Jacques Simonin*, Julie Soulas and Philippe Lenca

Activity Monitoring Process based on Model-Driven Engineering – Application to Ambient Assisted Living

Abstract: The supervisor of the activities of a system user should benefit from the knowledge contained in the event logs of the user. They allow the monitoring of the sequential and parallel user activities. To make event logs more accessible to the supervisor, we suggest a process mining approach, including first the design of an understanding model of the activities of a system user. The model design is based on the relationships between the event logs and the activities of a system user. An intervention model completes the understanding model to assist the supervisor. The intervention model enables an action of the supervisor on the critical activities, and the detection of anomalies. The models are automatically designed with a model-driven engineering approach. An experiment on a smart home system illustrates this tooled design, where the supervisor is a medical or paramedical staff member.

Keywords: Monitoring process, event log, understanding model, intervention model, model-driven engineering.

2010 Mathematics Subject Classification: 93A30, 91B74, 97K80.

DOI 10.1515/jisys-2014-0143
Received September 30, 2014; previously published online February 21, 2015.

1 Introduction

This paper focuses on the monitoring of the activities of a system user. The objective is to assist the supervisor of these activities with an as-is description (what activity is carried out, and when) and a to-be assistance (detection of anomalies in relation to the as-is description). This assistance is based on event logs specifying the observed start and end times of the user activities.

A typical application field is the monitoring of health-related behaviours of elderly people in their home, in order to assess their ability to keep living on their own. Indeed, with the development of sensor technologies, Smart Homes and Ambient Assisted Living systems have attracted a lot of attention during the last decade (see, e.g., the surveys [3, 13], and the projects CASAS [2] and Domus [12]). The sensors disseminated in such apartments produce a trace of the activities occurring there. This trace can be used for the monitoring of the elderly: the supervisor, a physician for example, can understand what is going on in the home, and whether the activities characterise a normal and safe behaviour. In particular, the search for frequent and periodic patterns or sequences is of great interest [5, 16, 17], as they highlight relationships between activities.

A system can be considered as supporting one or more activities of a user. This is the case in the Enterprise Architecture framework of Zachman, where the system is specified in the system viewpoint and the

***Corresponding author: Jacques Simonin,** Institute Mines-Telecom, Télécom Bretagne, UMR CNRS 6285, Lab-STICC, Université Européenne de Bretagne, 29238 Brest, France, e-mail: jacques.simonin@telecom-bretagne.eu
Julie Soulas and Philippe Lenca: Institute Mines-Telecom, Télécom Bretagne, UMR CNRS 6285, Lab-STICC, Université Européenne de Bretagne, 29238 Brest, France

activities in the business viewpoint [22]. The support relationship established between these viewpoints may be considered as an alignment between a model of the system, and a model of the activities making up the process [15]. The models of the processes enable an evaluation of the performances of the organisation [6].

The performance of a monitored process needs an evaluation: amount of time between two activities, number of activities carried out in parallel, etc. Such an evaluation is based on the transitions between activities [1], which are collected in the event logs. However, although they are relevant for a process mining expert, these logs are likely overwhelming for the supervisor of activities of a system user [7].

Process mining is an approach centred on the processing of event logs [18]. It is more accessible than a data mining approach for the supervisor, as it is closer to the processes. Process mining is, moreover, associated with tooling for the extraction of process-related knowledge [10]. Dealing with the complexity of the event logs is, however, the main difficulty for this kind of tool.

The complexity of event log processing could benefit from an understanding model and an intervention model [8]. The understanding model (an as-is model, describing what the user did) represents a process and is deduced from the alignment between an event log model (system viewpoint) and a process model (business viewpoint). The intervention model is a to-be model (what is monitored by the supervisor), enabling the supervisor to analyse the activities of the system user. The understanding and intervention models are designed through rules, aiming at the design of relevant models for process mining [19]. These models contain some instantiation of concepts (the concept of activity is, e.g., instantiated by the *go to bed* activity) that are represented in a meta-model, following a Model-Driven Engineering (MDE) approach. The rules are parameterised by these concepts. The supervising tool is based on the implementation of these rules.

Figure 1 presents the understanding and intervention models. The purpose of the monitoring tool is the evaluation of the distance between the observed activities or activity sequences of the system user, compared to the expected ones:

– {understanding model}∩{intervention model} = {both observed and expected activities or sequences};
– {understanding model}-{intervention model} = {real, but unexpected activities or sequences: the system user did something that was not expected (this behaviour can be added to the next intervention model)};
– {intervention model}-{understanding model} = {expected activities or sequences, which are not observed: the system user did not carry out the expected activities (this behaviour can be removed from the next intervention model)}.

