Transfer Learning Operators for Process-oriented Cases*

Mirjam Minor, Miriam Herold, Julius Rubbe, Stefan Dufner and Georgios Brussas
Department of Business Informatics
Goethe University
Frankfurt, Germany
minor@cs.uni-frankfurt.de

***Abstract*—Transfer learning aims to reuse knowledge in a context that is different from the context for which it has been acquired. In Case-based Reasoning (CBR), experiential knowledge is recorded in form of cases. It is a challenging task to port cases across the boundaries of the application domain. In this paper, we propose a framework for the transfer of process-oriented cases from a source to a target domain, including workflows contained in the cases. Adaptation methods for workflows in a classical CBR sense are combined with analogical knowledge learned by means of word embeddings. A set of transformation operators is specified that apply transfer knowledge to a workflow. An experiment with the three operators 'analogical substitution', 'generalisation' and 'abstraction' is conducted. 30 workflows in BPMN from the source domain *passenger and baggage handling at the airport* are transferred into target domain *SAP warehouse management*. The experimental results are discussed by means of the 3QM framework which provides metrics to assess the quality of business process.**

***Index Terms*—Transfer Learning, Ontological Knowledge, Process-oriented CBR, Business Process Management**

I. Introduction

Well-defined business processes are a key prerequisite of successful digitisation projects in organisations [1]. Companies who aim to implement Robotic Process Automation (RPA) or AI analytics, for instance, may find that they do not have appropriate process knowledge so far or that their process models are outdated. A variety of process-oriented reference models can be consulted for standard business goals, for example in the fields of supply chain management [2] or business application management [3]. In the literature on process-oriented case-based reasoning (POCBR) [4] [5] [6] [7] [8], retrieval and adaptation methods have been investigated in order to taylor and reuse workflows[1]. Fig. 5 depicts a sample workflow that is part of a process-oriented case. Many application areas, however, suffer from a lack of sample cases or process repositories in sufficient quantity. Reuse is very difficult or impossible if the amount of process-oriented cases is too small.

*The publication is a contribution to the project EVER2, funded by Deutsche Forschungsgemeinschaft (DFG) under project no. MI 1455-2-3. The authors would also like to thank Xinyuan Cai for her valuable contribution to the development of part of the experiments.

[1]Workflows are "the automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules" [9].

Transfer learning (TL) addresses the "question of how the things that have been learned in one context can be reused and adapted in related contexts" [10, p. 5]. In machine learning, TL is frequently understood as the problem to transfer a "collection of patterns observed across tasks" [11] also known as inductive transfer. For future directions in TL, it seems clear that "proper treatment of the inductive transfer problem requires more than just statistical or mathematical techniques" [11]. Embedding statistic-oriented AI methods into knowledge-oriented approaches like Case-based Reasoning (CBR) [12] is a promising research strand for TL to fill this gap [13]–[15].

This paper presents a novel framework on operator-based transfer learning in POCBR. It includes

- the representation of transfer knowledge for workflows in an ontology,
- a set of basic transformation operators for workflows, and
- an exemplary TL process for workflows with the three operators for analogical substitution, generalisation, and abstraction implementing the transfer for two example domains 'passenger and baggage handling at the airport' and 'SAP warehouse management'.

Please note that we have chosen the 'SAP warehouse management' processes as a target domain despite the fact that reference models are available for those processes since this facilitates assessing the experimental results. The paper is an extension of previous work on the transferability of process-oriented cases in principle [14], a more detailed introduction of the sample application domains [16], and a knowledge discovery approach using word embeddings to learn analogical knowledge from a text corpus [15]. In the work, the result of the latter is used by the operators as a part of the ontological knowledge.

The remainder of the paper is organised as follows. Related work is discussed in Section II. In Section III, the transfer learning framework for workflows is introduced including the ontology, the basic transformation operators, and a transfer strategy. Section IV reports on the experiments. Finally, a conclusion is drawn in Section V.

II. Related work

Transfer learning approaches in CBR use knowledge from a source domain "to enhance an agent's ability to learn to solve

tasks from a target domain" [13, p. 54]. In the following, we will discuss related work on analogy and CBR, generalisation, and abstraction as well as some work that is loosely related to our transfer strategy.

Klenk et al. [13] present a case-based application using analogical mapping to support students in solving physics-style problems for an exam. In our work, we consider also analogical relations to be used for the transfer of workflow tasks. Hong & Lepage [17] create analogical clusters that go across domains as part of knowledge acquisition for CBR. In contrast to such clusters comprising many terms of the vocabulary to describe a case base each, our approach considers cross domain analogy in binary relations between two analogical elements only.

Generalisation of process-oriented cases has been used for adaptation [18]. Our generalisation operator uses also generalisation of the semantic descriptions of tasks.

Like the generalisation operator, our abstraction operator has been inspired by work on case-based adaptation. Dufour-Lussier et al. [19] employ formal concept analysis to abstract linguistic entities that have been extracted from text. The formal concepts serve as abstract artifacts that can be used to substitute a particular linguistic entity in this context, such as an ingredient of a dough in a cooking recipe. In POCBR, abstraction operators for workflows have been published by Müller et al. [8] using workflow streams. A workflow stream subsumes a transitively data-flow-connected partial workflow and can be represented by an abstract task. The identification of a workflow stream is based on data items consumed or produced by workflow tasks. In many business process scenarios, however, including the sample domains for our experiments, the workflows are rather control flow oriented than data-driven.

