What Shall I Do Next?

Intention Mining for Flexible Process Enactment

Elena V. Epure[1, 2], Charlotte Hug[1], Rebecca Deneckère[1], Sjaak Brinkkemper[2]

[1] Centre de Recherche en Informatique, Université Paris I Panthéon-Sorbonne, France
{charlotte.hug, rebecca.deneckere}@ univ-paris1.fr

[2] Department of Information and Computing Sciences, Utrecht University, the Netherlands
{E.V.Epure, S.Brinkkemper}@ uu.nl

Abstract. Besides the benefits of flexible processes, practical implementations of process aware information systems have also revealed difficulties encountered by process participants during enactment. Several support and guidance solutions based on process mining have been proposed, but they lack a suitable semantics for human reasoning and decisions making as they mainly rely on low level activities. Applying design science, we created *FlexPAISSeer*, an *intention mining* oriented approach, with its component artifacts: 1) *IntentMiner* which discovers the intentional model of the executable process in an unsupervised manner; 2) *IntentRecommender* which generates recommendations as intentions and confidence factors, based on the mined intentional process model and probabilistic calculus. The artifacts were evaluated in a case study with a Netherlands software company, using a Childcare system that allows flexible data-driven process enactment.

Keywords: intention mining, process mining, flexible processes, process aware information systems, process recommendations

1 Introduction: Intention Mining

Process Aware Information Systems (PAIS) form a category of information systems, highly adopted by organizations, defined by van der Aalst as "software systems that manage and execute operational processes involving people, applications, and/or information sources on the basis of process models" [3]. In flexible PAISs which support process changes and variations as result of the external and internal environment, the primacy of humans has been highly acknowledged [8, 18, 26]. The agency characteristic of process participants, entailing their freedom of decision making during process enactments becomes thus central as it impacts the process outcomes. For instance, let's consider an e-commerce application: when a net surfer adds a product to his basket, several choices are offered: he can select another product, handle his basket, create his customer account etc. Following the flexibility of the studied process, the decision making complexity can increase rapidly. An experienced process participant who is highly aware of the process is able to make a better decision about the action to execute next under specific constraints or how to model a process fragment at run-time. In contrast, this can be very challenging for a less experienced process participant or for a process participant who faces a very dynamic and complex process environment [2, 23, 26]. If the resulting problem-prone situation is ignored, the adoption of flexible processes can instead have a negative impact on organizations.

Consequently, in this paper, we focus on tackling the difficulties of process participants when enacting flexible processes in PAISs by proposing *FlexPAISSeer*, a solution based on *intention mining* [12]. A PAIS enables the process discovery in a bottom-up manner by capturing events during enactment. Practically, this is realized by process mining, whose main goal is “to discover, monitor and improve real processes” by transforming the event logs data in valuable knowledge [3]. The mining result is most often a process model. Additionally, process mining has been used as a key technology in several approaches to support process participants during enactment [1, 22, 26]. While these solutions integrate process mining successfully, we consider the recommendations semantically not rich enough to support effective decision making meaning *effective criteria identification, development, and analysis of alternatives* [13]. The recommendations are formulated based on the mined process models which are frequently represented as control flows of low level activities. Therefore, to semantically enrich the recommendations, the mined process models must be enriched.

Through intention mining, we have the ambitious goal of extending process mining with a more suitable perspective for supporting humans in decision making, by mining the intentional process model from event logs and by using it for providing recommendations as intentions and confidence factors. We consider the intention a higher abstraction and a logical grouping of activities which captures their hidden goal: what the user wanted/want to achieve by following those activities. Human behavior is intentional by nature. Hence, making decisions based on intentions is closer to his natural reasoning mechanism. This topic has been extensively discussed in philosophy [4, 10], artificial intelligence [7, 16] and various areas of information systems, as requirements and enterprise engineering [15, 17, 18, 25, 31], and data mining [5, 27].

Once the process participant adopts an intention, he acts accordingly to achieve it [4]. Hence, the event log contains data about his intention. The research objectives regarding the unsupervised intention mining technique are: *the identification of the data which provides information about intentions* and *the identification of the intentional cluster of events associated with an intention and its naming*. We propose a general definition of *IntentMiner*, applicable for multiple systems while we also identify domain-specific aspects as the cost function in clustering and the intention naming. We propose *IntentRecommender* to *predict a set of intentions* based on the process model and the process participant trace, *each having associated a confidence factor*: a numerical value *aggregating the probability of the past occurrence of the full or partial sequence of intentions* (the trace and each predicted intention).

We used design science [11] collaborating with 42windmills, a software company located in Leiden, the Netherlands. We chose this research method as it addresses the relevance and acceptance of our created artifacts in the application domain. Accordingly, this paper is organized as follows: Section 2 describes the *FlexPAISSeer* approach and its artifacts design, Section 3 presents the artifacts development and demonstration in the case study context, Section 4 details the artifacts evaluation. Finally, Section 5 presents the conclusions and future works.

2 FlexPAISSeer: Enactment Support in Flexible PAISs

We identified the problem situation could be best tackled with a knowledge management approach. Thus, we chose the knowledge management cycle proposed by Wiig [28] for the FlexPAISSeer design which distinguished four phases: *Build* knowledge, *Hold* knowledge, *Pool* knowledge and *Use* knowledge.

IntentMiner is the central component of the *Build* and *Hold* knowledge phases. It consists of the intention mining technique that creates and embeds knowledge as follows: it mines all the existing event logs and generates the *intentional process model* enriched with meta-data regarding the frequencies of various process instances (steps 9, 4, 10-11 in Fig. 1); *IntentMiner* also transforms the current process instance in the *intentional process instance* to feed *IntentRecommender,* and uses it for updating the *intentional process model* (steps 1-5 in Fig. 1).

