Modeling and Mining of Collaborative Learnflows

Robin Bergenthum, Andreas Harrer, Sebastian Mauser

Katholische Universität Eichstätt-Ingolstadt, Fachgebiet Informatik
forename.name@ku-eichstaett.de

Abstract. This short paper first presents a modeling language for collaborative learnflows. This language is based on ideas from the area of business process modeling. Second, it is shown how to automatically generate respective learnflow models from log files of learning systems.

1 Introduction

Business processes have been established in the research and application field of *business process resp. workflow engineering* with matured methods, representations and computer support with tools. In contrast, the closely related learning and teaching processes only recently gained attention in research and practice and have not yet created shared terms, methods, and representations. This is especially true for the field of Computer-supported Collaborative Learning (CSCL), a discipline that investigates in the affordances and effects of computer applications supporting groups of students in knowledge construction and skill development. Thus, while we will call the formal representation of learning / teaching processes *learnflow engineering* in this paper, this term is neither firmly established nor fully deserves the engineering character, but is approaching this currently with the work of our colleagues and our own contributions. To reach this goal we compared in earlier work [1] commonalities, specifics, and potential methodological transfer between workflow and learnflow engineering. We also introduced first approaches for formal modeling of collaborative learning processes by means of Petri nets and the synthesis of nets from protocols and log files via process mining algorithms.

Of particular interest are the explicit representation of collaborative learning and of roles and learning groups. In comparison to business processes [2] the following specifics of learning processes have to be taken into account:

– For business processes the ultimate goal of performing / enacting a process with its associated activities is the achievement of a product with guaranteed quality criteria, while the participation of individual actors is only a minor concern. However, for learning processes the priority is that the *learners* involved in the activities get the opportunity to gain knowledge and experience: an objectively measurable result of the process is - besides the use of formative evaluations / exams - much less important than the completion of the learning experience for each participant and its implicit result of constructed knowledge in the student's mind. This requires that in the modeling of learning processes the individual actors have to be modeled thoroughly and explicitly.

Recent Advances in Petri Nets and Concurrency, S. Donatelli, J. Kleijn, R.J. Machado, J.M. Fernandes (eds.), CEUR Workshop Proceedings, ISSN 1613-0073, Jan/2012, pp. 151–159.

- The concept of *role* in workflow engineering is mainly based on responsibility and ability for a set of activities, which is static after an initial assignment of roles to actors. Dynamic constraints on the activities (e.g. the same actor that created a proposal should also fix the contract) and special rules for allocation of actors to activities (e.g. the actor of a given role that has a minimum number of other roles should be chosen to distribute the workload) have been formulated in respective work by means of additional model constructs and notations. In contrary to this rigid role concept, the usage of roles in learning processes is frequently guided by exercising specific skills where roles are changed and acquired during a learning process. Therefore, an extension of static role concepts to dynamic ones that are capable of taking into account the learning history are needed for learnflow processes.
- Each activity, potentially even the whole learning process, may require *group work* and discussion and especially these group activities can have a high importance for the learning experience. Thus, the flexible and explicit representation of groups with required roles and - if needed - the dynamic re-arrangement of groups is one more requirement for learnflow engineering methods.

In this paper we will present a modeling language for learning processes that takes specifically into account the requirements identified above. To make use of expertise and experience from workflow engineering we will build the proposal upon sound existing approaches from this field [2]. Besides the intended applicability for learning and teaching processes we also consider our approach useful for business processes with dynamic roles, cooperative and collaborative activities such as in adaptive workflows where a static process structure is not satisfactory.

A first modeling approach in this direction has been proposed in [1]. There, we represented learning processes as Petri nets as is usual in the field of workflow management [2]. The allocation of actors was made by assignment of the required roles to activities and the usage of a global pool of actors that have the possible roles they can take associated to them. The established workflow concepts were extended by a mechanism to allow the change of actor roles while performing an activity.

Because of the significance of role changes in the modeling of learning processes in this paper we propose a refinement of that approach. We explicitly model dynamic role assignments and changes in a state diagram: actors can change their assigned roles when performing activities, which means that the role changes in the state diagram are synchronized with the activities in the process model (Petri net). Learning groups are taking into account by collaborative activities performed by several roles jointly.

We will discuss the potential and methods for the automated synthesis of these models from real protocol instances as an approach for *collaboration flow mining*. For this we extend our first proposal of a learnflow mining approach presented in [1]. This previous work focused on the discovery of the structures for the control flow using methods from the area of workflow mining [3, 4], but in this paper we will address additionally the challenge how to gain information about dynamic roles and collaboration situations from the protocol instances. Related work to this is organizational mining [5] that is limited to static roles and organizational units.