Our transfer strategy is loosely related to adaptation strategies reported in the CBR literature. In accordance with Fuchs et al.'s work on differential adaptation [20], we consider the adaptation process a search process in the space of solutions, resulting in a sequence of appropriate adaptation steps. The main idea of differential adaptation is to chose the next adaptation step by estimating the contribution of the step to reduce the distance to the solution of the problem. Our transfer strategy uses utility values to select the most promising transformation operators. Leake & Le [21] propose an adaptation strategy that combines adaptation rules in an adaptation path comprising multiple adaptation steps. If the baking powder in a pancake recipe, for instance, has been substituted by botter milk the latter can be further replaced by milk and vinegar. Like Fuchs' work and in contrast to our work, Leake's adaptation strategy has not yet been applied across the boundaries of domains.

III. Transfer Framework

The goal of the transfer framework is to implement the transfer of process-oriented cases across the boundaries of application domains. The *source domain* D_S denotes the context in which knowledge is available at a rich, mature level. The *target domain* D_T provides a context where the knowledge is sparse. D_S comprises a large repository of workflows called WF_S. The workflow repository WF_T in the target domain is small and is to be enriched by a set of transferred workflows denoted by $WF_{T'}$.

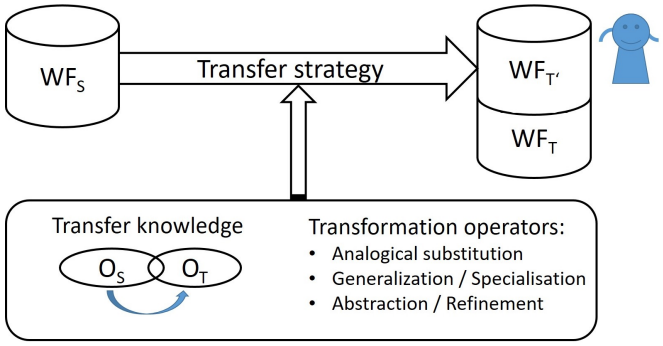

Fig. 1. Framework for the transfer of workflows.

As depicted in Fig. 1, the transfer process is performed by a *transfer strategy* in an interactive manner. The user is a workflow designer who aims to model workflows in D_T. The transfer process creates entire workflows or parts of workflows in $WF_{T'}$ from reusing workflows in WF_S, which are approved and further developed by the user. A set of transformation operators is specified to adapt workflows within the source domain, port them across the boundaries from source to target domain by analogical adaptation, and further adapt them within the target domain.

A. Representation of transfer knowledge in an ontology

We assume that an ontology O is available (or can be created) as vocabulary for both domains and for the representation of transfer knowledge that is required by the transformation operators. $O_S \subseteq O$ denotes the part of the ontology whose concepts belong to the source domain D_S. $O_T \subseteq O$ denotes the concepts of the target domain D_T.

O covers the workflow tasks and the data items of the workflows in WF_S and WF_T, as well as their control flow and data flow. As part of the ontology, a particular workflow forms a graph $W = (N, E)$ where N is a set of nodes and $E = N \times N$ is a set of edges. N subdivides in different node types $N = N_T \cup N_E \cup N_G \cup N_D$ where N_T represents the workflow tasks labeled by their task names. N_E represents the events, N_D the data objects, and N_G the gateways labeled by their type, such as 'XOR split', 'XOR join', 'AND split'. $N_{CF} = N_T \cup N_E \cup N_G$ subsumes all nodes that belong to the control flow. $E = E_{CF} \cup E_{DF}$ contains edges for the control flow $E_{CF} \subseteq N_{CF} \times N_{CF}$ and edges for the data flow $E_{DF} = (N_T \times N_D) \cup (N_D \times N_T)$. The example workflow in Fig. 5 depicts the check-in procedure for an airport that takes place at a check-in desk in a hotel located in close proximity to the airport. It comprises several workflow tasks, such as 'Enter hotel check-in desk', 'Check identity card', etc. in N_T, an 'XOR split' and an 'XOR join' as part of N_G and two

events for 'Start' and 'End' in N_E. The workflow requires the data objects 'Id card' and 'Boarding pass'.

A further part of the ontology represents transfer knowledge. Analogical knowledge between workflow tasks is expressed by the relation `analogTo`. Fig. 2 depicts an example for two analog tasks. Taxonomical knowledge on workflow tasks is expressed by the `subClassOf` relation (cmp. Fig. 3). The taxonomy is used during generalisation of workflows. Abstraction knowledge is used to analogise workflow fragments. Polyvyanyy defines abstraction of a workflow as "a function that ... hides process details and brings the model to a higher degree of abstraction" [22]. In contrast to generalisation, abstraction modifies the structure of a workflow. An abstracted workflow task aggregates the control flow and the data flow of a subgraph of a workflow. The abstract task is represented by an additional task node in the ontology. Fig 4 depicts a sample abstract task node. The workflow nodes subsumed by the abstract task node are connected to it by means of the relation `belongsTo`. Further, the abstract task node relates to the workflow node predecessing the workflow part to be abstracted by `hasEntry` and to the succeeding workflow node by `hasExit`.

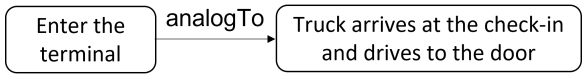

Fig. 2. Sample analogical relation.

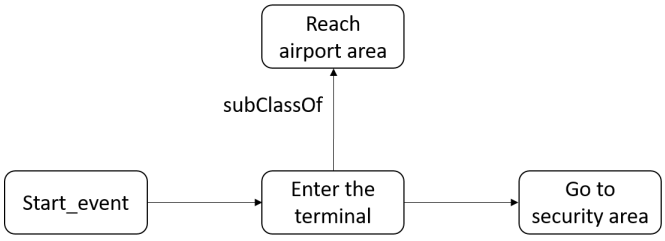

Fig. 3. Sample taxonomical relation.