The understanding model is the transformation of an event log model into a process model. The intervention model results from a transformation of the understanding model based on the monitoring needs of the supervisor. The MDE approach, especially model transformation [14], is thus a powerful tool for process mining. As

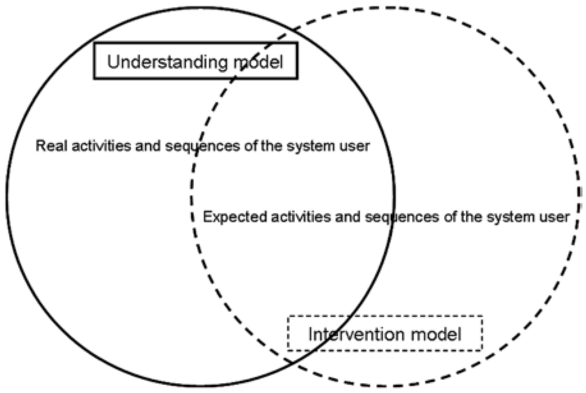
Understanding model
Real activities and sequences of the system user
Expected activities and sequences of the system user
Intervention model

Figure 1: Understanding Model and Intervention Model.

recommended of the MDE approach [9], the model transformation implementations are based on the operational QVT language (Query/View/Transformation). Our contribution allows the supervisor of the activities of a system user to monitor the following sequences of activities:

– The orderly succession of activities of a system user;
– The parallel completion of activities of a system user.

We present in Section 3 the automatic design of an understanding model for the monitored activities of a system user, based on event logs. The intervention model is presented in Section 4 and allows the action of the supervisor on the critical activities of the system user. Throughout Sections 3 and 4, an experiment illustrates the understanding and intervention models. The event logs [21] come from a smart home system where the behaviour of the user is monitored by a medical or paramedical staff member. Section 5 contains the conclusions and perspectives.

2 Related Work

The use of process mining is a classical approach to get an understanding of event logs, and to intervene on what is discovered [19]. The process is a set of activities, traditionally represented by oriented graphs: each vertex is an activity, and a directed edge from an activity A1 to an activity A2 represents a temporal sequence (A1 occurs before A2). Two disjoint vertices mean that there is no constraint on the relative order of the activity instances. In particular, the search of fully ordered sequences [11] and unordered (parallel) sequences [5] received most of the attention, allowing the detection of frequent patterns in the activities of a system user. The sequences can be constrained on their length (number of activities) or on the time span of each of its instances. A simple intervention model could be based on a pruning targeting the activities and the sequences with an insufficient number of instances in the set of event logs covering the activity set. Such an intervention model would allow the detection of anomalies when the system is used and records event logs that do not follow the usual course of activities. The main issue with this unsupervised approach is that the activities of the system user lack specification. The critical activities and sequences are defined solely based on their statistical distributions, and the supervisor's knowledge is not exploited. This makes activity analysis harder, and less adapted to the actual supervision needs.

Process mining, however, takes into account the process analysis, and enables an audit of the business viewpoint [20]. It helps deal with the lack of structure in the event log that prevents a supervisor from understanding the event logs. This is why we propose and describe in the next sections a process mining approach, assisting the supervisor for the monitoring.

This approach is complementary to the classical data mining approach. Data mining also produces some patterns based on event logs. However, most of the time, these patterns come only from machine learning algorithms without taking into account the knowledge and goals of the supervisor. We propose here to enhance the role of the system supervisor with the help of MDE: the supervisor can define some rules specifying the expected behaviours of the system user.

3 System User Activities Understanding Model for Monitoring

The understanding model needs to be useful to the supervisor of the activities of the system user. Unfortunately, the event logs do not represent unambiguously the real behaviours of the system user. For example, several activities can be interleaved, which is not easily detected by the human eye among a big set of event logs. This entails the transformation of an event logs model, constrained by the system user activities, into a process. The transformation results in a process model, highlighting the activity sequences composing it.

3.1 Event Logs and Activities

The model in the input of a transformation model must conform to a meta-model representing the requirements for the model. The Ecore model (see http://www.eclipse.org/modeling/emf/) specifies the concepts used to design this understanding model, as well as their relationships:

(i) The “Activity” Eclass, specified by the supervisor, with
  – an identifier (“id” EAttribute);
  – a name (“name” EAttribute);
  – some event logs, aligned with an activity (“alignsEvent” EReference).

(ii) The log “Event” Eclass, defined by
  – a start time (“beginning” EAttribute);
  – an end time (“end” EAttribute);
  – an activity, aligned with some event logs (“isAligned” EReference).

A dataset containing 245 event logs of one person living in a smart house [21] is used throughout this contribution for illustration purposes. It is investigated in greater detail in Sections 3 and 4. The event entries are produced by sensors installed in the house. The system is the house, and the system user is the person living there. The supervisor can be a doctor or anyone in charge of taking care of the system user. The technical view of the system is based on a wireless sensor network. A node sends an event every time a change is detected by the corresponding sensor. The sensors used in the experiment are mostly magnetic switches on the cupboards and motion detectors. The 245 event logs correspond to annotated events grouped in seven activities: *1: leave house*, *4: use toilet*, *5: take shower*, *10: go to bed*, *13: prepare breakfast*, *15: prepare dinner*, and *17: get drink*. Event logs are represented by a “Start time”, an “End time”, and the ID of the corresponding activity, as illustrated in Table 1.