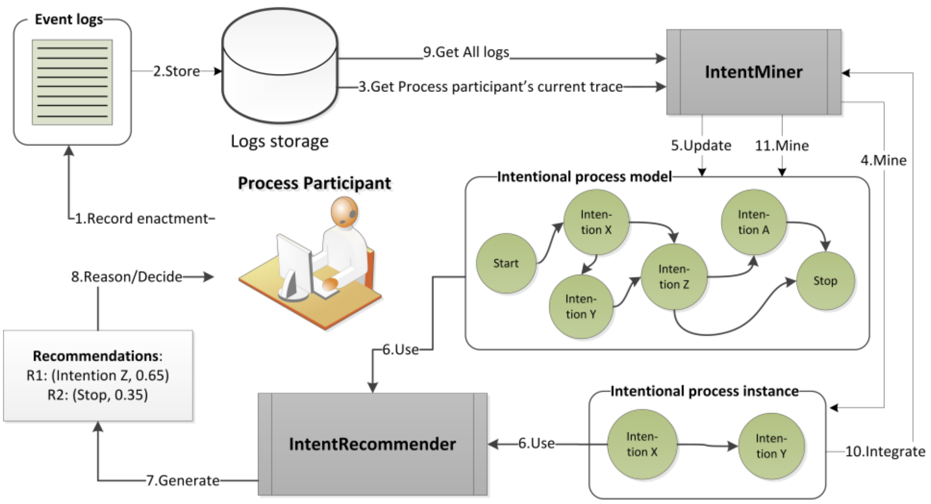


Fig. 1. FlexPAISSeer approach

IntentRecommender is the central component for the *Pool* knowledge phase: it assembles and reconstructs the intentional process instance and the process model as recommendations composed of *intentions* and *confidence factors* (steps 6-7 in Fig. 1). The *Use* knowledge phase concerns the *Process Participant* who can decide to enact considering the given recommendations (steps 8). However, the recommendations are not enforced, the Process Participant being free to enact the process differently when required by the situation at hand.

Further, we present *IntentMiner* and *IntentRecommender* with a focus on our design decisions and algorithms. The design decisions were created based on extensive literature review and interviews with the company before and during the project [9].

2.1 IntentMiner

The main design goal of *IntentMiner* is to discover the intentional process model from the traces of the process participants, by both mining their intentions and the flow between these intentions. We group these design decisions in the following categories: input-related and algorithm-related design decisions.

The *input-related design* focuses on the identification of the relevant data for mining the intentional process, a logging mechanism and a data extraction mechanism. After analyzing other process mining techniques [3], we decided to structure the event logs as in Table 1. Moreover, the mechanism extracting the data from the data source should produce event logs compliant with the XES standard for storing and exchanging logs [3] as it is the most used in the process mining domain.

Table 1. The definition of the event structure

Attribute	Description	Standard XES extension
Event Id	The event's unique identifier	Yes
Originator	The process participant's identifier (username or user Id)	Yes
Operation	The name of the operation identified by a verb	Yes
Timestamp	The date and time information of the produced event	Yes
Entity	The name of the entity type handled in the event	No
Trace Id	The trace's unique identifier	Yes
Lifecycle Transition	The name of the event' state during its lifecycle (applicable only for non-momentary events)	Yes
Process Context	Extra information, extracted from the system as key/value, relevant for intentions discovery (for example the entity Id)	No

The *algorithm-related design* is built to mine *elementary* intentions. We plan to extend *IntentMiner* to mine higher level intentions in the future. The *IntentMiner* algorithm consists of six steps, as shown in Fig. 2.

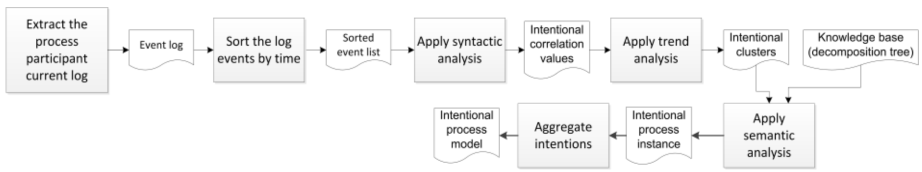


Fig. 2. The *IntentMiner* algorithm steps

We explain the *IntentMiner* algorithm by means of a semi-formal presentation. A full example is shown in section 3.2. Let P be the set of process participants for a specific PAIS. An intention I is said to be *elementary* if there exists a set of activities $A_I = \{a_1, a_2 ... a_t\}$, executed by a process participant $p \in P$ such that their *consecutive execution* leads to the achievement of I and only I.

Each activity $a_i \in A$ is associated to an event $e_i \in E$ which is logged during its execution. Thus, we define the *intentional cluster* as the set of events $C_I = \{e_1, e_2 ... e_t\}$

logged during the consecutive execution of their corresponding activities $A_I = \{a_1, a_2... a_t\}$, which leads to the achievement of the elementary level intention I.

Let L be ***a log of events ordered by time***, recorded ***for a process participant*** $p \in P$. Practically, L represents a series of events corresponding to a series of activities which were executed for realizing a series of intentions. Therefore, the log can be transformed in a series of intentional clusters $L = \{C_{Ik}: C_{Ik}$ *identifies* I_k, *for every* k, $1 \leq k \leq n\}$. Consequently, the first goal of the *IntentMiner* algorithm is to mine the intentional clusters and to extract the associated intention out of each cluster.