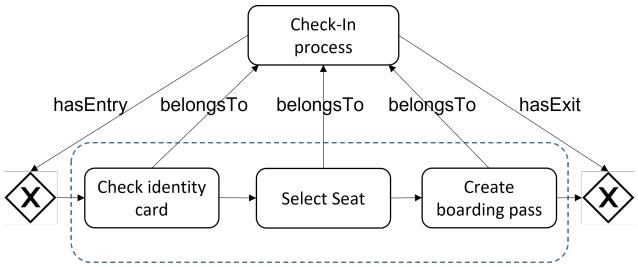

Fig. 4. Sample abstract task with its relations.

B. Transformation operators

A set of basic transformation operators OPS is defined where $o \in OPS$ describes an adaptation step for a workflow. An operator description o comprises an operator name, a set of pre-conditions, and a set of post-conditions (effects). The pre-conditions are read from the ontology of workflows and relations expressing transfer knowledge (cmp. the above description of the ontology).

The following preliminaries are required by the operator descriptions.

$\Gamma^+_W(n)$ The *successor set* $\Gamma^+_W(n) = \{m \in N | (n,m) \in E\}$ of a node $n \in N$ denotes the nodes succeeding n in both, the control flow or data flow of the workflow.

$\Gamma^-_W(n)$ The *predecessor set* $\Gamma^-_W(n) = \{m \in N | (m,n) \in E\}$ denotes the preceding nodes for n.

$W[S]$ The *induced subgraph* $X = W[S]$ denotes the subgraph X induced by W with respect to node set S where $N(X) = S, S \subseteq N(W)$, $E(X) = E(W) \cap \binom{S}{2}$.

A set of basic operators g, a, p is defined as follows:

- $g(n)$ - *generalise* W_i into W_{i+1} at task n
  - pre-conditions: $n \in N_T(W_i)$, $\exists \hat{n} \in O$ with $subClassOf(n, \hat{n})$
  - post-conditions:
    - $n \notin N_T(W_{i+1})$, $\hat{n} \in N_T(W_{i+1})$,
    - $\forall v \in \Gamma^-_{W_i}(n) : (v, \hat{n}) \in E(W_{i+1})$, $\forall v \in \Gamma^+_{W_i}(n) : (\hat{n}, v) \in E(W_{i+1})$
- $a(X)$ - *abstract* W_i into W_{i+1} at SESE region X [2]
  - pre-conditions:
    - $X = W_i[S]$, $S \subset N(W_i)$ is SESE region between $n, m \in N_{CF}(W_i)$,
    - $\exists \hat{x} \in O$: $hasEntry(\hat{x}, n)$, $hasExit(\hat{x}, m)$, $\forall s \in S : belongsTo(s, \hat{x})$
  - post-conditions: $\forall s \in S : s \notin N(W_{i+1})$, $\hat{x} \in N_T(W_{i+1})$
  - $(n, \hat{x}) \in E(W_{i+1}), (\hat{x}, m) \in E(W_{i+1})$
- $p(n, n')$ - *port* task n from W_i in source domain to W_{i+1} with n' in target domain by analogical substitution
  - pre-conditions: $n \in N_T(W_i)$, analogTo(n,n')
  - post-conditions:
    - $n \notin N_T(W_{i+1})$, $n' \in N_T(W_{i+1})$
    - $\forall v \in \Gamma^-_{W_i}(n) : (v, n') \in E(W_{i+1})$, $\forall v \in \Gamma^+_{W_i}(n) : (n', v) \in E(W_{i+1})$

The abstraction operator a covers different types of abstract tasks. The type 'sequence abstraction' aggregates task nodes of a SESE region that forms a sequence. We denote an abstraction operator of this type by a_{seq}. The sample abstract task from Fig. 4 belongs to this category since three tasks are conducted into the task 'Check-In process'. a_{role} denotes a 'role abstraction' operator that aggregates a SESE region whose tasks are assigned a particular actor or role. Further types include a_{block} for XOR or AND blocks, a_{loop} for loop blocks, and a_{dead_end} for exceptions from the SESE paradigm

[2] A SESE region of a workflow graph denotes a Single-Entry, Single-Exit fragment [23] of a workflow W that is a control-flow connected workflow subgraph $X = W[S]$ induced by the set $S \subset N$ of control flow elements and data objects that are enclosed between an entry point $n \in N_{CF}(W)$ and an exit point $m \in N_{CF}(W)$.

where a split nodes is not followed by an according join node but one ore more branches lead directly into an end event.

Any basic transfer operator is implemented by a sequence of atomic operators from the set of atomic operators $\{del_e, del_v, add_e, add_v, rename_e, rename_v\}$ which delete, add, or change the label of a node or an edge within the workflow graph. The deletion of a node deletes automatically the edges to and from this node, i.e. in some operators, the neighborhood of nodes in the workflow to be adapted is cached to reconnect an newly inserted workflow fragment to the remainder of the workflow graph. The set of basic operators is extensible by further operators in future.

C. Transfer process

The transfer process for a workflow graph W_0 is a search for operators o_1, o_2, ..., o_n to form a *transformation path* $W_0 \Rightarrow^{o_1} W_1 \Rightarrow^{o_2} ... \Rightarrow^{o_n} W_n$ with the goal that the resulting workflow W_n uses only vocabulary that is aligned to the target domain. Such a workflow is called fully transferred workflow. Please note that some workflows may be transferable only in parts. Further, multiple transformation paths may exist for a workflow.

