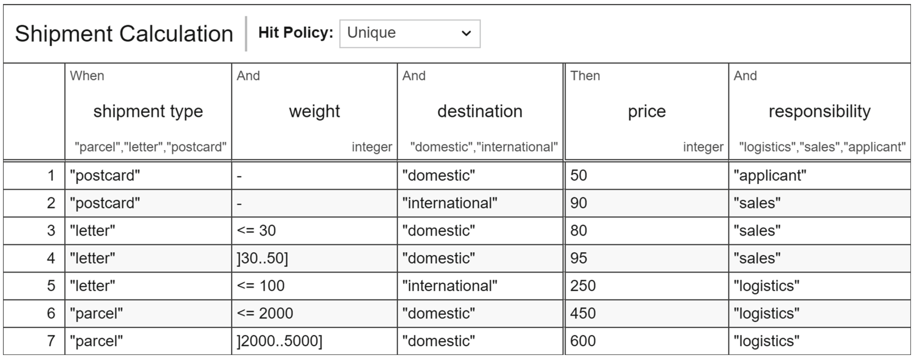
Fig. 2. RPA bot modeled with conventional decision elements (intentionally not readable) (Color figure online)

So far, the company has used a paper-based approach, i.e., the secretary received requests for a certain campaign, calculated the costs, and arranged the handover to the appropriate department. To simplify and streamline the communication process as well as to track requests, the company recently switched to digital documents, such that applications are now sent as PDF files. Still, the processing of the application remained a manual and humdrum task.

With RPA, automating the secretary's workflow described above becomes feasible, as RPA is able to read and send e-mails with attachments and analyze PDF files, especially if they are uniformly formatted, like forms.

However, the resulting RPA process is complex and lengthy as the decision logic to determine the price and the department has to be modeled using the above-described elements, like *if/else*, and the decision part, therefore, takes up a majority of the model as shown in Fig. 2 (blue-framed box). Once created, the bot is of course functional, however, its maintenance is difficult. For example, as soon as the production and shipment costs for the material change, the RPA bot needs to be updated, but the nested structure hampers quick adjustments, not to mention more fundamental changes in the decision logic. Consequently, it is very time-consuming to create and maintain the RPA bot, and it is very likely that the use case is soon discarded for automation with established RPA solutions.

Using DMN, however, the previously described decision can be modeled in a comprehensible and compact way, as shown in Fig. 3. Here, the individual rules



Shipment Calculation | Hit Policy: Unique

	When	And	And	Then	And
	shipment type	weight	destination	price	responsibility
	"parcel","letter","postcard"	integer	"domestic","international"	integer	"logistics","sales","applicant"
1	"postcard"	-	"domestic"	50	"applicant"
2	"postcard"	-	"international"	90	"sales"
3	"letter"	<= 30	"domestic"	80	"sales"
4	"letter"	]30..50]	"domestic"	95	"sales"
5	"letter"	<= 100	"international"	250	"logistics"
6	"parcel"	<= 2000	"domestic"	450	"logistics"
7	"parcel"	]2000..5000]	"domestic"	600	"logistics"

Fig. 3. Illustrative decision table for determining shipment costs and the responsible department in the example

for determining the correct costs and the department are defined, for example, that internationally sent postcards of any weight cost 80 ct per piece and must be ordered by the sales department. If now the RPA bot could make the decision using the decision table instead of using the control flow, the model, as well as the maintenance, could be facilitated.

4 Integration Concepts

In this section, the feasibility of integrating DMN, with focus on decision tables, within RPA is explored. For this, each phase of the RPA lifecycle, as introduced in Sect. 2.2, is examined for potential barriers and problems, and solution concepts are presented, with particular focus on the prominent phases of design and development as well as the execution time.