From a supervisor point of view, the size of the event log file is overwhelming and prevents a good understanding of the system user activities. A first assistance is the representation of the activities and the activity sequences, aligned with the event logs. A sequence is an activity relationship specifying that an activity starts just after another activity.

3.2 Understanding Model Automated Design

The understanding model results from a model transformation composed of rules. Applied on the set of event logs, these rules allow the automated design of the understanding model activities, and the sequences between these activities. The terms “mined activity” and “mined sequence” are used below to denote, respectively, an activity and a sequence in the understanding model.

3.2.1 Mined Activity and Sequence Design

MADR-AR (**A**lignment **R**ule). A mined activity is an activity that is aligned with at least one event log.

The **M**ined **A**ctivity **D**esign **R**ule (**MADR**) must be satisfied during the mined activity design. The **MADR-AR** basic alignment rule means that an activity of a system user is taken into account in the understanding model

Table 1: Event Logs Dataset Structure.

Start Time	End Time	ID
25-Feb-2008 00:22:46	25-Feb-2008 09:34:12	10
25-Feb-2008 09:37:17	25-Feb-2008 09:38:02	4
25-Feb-2008 09:49:23	25-Feb-2008 09:53:28	13

if at least one event entry logs it. This rule allows the supervisor to restrict the activities of the system user to those that are actually performed. The activities that are not transformed into mined activities need to be investigated by the supervisor: they can be linked to a technical problem affecting the event log production. They can also result from the user, who does not respect the procedure.

The following operational QVT code (see Algorithm 1) is an illustration of an **MADR-AR** rule implementation where *IM* is the input model (*IM* conforms to the input meta-model in Section 3.1 and, especially, the “alignsEvent” EReference of the “Activity” EClass), and *OM* is the output model (that conforms to the understanding meta-model further in this section).

A mined sequence represents a temporal sequence from a mined activity (called source activity) to a mined activity (another one or itself, called target activity). For example, in the temporal sequence from the *prepare breakfast* activity to the *take shower* activity, the source activity is *prepare breakfast* and the target activity is *take shower*. A sequence suggests that there is a relationship between two logged activities, thus helping the supervisor understand the recorded data. The mined sequences are discovered from the event log set through the following four **M**ined **S**equence **D**esign **R**ules (**MSDR**).

3.2.2 Mined Sequence Design based on Coherence

MSDR-CH (**C**o**H**erence). A mined sequence designed from the mined activity A1 (aligned with event log E1) to the mined activity A2 (aligned with event log E2) has a coherence property if A2 is coherent with A1:
(E1.beginning<E2.beginning) AND (E2.end<E1.end)

Coherence helps the supervisor during the monitoring of parallel activities of the system user (see Section 1). The first rule (**MSDR-CH** rule) defines the coherence of a mined sequence from a mined activity of a system user to another activity that is embedded in the first one. This rule highlights a scenario where two activities are interleaved (parallel). Knowing that two activities can be simultaneous is significant for the supervisor. The **MSDR-CH** rule is represented by a graph in Figure 2. Each oriented edge of the graph is a component of the rule. A guard is associated to each sequence of activities. The (E1.beginning<E2.beginning) guard is, for example, associated to the sequence from the A1 activity to the A2 activity.

Algorithm 1: MADR-AR rule operational-QVT implementation (“--”: comment)

```
mapping IM::createActivityModel(): OM {
  log (“Activity creation”);
  -- call of activities creation with the new OM model as parameter
  self.map createActivities(result);
}
mapping IM::createActivities(inout oms : OM) : OrderedSet(MiningActivity) {
  -- a mining activity is created if
  --   there is at least one event log aligned with one activity of the IM model
  --      ⇔ alignsEvent→size()>0
  oms.miningActivity:= self.activity→
        select(e : Activity|e.alignsEvent→size()>0)→asSequence()→
          map createActivity(oms)→asOrderedSet();
}
mapping Activity::createActivity(oms : OM) : MiningActivity {
  -- creation of a mining activity having the same name than
  --   an activity of the IM model satisfying the MADR-AR rule
  name:= self.name;
}
```

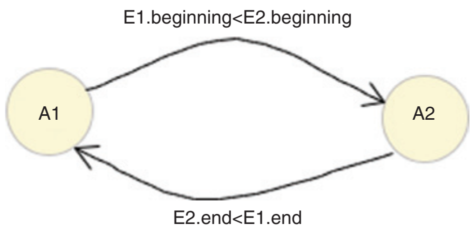

Figure 2: **MSDR-CH** Rule Represented by a Graph.