In Section 2 we will present our new modeling language by an example learning process. By means of this example we also illustrate the core ideas of collaboration flow mining in Section 3.

2 Modeling Language

As an example we consider the following computer supported learning scenario. Groups of three students use the tool Freestyler (www.collide.info) to learn the effect of different factors such as lightning conditions and CO2-concentration on the growth of plants. For this purpose, Freestyler provides several tabs with different functionalities. There are for instance tabs to formulate questions, to create simple models or to import data from a simulation tool. In this example, the set of tabs corresponds to the set of supported learning activities. Namely we consider the following activities: In = **In**troduction, Qu = Elaboration of the Research **Qu**estion, Pl = **Pl**anning, Mo = **Mo**deling of the Relations of the Different Factors, Hy = **Hy**pothesize, E1 & E2 = **E**xperiment One resp. Two for Hypothesis Testing, Da = Study of Existing Experimental **Da**ta for Hypothesis Testing, An = **An**alysis of Experimental Results Including Comprehensive Hypothesis Testing, Pr = **Pr**esentation of the Research Results. Some of these learning activities require collaboration (In, Hy, An require all three learners and Pl, Mo, Pr each require two learners).

In this context a learnflow model first describes the order in which the tabs have to be processed by the learners. Second, it determines who is allowed to work on a tab. This depends on the dynamic roles of the learners within the learning group. The learners may have the role Student, Modeler, ExModeler, Recorder and ExRecorder. Student is the default role. The other roles encode the learning history of the learners, e.g. Recorder and ExRecorder store the information that the learner has performed the activity Question.

Figure 1 together with Figure 2 show a possible learnflow model for the example scenario. Figure 1 illustrates the process aspect in the form of a Petri net with transition annotations. The annotations refer to the numbers of roles required for an activity. The state chart of Figure 2 complements the Petri net model. It represents a consistent role diagram, which models the dynamic roles of the three learners in the pool of actors. Each learner corresponds to one instance of the state chart. Instead of state charts, it would also be possible to consider state machine Petri nets to model the roles of learners, but state charts are in our opinion the more natural modelling approach.

The dynamic of the learnflow model is as follows. An activity of the net can only be accomplished, if the pool of actors contains learners having the roles annotated at the transition. For instance, the collaborative activity Pl requires two actors with the role Student. The occurrence of a transition can change the roles of the involved actors. Such change is modeled in the role diagram by a state transition with the activity name as the input symbol. Therefore, in our example the activity Pl causes the two students to switch over to the role Modeler. This role is later on required to perform the activity Mo. For simplicity of the role diagrams, we use the following convention. If an actor performs an activity, which does not explicitly cause a change of his role in the state

chart, he preserves his role. For instance, the activity Mo does not change the role of an actor with the role Modeler and therefore can be neglected in the state chart.

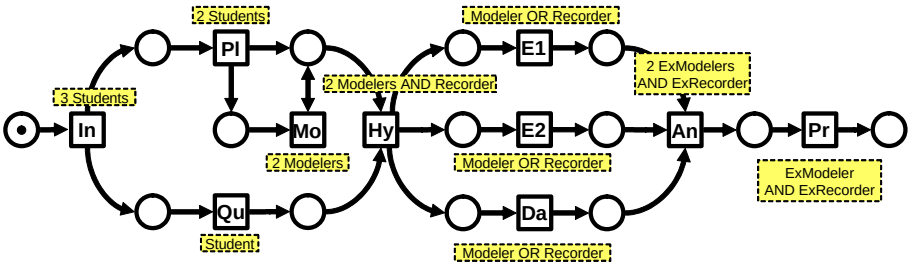

Fig. 1. Petri net model with role annotations

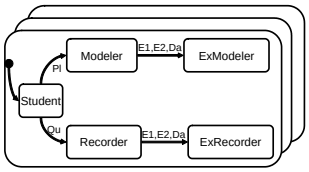

Fig. 2. State chart representing a role diagram

The example shows, that the presented modeling language allows to comprehensibly represent collaborative activities as well as dynamic roles and the learning progress of learners within a learning process. Still the language is simple and intuitive. It naturally extends well established modeling approaches from the domain of business process management. The approach supports a clear separation of the process perspective and the role perspective of a learn flow.