For the *analysis* phase, i.e., the selection of processes that are suitable for automation with RPA, different frameworks were proposed. In general, RPA is found especially useful for less complex processes to allow for short implementation times [23]. In terms of decision complexity, the integration of DMN can increase the number of suitable processes, since decisions that previously had to be laboriously modeled using control flow elements can then be created independently using DMN. Of course, not all decisions are suitable for automation with RPA and DMN. DMN is mainly suitable for modeling and making operational decisions that are well-defined, frequently executed, and rather have a local and short impact [6]. However, these requirements fit well with the characteristics of RPA processes, such as a high volume and degree of standardization, and the "digitized structured data input" [23], important for an automated evaluation of decisions.

4.1 Design

After the selection of a suitable process, the *design* phase involves creating a visual RPA process model that defines the RPA agent's relevant activities, structure, and data flow [16]. These models also enable communication and exchange about the behavior that the RPA bot should exhibit.

The overall goal of integrating DMN into RPA is to separate the decision logic from the control flow. This separation is therefore particularly apparent and significant in the design phase. The logic for decision-making, previously defined using the available bot building blocks such as if/else, should now be extracted into a single RPA activity dedicated to decision-making by evaluating a corresponding decision table.

In general, RPA bots are created in an RPA vendor-specific model notation that provides the activities available for automation and allows selecting and arranging them in a process-like sequence. But, as mentioned before, there is no standardized modeling language for RPA. Instead, each provider of RPA tools maintains its specific solution of a graphical or textual notation to represent the model. However, it is recommended to prefer intuitive visual modeling tools, as they do not require specific IT development skills and thus make the creation of RPA process models accessible to domain experts [12]. For the integration of DMN in RPA, graphical models are primarily suitable since DMN allows decisions to be represented by graphical elements and aims at being accessible to non-IT users as well [21].

For creating RPA models in the respective model notation, several ways are conceivable. Recent approaches suggest mining or learning RPA bots from past executions (i.e., [2,13]). However, these approaches are currently at the beginning of their development and are rarely adopted in industry tools. Hence, they will not be considered further here.

Consequently, the traditional manual modeling of RPA bots is still the predominant way. It involves human workers, domain experts, as well as technical experts [23]. The integration of DMN benefits from this setting, as all stakeholders can participate in the modeling process, and thus all decision-relevant factors can be considered due to the broad circle of participants. Decision activities can easily be added to the RPA process by hand just as any other type of activity is added to the process.

Besides the manual modeling, most RPA tools offer a recording mode that tracks all interactions that a user performs on a computer system [18], which is a quick way to capture an RPA process [12]. This method, called screen recording, uses the observed user interactions to create a model that the RPA bot can then repeat exactly. However, screen recordings can only capture one execution path, i.e., a case-specific, linear workflow performed by a human user that does not include any choices regarding control flow or data. Therefore, this model creation method is suitable for repetitive processes without variation but not for processes with extensive decisions.

At the current state, the conventional modeling tends to be the most robust way to obtain an RPA process model with decision points. To enable better decision management for the screen recording approach as well, the recording functionalities would need to be adapted to support alternative execution branches.

In addition to the RPA model, the decision logic must now also be modeled, since, as already mentioned, the integration of DMN into RPA aims at separating the decision logic from the control flow. Here, the separation facilitates the communication regarding the decision logic, since it is not covered in the RPA bot model but explicitly represented using the DMN standard, enabling the collaboration of both business users and IT experts.

Similar to the RPA process, the modeling of the decision logic is mostly done manually. However, there are also approaches for mining decision logic from event logs [4,8]. While this could result in beneficial synergies with the previously mentioned mining methods of RPA models, it is still a complex challenge to extract both the process and decisions together from an event log [10] and requires more research.

Rather than testing robots only in their execution environment [16], the proposed integration enables initial verifications of RPA applications already at design-time using an existing formal property for decisions in business processes, decision soundness [5]. Based on decisions defined in DMN decision tables, asserting decision soundness increases the quality of RPA bots. As a result, run-time problems such as deadlocks will not occur. Decision soundness is based on the following criteria [5]:

- *Table Completeness:* any combination of inputs can be assigned to an output. E.g., the table in Fig. 3 is not complete as not every weight can be sent and the combination of *parcel* and *international* is not covered.
- *Output Coverage:* the process can handle all outputs of the decision. E.g., the bot can handle all possible values for the responsible department.
- *Dead Branch Absence:* any branch of the process flow after the decision point is reachable. E.g., check that there is no control flow branch in the bot for handling a value for the responsibility (like *promotion*), which was not defined in the table and could therefore never be reached.