As mentioned earlier, each event e is described by a set of attributes, $AT_e = \{at_{ek}$: *for some* k, $1 \leq k \leq m\}$. This data, contained in the event structure, gives information about the realized intention. We define the *intentional correlation* as a function [23] applied on two consecutive events for quantifying the similarity with regard to the unknown intention I: $f(at_i, at_{i+1}) = \sum_{i=1..m} \alpha_k * g(at_{e_ik}, at_{e_{i+1}k})$, e_i, $e_{i+1} \in L$ where $g(at_{e_ik}, at_{e_{i+1}k}) = 1$ *if* $at_{e_ik} \equiv at_{e_{i+1}k}$, *0 otherwise* and $0 \leq \alpha_k \leq 1$. The coefficient α_k is introduced to differentiate the contribution of two attributes to the total correlation value. For example, two consecutive events that refer to the same *entity instance* have a much stronger correlation than two events that refer to the same *entity type*.

This introduces the third step of the algorithm: discovering the intentional clusters [24] with ***syntactic analysis*** which consists in the application of the function f on each pair of consecutive events belonging to the input log L. In this way, the log is transformed in a series of intentional correlation values. Then, the normalization of the series is realized by subtracting from each correlation value the minimum correlation value discovered in the set, until this minimum becomes null.

The fourth step is the ***trend analysis*** built on the observation that the progressive achievement of the intention [4, 7] is captured by the trend in the correlation values as follows: an *increasing trend* marks the progressive realization of an intention while a *change in trend* from increasing to decreasing or a *null correlation* value delineates two intentions. We analyzed multiple event logs of different applications and observed that two consecutive events belonging to an intention had a similar process context and a higher correlation value. Contrarily, if two events were triggered as a result of achieving two different intentions, they had different process context and a low or null correlation value. The result is the discovery of the intentional clusters.

Once the intentional clusters are identified, the further step is the intention extraction and naming by applying the ***semantic analysis*** [20] for each C_{Ik}, $1 \leq k \leq n$. A predefined knowledge base is created as a decomposition tree (see example in Fig. 4) populated with a starting set of known intentions and activities. The activities are always positioned in leaves and they could belong to multiple intentions. An intention could be standalone or a sub-intention of another intention (high level intention). The extracted intention for a cluster is the one on the *lowest level in the tree* that covers the *maximum number of known activities* of that cluster. The first implication is that an intention can be discovered even if not all activities are known in the knowledge base. The second implication is that a cluster could represent a different intention which is not yet known and stored in the knowledge base. An expert as a process administrator being responsible for process definition and implementation should review the mined process instance and the intentional clusters to decide if the knowledge base should be updated with new intentions or activities. The flow between intentional clusters describes the *flow between intentions*, thus obtaining the *intentional process instance*.

The final step is the ***aggregation*** of the mined process instance in the intentional process model: new mined intentions and transitions are added, and the transitions frequencies are increased. Thus, we obtain the updated intentional process model (step 5 in Fig. 1) that is further used by *IntentRecommender* for providing up-to-date recommendations to the process participants.

2.2 IntentRecommender

The leading design decision of *IntentRecommender* was to provide recommendations at *the intentional level* as we considered it could offer a more effective support to process participants in making decisions. This enables a more effective support for the identification of the decision criteria, the developing of the decision alternatives and the analysis of the decision alternatives.

The second design decision was to provide recommendations according to the *as-is intentional process model*, discovered by *IntentMiner* instead of using a pre-defined process model which might not be exactly followed by the process participants in practice. Moreover, *IntentMiner* transforms the process participant's partial trace of events in a flow of intentions which is given as input to *IntentRecommender* and is also used for *updating* the intentional process model.

The third design decision was to provide recommendations that contain information about the behavior of *other process participants* in a similar or identical process enactment situation, through *a confidence factor* [6]. The confidence factor is a numerical value attached to the recommendation, which quantifies *the match* and *the frequency* of the current process participant log based on the known process data.

Providing recommendations starting from a flow of intentions $F = \{I_1 \rightarrow \ldots \rightarrow I_n\}$, $n \geq 1$ is a matter of prediction, having, as prior knowledge, the intentional process model. A recommendation is the next predicted intention, $I_{predicted}$, which has attached the confidence factor $CF_{Ipredicted}$. We focus further on describing the two main parts of *IntentRecommender*: the prediction and the confidence factor computation.

The *prediction* is the identification of the next intentions based on the input flow of intentions, *F*, and the process model. *IntentRecommender* consists of three steps:

- **Discover the set of intentions**, $SI_{predicted}$, that are directly reachable from the last intention I_n, $n \geq 1$ of the flow *F*: $SI_{predicted} = \{I_{predicted} : I_n \rightarrow I_{predicted}, n \geq 1$ *exists in the intentional process model*$\}$.
- For each $I_{predicted} \in SI_{predicted}$, **create the set of predecessors** consisting of the intentions found in the flow, sorted by time in descending order, $P_{Ipredicted} = \{I_n \ldots I_1\}$ $n \geq 1$. However, there are two possible issues. First, the path described by *F* cannot be fully found in the intentional process model. In this case, $P_{Ipredicted}$ is modified to contain only those intentions which describe an existing flow to $I_{predicted}$ in the intentional process model: $P_{Ipredicted} = \{I_n \ldots I_k\}$, $n, k \geq 1$ *and* $I_k \rightarrow \ldots \rightarrow I_n \rightarrow I_{predicted}$ *exists in the intentional process model*. Second, an intention could appear several times in

$P_{Ipredicted}$. In this case, the interpretations could be: (i) an intention was among its list of predecessors, thus influencing its future occurrence; or (ii) the flow exposed different ways of achieving that intention. By invoking Occam's razor [18], which specifies that the model with the simple assumptions should be selected, we chose the interpretation (ii). This implies another constraint on $P_{Ipredicted}$: each intention in the sequence of predecessors must be *unique* and *different* from $I_{predicted}$.

