Resource-Based Adaptive Robotic Process Automation

Formal/Technical Paper

Renuka Sindhgatta[1](✉), Arthur H. M. ter Hofstede[1], and Aditya Ghose[2]

[1] Queensland University of Technology, Brisbane, Australia
{renuka.sr,a.terhofstede}@qut.edu.au
[2] University of Wollongong, Wollongong, Australia
aditya.ghose@uow.edu.au

Abstract. Robotic process automation is evolving from robots mimicking human workers in automating information acquisition tasks, to robots performing human decision tasks using machine learning algorithms. In either of these situations, robots or automation agents can have distinct characteristics in their performance, much like human agents. Hence, the execution of an automated task may require adaptations with human participants executing the task when robots fail, to taking a supervisory role or having no involvement. In this paper, we consider different levels of automation, and the corresponding coordination required by resources that include human participants and robots. We capture resource characteristics and define business process constraints that support process adaptations with human-automation coordination. We then use a real-world business process and incorporate automation agents, compute resource characteristics, and use resource-aware constraints to illustrate resource-based process adaptations for its automation.

Keywords: Robotic process automation · Declarative constraints · Resource characteristics

1 Introduction

Business process automation (BPA) provides the ability to coordinate tasks and distribute them to resources (humans or software systems) according to certain logical or temporal dependencies [1]. Tasks in a business process are often either *manual* and performed by human participants, or *system-supported* and executed by software systems. Robotic process automation (RPA) strives to automate frequent and repetitive manual tasks performed by human participants using *robots* by mimicking their interactions with IS systems [2,3].

Until recently, one of the criteria for the selection of tasks for automation has been a high level of repetition requiring limited human judgment. However, advances in artificial intelligence and learning algorithms have extended the ambit of automation capabilities [3]. The type of automation can vary

S. Dustdar et al. (Eds.): CAiSE 2020, LNCS 12127, pp. 451–466, 2020.
https://doi.org/10.1007/978-3-030-49435-3_28

in complexity with an automation agent or robot simply mimicking a human information acquisition activity (such as logging into a website and retrieving information for a warehouse management system), to providing decision support to human participants (e.g. a learning algorithm predicting inventory in a warehouse), to carrying out the necessary action (e.g. ordering and updating inventory). RPA technology focuses on the development of robots or 'software programs' having limited support for the design and characterisation of robots performing a task. As an example, in a real world process, a robot (or bot) extracting educational qualifications from a tax exemption request document may have lower accuracy when dealing with acronyms entered by claimants resulting in erroneous output. In such situations, the case execution needs to adapt suitably by having a human participant supervise the task performed by the robot.

There are several reasons why it is important to characterize robots in addition to human participants. First, the characteristics of robots and human participants influencing the execution of a process can be distinct and it is necessary to acquire an understanding during the design phase. For example, considering and characterizing the resource based on the ability to manage workload is critical for human participants but is of little significance for the design of robots (given their capacity is much larger than humans). Second, in line with human-automation studies that detail human interactions with automation agents [4,5], business process execution requires adaptation based on the resources and their characteristics to support different levels of automation: 1) with a robot not capable for performing a task, or 2) capable of performing a task with human assistance or 3) act autonomously and perform a task independently. Process tasks, process participants (humans and robots), and the coordination between process participants needs to be modelled as part of the design phase of the RPA development life-cycle [6]. Third, by taking resource characteristics and process adaptation into account during process design one is able to systematically determine the degree to which automation of the process is feasible. The paper makes the following contributions:

– It outlines a design approach that considers distinct resources (humans and robots) and their characteristics to support different levels of automation, and
– It describes a real-world business process based on a process event log and realizes it using different robots.

The paper is organised as follows. A brief overview of previous studies on resource characteristics and their extension in the context of RPA (Sect. 2) is followed by the introduction of distinct levels of automation and resource-based constraints required to support such levels of automation (Sect. 3). Different levels and types of automation are presented using a real world process and its event log (Sect. 4). Related work (Sect. 5) is followed by a brief conclusion and avenues for future work.

2 Resource Types and Characteristics

The organizational perspective of a business process tends to focus on the human participants and the constraints that need to be met at both design time (assignment) and run time (allocation) for tasks to be performed by certain human resources [7]. Our work broadens the typical focus of the organizational perspective by considering various other types of resources and how they can be involved in the execution of tasks.

Resource Types. We consider three types of resources participating in a business process:

- *Human Agent (HA)*: A human resource is capable of executing all types of (manual) tasks of the process.
- *Robotic Agents (RA)*: Robotic agents are robots, i.e. specialised software programs, that automate *information acquisition* tasks or information gathering tasks. In many scenarios, the RA mimics human interactions on user interfaces by reading the output of interface screens and entering values into such screens [3]. The RA functions like any software system and does not change (or learn) unless the software program is re-written. As observed by Scheepers, Lacity, and Willcocks [3], RAs often automate a subset of tasks and hence are used in conjunction with other resources.
- *Intelligent Agents (IA)*: The notion of an intelligent agent has been envisioned for over three decades. Intelligent agents automate *information analysis* and decision-making tasks. IAs improve their performance through learning [8]. In the scope of this paper, we refer to intelligent agents as agents that use statistical machine learning techniques and learn from observed data [9]. However, the approach scales to agents using other approaches to learning and decision making.