MSDR-SP (**SP**litting). If a mined sequence from the mined activity A1 (aligned with event log E1) to the mined activity A2 (aligned with event log E2) has a coherence property, then the A1 activity is split into two activities A1_begin and A1_end, respectively, aligned with E1.beginning and E1.end.

When a mined sequence has a coherence property, the following **MSDR-SP** rule completes the coherence property definition with an appropriate sequence design. If a sequence has a coherence property, **MSDR-SP** implies the creation of new sequences. The graph illustrates this rule in Figure 3 with the three new sequences:
- A sequence from the A1_begin activity to the A1_end activity;
- A sequence from A1_begin to A2 (wider edge from A1_begin to A2);
- A sequence from A2 to A1_end (wider edge from A2 to A1_begin).

3.2.3 Mined Sequence Design based on Coupling

The monitoring of the consecutive activities of the system user (see Section 1) is based on coupling. The coupling property of a mined sequence is specified by the **MSDR-CP** rule. This property specifies that there is a coupling between two activities of a system user when no activity beginning or end is logged between the end of the first activity and the beginning of the second one.

MSDR-CP (**C**ou**P**ling). A mined sequence, designed from the mined activity A1 (aligned with event log E1) to the mined activity A2 (aligned with event log E2), has a coupling property if:

$$(\text{E1.end} < \text{E2.beginning}) \text{ AND } \neg\exists\ \text{E3} \in \{\text{event logs}\} \text{ such that}$$
$$(\text{E1.end} < \text{E3.end} < \text{E2.beginning}) \text{ OR } (\text{E1.end} < \text{E3.beginning} < \text{E2.beginning})$$

The objective is to make the event logs as explicit as possible, with regard to the activity succession for the system user. The coupling rule highlights the absence of parallelism between activities of the system user. The graph illustrating this rule is in Figure 4. The sequences from an A3 activity or to an A3 activity do not satisfy the rule. This is represented by a cross on every such A3 activity.

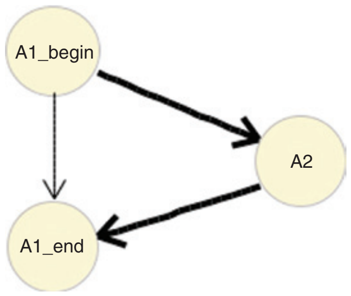

Figure 3: **MSDR-SP** Rule Represented by a Graph.

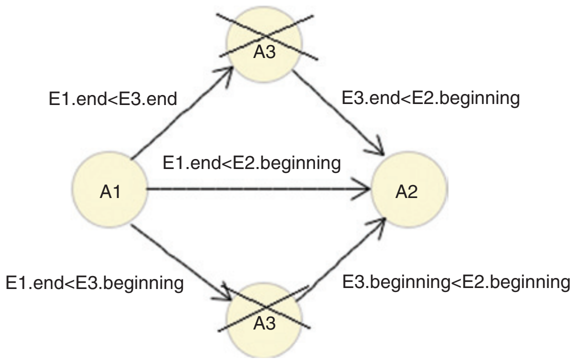

Figure 4: **MSDR-CP** Rule Represented by a Graph.

MSDR-LP (**L**oo**P**). A mined sequence, designed from the mined activity A1 (aligned with the event logs E11 and E12) to itself, has a loop property if:

(E11.end<E12.beginning) AND
¬∃ E3∈ {event logs} such that (E11.end<E3.end<E12.beginning) OR
(E11.end<E3.beginning<E12.beginning).

The last rule, **MSDR-LP**, defines the loop property of a mined sequence. The sequence from an activity of a system user to the same activity is a loop if it verifies the coupling property. A loop allows the supervisor to group the instances of one activity. The **MSDR-LP** rule prevents the presence of other activities in parallel of the considered mined sequence. As previously, the A3 activities of the graph in Figure 5 show the cases where the rule is not satisfied. Like the coherence and coupling properties, the loop property is significant in order to help the supervisor understand the behaviour of the system user.

The meta-model of the understanding model defines a mined process, based on the following concepts conforming to the **MADR** and **MSDR** rules:

(i) The “MinedActivity” Eclass, specified by
 - a name: this of the initial activity in the input model (“name” EAttribute);
 - the number of sequence instances where the mined activity is the target activity (“inputWeight” EAttribute);
 - the number of sequence instances where the mined activity is the source activity (“outputWeight” EAttribute).

(ii) The “MinedSequence” Eclass, defined by
 - an identifier, automatically created by the transformation (“identifiant” EReference);
 - a source activity (“source” EReference);
 - a target activity (“target” EReference);
 - the number of sequence instantiations (“weight” EAttribute);

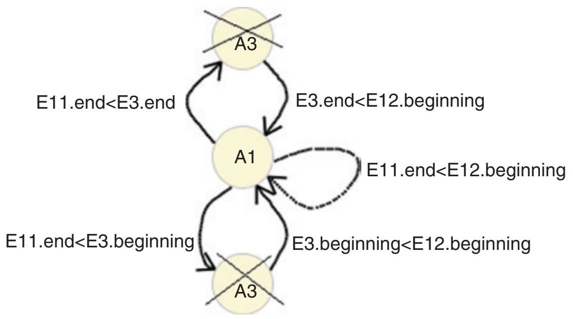

Figure 5: **MSDR-LP** Rule Represented by a Graph.