A transfer strategy guides the decision which transformation path to choose for the source workflow W_0. A *utility function* measures the amount of the reduction of the distance to the goal when applying transformation operators. The intuition behind is to estimate how useful a (partially or fully) transferred workflow is with respect to support modeling in the target domain. The utility function $utility$ for a workflow W comprises the utility values for its workflow tasks.

$$utility(W) = \sum_{n \in N_T(W)} utility(n) \tag{1}$$

The utility function for a workflow task n considers the operator chain $c = o_k \circ ... \circ o_2 \circ o_1$ it origins from.

$$utility(n) = \begin{cases} \epsilon & , n \notin D_T \\ u(c) & , c = o_k \circ ... \circ o_1, o_k = p(., n), n \in D_T \end{cases} \tag{2}$$

It takes the value ϵ in case a workflow task is still out of target domain. The value u for the result of an operator chain c is calculated by the following recursive function.

$$u(c) = \begin{cases} \alpha_p \cdot tp(n) & , c = p(n, n') \\ \alpha_g \cdot tp(n) & , c = g(n) \\ \alpha_a \cdot tp(X) & , c = a(X) \\ \alpha_o \cdot u(\tilde{c}) & , c = o \circ \tilde{c} \end{cases} \tag{3}$$

where tp denotes the *initial transfer potential* of a workflow task n or of set of tasks X.

$$tp(n) = \begin{cases} 1 & , n \in N_T(W_0) \\ 0 & , \ else \end{cases} \tag{4}$$

$$tp(X) = \sum_{n \in W[X]} tp(n) \tag{5}$$

The transfer potential of a workflow task that occurs in the original workflow W_0 is set to one. In the special case that $p(n, n)$ is applicable, the full transfer potential is maintained. In the most cases, the transfer potential is reduced by the *attenuation factors* $\alpha \in [0, 1]$ when applying transformation operators. α_o is set to the attenuation factor of the operator o as follows. α_g and α_a are a configurable constant values. α_p is set to the individual mapping value mv of the pair of tasks to which the port operator p is applied to. mv expresses the confidence of an analogical mapping. In case of a computational analogy (cmp. previous work [15]) the value is provided by the learning or discovery algorithm.

For instance, the task 'Check identity card' in Fig. 5 is involved in multiple transformation paths considering Tables I and II. One of those paths is $p(`Check\ identity\ card', `Load\ shipping\ units')$ with a utility of 0.3181. An alternative path, for example, applies $p \circ g \circ g$ to $`Check\ identity\ card'$ resulting in $c = p(g(g(`Check\ identity\ card')), `Control')$. Assuming $\alpha_g = 0.75$ and $\alpha_p = 1.0$ for porting a task to itself, the latter path creates a utility for 'Control' of $1 \cdot 0.75 \cdot 0.75 \cdot 1 = 0.5625$. Thus, the second path is preferred over the first path. Further paths are applicable to the sample task using abstraction (cmp. Fig. 6).

The transfer strategy is implemented as a means end reasoning procedure, i.e. each application of an operator chain ending with an analogical substitution by p reduces the distance to the goal of an optimally transferred workflow. Optimal transfer means that the transferred workflow supports a real business process in the target domain in the most promising quality (cmp. the quality criteria discussed for the experimental evaluation in Section IV). Since this is hardly achievable fully automatic in practice, the transfer strategy allows interaction with the user.

IV. Experiments

The implementation of the transfer framework is written in Java[3]. The three transformation operators g, a and p have each been implemented in a separate Java class. Sample transfers with experimental workflow data have been executed. The experiments aim to provide a proof of concept that each of the three implemented operators are applicable and potentially beneficial. The results highlight the feasibility of the proposed transfer framework.

The experimental setup comprises four variants of transfer strategies. Three variants follow a very straightforward strategy to use only one type of operator as many times as applicable. GEN uses only the generalisation operator g. A_{all} uses abstraction operators a of all types. ANA uses the operator for analogical substitution p. The fourth variant $GANA$ concatenates generalisation g and analogical substitution p. The strategies are following the utility-based procedure as described above in Section III-C. The four transfer strategies have been applied to a repository with

[3]The source code is available under the creative common licence CC BY-NC 4.0- at https://github.com/BusinessInformationSystemsLab/TL_Framework.

experimental data described in Section IV-A. Illustrating samples from this repository are discussed using the 3QM framework [24]. Following semiotic theory, it distinguishes between syntactics, semantics and pragmatics as quality categories of business process models [25]. Syntactical quality describes whether a process model has a well-formed structure. Semantic quality deals with the meaning of the elements of a business process, such as their relevance (whether it contains superfluous elements), or completeness. Pragmatical quality focuses on the interpretation of elements, such as unambiguity or understandability. Redundancy is one of the key criterion of unambiguity, it can, for example, be measured by a metrics counting "the number of carriers of meaning which are unnecessarily depicted repeatedly in a business process" [24, p. 238]. These categories have been confirmed in empirical studies to be necessary, sufficient and independent of each other for assessment [25].

A. Experimental data

As mentioned in the introductory section, we propose two different domains for transfer learning in business processes. The first domain is *passenger and baggage handling at the airport*. The second domain is *SAP warehouse management*. For the repository in the airport passenger and baggage handling domain we extracted 30 workflows, mainly different kinds of passenger check-in, baggage handling, security checks (for passengers and baggage), customs clearance and various workflows for transport and loading of baggage into the aircraft. Most of the workflows are based on a textbook [26] and modelled in BPMN 2.0. Fig. 5 shows an example of a partial workflow from the airport domain.