- For each $I_{predicted} \in SI_{predicted}$, **compute the confidence factor** $CF_{Ipredicted}$ (1) having the possibility to tune it through the coefficients α and β, $0 \leq \alpha, \beta \leq 1$. The process administrator can decide the frequency of a certain path is more important through α's value, or the match of a certain path is more important, through β's value.

$$CF_{Ipredicted} = \alpha * P(\{I_{predicted}\} + P_{Ipredicted}) + \beta * L(\{I_{predicted}\} + P_{Ipredicted}) / L(F) \quad (1)$$

$$P(\{I_{predicted}\} + P_{Ipredicted}) = Probabability\ of\ I_k \rightarrow \ldots \rightarrow I_n \rightarrow I_{predicted}\ n,\ k \geq 1\ occurs \quad (2)$$

$$L(\{I_{predicted}\} + P_{Ipredicted}) = n - k + 2 = Length\ of\ I_k \rightarrow \ldots \rightarrow I_n \rightarrow I_{predicted}\ n,\ k \geq 1 \quad (3)$$

$$L(F) = n = Length\ of\ I_1 \rightarrow I_2 \rightarrow \ldots \rightarrow I_n \quad (4)$$

Every time a new process instance is mined, *IntentMiner* updates the tree T_I of each intention I with all the paths that lead to it and their frequencies. Based on the data maintained in T_I we compute the probabilities. The tree has a specific structure: a full discovered process instance that describes a path to I is stored in a leaf; then this path is recursively decomposed in shorter paths to I by removing one intention from the tail until there is nothing left to be removed. For example, let's consider *IntentMiner* discovers the following process instance: $I_1 \rightarrow I_2 \rightarrow \ldots \rightarrow I_k \rightarrow \ldots \rightarrow I_n$. The tree corresponding to the intention I_k is updated as follows: the leaf node $n_1 = I_1 \rightarrow I_2 \rightarrow \ldots \rightarrow I_k$ is created, then a new node $n_2 = I_2 \rightarrow \ldots \rightarrow I_k$ is created and linked to n_1 and so on until the root $r = I_k \rightarrow null$ is reached. During the path decomposition, it might happen that a node is already in the tree in which case only the link is created and the node frequency is incremented. Considering, $\#T_I$ the total number of mined paths that lead to the intention I, we have:

$$P(\{I_{predicted}\} + P_{Ipredicted}) = \text{Frequency}(\{I_{predicted}\} + P_{Ipredicted}) / \#\text{T}_{\text{Ipredicted}} \quad (5)$$

We compute the confidence factor (1) by using (2-5) and create the recommendation. The computation is realized for each intention of $SI_{predicted}$ (step 7 in Fig. 1).

3 The Demonstration of FlexPAISSeer

3.1 Case Study of an Enterprise Software Product

To demonstrate the validity of our *FlexPAISSeer* approach, we conducted a revelatory single case study [30]. We selected the case company considering its suitability (the support of flexible processes through its software product): the Childcare system developed by 42windmills used by several child day care centers in the Netherlands.

Childcare is created with the company's main product: a platform which generates software following a model driven approach. The platform together with a Web-based application designer enables the customers to design, preview, generate, re-design and deploy a wide variety of business applications.

Even if some processes of the created business application can be automated, most of them are flexible, being enacted in a data-centered, human-driven manner. In a data-centered approach, the elements that influence the process enactment are entities, entity attributes and entity relationships (as shown in Fig. 3). A transition in the process enactment is triggered by a change in the entity state through user forms [23]. An exploratory interview reported that the high Childcare's complexity combined with the flexible processes support created problems: inexperienced process participants often enacted inefficiently the processes or made mistakes because of the scenario complexity.

To ensure the research reliability, construct and internal validity, we defined a case study protocol beforehand and we used multiple sources of evidence which were carefully documented in a case study database. We conducted exploratory interviews with the CTO, the Childcare consultant and the platform architect to deepen the understanding of the problem the company was facing, to study more thoroughly the technical aspects of the product and to validate the suitability of the proposed solution. The external validity, concerning the generalization of the results, is more difficult to guarantee after a single-case study. However, given the generic type of the administrative application and the standard technology employed, we can consider the case settings as a good representative for an enterprise software product [29].

We developed prototypes for both *IntentMiner* and *IntentRecommender* using Microsoft C#.NET language, Visual Studio 2012 and Microsoft SQL Server 2005. We choose these technologies to ensure an easier integration of the artifacts with the company's product. Though generic, the prototypes are not officially released as they must be integrated and some parts still need improvements.

3.2 IntentMiner's demonstration

IntentMiner is demonstrated for the *Request child care* process. In Fig. 3, we present a partial entity model involved in the registration process. As mentioned, the enactment of the Childcare's processes is based on the entity states transitions.