- the coherence property of the sequence, in relation to the **MSDR-CH** rule respect (“coherence” EAttribute);
- the coupling property of the sequence, in relation to the **MSDR-CP** rule respect (“coupling” EAttribute);
- the loop property of the sequence, in relation to the **MSDR-LP** rule respect (“loop” EAttribute);
- the sum of sequence instances durations (“timeSigma” EAttribute);
- the sum of the sequence instances squared durations (“timeSquare” EAttribute);
- the minimum value of the sequence instances durations (“timeMinimum” EAttribute);
- the median value of the sequence instances durations (“timeMedian” EAttribute);
- the maximum value of the sequence instances durations (“timeMaximum” EAttribute);
- the expected value of the sequence instances durations (“expectedValue” EAttribute);
- the standard deviation of the sequence instances durations (“standardDeviation” EAttribute).

The understanding model can be represented by a graph where every vertex is a mined activity. Every directed edge is a mined sequence, automatically generated by the model transformation implementing the previous **MSDR** rules. An oriented edge from the mined activity A1 toward the mined activity A2 means that A1 comes before A2. As defined in the meta-model, an edge can represent a mined sequence having a coherence property, a coupling property, or a loop property. The number of instances of each sequence is given in the label of each edge.

The first log entry of the excerpt (see Table 1) specifies a 25-Feb-2008 00:22:46 start time and a 25-Feb-2008 09:34:12 end time for the *go to bed* activity. We implement a specific transformation having as input the event logs stored in the format shown in Table 1, and resulting in a model conforming to the previous understanding meta-model.

The analysis of the activities of the system user is under the responsibility of the supervisor. The activities monitored during the experiment are those that are classically supervised for elderly people. The understanding model is deduced from the application of the **MSDR** rules, implemented by a model transformation.

In the dataset [21], the **MSDR-SP** rule is applied twice:

- The *prepare Dinner* activity is split because of its coherence with the *use toilet* activity.
- The *go to bed activity* is split because of its coherence with the *use toilet* activity.

Figure 6 presents the understanding model that is deduced from the 245 activity logs. This model is a graph as defined previously. The keys specifying the properties of the sequences are given at the top right corner. The oriented edge (coupling property) from the *prepare breakfast* vertex to its successor vertex (*take shower*) means that a mined sequence from the *prepare breakfast* mined activity to the *take shower* mined activity is discovered in the set of event logs, this mined sequence is supported by 11 instances (edge weight of 11), and verifies the coupling property.

4 System User Activities Intervention Model for Monitoring

The supervisor of the activities of the system user chooses some sequences that he or she considers as critical. The criticality of an activity (resp. a sequence of activities) means that the monitoring of the activity (resp. the sequence) is required by the supervisor of the activities. The intervention model contains the critical sequences as well as all the mined sequences that have at least the same importance in the understanding model. The importance is defined by various statistical characteristics, detailed in the next subsections.

4.1 Mined Sequence Pruning from Critical Sequence Specification

In the intervention meta-model, a “critical” EAttribute and a “pruned” EAttribute are added to the understanding meta-model in the “MinedSequence” EClass (see Section 3.2). The supervisor can choose statistical