There can be other types of resources needed for executing a business process such as data resources, hardware resources, and other information systems. The focus of this work is limited to the resources automating tasks performed by human participants.

Resource Characteristics. Previous studies have presented various characteristics of human resources for allocation of tasks (referred to as criteria, ability, or profiles in existing literature). Table 1 summarizes different characteristics of a resource based on a prior systematic literature review [14]. Pika, Leyer, Wynn, *et al.* [10] present a detailed and fine-grained definition of various resource characteristics or behaviour, but for our study, we refer to those presented in the aforementioned literature review.

The significance of each of the resource characteristics for different types of resources is presented in Table 1. By significance, we mean the importance of the various characteristics when allocating a task to a resource. *Preference* and *Collaboration* are not primary characteristics for an IA or an RA as automation

Table 1. Literature-based resource characteristics for assigning or allocating tasks

Resource characteristics	Significance			Description
	HA	*IA*	*RA*	
Expertise, skills	✓	✓	✓	The demonstrated capability of a resource to perform a task [10]
Preference	✓	✗	✗	The tendency for choosing particular types of work or for involving particular resources [11]
Collaboration	✓	✗	✗	The degree to which resources work well together [12]
Workload	✓	✗	✗	The average number of activities started by a given resource but not completed at a moment in time [10]
Availability	✓	✗	✗	The resource is available to perform an activity within a specific time frame [12]
Suitability	✓	✓	✓	The inherent qualification of the resource to perform a task [13]
Authorization	✓	✓	✓	Constraints on, or privilege of, a specific person or role to execute a task or case [14]
Experience	✓	✓	✓	Experience is collected by performing the task [15]
Performance (quality)	✓	✓	✓	Number of activities/cases completed with a given outcome by the resource [10]
Duration (time)	✓	✗	✗	The average duration of activities or cases completed by a resource [10]

resources do not have a personal preference or choice (unless programmed as a part of the software code). *Duration* and *Workload* are constant and known at design time for an IA and RA, as compared to human participants with varying completion times [16]. Furthermore, unlike human participants with work schedules, an IA or an RA is always available. Resource characteristics such as expertise, preference, workload, or suitability may influence other resource characteristics such as experience, performance (quality) and duration (time).

In the context of automation, we present a subset of important resource characteristics relevant for the three types of resources to enable human-automation interactions. These measures can be determined using distinct data sources: process event logs, IA test results, or RA software specifications. Given a set of tasks A, a set of resources R executing the tasks, and a set of process attributes $\mathcal{D} = \{d_1, \ldots, d_{|\mathcal{D}|}\}$, a subset of resource characteristics are presented:

Suitability is the inherent quality of a resource $r \in R$, to perform a task $a \in A$. The suitability of a resource [13] can be determined for a process attribute value $d_i \in \mathcal{D}$, with a value $D_{val}(d_i) = v_i$.

$\mathit{suitability}(r, a, d_i, v_i) \rightarrow [0, 1]$, is the suitability of resource r for task a, for an attribute d_i with its value v_i

For example, in an IT support process, a resource of type RA may not be suitable to perform the task ‘apply patch’ for a specific operating system. ‘Operating system’ is the process attribute which could take its value from the set

$\{ubuntu, redhat, \ldots, windows10\}$. Similarly, in a loan application process, a loan approval task for a higher amount may mandate an HA due to a business requirement. Suitability can be determined based on agent specification and implementation, or can be determined based on the organization model attributes such as role, department, or cost of the resource.

Experience: Performing a task activates the experience of a resource [15]. "Experience can be possessed and increased by a [resource] through performing that task"[15]. Event logs can be used to compute the experience of a resource. Consider an event log $\mathcal{L}$ consisting of a set of events occurring during window length κ. Each event $e \in \mathcal{L}$, is associated with a resource $res(e) = r$, task $task(e) = a$ and process attribute values $attr\_val(e, d_i) = v_i$. The number of task completions a by resource r, having a process attribute d_i with value v_i indicates the experience of the resource.
$experience(r, a, d_i, v_i) = |\{e | res(e) = r \wedge task(e) = a \wedge attr\_val(e, d_i) = v_i\}|$

Performance: Automation agents are more susceptible to resource specific errors i.e. errors made by resources when performing a task [17]. Performance measure of an IA can be computed based on the algorithms implemented such as F1-score, root mean square error, precision, or precision@k [18]. These measures can be computed during the training and testing of the algorithms. The performance of agent r, on task a, with a process attribute d_i having value v_i can be computed using the measure specific to the implemented algorithms. $performance(a, r, d_i, v_i) \rightarrow [0, 1]$.