While it is conceivable that these criteria could be checked in the context of RPA, it is, due to a lacking standard, heavily dependent on the chosen RPA vendor and requires further research. Nevertheless, such a formal verification at design-time could prevent avoidable errors during execution.

4.2 Development

The models created in the design phase are converted into executable code during the *development* phase. For the integration of DMN in RPA, additional requirements for the RPA tool's infrastructure must be met here, such as creating, storing, and evaluating decision tables.

DMN models are typically created within a separate modeling software [10]. To enable the handling of DMN decisions within RPA, either a DMN modeling tool needs to be developed, or an existing one needs to be accessed from the RPA software architecture.

As now separate models for the decisions are created, these models must be managed and stored next to the RPA bot models so that they are accessible from the RPA system architecture. A decision model repository provides this functionality. Within this, previously created decision models can be accessed, and ideally also versioned, allowing for updates and rollbacks. Also, a repository might be created as a central component for models which facilitates the reuse of decision tables within different RPA agents.

4.3 Deployment, Testing, and Operation

At run-time, not only the RPA bots must be enacted, but also decisions must be evaluated as soon as a bot reaches a decision point. Therefore, a so-called decision engine is required that can evaluate decision tables.

Evaluation means that for given input variables the corresponding output is calculated according to the rules in the table. For this purpose, the decision table must be parsed and processed at the software robot's run-time.

In general, for the evaluation of decision tables in RPA, two alternatives are conceivable, either a local embedding of a decision engine within the RPA tool or the connection to an external decision service. When the decision engine is directly embedded in the RPA software, no further, external dependencies are needed and decisions are created and evaluated locally. However, it requires the vendor to implement DMN functionalities that might already be present in potentially used BPMS systems.

Therefore, another approach is to outsource the modeling and evaluation of decisions to an external decision service. Such decision services usually already provide a modeling tool for defining decisions and a network interface through which third-party software can request their evaluation. To enable the processing of decision tables in RPA, the bot must only be able to connect to this interface to provide the required input values and obtain the decision's outputs. As the decisions are not defined within the RPA tool anymore, unlike the local embedding, this external approach facilitates sharing the same decision logic between RPA bots or even business processes of the company.

If the second approach is chosen, the external decision engine needs to be available beginning with the *deployment* phase of the RPA bot to ensure a connection can be established when required.

For the *testing* and *operation* phases, when the RPA bots are actually executed, the procedure for evaluating a decision, independent of an external or internal decision engine, is described in more detail in Sect. 4.4 (cf. Fig. 5).

In the *testing* phase, special attention should be paid to the decision points to ensure that required input data is available and also in the correct format so that the decisions can be evaluated as planned. During *operation*, where the performance metrics of the RPA bot are measured [16], additional performance indicators can now be derived, such as the distributions of decision outcomes, e.g., what share international letters account for in the example, which can be used to further improve the process in the subsequent passes of the lifecycle.

4.4 Generic Architecture

So far, two different approaches for integrating a DMN-based decision service into RPA were discussed, either directly embedded in the RPA bot or by using an external service.