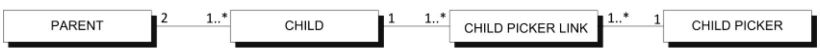


Fig. 3. Entities involved in the registration process

In Table 2a, we present a possible process instance of the *Request child care* process. We defined the intentional correlation function (used in Table 2b) to take into account the following event attributes: the trace Id (for Childcare being the Child entity Id), the entity type and the entity Id (which is stored as contextual information):

$$f(e_i, e_{i+1}) = 0.5 * g(EntityId_{e_i}, EntityId_{e_{i+1}}) + 0.3 * g(EntityType_{e_i}, EntityType_{e_{i+1}}) + 0.2 * g(TraceId_{e_i}, TraceId_{e_{i+1}})$$

The intentional correlation for each pair of consecutive events is calculated (*syntactic analysis*, Table 2b). According to the defined rules of trend analysis, the intentional clusters are formed (*trend analysis*, Table 2c).

Table 2. Exemplification of the *IntentMiner* algorithm

(a) Extract process participant log and sort by timestamp	$E_1 \rightarrow e_2 \rightarrow e_3 \rightarrow e_4 \rightarrow e_5 \rightarrow e_6 \rightarrow e_7$ **e_1:** Read the list of Child entities **e_2:** Read the Child entity with Id C1 **e_3:** Update the Child entity with Id C1	**e_4:** Read the list of Child entities **e_5:** Read the Child entity with Id C2 **e_6:** Read the list of Parent entities **e_7:** Read the list of Child Picker entities
(b) Apply syntactic analysis	$f(e_1, e_2) = 0.5*0 + 0.3*1 + 0.2*0 = 0.3$; $f(e_2, e_3) = 0.5*1 + 0.3*1 + 0.2*1 = 1$; $f(e_3, e_4) = 0.5*0 + 0.3*1 + 0.2*0 = 0.3$;	$f(e_4, e_5) = 0.5*0 + 0.3*1 + 0.2*0 = 0.3$; $f(e_5, e_6) = 0.5*0 + 0.3*0 + 0.2*0 = 0$; $f(e_6, e_7) = 0.5*0 + 0.3*0 + 0.2*0 = 0$;
(c) Apply trend analysis	$C_{I1} = \{e_1, e_2, e_3\}$ $C_{I2} = \{e_4, e_5\}$ $C_{I3} = \{e_6\}$ $C_{I4} = \{e_7\}$ e_1 and e_2 have a correlation higher than 0 and are grouped in C_{I1}. The correlation of e_3 with e_2 is higher than its correlation with e_4, thus e_3 is added to C_{I1} too. The first change in trend is identified (the decrease from 1 to 0.3) so C_{I2} is formed, to which e_4 is added. Further, the correlation of e_5 with e_4 is higher than its correlation with e_6 so e_5 is added to C_{I2}. The change in trend (the decrease from 0.3 to 0) marks the creation of C_{I3} consisting of e_6. Finally, because the correlation of e_6 with e_7 is zero, C_{I4} consisting of e_7 is built.	
(d) Apply semantic analysis	I_1 (Update Child entity) $\rightarrow$ I_2 (Read Child entity) $\rightarrow$ I_3 (Read Parent entities) $\rightarrow$ I_4 (Read ChildPicker entities)	

Once we discover the intentional clusters, we identify the intention associated with each of them (*semantic analysis*, Table 2d). For this, we pre-defined a knowledge base during the Childcare analysis. For each entity type, a decomposition tree based on Fig. 4 was created. The intention composition is generic for all the Childcare entities because of the software's nature, being model driven generated.

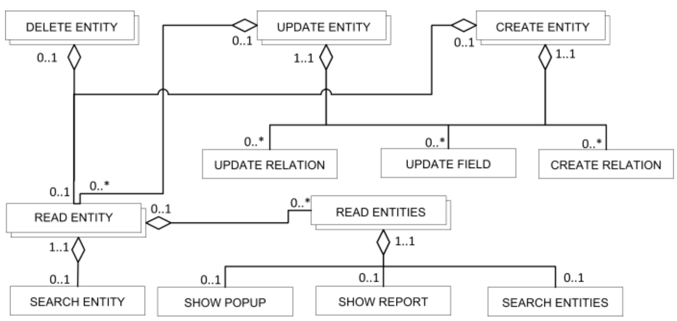


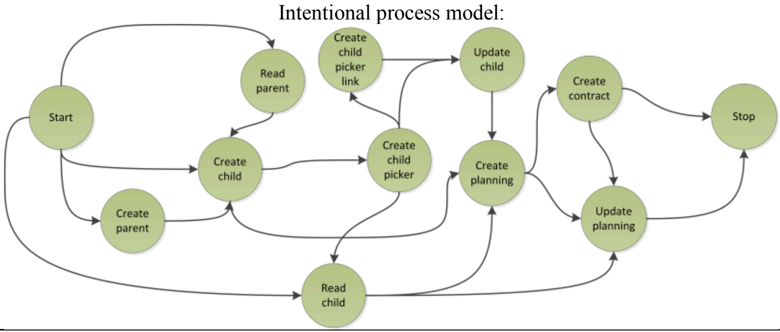
Fig. 4. Intention composition for semantic analysis

The tree contains five elementary intentions (*Create entity*, *Read entity*, *Read entities*, *Update entity*, and *Delete entity*) and seven activities (*Update relation*, *Update field*, *Create relation*, *Search entity*, *Search entities*, *Show popup* and *Show report*).

3.3 IntentRecommender's demonstration

The *IntentRecommender* algorithm is also demonstrated further. The inputs consist of the intentional process model in Table 3, and the trees associated to each intention (discovered with *IntentMiner*). The process relates to the Childcare registration, as only this part was mined during the experiments. When the process participant invokes *IntentRecommender*, the input trace is extracted. The intentional process instance does not necessary match the intentional process model as our goal is process discovery and not conformance checking [3].