We illustrate the computation of the performance measure for an IA that uses a supervised classification algorithm [9]. A common metric for evaluating the performance of a classifier is the F1-score which considers precision and recall measures. These measures are computed on a test data set where the predictions of the classifier are compared to the true values to arrive at the confusion matrix. Table 2 shows the confusion matrix of the classifier, where each row represents the actual true value, and each column contains the predicted value. Hence, the values of the diagonal elements represent the degree of correctly predicted classes. The confusion is expressed by the false predicted off-diagonal elements, as they are confused for another class or value. Based on the confusion matrix, it is evident that the classifier performance is poor when identifying the label or class 'B'. In scenarios where the IA predicts class 'B', it would be necessary for an HA to intervene and verify the task completion. Hence, this section highlights

Table 2. Illustrative confusion matrix of a classifier

	Predicted class				
		A	B	C	F1-score
Actual class	A	25	2	3	0.69
	B	10	10	10	0.47
	C	7	0	23	0.69

the need for capturing fine grained resource characteristics for specific ***domain attributes and their values*** (d_i and v_i), as the automation support could vary with these characteristics.

3 Process Adaptations for Levels of Automation

The notion of human participants and robots working together and requiring suitable interventions has been presented in human-automation studies acknowledging that "... automation is not all or none but can vary across a continuum of levels, from the lowest level of fully manual performance to the highest level of full automation" [4]. Taxonomies proposing the categorization of automation on different point scales, referred to as Levels of automation (LOA), have lower levels representing manual or no automation and higher levels representing increased automation [5]. While these scales vary from 3 LOA to 11 LOA, they can generally be broken down into three broad categories: (i) levels where the task is primarily performed by a human, (ii) levels where the human-agent interaction is high during task execution and (iii) levels with low human involvement. Table 3 summarizes the broad categories of automation levels.

Table 3. Levels of automation

Scale	Description
Full automation	The automation agent carries out the action
Supervisory control	The automation agent carries out the action, the human may intervene if required
Decision support	The automation agent cannot perform the action but can provide support to the human
Manual	The automation agent offers no assistance

In the context of RPA, at lower levels of automation (decision support), an HA would often be required to execute the task again after its completion by an IA or RA. At higher levels of automation, HAs exercise a supervisory role intervening only if necessary (failures, errors, or poor execution quality). Progress through different levels of automation is dependent on certain characteristics or attributes of the robots such as their performance and experience. Human verification tasks may be added dynamically during process execution based on resource characteristics of the IAs or RAs. Consequently, one or more resource characteristics can be used to design conditions for an HA to intervene.

3.1 Declarative Constraints for Process Adaptation

We use Declare, a declarative business process specification language to illustrate the process adaptations to support different levels of automation and interactions between the different types of resources [19]. Our approach can be supported

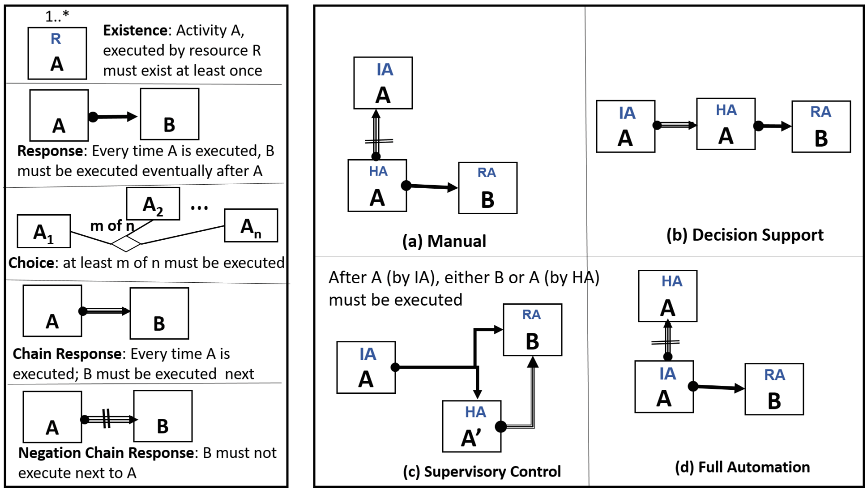
1..*
R
A
Existence: Activity A, executed by resource R must exist at least once
A
B
Response: Every time A is executed, B must be executed eventually after A
A1
A2
...
An
m of n
Choice: at least m of n must be executed
A
B
Chain Response: Every time A is executed; B must be executed next
A
B
Negation Chain Response: B must not execute next to A
IA A
HA A
RA B
(a) Manual
IA A
HA A
RA B
(b) Decision Support
After A (by IA), either B or A (by HA) must be executed
IA A
RA B
HA A'
(c) Supervisory Control
HA A
IA A
RA B
(d) Full Automation

Fig. 1. Some Declare constraints and constraints supporting different LOA.

using any declarative specification that orchestrates control-flow through a definition of constraints and is not limited by the specification language. Declare constraints are grouped into four categories, as shown in Fig. 1: (i) Existence constraints, (ii) Choice constraints, (iii) Relation constraints, and (iv) Negation constraints. The different levels of automation for activity A is shown in Fig. 1:

- *Manual:* An HA executes activity A, which is followed by activity B eventually. Activity A must not be immediately executed by an IA after it was just executed by an HA. Activity B can be performed by any resource type. An RA is chosen in the example to illustrate the interplay between different resource types.
- *Decision Support:* An IA executes activity A, which is immediately followed by an HA executing A, and is eventually followed by the execution of activity B. Thus, the execution by an IA is overridden by an HA (as the IA may not have much experience with, or a poor track record, performing A).
- *Supervisory Control:* An IA executes activity A which can be followed by either activity B, or an activity strongly related to A (A' in the figure), e.g. a redo or a quality check, this time performed by an HA. Activity A' is performed by an HA, and this has to be followed by activity B.
- *Full Automation:* An IA executes activity A, which is followed by activity B eventually. Activity A must not be immediately executed by an HA after it was just executed by an IA.