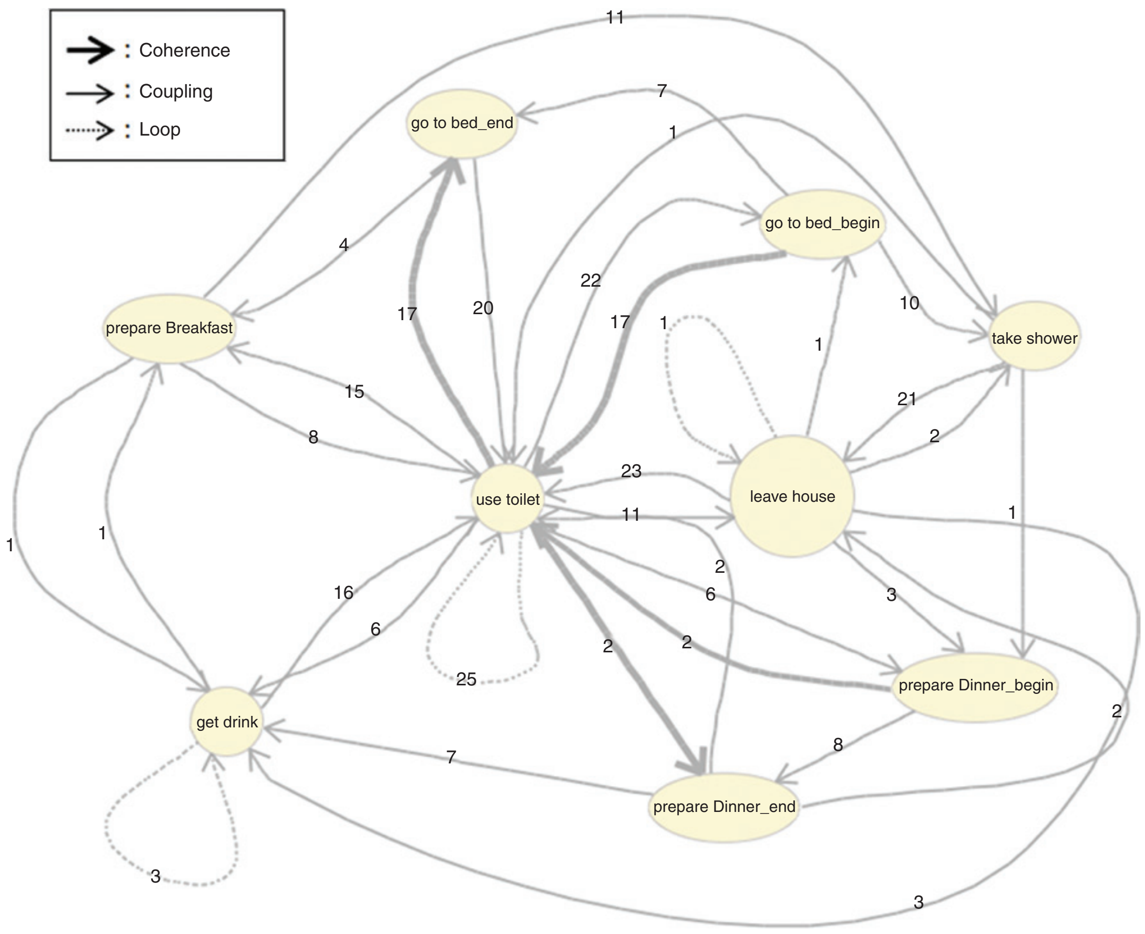

Figure 6: Understanding Model of the Experiment Represented by a Graph.

rules to specify useful thresholds for the pruning of some activity sequences. We select two statistical thresholds each defined by a **M**ined **S**equence **P**runing **R**ule (**MSPR**). The thresholds must be estimated in relation to the critical sequences: a critical sequence cannot be pruned. The intervention model contains at least the critical sequences and the not-pruned mined sequences.

The **MSPR-IW** rule is based on the number of instances of a mined sequence. An instance means that two event logs satisfy one of the **MSDR** rules. The number of instances is specific to a property (coherence, coupling, or loop).

MSPR-IW (**I**nstance **W**eight). A mined sequence is maintained in the intervention model if its number of instances ("weight" EAttribute of the "MinedSequence" EClass) divided by the number of instances of all the mined sequences having the same property is greater than a T_IW threshold.

The supervisor can ignore the mined sequences of the understanding model that are not frequent enough with the **MSPR-IW** rule. When a mined sequence is pruned in the intervention model, the supervisor can also analyse the reasons for this pruning: it can be an anomaly in relation to the activity order.

The next rule, **MSPR-MD**, is based on the duration of a sequence instance, which is equal to the time interval between the beginning of the target activity and the end of the source activity of the sequence. The maximum duration of each sequence is estimated from the sequence instances.

MSPR-MD (**M**aximum **D**uration). A mined sequence is maintained in the intervention model if the maximum duration of the sequence ("timeMax" EAttribute of the "MinedSequence" EClass) is lower than a duration threshold.

On the one hand, the supervisor can ignore the mined sequences classified as too long by the **MSPR-MD** rule. This is relevant when the intervention model is designed to represent the major behaviours, statistically speaking, of the system user. On the other hand, the supervisor can also consider these sequences because of their excessive duration. Indeed, excessive duration may be considered as an anomaly by the supervisor.

4.2 System Using Anomaly Detection

Anomalies are searched for critical mined sequences of the intervention model and are conditioned by the understanding model. We suggest defining the detection of an anomaly through **M**ined **S**equence **A**nomaly **R**ules (**MSAR**).

The first **MSAR-SW** rule focuses on the weight of a critical mined sequence, conditioned by all the mined sequences sharing the same mined activity as source. The supervisor can thus obtain information about the relevance of the critical mined sequence when the system user participates in the source activity of this sequence.

MSAR-SW (**S**ource **W**eight). There is an anomaly for a critical mined sequence S in the intervention model when the instance count of S (“weight” EAttribute of the “MinedSequence” EClass) divided by the sum of the weights of the mined sequences having the same source activity is lower than a threshold T_SW.