In the SAP warehouse management domain, we extracted 20 different workflows from the SAP website [27]. The website is a collection of best practices for integration of SAP Extended Warehouse Management and SAP S/4HANA rapid deployment solution. It contains different process models, primarily designed for the integration of SAP modules, but they offer a solid amount of business procedures to extract workflows with the appropriate control flow. The processes cover, for example, inbound and outbound of products in/from various kinds of warehouses, replenishment, scrapping, inventory as well as consumption of products during production. We also transformed these workflows into BPMN. Further, we created an ontology for both domains, including the workflows, the taxonomy of tasks, and the analogical relation between tasks. In total we have about 125 different tasks from the source domain. They are organised in a taxonomy with four different hierarchy levels. The analogical relation between tasks (cmp. Table I) is resulting from a knowledge discovery method [15].

B. Experimental results

For the experiment, the four different methods – analogical substitution ANA, generalisation GEN, abstraction A_{all}, and concatenation $GANA$ – were applied. The four strategies were applied to all 30 workflows. The results can be found in the repository[4]. All experimental workflows could be fully transferred. In the following section, the results of the individual operators are evaluated in relation to the 3QM framework; in particular, the workflow presented in Fig. 5 is considered as a running example. This workflow is well suited to illustrate the general advantages and disadvantages of the individual strategies. Thus, the workflow is representative for the findings from all workflows. The experimental data is in German, for a better understanding all results are translated into English.

Airport domain	**SAP Warehouse domain**	**mv**
Enter hotel check-in desk	Reach truck check-in and drive up to the door	0.4318
Check identity card	Load shipping units	0.3181
Select seat	Transfer stock from storage bin 'ROD' to storage bin 'AFS'	0.2413
Create boarding pass	Create Customer return document	0.5652
Drive to airport	Truck drives off	0.2000

TABLE I
(`analogTo`) RELATION BETWEEN SAMPLE WORKFLOW TASKS.

We will focus on the peculiarities of semantics and pragmatics, since the syntactical correctness is given by construction.

The analogical knowledge used by the transfer strategy ANA for the sample tasks can be found in Table I. In this particular workflow, an analog task of the target domain could be assigned to each task. When looking at the results, there are mostly good transfers, this can for example also be seen in the transfer, 'Enter hotel check-in desk' to 'Reach truck check-in and drive up to the door'. These are comparable tasks in regards of their domain, which we could observe in the majority of tasks across all workflows in the repository.

Since `analogTo` origins from a knowledge discovery approach, the mapping value mv represents the confidence of the computational analogy. The mapping example 'Select seat' to 'Transfer stock from storage bin 'ROD' to storage bin 'AFS'' would not lead to a convincing substitution.

With regards to the generalisation strategy GEN, all tasks could be generalised, and the semantics remained complete. The generalisation of the example workflow can be found in Table II, where all individual tasks were generalised up to two levels. The tasks are generalised into the parent class, whereby several tasks can be transferred into the same generalised class. The higher the level of generalisation the more common the task, and thus more tasks are assigned the same category. This limitation of pragmatics can be seen in the workflow example on the second generalisation level, where both 'Enter hotel check-in desk' and 'Drive to airport' are allocated into the class 'Move person', and thus create a redundancy. From a certain level on, most tasks are funnelled into the same class. To counteract this problem, we have restricted the number of generalisations per task. A chain can comprise two generalisation operators at maximum. This can be seen in the example, where 'Edit document' is not further generalised,

[4]Repository with detailed experimental results: https://hessenbox-a10.rz.uni-frankfurt.de/getlink/fiGqd6E124uJCpkideRsNegN/