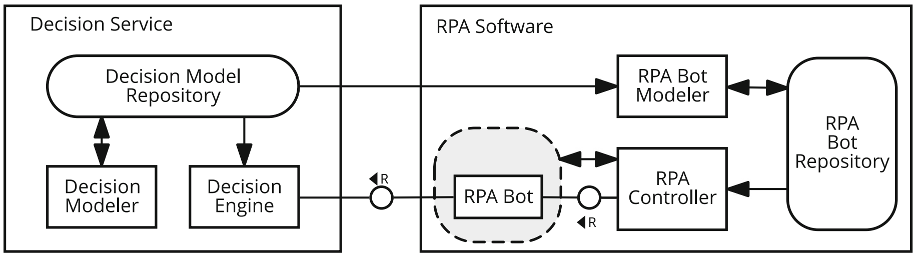


Fig. 4. Architecture with an external decision service

In Fig. 4, a conceivable, generic architecture for the external approach is given. First, at design-time, the decision logic is defined using the *decision modeler* of the external decision service and subsequently saved in its *decision model repository*. Now, whenever a new *RPA bot* is created in the RPA software, a DMN decision point can be added to the bot's workflow, provided that such a DMN activity is available in the RPA tool. This DMN activity is then linked to a decision model stored in the decision model repository. Additionally, it requires configuration of variables that should be passed from the bot to the decision service for evaluation, i.e., values available in the bot need to be mapped to input values required by the decision.

At run-time, the two components communicate as shown in Fig. 5. As soon as a bot, started and operated by the *RPA controller*, reaches a decision point, it requests the *decision engine* of the external decision service using the linked decision identifier and provides the required decision variables as input as configured at design-time. The decision engine then requests the decision table from the decision model repository using the identifier and subsequently calculates the decision result based on the input data and the decision table. The output of the decision (*decisionResult*) is then returned to the bot so that the RPA process can continue accordingly and use the decision result.

When using a locally embedded decision engine, a similar flow of communication would be applicable. With regard to the architecture, however, the RPA software would need to be extended by the components required for providing DMN capabilities, i.e., a modeler (if not integrated in the bot modeler), an engine for evaluating decisions, and a repository for storing the decision models, to substitute the decision service.

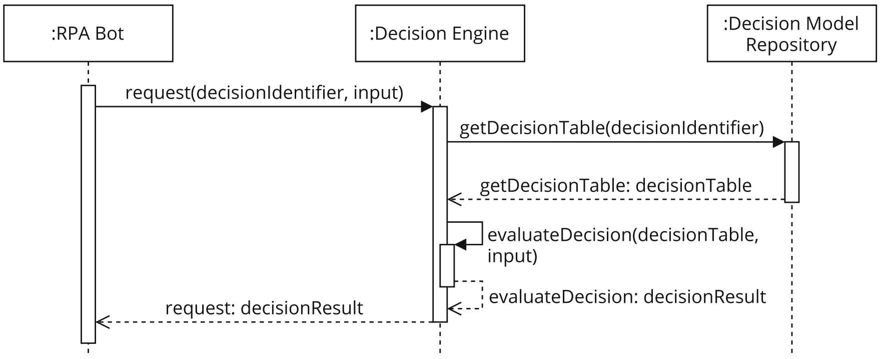

:RPA Bot
:Decision Engine
:Decision Model Repository
request(decisionIdentifier, input)
getDecisionTable(decisionIdentifier)
getDecisionTable: decisionTable
evaluateDecision(decisionTable, input)
evaluateDecision: decisionResult
request: decisionResult

Fig. 5. Communication between RPA bot, decision engine, and decision model repository

5 Evaluation

The analysis of the lifecycle has shown that DMN and its decision tables, can, with some adjustments, be integrated into RPA. In this section, a potential realization of the integration, based on the generic architecture introduced in Sect. 4.4, is presented.

5.1 Proof of Concept Prototype

To demonstrate the feasibility of the proposed approach, we implemented[1] a DMN activity for the RPA tool UiPath[2] using the external decision service approach. For the decision service, Camunda[3] with its open-source modeler and decision engine is used.

The new DMN activity is available to the RPA bot creator as a normal building block and can be added to the workflow as usual, similar to the business rule task available in BPMN. When using the activity, the creator has to specify the internet address of the decision service and the identifier of the decision that should be evaluated (provided by the decision model repository). Furthermore, the variables of the RPA bot that should be passed to the decision engine at run-time must be provided, as well as the variables in which the output of the decision should be saved.