For a more complete overview of the system use, a similar rule is suggested in relation to the target activity of the critical mined sequence.

MSAR-TW (**T**arget **W**eight). There is an anomaly for a critical mined sequence S in the intervention model when the instance count of S (“weight” EAttribute of the “MinedSequence” EClass) divided by the sum of the weights of the mined sequences having the same target activity is lower than a threshold T_TW.

The next two rules allow a comparison between the different paths, composed of mined sequences (paths possibly longer than one sequence) that connect the source and the target of a critical mined sequence. The **MSAR-TF** rule specifies a comparison in terms of time and the **MSAR-WF** in terms of weight. These rules provide information to the supervisor about the indirect ways to go from the source to the target activity of the critical mined sequence. Thus, we obtain a more flexible constraint than the critical mined sequence, allowing the supervisor to know when the path between the source and the target of a critical sequence takes too much time, or has too many steps. With the **MSAR-TF** rule, the supervisor has information when too much time is spent between the source and the target activity of a critical mined sequence, compared to the duration of the indirect paths.

MSAR-TF (**T**ime **F**actor). There is an anomaly for a critical mined sequence S in the intervention model if the average time span of the mined sequence paths (“expectedValue” EAttributes of the mined sequences used in the paths) from the source of S to the target of S divided by the average time span of S is greater than a threshold T_TF.

With the **MSAR-WF** rule, the supervisor has information about the excessive weight of indirect paths between the source and the target activity of a critical mined sequence. This information enables the detection of an anomaly when the critical mined sequence is less used by a system user than some indirect paths.

MSAR-WF (**W**eight **F**actor). There is an anomaly for a critical mined sequence S in the intervention model if the average weight (“weight” EAttribute) of the indirect paths from the source of S to the target of S divided by the weight of S is greater than a threshold T_WF.

For the experiment, two mined sequences were chosen as critical: from *go to bed_end* to *use toilet* and from *prepare breakfast* to *take shower*. The **MSAR-TF** and **MSAR-WF** rules are implemented with minimum thresholds of 5.0 for the time factor and 0.15 for the weight factor, fixed based on the needs of the supervisor. Here are some remarks on the automatic detection of anomalies resulting from the implemented transformation:

(i) The SW ratio (see **MSAR-SW** rule) shows that the *go to bed_end* to *use toilet* mined sequence is the main sequence having the *go to bed_end* mined activity as source: 20/(20 + 4) = 83% (the other sequence is from *go to bed_end* to *prepare breakfast* and has a weight of 4).

(ii) The SW ratio (see **MSAR-SW** rule) shows that the *prepare breakfast* to *take shower* is the most important mined sequence having the *prepare breakfast* mined activity as source: 55%.

(iii) The TW ratio (see **MSAR-TW** rule) shows that the *go to bed_end* to *use toilet* mined sequence is not a very significant activity sequence in relation to the eight sequences having *use toilet* mined activity as target: 18%.

(iv) The TW ratio (see **MSAR-TW** rule) shows that the *prepare breakfast* to *take shower* mined sequence is significant among the two sequences having the *take shower* mined activity as target: 48%.

(v) The average time of the *go to bed_end→prepare breakfast→take shower→leave house→use toilet* sequence path has a 8.17 factor (see **MSAR-TF** rule) in relation to the average time of the *go to bed_end* to *use toilet* critical mined sequence. The weight of the *go to bed_end→prepare breakfast→take shower→leave house→use toilet* sequence path has a 0.2 factor (see the **MSAR-WF** rule) in relation to the average time of the *go to bed_end* to *use toilet* mined sequence. The direct path spans much less time and is more frequent than the indirect path.

(vi) The average time of the *go to bed_end→prepare breakfast→use toilet* sequence path has a 5.37 factor (see the **MSAR-TF** rule) in relation to the average time of the *go to bed_end* to *use toilet* mined sequence. The weight of the *go to bed_end→prepare breakfast→use toilet* sequence path has a 0.2 factor (see the **MSAR-WF** rule) in relation to the average time of the *go to bed_end* to *use toilet* mined sequence.

5 Conclusion and Perspectives

In this contribution, we propose some rules to automatically design the understanding and the intervention models of the activities of a system user. This support tool targets the sequential and parallel activities of the user. The rules are implemented using an MDE approach with the operational QVT language. This implementation provides first an understanding model representing the behaviour of the system user. When the supervisor chooses critical sequences, the rules for intervention model allow the pruning of irrelevant mined sequences from the understanding model, thus producing a custom intervention model, adapted to the supervisor's needs and expertise. The anomalies in the critical mined sequences are detected with appropriate statistical measures (the set of measures could also be further enriched).