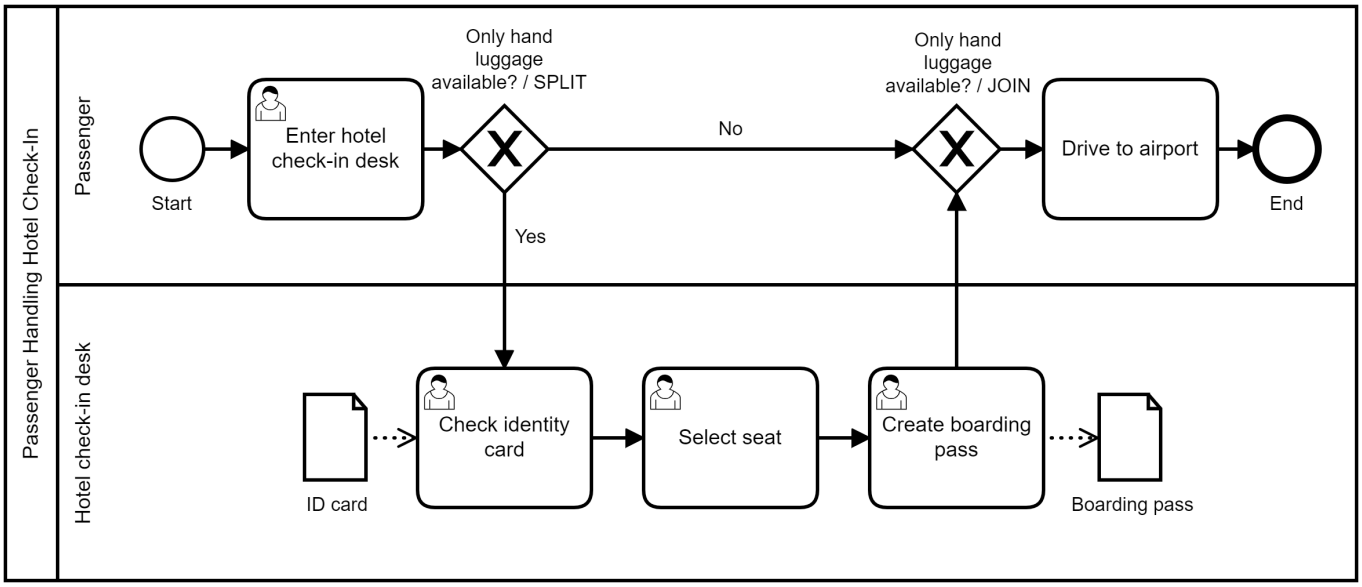

Fig. 5. Workflow 'Passenger Handling Hotel Check-In'

Original	1. Generalisation	2. Generalisation
Enter hotel check-in desk	Reach not airport area	Move person
Check identity card	Control document	Control
Select seat	Other tasks	Other tasks
Create boarding pass	Edit document	Edit document
Drive to airport	Reach airport area	Move person

TABLE II
GENERALISATION OF WORKFLOW TASKS.

since it would otherwise be in the most general class, that includes all tasks.

In the strategy A_{all}, a combination of all five abstraction operators is investigated. The abstraction of our workflow is depicted in the Fig. 6. The machine-generated versions of all abstractions can be found in the repository. In the particular workflow only two of the five abstraction operators were applied. This is the case because there is no deployment scenario for the other operators in this small workflow. The two operators a_{seq} and a_{block} were utilised. It can be seen well how operators are nested. First three tasks are consolidated through the sequential operator. Furthermore, these are in a XOR block and can therefore be grouped together again by the block-operator. Despite the higher complexity of some of the other sample workflows, there are no redundancies in terms of 3QM observed and the nesting of the operators works without errors. Unsurprisingly, detailed information is lost during abstraction because the structure of the workflows changes which slightly decreases the semantics criterion of 3QM. In the shown workflow, three tasks are combined into one abstract task.

In the fourth strategy $GANA$, generalisation and analogical substitution are combined. This happens through the utility function. The machine-generated result is displayed in Fig. 7 and shows the same workflow, that we used to illustrate the other strategies. The combination through the utility function worked well, all tasks could be transferred according to the algorithm. The generated tasks are all within the same logistic use case. Specifically, the first and last task provide a consistency. In the first task the truck arrives in the last it drives off. While this is a rather general workflow it shows the potential and provides a good initial framework for the designer. Regarding the three tasks in between, these tasks have different levels of detail and various advantages for the designer. In the second task 'Check recounted inventory documents' existing documents are reviewed and in the fourth task 'Create customer return documents' documentation is created. These are very common tasks in a logistics process. Although both tasks are very specific, they can provide the designer a good thought-provoking impulse. In contrast, the task 'Other tasks' is very general due to the generalisation and require the designer to specialise. Within the three-part sequence, there is thus an acceptable consistency.

The difference between the generalisation and just the analogical substitution by the utility function can be seen looking at the second task. Originally this task was 'check identity card', through the analogical mapping strategy ANA this turns into 'Load shipping units'. Since the utility function also considers the mapping value mv on the generalised level, the result changes. The task 'Check recounted inventory documents' which is in an analogical relation with 'Control document' is preferred. This also occurs for the task 'Select Seat'. For the other tasks, the simple analogical substitution prevails due to the higher utility value. By the combination of the two operators the overall redundancy is smaller than in the other three strategies.