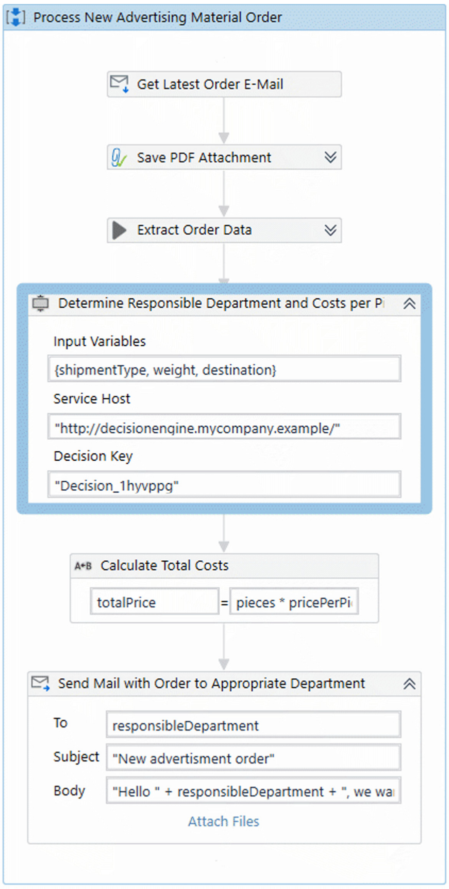
At execution-time, the activity requests the evaluation of the configured decision and supplies the current data stored in the variables to the Camunda engine. After evaluation, the result is interpreted and provided to the RPA bot in the specified variables, which can then be used in the subsequent flow.

[1] The prototype is open source and can be found on GitHub: https://github.com/bptlab/rpa-dmn-operation.
[2] https://www.uipath.com/.
[3] https://camunda.com.

In Fig. 6, the same process as described in Sect. 3 is modeled again, but now using the prototype. The new activity, highlighted by the blue frame, replaces the formerly required extensive decision logic. Comparing Fig. 2 and Fig. 6, it becomes apparent that the new model comprises significantly fewer elements than before and exhibits a considerably reduced nesting level.

In the example, the activity calls the decision engine to evaluate the decision table given in Fig. 3. The output values of the decision, *pricePerPiece* and *responsibleDepartment*, are used subsequently to calculate the total costs and notify the appropriate departments. But the values could, for example, also be used to trigger different control flows depending on the *responsibleDepartment*, e.g., by using the *switch* statement.

By using the external decision service, i.e., a centralized solution, decisions can be reused in other bots as well, or existing decisions already used in business processes become available for use in RPA bots. Furthermore, this eases the maintenance of decisions, as the decision logic only has to be updated in one central place, instead of in all bots separately.


Process New Advertising Material Order

Get Latest Order E-Mail

Save PDF Attachment

Extract Order Data

Determine Responsible Department and Costs per Pi

Input Variables: {shipmentType, weight, destination}

Service Host: "http://decisionengine.mycompany.example/"

Decision Key: "Decision_1hyvppg"

Calculate Total Costs

totalPrice = pieces * pricePerPi

Send Mail with Order to Appropriate Department

To: responsibleDepartment

Subject: "New advertisment order"

Body: "Hello " + responsibleDepartment + ", we wa

Attach Files

Fig. 6. Same RPA bot as shown in Fig. 2, but now modeled using the new DMN activity

5.2 Limitations

The presented prototype already enables RPA developers to separate decision logic and control flow. However, in the chosen approach, an external dependency is introduced. In this case, the decision logic is neither modeled nor evaluated within the RPA software, but relies on third-party software. This could be mitigated by directly integrating DMN capabilities into the RPA tool. This way, decisions could be modeled in the same tool as the RPA bot.

Overall, the integration of DMN into RPA not only increases the number of potential use cases, but also the complexity. Robotic process automation thrives on being easily accessible and quickly employed, without the need for extensive training. With DMN and its decision tables, another modeling standard must be mastered if this extension is to be used. However, especially for companies already employing BPMN, the DMN standard might already be familiar.

Furthermore, this approach is limited to data-based and rule-based decisions, as it inherits the limitations of DMN. Therefore, decisions that, for example, are of strategic nature or require human intuition, cannot be covered. Additionally, like in BPMN, the decision task itself does not branch the control flow, but, based on the decision result, the branching must still be modeled in the RPA bot using the available concepts. Nevertheless, encapsulating the decision-making process already reduces the complexity of the model to some extent.