3.2 Syntax for Resource-Aware Declarative Constraints

To support automation based on types of resources and resource characteristics, we extend Declare with fine-grained resource-aware constraints. Existing work extends Declare with multiple perspectives beyond the control-flow perspective and includes data, resource roles, and time [20,21]. We need additional constraints to support human and automation agent interactions as well as process adaptations based on resource types and their characteristics. Hence, we extend Declare and define the syntax of resource-aware declarative specification through key extensions to Declare using abstract syntax notation [22].

A process model consists of an *ActivitySet*, and a *ConstraintSet* applied to the activities. An *ActivitySet* has one or more *Activity*. Each *Activity* has a label, a set of one or more *Resource* permitted to perform the activity, input and output *Data*. We use $\mathbb{N}$ and $\mathbb{S}$ to represent Integer and String types respectively.

Each *Resource* has a *Role* corresponding to role, a *ResourceType*, and a set of resource characteristics, *ResourceChar*.

$$\begin{array}{rlrl} \mathit{Resource} \triangleq & \mathit{ro} : \mathit{Role}, \mathit{rt} : \mathit{ResourceType}; & \mathit{ResourceCharSet} \triangleq & \mathit{ResourceChar}^{*} \\ & \mathit{rcs} : \mathit{ResourceCharSet} & & \\ \mathit{Role} \triangleq & \mathbb{S} & \mathit{ResourceType} \triangleq & \mathit{HA} \mid \mathit{RA} \mid \mathit{IA} \end{array}$$

A resource characteristic *ResourceChar* is an attribute name and a value pair as discussed in Sect. 2.

$$\begin{array}{rlrl} \mathit{ResourceChar} \triangleq & \mathit{ra} : \mathit{ResourceAttribute} & \mathit{ResourceAttribute}; \triangleq & \mathit{identifier} \\ & \mathit{rv} : \mathit{Value} & \mathit{Value} \triangleq & [0...1] \end{array}$$

The *ConstraintSet* is a set of Declare *Constraint*. A constraint can be a unary or a binary constraint. Constraints operate on a set of *ActivityContext*.

$$\mathit{Constraint} \triangleq \mathit{UnaryConstraint} \mid \mathit{BinaryConstraint}$$

A *UnaryConstraint* refers to the (i) existence, and (ii) choice constraints. *Degree*, states the number of times, an *ActivityContext* must be executed for *existence* or *absence* constraint. For *choice* and *exclusive_choice*, it is the number of activities to be executed from *ActivityContextSet*.

$$\begin{array}{rlrl} \mathit{UnaryConstraint} \triangleq & \mathit{uc} : \mathit{UConstraint}; & \mathit{UConstraint} \triangleq & \mathit{Exist} \mid \mathit{Absence} \\ & \mathit{acset} : \mathit{ActivityContextSet}; & & \mid \mathit{Exactly} \mid \mathit{Choice} \\ & n : \mathit{Degree}; & & \mid \mathit{Exclusive\_Choice} \\ \mathit{ActivityContextSet} \triangleq & \mathit{ActivityContext}^{*} & \mathit{Degree} \triangleq & \mathbb{N} \end{array}$$

BinaryConstraint represents the Declare binary constraint and comprises of a *BConstraint*, a source *ActivityContext* and a set of target *ActivityContext*.

$$\begin{array}{rlrl} \mathit{BinaryConstraint} \triangleq & \mathit{bc} : \mathit{BConstraint}; & \mathit{BConstraint} \triangleq & \mathit{Response} \mid \mathit{Neg\_Response} \\ & \mathit{ac} : \mathit{ActivityContext}; & & \mid \mathit{Chain\_\ Response} \\ & \mathit{acset} : \mathit{ActivityContextSet} & & \mid \mathit{Neg\_Chain\_Response} \end{array}$$

The *ActivityContext* is composed of an *Activity*, an *ExpressionSet*. Each *Expression* describes conditional expressions on the *ResourceAttribute* of the

ResourceChar. The Expression can be a Constant or another *Binary* expression. A *Binary* expression contains two expressions and an Operation.

$$\begin{aligned} \mathit{ActivityContext} &\triangleq \mathit{a{:}\ Activity};\ \mathit{exps{:}\ ExpressionSet} \\ \mathit{ExpressionSet} &\triangleq \mathit{Expression}^{*} \\ \mathit{Expression} &\triangleq \text{Constant} \mid \mathit{Binary} \mid \mathit{ResourceAttribute} \\ \mathit{Binary} &\triangleq \mathit{exp\_1, exp\_2{:}\ Expression};\ \text{op: Operation} \end{aligned}$$

4 Evaluation and Results

The objective of this section is to illustrate our approach using two scenarios. In both scenarios, an IA is simulated, and the resource characteristics of the IA are computed using historical data. The levels of automation are configured by defining constraints on resource characteristics. The achievable levels of automation are illustrated based on the resource constraints. For the first scenario, we choose a business process event log and identify a task that can be automated by an IA. We measure two resource characteristics: performance and experience and illustrate the four levels of automation. The second scenario depicts a service-oriented chatbot (or IA) deployment considering bot-human partnerships. Here, we measure the performance of the chatbot and define two levels of automation. Lack of temporal information in the data limits our ability to measure other resource characteristics such as experience. In this experiment, we illustrate the flexibility of defining levels of automation and identifying achievable levels of automation with the chatbot.