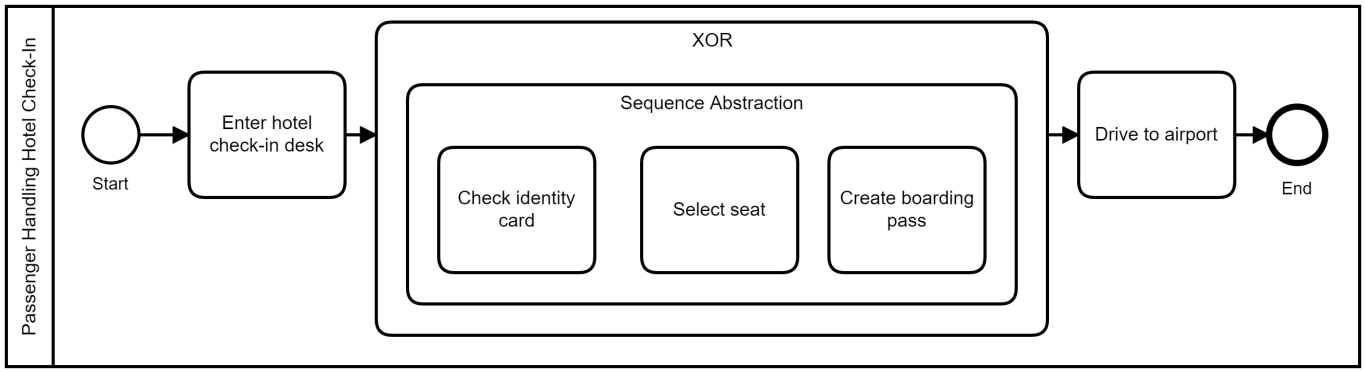

Fig. 6. Workflow with abstracted tasks

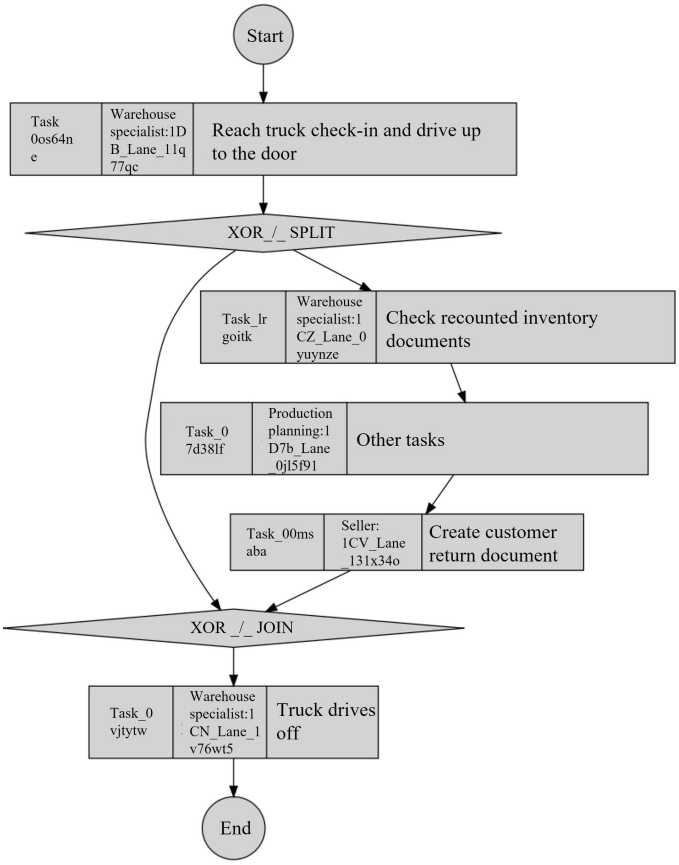

Fig. 7. Workflow after generalisation and analogical mapping.

To summarise, the transferred workflows provide plausible first drafts of workflows in the target domain that can be interactively modified by the designers.

V. Conclusion and Future Work

In this paper we introduce a novel framework for the transfer of process-oriented cases. A set of three transformation operators has been specified and evaluated in an experiment with real process models. In particular, we show results for different transfer strategies and discuss the shortcomings and advantages. The combination of operators has achieved the most promising results.

With the generalisation strategy GEN, the semantics remained complete and all tasks could be generalised. However, it is problematic that the higher the level of generalisation, the more tasks end up in the same class. To prevent this, a restriction of the number of generalisations has been included in the strategy. The strategy A_{all} allows the combination of five different abstraction operators. The combination and nesting of abstraction operators worked well, however it is difficult to specify the α factors for the transfer strategy. Further experiments are required to investigate the appropriate utility values for the contribution of the particular type of abstraction. The ANA-strategy allowed most tasks to be mapped to the target domain. Specifically the introduced mapping value mv allowed for an optimisation of the results. This strategy can potentially be further refined in the future. The strategy $GANA$ combines analogical substitution and generalisation through a utility function. By calculating all potential transformation paths, different variants to port each task are compared. In future work, the abstraction operator can be incorporated into the $GANA$ strategy.

According to the quality criteria from the 3QM framework, the experiments clearly show the benefit of our approach. A designer who is about to model new workflows in a target domain can benefit from all four strategies. Especially $GANA$ can provide a valid draft. However, the results also show that the workflows require further adaptations from the designer. Thus, the automatically transferred workflows serve as an inspiration for the human designer.

In the future, further refinements and specialisations can be made so that the result of the transfer process is even closer to full-fledged workflows, thereby reducing the needed adaptions by designers. Further, it may also be interesting to define more operators transforming the workflows within the target domain. For instance, an additional operator might use the (sparse) taxonomical knowledge in the target domain as well as the position and context of the task in the original workflow to recommend a specialisation for a transferred workflow. We are planning to investigate whether such situation-specific

operators are feasible in addition to the process independent analogical relations that we have already considered between workflow tasks. In summary, our experiments have shown that the proposed transfer strategy is very promising and has a high flexibility to be extended further. We have successfully demonstrated that classical adaptation in POCBR can be combined with analogical knowledge and knowledge discovery to achieve transfer learning across the boundaries of application domains.

REFERENCES

[1] T. Hess, C. Matt, A. Benlian, and F. Wiesböck, "Options for Formulating a Digital Transformation Strategy," *MIS Q. Executive*, vol. 15, no. 2, pp. 123–139, 2016.

[2] G. Stewart, "Supply-chain operations reference model (SCOR): the first cross-industry framework for integrated supply-chain management," *Logistics information management*, 1997, publisher: MCB UP Ltd.

[3] M. B. Carter, J. Lange, F.-P. Bauer, C. Persich, and T. Dalm, *SAP Extended Warehouse Management: Processes, Functionality, and Configuration*. Galileo Press, 2010.

[4] M. Minor, A. Tartakovski, and R. Bergmann, "Representation and Structure-based Similarity Assessment for Agile Workflows," in *Case-Based Reasoning Research and Development, 7th International Conference on Case-Based Reasoning, ICCBR 2007, Belfast, Northern Ireland, UK, August 2007, Proceedings*, ser. LNCS 4626, R. O. Weber and M. M. Richter, Eds. Springer, 2007, pp. 224 – 238.

[5] R. Bergmann and Y. Gil, "Retrieval of Semantic Workfows with Knowledge Intensive Similarity Measures," in *Case-Based Reasoning. Research and Development, 19th International Conference on Case-Based Reasoning, ICCBR 2011*, ser. Lecture Notes in Computer Science, vol. 6880. Springer, 2011, pp. 17–31.