Table 3. Recommendation algorithm – running example

Intentional process model: 
(a) Process participant's intentional process instance: **F**: $\mathbf{I_{t1}}$ (Read parent list) → $\mathbf{I_{t2}}$ (Read parent) → $\mathbf{I_{t3}}$ (Create child) → $\mathbf{I_{t4}}$ (Create child picker)
(b) Discover the set of intentions directly reachable from the last intention, $\mathbf{I_{t4}}$: $\mathbf{SI_{predicted}}$ = { $\mathbf{I_{p1}}$ (Create child picker link), $\mathbf{I_{p2}}$ (Update child), $\mathbf{I_{p3}}$ (Read child) }
(c) Compute the confidence factor exemplified for $\mathbf{I_{p3}}$ (Read child) : $\mathbf{P_{Ip3}}$ = { I_{t4}, I_{t3}, I_{t2} } $\mathbf{CF_{Ip3}}$ = 0.5 * P ({I_{p3}} + P_{Ip3}) + 0.5 * L ({I_{p3}} + P_{Ip3}) / L (F) = 0.5 * P ($I_{p3} \leftarrow I_{t4} \leftarrow I_{t3} \leftarrow I_{t2}$) + 0.5 * 3 / 4 where P ($I_{p3} \leftarrow I_{t4} \leftarrow I_{t3} \leftarrow I_{t2}$) is calculated according to the formula (5), considering the information extracted from T_{Ip3} (Frequency ($I_{p3} \leftarrow I_{t4} \leftarrow I_{t3} \leftarrow I_{t2}$) and $\#T_{Ip3}$)

The first step of the algorithm consists in the identification of the last intention of the process participant: I_{t4} (Table 3a). Further, the intentions that are directly reachable from I_{t4} are identified in the model ($SI_{predicted}$ in Table 3b). The path to I_{p3} is formed according to the input trace and, then, the longest sub-sequence of this path found in the model is extracted (P_{Ip3} in Table 3c). Based on this maximal sequence, the confidence factor is calculated and the first recommendation $\boldsymbol{R_3}$**:** $\boldsymbol{(I_{p3}, CF_{Ip3})}$ is formulated. We repeat step (c) for the other left intentions – I_{p1} and I_{p2}, in a similar manner.

4 Preliminary evaluation of FlexPAISSeer

The evaluation of the artifacts consisted in an experiment with 10 participants, interacting with Childcare [9]. Previous experience was not required, though we provided a tutorial about the application usage in advance. The participants had to be able to express themselves in English and to have basic computer skills. An experiment lasted around two hours and consisted of two parts. In the first part, we evaluated *IntentMiner*. The process participants were asked to perform different tasks while they were verbalizing their intentions in the presence of the interviewer. The second part focused on the *IntentRecommender*'s evaluation through structured interviews.

4.1 IntentMiner's evaluation

We evaluated *IntentMiner* following the Confusion matrix approach, built on the concept of instances classification, realized by a classifier system [14]. In our context, the classifier system was *IntentMiner* and the instance was the discovery/existence of an intention. An intention discovery was classified as positive when *IntentMiner* discovered it from event logs and negative otherwise (*Classified instance,* Table 4). An intention existence was positive if the process participant confirmed he had that intention and negative otherwise (*Actual instance*, Table 4).

Table 4. Confusion matrix for intention mining

Results of the case study		**Classified instance**	
		Negative: an intention I is not discovered	*Positive*: an intention I is discovered
Actual instance	*Negative*: the process participant does not have the intention I	#TN (the number of *true negative* instances): **0**	#FP (the number of *false positive* instances): **47**
	Positive: the process participant has the intention I	#FN (the number of *false negative* instances): **3**	#TP (the number of *true positive* instances): **105**

The participants verbalized 108 intentions out of which 105 (#TP) were correctly discovered by *IntentMiner* and 3 (#FN) were not. *IntentMiner* discovered 152 intentions out of which 47 (#FP) were negative as the process participants did not have those intentions. The number of true negative instances was always 0. Since a process participants had no intention and did not act accordingly, there were no logs based on which the intention could be mined.

Given an intention discovered by *IntentMiner*, the **average precision** (**Precision = #TP / (#TP + #FP)**) of *being correct* was **0.69**. Furthermore, *IntentMiner* mined the process participants' intentions in **0.97** cases. This was measured by the **average recall** (**Recall = #TP / (#TP + #FN)**). These results are very satisfactory for a first time use of our *unsupervised* intention mining technique. Khodabandelou et al. [12] reported an average recall of 0.93 and an average precision of 0.97 for their *supervised* intention mining technique based on Hidden Markov Models. The precision was considerably better given the fact the classifier was trained in advance.

For getting more insights into how we could improve *IntentMiner*, we analyzed thoroughly each log and noticed two recurring issues. First, *IntentMiner* discovered several intentions even if the activities behind them were not intended for that, but for higher intentions. For example, *Explore the Childcare application* was mined as reading different entities. Second, several activities were triggered by the system on behalf of the process participant thus were mined as process participant's intention. Every time a new *Child* entity was created, an empty *ChildPicker* entity was also created by the system; these events were mined as two separate intentions but in reality it was only one intention: *Create Child entity*.

In conclusion, the functional requirements of *IntentMiner* were completely satisfied as proved by its usage without errors in the experiments. *IntentMiner* can be used for mining intentional processes but a further review of the results by the process administrator is required as they might not be completely precise.