6 Conclusion

Even though RPA promises to take over rule-based and routine tasks, the rudimentary support for making decisions in workflows may be an exclusion criterion for many decision-intensive processes. Future combinations with artificial intelligence are also expected to provide opportunities for improved and intelligent decision-making [7,23,25], such as learning decisions from past executions. However, it is unlikely that they will completely replace manual modeling and no longer require human intervention.

In this paper, we examined the integration of DMN, a standard for modeling and evaluating decisions, into RPA to address this limitation of current tools and analyzed the RPA lifecycle accordingly. Furthermore, an implementation for an RPA software was presented that allows bot creators to embed DMN decision points in bot workflows and subsequently use the decision result for further actions.

The integration of DMN in RPA offers several benefits. The size of bot models decreases as the decision logic does not have to be realized using control flow elements but is encapsulated in a decision task. This not only facilitates the modeling process itself, but also ensures better maintainability later, as the control flow logic and decision logic can be updated independently. In addition, especially if BPMN and DMN are already in use, it allows reusing decision logic in other bots or business processes, thus having a central place for decision logic. Overall, it may further increase the adoption of RPA, as the barriers for automating workflows with complex, data-based decision logic are lowered.

So far, the approach requires the decision tables to be crafted manually. However, the use of already existing approaches for mining decision logic from data could be evaluated further in the future. This would coincide with the recent endeavors to mine RPA bots from logs. Furthermore, we concentrated on decision tables in this work, but DMN provides more advanced concepts for decision management, such as decision requirements graphs, that could be considered in the future. Other interesting points for future research are checks for correctness or soundness of RPA bots in conjunction with DMN activities, as it has been done for BPMN and DMN. This is especially important since RPA bots are usually not tested in a separate environment, but are directly deployed to the live systems.

References

1. van der Aalst, W.M.P., Bichler, M., Heinzl, A.: Robotic process automation. Bus. Inf. Sys. Eng. **60**(4), 269–272 (2018). https://doi.org/10.1007/s12599-018-0542-4
2. Agostinelli, S., Lupia, M., Marrella, A., Mecella, M.: Automated generation of executable RPA scripts from user interface logs. In: Asatiani, A., et al. (eds.) BPM Blockchain and RPA Forum 2020. LNBIP, vol. 393, pp. 116–131. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58779-6_8
3. Aguirre, S., Rodriguez, A.: Automation of a business process using robotic process automation (RPA): a case study. In: Figueroa-García, J.C., López-Santana, E.R., Villa-Ramírez, J.L., Ferro-Escobar, R. (eds.) WEA 2017. CCIS, vol. 742, pp. 65–71. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66963-2_7
4. Batoulis, K., Meyer, A., Bazhenova, E., Decker, G., Weske, M.: Extracting decision logic from process models. In: Zdravkovic, J., Kirikova, M., Johannesson, P. (eds.) CAiSE 2015. LNCS, vol. 9097, pp. 349–366. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19069-3_22
5. Batoulis, K., Weske, M.: Soundness of decision-aware business processes. In: Carmona, J., Engels, G., Kumar, A. (eds.) BPM Forum 2017. LNBIP, vol. 297, pp. 106–124. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-65015-9_7
6. Biard, T., Le Mauff, A., Bigand, M., Bourey, J.-P.: Separation of decision modeling from business process modeling using new "decision model and notation" (DMN) for automating operational decision-making. In: Camarinha-Matos, L.M., Bénaben, F., Picard, W. (eds.) PRO-VE 2015. IAICT, vol. 463, pp. 489–496. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24141-8_45
7. Chakraborti, T., et al.: From robotic process automation to intelligent process automation. In: Asatiani, A., et al. (eds.) BPM Blockchain and RPA Forum 2020. LNBIP, vol. 393, pp. 215–228. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58779-6_15
8. De Smedt, J., Hasić, F., vanden Broucke, S.K.L.M., Vanthienen, J.: Towards a holistic discovery of decisions in process-aware information systems. In: Carmona, J., Engels, G., Kumar, A. (eds.) BPM 2017. LNCS, vol. 10445, pp. 183–199. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-65000-5_11
9. Enriquez, J.G., Jimenez-Ramirez, A., Dominguez-Mayo, F.J., Garcia-Garcia, J.A.: Robotic process automation: a scientific and industrial systematic mapping study. IEEE Access **8**, 39113–39129 (2020)
10. Figl, K., Mendling, J., Tokdemir, G., Vanthienen, J.: What we know and what we do not know about DMN. EMISAJ **13**(2), 1–16 (2018)
11. Flechsig, C., Lohmer, J., Lasch, R.: Realizing the full potential of robotic process automation through a combination with BPM. In: Bierwirth, C., Kirschstein, T., Sackmann, D. (eds.) Logistics Management. LNL, pp. 104–119. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-29821-0_8
12. Galusha, B.: Considering RPA? Ask smart questions for long-term success. Database Trends Appl. **31**, 44–45 (2017)
13. Gao, J., van Zelst, S.J., Lu, X., van der Aalst, W.M.P.: Automated robotic process automation: a self-learning approach. In: Panetto, H., Debruyne, C., Hepp, M., Lewis, D., Ardagna, C.A., Meersman, R. (eds.) OTM 2019. LNCS, vol. 11877, pp. 95–112. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-33246-4_6
14. Hofmann, P., Samp, C., Urbach, N.: Robotic process automation. Electron. Mark. **30**(1), 99–106 (2019). https://doi.org/10.1007/s12525-019-00365-8