[6] M. Minor, S. Montani, and J. A. Recio-García, "Editorial: Process-oriented Case-based Reasoning," *Inf. Syst.*, vol. 40, pp. 103–105, Mar. 2014.

[7] S. Montani, G. Leonardi, S. Quaglini, A. Cavallini, and G. Micieli, "Mining and Retrieving Medical Processes to Assess the Quality of Care," in *Case-Based Reasoning Research and Development - 21st International Conference, ICCBR 2013, Saratoga Springs, NY, USA, July 8-11, 2013. Proceedings*, ser. Lecture Notes in Computer Science, S. J. Delany and S. Ontañón, Eds., vol. 7969. Springer, 2013, pp. 233–240.

[8] G. Müller and R. Bergmann, "Workflow Streams: A Means for Compositional Adaptation in Process-Oriented CBR," in *Case-Based Reasoning Research and Development*, ser. Lecture Notes in Computer Science, L. Lamontagne and E. Plaza, Eds. Springer International Publishing, Sep. 2014, no. 8765, pp. 315–329.

[9] {Workflow Management Coalition}, "Workflow Management Coalition Terminology & Glossary (Document No. WFMC-TC-1011)," 1999, last access 05-23-2007. [Online]. Available: http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf

[10] D. Kudenko, "Special Issue on Transfer Learning," *KI*, vol. 28, no. 1, pp. 5–6, 2014.

[11] R. Vilalta, C. Giraud-Carrier, P. Brazdil, and C. Soares, "Transfer Learning," in *Encyclopedia of Machine Learning and Data Mining*, C. Sammut and G. I. Webb, Eds. Springer US, 2017, pp. 1283–1283.

[12] M. M. Richter and R. Weber, *Case-Based Reasoning: A Textbook*. Springer, 2013.

[13] M. Klenk, D. W. Aha, and M. Molineaux, "The Case for Case-Based Transfer Learning," *AI Magazine*, vol. 32, no. 1, pp. 54–69, 2011.

[14] M. Minor, R. Bergmann, J.-M. Müller, and A. Spät, "On the Transferability of Process-Oriented Cases," in *Case-Based Reasoning Research and Development - 24th International Conference, ICCBR 2016, Atlanta, GA, USA, October 31 - November 2, 2016, Proceedings*, ser. Lecture Notes in Computer Science, A. K. Goel, M. B. Díaz-Agudo, and T. Roth-Berghofer, Eds., vol. 9969. Springer, 2016, pp. 281–294.

[15] M. Minor and M. Herold, "Discovery of Analogical Knowledge for the Transfer of Workflow Tasks," in *Artificial Intelligence & Knowledge Engineering (AIKE 2020)*. IEEE Computer Society Press, 2020, pp. 69–75.

[16] M. Herold and M. Minor, "Ontology-based transfer learning in the airport and warehouse logistics domains," in *Workshop Proceedings of ICCBR 2019*, Otzenhausen, Germany, 2019.

[17] Y. Hong and Lepage, Yves, "Production of Large Analogical Clusters from Smaller Example Seed Clusters Using Word Embeddings." in *Case-Based Reasoning Research and Development - 26th International Conference, ICCBR 2018, Stockholm, Sweden, July 9-12, 2018, Proceedings*, vol. LNCS 11156. Springer, 2018, pp. 548–562.

[18] G. Müller and R. Bergmann, "Generalization of workflows in process-oriented case-based reasoning," in *The Twenty-Eighth International Flairs Conference*. AAAI Press, 2015, pp. 391–396.

[19] V. Dufour-Lussier, J. Lieber, E. Nauer, and Y. Toussaint, "Text adaptation using formal concept analysis," in *Case-Based Reasoning. Research and Development*, 2010, pp. 96–110.

[20] B. Fuchs, J. Lieber, A. Mille, and A. Napoli, "Differential adaptation: An operational approach to adaptation for solving numerical problems with CBR," *Knowledge-Based Systems*, vol. 68, pp. 103–114, Sep. 2014.

[21] D. Leake and X. Ye, "On Combining Case Adaptation Rules," in *International Conference on Case-Based Reasoning*. Springer, 2019, pp. 204–218.

[22] A. Polyvyanyy, S. Smirnov, and M. Weske, "Business Process Model Abstraction," in *Handbook on Business Process Management 1*, J. vom Brocke and M. Rosemann, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 147–165.

[23] M. Reichert and B. Weber, *Enabling Flexibility in Process-Aware Information Systems: Challenges, Methods, Technologies*. Springer, 2012.

[24] S. Overhage, D. Q. Birkmeier, and S. Schlauderer, "Quality Marks, Metrics, and Measurement Procedures for Business Process Models: The 3QM-Framework," *Business & Information Systems Engineering*, vol. 4, no. 5, pp. 229–246, Oct. 2012.

[25] S. Overhage, S. Schlauderer, and D. Q. Birkmeier, "Sind Ereignisgesteuerte Prozessketten besser für Fachanwender geeignet als UML Aktivitätsdiagramme? Eine empirische Untersuchung," in *10th International Conference on Wirtschaftsinformatik, A. Bernstein and G. Schwabe*, 2011, pp. 745–755.

[26] A. Richter, *Gepäcklogistik auf Flughäfen: Grundlagen, Systeme, Konzepte und Perspektiven*. Springer-Verlag, 2012.

[27] "SAP Best Practices Explorer," 2019, last visit: May 04, 2021. [Online]. Available: https://rapid.sap.com/