1. Business Process Intelligence Challenge 2014 (BPIC 2014): The BPIC 2014 event log[1] comprises of events capturing the interaction management and incident management of a large bank. The interaction management process is triggered when a customer has an IT issue and calls a service desk agent (SDA). The SDA identifies the relevant IT element having the issue (known as configuration item or CI), the urgency and priority of the issue. If the SDA is unable to resolve the issue, an incident is created, thus initiating the incident management process. The incident is assigned to a team suitable for resolving the incident. Given the data available in the event log, we choose the automation of the activity 'Assign the team' of an incident which in the current process is manually done by an SDA. Identification of the activities for automation is carried out during the design phase of an automation life-cycle [6] and is out of the scope of this work. We choose this task for the purpose of illustration.

Resource Characteristics: In this study, the performance and experience (both resource characteristics) of the IA executing the task 'Assign Team' are presented. We use the BPIC 2014 interaction log and incident activity log for this purpose. Each event in the interaction log represents an interaction and

[1] https://www.win.tue.nl/bpi/doku.php?id=2014:challenge.

contains information entered by a SDA: the type and sub-type of CI, the component, the urgency and priority of the interaction. An incident is created using details of the corresponding interaction. Assignment of an incident to a team is captured in the incident activity log. The incident activity log may contain multiple teams corresponding to a single incident. Such a scenario occurs if there are multiple assignments, caused by an error made by an SDA, or an incident requires several teams to resolve. For this study, we consider incidents involving a single team. The event log contains 46086 incidents resolved over a period of 6 months, of which 13257 are handled by a single team and used to build the IA. To simulate a real-life situation where an IA is trained using historical data and applied to executing process instances, we split the data temporally, with the first 10,000 incidents (80%) used for building the IA. The remaining 20% of the data (2651 incidents) is considered to be unseen data replicating the scenario of an IA when deployed. The IA is a Random Forest classifier, trained using a training, cross-validation and test data. The overall accuracy of the classifier is 72.6%. Any suitable machine learning classifier can be used, as depicting the best performing classifier is not the goal of our experiment. The input features of the classifier based on the information available in the event log, are the CI name, CI type, and component of the incident. The team of the incident is the class or output feature. The F1-score representing the performance of the IA is computed. Figure 2 shows the normalized confusion matrix for a subset of the teams (due to space constraints) on the test data as detailed in Sect. 2. The *performance* of the classifier is high when predicting certain teams (e.g. TEAM0003, TEAM0015) vis-a-vis others (notably TEAM0031, TEAM0044).

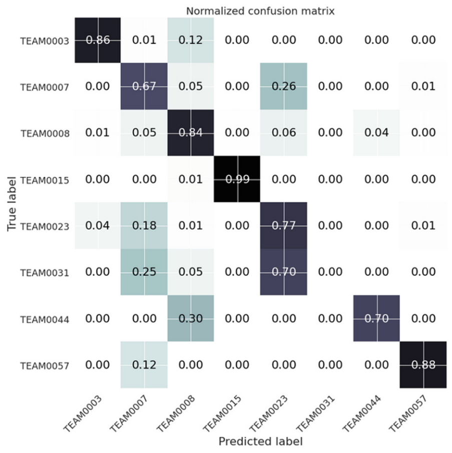

Fig. 2. Confusion matrix of the random forest classifier predicting teams

The experience of the IA is computed as the number of incidents handled by the IA, having a specific type, sub-type of the CI and the component (indicating a specific task). The IA would start with zero experience and accumulate experience as time progresses. We use min-max normalisation to limit experience values to a [0, 1] range. The pre-processed logs, the training data and the source code for replicating the results are available at https://git.io/JePaI.

Resource-aware Constraints: To illustrate different levels of automation and a progression from a low level to a higher level of automation, we present resource-aware constraints for 'Assign Team' task.

When the IA is deployed, if the performance of IA is low on a task (or incident), an HA will execute the task again with no assistance from IA (Manual). In Fig. 3(a), this is captured as a *chain_response* between two 'Assign Team' tasks, first task performed by an IA, followed by another task performed by an HA. When an IA is deployed anew, experience is low and hence, the trust in automation is low. In such a scenario, if the performance is high, the IA may provide recommendations to an HA, but the task will be executed by an HA (Fig. 3(b)). An increase in the experience of IA will result in an increase in level of automation (Fig. 3(c)). Here, an HA performs a supervisory role and executes