15. Ivančić, L., Suša Vugec, D., Bosilj Vukšić, V.: Robotic process automation: systematic literature review. In: Di Ciccio, C., Staples, M., et al. (eds.) BPM 2019 Blockchain and CEE Forum. LNBIP, vol. 361, pp. 280–295. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30429-4_19
16. Jimenez-Ramirez, A., Reijers, H.A., Barba, I., Del Valle, C.: A method to improve the early stages of the robotic process automation lifecycle. In: Giorgini, P., Weber, B. (eds.) CAiSE 2019. LNCS, vol. 11483, pp. 446–461. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-21290-2_28
17. Lacity, M.C., Willcocks, L.P.: A new approach to automating services. MIT Sloan Manage. Rev. Fall **58**, 41–49 (2017)
18. Leno, V.: Multi-perspective process model discovery for robotic process automation. In: CAiSE 2018 Doctoral Consortium. CEUR-WS.org (2018)
19. Martínez-Rojas, A., Barba, I., Enríquez, J.G.: Towards a taxonomy of cognitive RPA components. In: Asatiani, A., García, J.M., Helander, N., Jiménez-Ramírez, A., Koschmider, A., Mendling, J., Meroni, G., Reijers, H.A. (eds.) BPM Blockchain and RPA Forum 2020. LNBIP, vol. 393, pp. 161–175. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58779-6_11
20. Object Management Group: Business Process Model and Notation (BPMN) (2014). https://www.omg.org/spec/BPMN/
21. Object Management Group: Decision Model and Notation (DMN) (2021). https://www.omg.org/spec/DMN
22. Siegert, S., Völker, M.: Towards decision management for robotic process automation. In: ZEUS 2021, pp. 9–13. CEUR-WS.org (2021)
23. Syed, R., et al.: Robotic process automation: contemporary themes and challenges. Comput. Ind. **115**, 103162 (2020)
24. van der Aa, H., Leopold, H., Batoulis, K., Weske, M., Reijers, H.A.: Integrated process and decision modeling for data-driven processes. In: Reichert, M., Reijers, H.A. (eds.) BPM Workshops 2015. LNBIP, vol. 256, pp. 405–417. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-42887-1_33
25. Viehhauser, J.: Is robotic process automation becoming intelligent? Early evidence of influences of artificial intelligence on robotic process automation. In: Asatiani, A., et al. (eds.) BPM 2020. LNBIP, vol. 393, pp. 101–115. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58779-6_7