4.2 IntentRecommender's evaluation

Unit tests were used for validating the *IntentRecommender* functionality. The non-functional evaluation of *IntentRecommender* was reduced to the following phases:

1. The non-functional evaluation of *IntentMiner* as the quality of the produced output (used as input for *IntentRecommender*) influences the quality of the recommendations. This was covered in section 4.1.
2. The analysis of the perceived effectiveness of recommendations as intentions and confidence factors on decision making support by the process participants.

The second phase consisted in a *structured interview based on a questionnaire.* It had various conceptual scenarios inspired from Childcare which required the process participants to make decisions. Besides, there were also general and confidence factors-related questions. The hypotheses guiding the evaluation of *IntentRecommender* were:

H1: The recommendations given as intentions improve the support for decision making by improving the support for *the criteria identification*.

H2: The recommendations given as intentions improve the support for decision making by improving the support for *the alternatives formulation*.

H3: The recommendations given as intentions improve the support for decision making by improving the support for *the alternatives analysis*.

H4: The confidence factors included in the *recommendations* improve the support for decision making.

In the first scenario, without any recommendations, the participants were asked to identify what they believed they should do next. The participants identified the high level intention (to update the child planning) without problems. When asked to give details about the specific process steps, they were able to cover only a part of them (even if they were revealed in the tutorial provided in the beginning). After the first set of recommendations as activities was given, most of the participants chose the option that was aligned with their previously identified intention except for two: one changed his intention from updating the child to updating the planning and the other stated that his new decision was based on the confidence factors.

After the intention behind the recommended set of activities was revealed, 9 of 10 participants agreed that the decision making was easier in that case motivating the answer as follows: the intention helped to clarify the activities to be performed, helped to validate an intention adopted in advance and provided information about the context. One participant disagreed with the added value by invoking the efficiency in following activities without reasoning about intentions (step by step guidance).

Consequently, it was shown that the recommendations as intentions improved the support for the criteria selection (*H1*) in two ways: *by the intention realization* when the process participants adopted the suggested intention and made the decision accordingly; *by the intention validation* when the process participants checked if the suggested intention was the same with the one they already formulated in their mind.

The aim of the next scenario was to compare the decision making support when recommendations were given as intentions and then as activities. 7 of 10 participants found the set of recommendations given as intentions helpful for supporting the decision making while 3 disagreed: two preferred a step by step guidance and one found it hard to make the decision because there were too many recommendations in the set. Analyzing the collected data, we noticed that most of the participants wanted support in interacting with the application and *preferred the recommendations as intentions* to recommendations as activities. Thus, *H2 and H3 seemed to be supported*.

The final questions were focused on the confidence factors. 6 of 10 participants disagreed that the numerical values attached to each recommendation influenced their decision. The main invoked reason was that there was no re-assurance the other participants enacted the process more efficiently or more effectively, to follow their behavior. Nevertheless, the other 4 participants agreed with the usefulness of the confidence factors and mentioned that their decision was influenced completely (following the others behavior) or partially (checking if the others reasoned similarly) by this. Consequently, *H4 could not be verified* based on the existing data.

5 Conclusion and future works

In this paper, our main goal was to create an improved approach for supporting process participants during flexible processes enactment, by offering recommendation based on an intentional process model. As process mining captures accurately how real life processes are enacted, we created *IntentMiner* to discover intentional process models automatically from event logs. The intentional process model was integrated in *IntentRecommender*, which after the evaluation in a case study, demonstrated its contribution to the problem solving. To sum up with, we consider the largest contribution of this research is the thorough study of the intentionality in the context of process enactment and its integration with process mining.

We intend to improve the evaluation of this approach as the evaluation of the artifacts was realized for only one case study with 10 participants. According to Yin [30] a more accurate evaluation should include at least 3 case studies. We will then conduct more case studies including other software products in different organizational settings. With more participants, we could do quantitative evaluation too.

IntentMiner can be improved to mine more accurately the intentions. The semantic analysis can be supported by ontologies and semantic annotations of the event logs which should also enable the mining of the non-functional intentions. Moreover, other machine learning algorithms for clustering can be explored, as self-organizing maps or genetic algorithms. *IntentMiner* in its current form requires several adaptations for being re-used by other applications (selection of event attributes relevant for the syntactic analysis, redefinition of the correlation function according to the selected event attributes, adaptation of the hierarchy of intentions for semantic analysis). These changes – triggered by specific cases – should be formalized in a method and supported by a tool to ease future adaptations. The intentional process models produced by *IntentMiner* are not as flexible as Map intentional process models [25]. We do not consider parallel intentions and refinement of intentions. Producing more complex intentional process models is one of our next steps. A ProM plugin for *IntentMiner* and *IntentRecommender* should be further developed. Official XES extensions also have to be proposed to integrate the concepts of process context and entity (Table 1).

Finally, *IntentRecommender* can be extended with an inference mechanism based on the Dynamic Bayesian Network [21], a more suitable probabilistic model for processes. This would allow an intention to be in its list of predecessors when calculating the confidence factors. The prototype should be released in a stable version and integrated in a PAIS to allow its runtime evaluation.