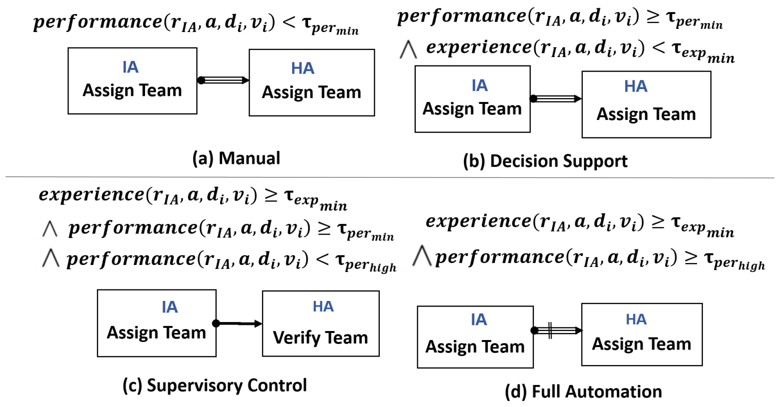
$performance(r_{IA}, a, d_i, v_i) < \tau_{per_{min}}$
IA Assign Team
HA Assign Team
(a) Manual
$performance(r_{IA}, a, d_i, v_i) \geq \tau_{per_{min}}$
$\wedge\ experience(r_{IA}, a, d_i, v_i) < \tau_{exp_{min}}$
IA Assign Team
HA Assign Team
(b) Decision Support
$experience(r_{IA}, a, d_i, v_i) \geq \tau_{exp_{min}}$
$\wedge\ performance(r_{IA}, a, d_i, v_i) \geq \tau_{per_{min}}$
$\wedge\ performance(r_{IA}, a, d_i, v_i) < \tau_{per_{high}}$
IA Assign Team
HA Verify Team
(c) Supervisory Control
$experience(r_{IA}, a, d_i, v_i) \geq \tau_{exp_{min}}$
$\wedge\ performance(r_{IA}, a, d_i, v_i) \geq \tau_{per_{high}}$
IA Assign Team
HA Assign Team
(d) Full Automation

Fig. 3. Configuring resource-aware constraints enables achieving different LOA

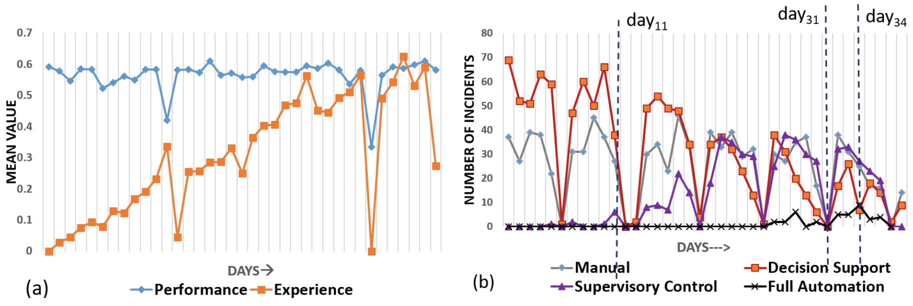

Fig. 4. (a) The mean values of resource characteristics for all incidents. (b) Levels of automation configured using resource characteristics.

a verification task 'Verify Team'. When the performance and experience of the IA are higher than a threshold, the highest level of automation is possible (Fig. 3(d)). Thus, the proposed approach suggests the possibility of having different levels of automation for different case attributes and their values. The thresholds for the resource characteristics are domain dependent and their choice results in a trade-off between the performance of the overall process and the levels of automation that can be achieved. In this experiment, we used threshold values to illustrate the ability to configure levels of automation. Generally speaking, such values are domain-dependent, but here they were heuristically determined. A methodological guideline to identify thresholds, which is yet to be developed, would take into account trade-offs between different resource characteristics and identify the implications on process performance and automation levels.

Levels of Automation: We simulate the scenario of deploying the IA. For every event in the new or unseen data, the performance of the IA is computed by considering the predicted team and the f1-score (or performance) for the predicted team. The normalized experience of the IA is computed considering the domain attributes (type, sub-type of CI, and component). The experience of the IA is updated daily for each incident, and the average experience is presented in Fig. 4(a). We observe a drop in the experience on day_{31}, when new tasks with different values of domain attributes are created by the SDA. This arrival of new tasks causes a decrease in the experience of the IA for those tasks, thus illustrating the need for measuring resource characteristics for specific tasks. Using the resource-aware constraints, levels of automation for the IA can be assessed for each incident (Fig. 4(b)). The distribution of levels of automation changes as time progresses. As shown by two markers, on day_{11}, the experience of the IA is be low for most domain attributes and values, and hence the automation of the task is distributed as 38% at `Manual`, 53% at `Decision Support`, 9% at `Supervisory Control`, and 0% at `Full Automation`. However, on day_{34}, the distribution changes with 37% of tasks executed manually, 10% at `Decision Support`, 40% at `Supervisory Control` and 13% at `Full automation`. The thresholds used for the purpose of illustration are: $\tau_{exp_{min}} = 0.4, \tau_{per_{min}} = 0.5, \tau_{per_{high}} = 0.8$. This example does not provide a benchmark on levels of automation that can be achieved but illustrates the need for configuring such automation levels.