The model-driven engineering approach enables us to increase the complexity of our process mining approach with new concepts in the input meta-model, thus opening perspectives for future work. The first concept is the modelling of the dependencies between event logs. The perspective is the design of a relevant event log model with a statistical pruning of the dependencies between event logs. This pruning should be consistent with the activity sequence pruning suggested here. The second concept is related to the technical nodes used to generate the events (e.g., sensors in our experiment). A second perspective would be to align the existence of a communication link between technical nodes with a dependency between event logs. The objective of a tool enabling the supervisor to easily monitor the activities of a system user remains the same. Currently, the system is used offline and a perspective is to extend it for data streams.

Bibliography

[1] J. Becker, P. Bergener, D. Breuker and M. Räckers, On measures of behavioural distance between business processes, in: *10th International Conference on Wirtschaftsinformatik*, pp. 665–674, Zurich, Switzerland, 2011.

[2] D. J. Cook, A. S. Crandall, B. L. Thomas and N. Chatapuram Krishnan, CASAS: a smart home in a box, *IEEE Computer* **46** (2013), 62–69.
[3] G. Demiris and B. K. Hensel, Technologies for an aging society: a systematic review of “smart home” applications, *Yearbook of Medical Informatics* **3** (2008), 33–40.
[4] R. A. Groeneveld and G. Meeden, Measuring skewness and kurtosis, *J. R. Stat. Soc., Series D (The Statistician)* **33** (1984), 391–399.
[5] E. Heierman, M. Youngblood and D. J. Cook, Mining temporal sequences to discover interesting patterns, in: *KDD Workshop on Mining Temporal and Sequential Data*, pp. 104–111, 2004.
[6] P. Kueng, Process performance measurement system: a tool to support process-based organizations, *Total Qual. Manage.* **11** (2000), 67–85.
[7] A. Makanju, S. Brooks, A. N. Zincir-Heywood and E. E. Milios, Logview: Visualizing event log clusters, in: *6th IEEE Conference on Privacy Security and Trust*, IEEE Computer Society, pp. 99–108, 2008.
[8] E. Morin, *Communication and complexity – Introduction to the complex thought*, ESF éditeur, 1990.
[9] OMG, *MOF 2.0 Query/View/Transformation*, OMG Technical Report, 2009, http://www.omg.org/spec/QVT, accessed on July, 2012.
[10] M. Pechenizkiy, N. Trcka, E. Vasilyeva, W. M. P. van der Aalst and P. De Bra, Process mining online assessment data, in: *International Working Group on Educational Data Mining*, pp. 279–288, 2009.
[11] J. Pei, J. Han, B. Mortazavi-Asl and H. Zhu, Mining access patterns efficiently from web logs, in: *Knowledge Discovery and Data Mining – Current Issues and New Applications*, pp. 396–407, Springer, Berlin, 2000.
[12] H. Pigot, When cognitive assistance brings autonomy in daily living: the DOMUS experience, *Gerontechnol.* **9** (2010), 71.
[13] P. Rashidi and A. Mihailidis, A survey on ambient assisted living tools for older adults, *IEEE J. Biomed. Health Inform.* **17** (2013), 579–590.
[14] D. C. Schmidt, Model-driven engineering, *IEEE Computer* **39** (2006), 25–31.
[15] J. Simonin, E. Bertin, Y. Le Traon, J. -M. Jézéquel and N. Crespi, Analysis and improvement of the alignment between business and information system for telecom services, *Int. J. Adv. Software* **4** (2011), 117–128.
[16] J. Soulas, P. Lenca and A. Thépaut, Monitoring the habits of elderly people through data mining from home automation devices data, in: *16th Portuguese Conference on Artificial Intelligence*, vol. 8154 of Lecture Notes in Artificial Intelligence, pp. 343–354, Springer, Berlin Heidelberg, 2013.
[17] M. Stikic, T. Huynh, K. Van Laerhoven and B. Schiele, ADL recognition based on the combination of RFID and accelerometer sensing, in: *Pervasive Computing Technologies for Healthcare*, 2008, IEEE Second International Conference on Pervasive-Health 2008, pp. 258–263, IEEE Computer Society, 2008.
[18] W. M. P. van der Aalst, A. J. M. M. Weijters and L. Maruster, Workflow mining: discovering process models from event logs, *IEEE Transactions on Knowledge and Data Engineering* **16** (2004), 1128–1142.
[19] W. M. P. van der Aalst, Business process management: A comprehensive survey, *ISRN Software Eng.* **2013** (2013), 1–37.
[20] W. M. P. van der Aalst, K. M. van Hee, J. M. van der Werf and M. Verdonk, Auditing 2.0: using process mining to support tomorrow’s auditor, *Computer* **43** (2010), 90–93.
[21] T. L. M. van Kasteren, G. Englebienne and B. J. A. Kröse, An activity monitoring system for elderly care using generative and discriminative models, *Personal Ubiquitous Computing* **14** (2010), 489–498.
[22] J. A. Zachman, A framework for information systems architecture, *IBM Syst. J.* **26** (1987), 276–292.