References

1. van der Aalst, W.M.P.: Decision Support Based on Process Mining. International Handbooks Information System, 637-657 (2008)
2. van der Aalst, W.M.P., Pesic, M., Schonenberg, H.: Declarative Workflows: Balancing Between Flexibility and Support. Computer Science - R&D, 99-113 (2009)
3. van der Aalst, W.M.P.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer (2011)
4. Bratman, M. E.: Intention, Plans, and Practical Reason. Cambridge University Press (1999)
5. Chen, Z., Lin, F., Liu, H., Liu, Y., Ma, W.-Y., Wenyin, L.: User Intention Modeling in Web Applications Using Data Mining. Journal of World Wide Web Internet And Web Information Systems 5(3), 181-191 (2002)
6. Cleger-Tamayo, S., Fernández-Luna, J.M., Huete, J.F., Tintarev, N.: Being Confident about the Quality of the Predictions in Recommender Systems. In: Serdyukov, P., Braslavski, P., Kuznetsov, S.O., Kamps, J., Rüger, S., Agichtein, E., Segalovich, I., Yilmaz, E. (eds.) ECIR 2013. LNCS, vol. 7814, pp. 411-422. Springer, Heidelberg (2013)
7. Cohen, P. R., Levesque, H.: Intention is choice with commitment. Journal of Artificial Intelligence 42(2), 213-261 (1990)
8. Ellis, C.A., Kim, K.: Process aware information systems: a human centered perspective. In: Dong, G., Lin, X., Wang, W., Yang, Y., Yu, J.X. (eds.) APWeb 2007 and WAIM 2007. LNCS, vol. 4505, pp. 39-49. Springer, Heidelberg (2007)
9. Epure, E.V., Hug, C., Deneckere, R., Brinkkemper, S.: Intention-mining: A solution to process participant support in process aware information systems. Technical report. Department of Information and Computing Sciences, Utrecht University (2013)

10. Fishbein, M., Ajzen, I.: Belief, Attitude, Intention, and Behavior: An Introduction to Theory and Research. Addison-Wesley, Reading MA (1975)
11. Hevner, A. R., March, S. T., Park, J., Ram, S.: Design Science in Information Systems Research. MIS Quarterly 28(1), 75-105 (2004)
12. Khodabandelou, G., Hug, C., Deneckère, R., Salinesi, C.: Supervised intentional process models discovery using Hidden Markov models. In: IEEE RCIS, pp.1-11. IEEE (2013)
13. Klein, G.A., Orasanu, J., Calderwood, R., Zsambok, C.E.: Decision making in action: Models and methods. Ablex Publishing (1993)
14. Kohavi, R., Provost, F.: On Applied Research in Machine Learning. Applications of Machine Learning and the Knowledge Discovery Process 30(2), 127-132 (1998)
15. Kueng, P., Kawalek, P.: Goal-Based Business Process Models: Creation and Evaluation. Journal of Business Process Management 3(1), 17-38 (1997)
16. Lacey, N., Hexmoor, H., Beavers, G.: Planning at the intention level. In: FLAIRS 2002, pp. 8-12. AAAI Press (2002)
17. van Lamsweerde, A., Letier, E.: From object orientation to goal orientation: A paradigm shift for requirements engineering. In: Wirsing, M., Knapp, A., Balsamo, S. (eds.) RISSEF 2004. LNCS, vol. 2941(1), pp. 325-340. Springer, Heidelberg (2004)
18. Lee, J., Kim, H., Seo, W., Kim, K., Kim, C.H.: Condition-based process patterns for modeling of human processes in Knowledge-intensive Business Services. Journal of Expert Systems with Applications 38, 4025-4038 (2011)
19. MacKay, D.J.C.: Information Theory, Inference and Learning Algorithms. Cambridge University Press (2003)
20. Medeiros, A.K.Alves, Pedrinaci, C., van der Aalst, W.M.P., Domingue, J., Song, M., Rozinat, A., Norton, B., Cabral, L.: An Outlook on Semantic Business Process Mining and Monitoring, In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM 2007 Workshops. LNCS, vol. 4806, pp 1244-1255. Springer, Heidelberg (2007)
21. Murphy, K.: Dynamic Bayesian Networks: Representation, Inference and Learning. UC Berkeley, Computer Science Division (2002)
22. Petrusel, R., Stanciu, P.L.: Making Recommendations for Decision Processes Based on Aggregated Decision Data Models. In: Abramowicz, W., Kriksciuniene, D., Sakalauskas, V. (eds.) BIS 2012. LNBIP, vol. 117, pp. 272-283. Springer, Heidelberg (2012)
23. Reichert, M., Weber, B.: Enabling Flexibility in Process-Aware Information Systems - Challenges, Methods, Technologies. Springer (2012)
24. Rokach, L.: A survey of Clustering Algorithms. Data Mining and Knowledge Discovery Handbook, 269-298 (2010)
25. Rolland, C., Prakash, N., Benjamen, A.: A Multi-Model View of Process Modelling. Journal of Requirements Engineering 4(1), 169-187 (1999)
26. Schonenberg, H., Weber, B., Dongen, B., van der Aalst, W. M. P.: Supporting flexible processes through recommendations based on history. In: Dumas, M., Reichert, M., Shan, M. C. (eds.) BPM 2008. LNCS, vol. 5240, pp. 51-66. Springer, Heidelberg (2008)
27. Song, I., Diederich, J.: Intention extraction from text messages. In: Wong, K.W., Mendis, B.S.U., Bouzerdoum, A. (eds.) Neural information processing: theory and algorithms 2010. LNCS, vol. 1, pp. 330-337. Springer-, Heidelberg (2010)
28. Wiig, K.M.: Case Study Research: Design and Methods. Journal of Knowledge Management 1(1), 6-14 (1997)
29. Xu, L., Brinkkemper, S.: Concepts of Product Software. European Journal of Information Systems 16(5), 531-541 (2007)
30. Yin, R. K.: Case Study Research: Design and Methods. Sage Publications (2009)
31. Yu, E. S. K.: Towards modelling and reasoning support for early-phase requirements engineering. In: IEEE Int. Symp. on Requirements Engineering, pp. 226-235. IEEE (1997)