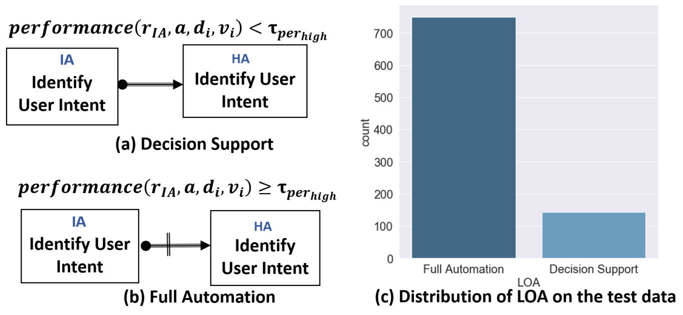

Fig. 5. (a), (b) resource constraints for LOA. (c) Distribution of LOA.

2. Chatbot Intent Classification: In this scenario, we illustrate use of resource constraints on an IA to customise LOA. Consider the scenario of a chatbot service answering the queries of customers. A chatbot would need to identify the goal or intent of the customer based on the customer query or utterance, a problem commonly referred to as 'intent classification'. If the performance of the chatbot in classifying the intent is high, the chatbot responds to the user query. If the performance is lower than the threshold, the customer query is handed over to a human participant. Hence, in this scenario there are two levels of automation - `Decision Support` or `Full automation` (Fig. 5). It would be useful to identify achievable levels of automation based on the frequency of distribution of user intents from past conversations and the performance achieved by the IA in relation to these user intents. To simulate this scenario, we consider the widely used public dataset ATIS (Airline Travel Information System) [23]. An IA is trained to classify intents and its performance is computed for each intent using the confusion matrix. The test data of ATIS is used, the user intents are predicted and the performance of IA is determined for each intent. Based on the performance of the IA on the predicted user intents, and a set performance threshold ($\tau_{per_{high}} \geq 0.95$), the achievable distribution of LOA is determined (Fig. 5(c)). The dataset, the training, and the testing of the classifier as well as the source code with further explanations are available at https://git.io/JeXmt.

5 Related Work

Recent studies on RPA have focused on the design phase presenting techniques to identify candidate tasks for automation [6]. This work similarly focuses on the design phase of the RPA development life-cycle and proposes to consider and broaden the organizational perspective of process automation design. Use of Artificial Intelligence (AI) and Machine learning to enable robots to do complex tasks has been discussed in previous work [3,24]. In this work, we distinguish types of resources and their characteristics in terms of suitability for execution of different types of tasks.

The need for robots and human participants to collaboratively work as part of BPA has been discussed [24] but has not gained sufficient attention. In this work, we present a domain-independent approach towards human-automation interactions, as such interaction requirements may vary across different domains [4,5] and thus need tailoring. The human-automation levels of interactions are supported using declarative process specification constraints.

Declarative specification supports ad-hoc compositions based on control-flow constraints of a business process [19,25]. Declare, a declarative language for process specification, has been further enhanced to support various perspectives by considering constraints on data [26], organizational roles [27], time [21], and all perspectives together [20]. Our work extends a declarative specification with additional resource-aware constraints that are important to support different automation levels and types of resources.

There have been extensive studies focusing on resources and their characteristics [10–12,14,15]. Resource characteristics are used for the allocation of tasks to resources [13,28,29]. The focus of these studies has been human participants. Furthermore, the primary objective of task allocation is to improve efficiency. In this work, we consider distinct types of resources and their characteristics from the perspective of supporting automation.

6 Conclusion and Future Work

The recent body of work on RPA has acknowledged the need to identify tasks that can be automated by robots and the need to consider the interplay between robots and human participants. In this paper, we introduce different types of resources and resource characteristics. We present declarative process constraints that enable interplay between resources taking their types and characteristics into account, thus supporting different levels of automation. This work provides a starting point for supporting more advanced forms of automation in business processes and for exploring more sophisticated ways of engaging resources in business processes.

Limitations and Future Directions: Choices made as part of run-time adaptation involving different resource types and different resource characteristics represents a complex *trade-off space*. While the current paper does not address this question, developing the machinery to support the identification of this space and reasoning over this space remains an interesting direction for future work. The monitoring machinery that flags the need for resource adaptation could, in principle, be quite sophisticated, involving the tracking of (potentially incremental) progress towards the achievement of functional goals and non-functional objectives (key performance indicators or KPIs). This too is something that our current proposal does not fully address and remains an avenue for future work.

References

1. Dumas, M., Rosa, M.L., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-33143-5. ISBN 978-3-642-33142-8
2. Lacity, M., Willcocks, L.P.: Robotic process automation at telefónica O2. MIS Q. Execut. **15**(1) (2016)
3. Scheepers, R., Lacity, M.C., Willcocks, L.P.: Cognitive automation as part of Deakin University's digital strategy. MIS Q. Execut. **17**(2) (2018)
4. Parasuraman, R., Sheridan, T.B., Wickens, C.D.: A model for types and levels of human interaction with automation. IEEE Trans. Syst. Man Cybern. Part A **30**(3), 286–297 (2000)
5. Vagia, M., Transeth, A.A., Fjerdingen, S.A.: A literature review on the levels of automation during the years. What are the different taxonomies that have been proposed? Appl. Ergon. **53**, 190–202 (2016)
6. 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

7. Cabanillas, C., Resinas, M., del-Río-Ortega, A., Cortés, A.R.: Specification and automated design-time analysis of the business process human resource perspective. Inf. Syst. **52**, 55–82 (2015)
8. Russell, S.J., Norvig, P.: Artificial Intelligence - A Modern Approach, 3rd edn. Pearson Education, London (2010)
9. Mohri, M., Rostamizadeh, A., Talwalkar, A.: Foundations of Machine Learning. MIT Press, Cambridge (2012). ISBN 978-0-262-01825-8
10. Pika, A., Leyer, M., Wynn, M.T., Fidge, C.J., ter Hofstede, A.H.M., van der Aalst, W.M.P.: Mining resource profiles from event logs. ACM Trans. Manage. Inf. Syst. **8**(1), 1:1–1:30 (2017)
11. Bidar, R., ter Hofstede, A., Sindhgatta, R., Ouyang, C.: Preference-based resource and task allocation in business process automation. In: Panetto, H., Debruyne, C., Hepp, M., Lewis, D., Ardagna, C.A., Meersman, R. (eds.) OTM 2019. LNCS, vol. 11877, pp. 404–421. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-33246-4_26
12. Huang, Z., Lu, X., Duan, H.: Resource behavior measure and application in business process management. Expert Syst. Appl. **39**(7), 6458–6468 (2012)
13. Kumar, A., et al.: Dynamic work distribution in workflow management systems: how to balance quality and performance. J. Manage. Inf. Syst. **18**(3), 157–194 (2002)
14. Arias, M., Munoz-Gama, J., Sepúlveda, M.: Towards a taxonomy of human resource allocation criteria. In: Teniente, E., Weidlich, M. (eds.) BPM 2017. LNBIP, vol. 308, pp. 475–483. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-74030-0_37
15. Kabicher-Fuchs, S., Mangler, J., Rinderle-Ma, S.: Experience breeding in process-aware information systems. In: Salinesi, C., Norrie, M.C., Pastor, Ó. (eds.) CAiSE 2013. LNCS, vol. 7908, pp. 594–609. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38709-8_38
16. Nakatumba, J., van der Aalst, W.M.P.: Analyzing resource behavior using process mining. In: Rinderle-Ma, S., Sadiq, S., Leymann, F. (eds.) BPM 2009. LNBIP, vol. 43, pp. 69–80. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12186-9_8
17. Reichert, M., Weber, B.: Enabling Flexibility in Process-Aware Information Systems - Challenges, Methods. Technologies. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30409-5
18. Sokolova, M., Japkowicz, N., Szpakowicz, S.: Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation. In: Sattar, A., Kang, B. (eds.) AI 2006. LNCS (LNAI), vol. 4304, pp. 1015–1021. Springer, Heidelberg (2006). https://doi.org/10.1007/11941439_114
19. Pesic, M., Schonenberg, H., van der Aalst, W.M.P.: DECLARE: full support for loosely-structured processes. In: 11th IEEE Conference on EDOC, pp. 287–300 (2007)
20. Burattin, A., et al.: Conformance checking based on multi-perspective declarative process models. Expert Syst. Appl. **65**, 194–211 (2016)
21. Ramirez, A.J., Barba, I., Fernández-Olivares, J., Valle, C.D., Weber, B.: Time prediction on multi-perspective declarative business processes. Knowl. Inf. Syst. **57**(3), 655–684 (2018)
22. Meyer, B.: Introduction to the Theory of Programming Languages. Prentice- Hall, London (1990). ISBN 0-13-498510-9
23. Tür, G., Hakkani-Tür, D., Heck, L.P.: What is left to be understood in ATIS? In: 2010 IEEE Spoken Language Technology Workshop, pp. 19–24 (2010)

24. van der Aalst, W.M.P., Bichler, M., Heinzl, A.: Robotic process automation. BISE **60**, 269–272 (2018). https://doi.org/10.1007/s12599-018-0542-4. ISSN 1867–0202
25. Schönig, S., Cabanillas, C., Jablonski, S., Mendling, J.: A framework for efficiently mining the organisational perspective of business processes. Decis. Support Syst. **89**(C), 87–97 (2016). ISSN 0167–9236
26. Montali, M., Chesani, F., Mello, P., Maggi, F.M.: Towards data-aware constraints in declare. In: 28th ACM SAC 2013, pp. 1391–1396 (2013)
27. Jiménez-Ramírez, A., Barba, I., del Valle, C., Weber, B.: Generating multi-objective optimized business process enactment plans. In: Salinesi, C., Norrie, M.C., Pastor, Ó. (eds.) CAiSE 2013. LNCS, vol. 7908, pp. 99–115. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38709-8_7
28. Kumar, A., Dijkman, R., Song, M.: Optimal resource assignment in workflows for maximizing cooperation. In: Daniel, F., Wang, J., Weber, B. (eds.) BPM 2013. LNCS, vol. 8094, pp. 235–250. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40176-3_20
29. Havur, G., Cabanillas, C., Mendling, J., Polleres, A.: Resource allocation with dependencies in business process management systems. In: La Rosa, M., Loos, P., Pastor, O. (eds.) BPM 2016. LNBIP, vol. 260, pp. 3–19. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45468-9_1