For a formal definition of the occurrence rule of the new modeling language, we consider a translation of respective learn flow models into a special class of colored Petri nets [6]. All the standard Petri net components are kept in the high-level Petri net model. The annotations of the transitions and the state diagrams (we require that a role occurs only once in a state diagram) are translated into one high-level place modelling the pool of ressources resp. actors. The color set of this place is given by the possible roles of learners. The initial marking contains for each state diagram one token of the kind given by the initial state of the state diagram. The place has an outgoing and an ingoing arc connected with each transition of the net. A transition consumes tokens from the place as given by its annotation. For each consumed token, there are two cases. Either the state corresponding to the token type in the state diagram allows a state transfer labeled with the name of the considered transition or it does not allow such transfer. In the first case, the transition produces a token of the kind given by the follower state of the state diagram in the ressource pool. In the second case, it produces a token of the same kind as the consumed token. Note that for classical business process models with static roles, an analogous translation is possible. But in this situation the first case neven occurs, i.e. each transition produces the same tokens in the ressource pool, that the transition consumes from the pool. This shows, that our new concept is more general than the classical approach.

Figure 3 illustrates the described translation for our example model (for the sake of clearness, the single ressource pool place is splitted in the illustration). The resulting high-level model on the one hand does not any more show explicitely the dynamic roles and the behavior of the learners. On the other hand it is quite difficult to read and understand. Therefore, for modeling purposes, the original representation should be prefered. The high-level view should only be used for formal considerations.

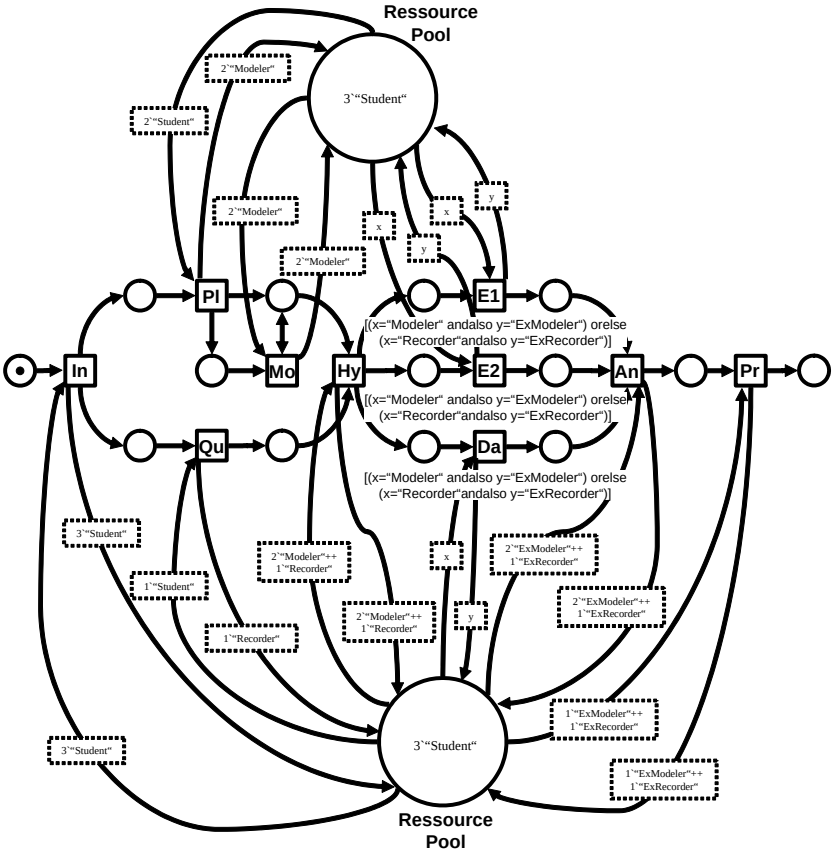

Fig. 3. High-level Petri net

There is the following important observation for our new modeling approach. The learning progress and the learning history of the actors are encoded in the role diagram. For instance, the activity E1 initiates a learning progress of the involved learner. This is represented by a change of the role of the learner. The new role then does not allow an execution of E2 and Da by this learner anymore. Thus, the three activities E1, E2 and Da have to be divided among the three learners. As an outlook, we also work on a

further similar modeling language which avoids such progress-dependend role changes by applying a nets-in-nets concept.

Lastly, our modeling approach regards learning groups implicitly by means of collaborative activities. In view of an explicit modeling of groups, there are two natural and useful possibilities. First of all, different groups can simply be represented by different process instances. However, in this case an extension of the modeling language which allows to model dependencies of process instances would be helpful to regard dynamic group structures. Second, groups can also be modeled analogously as roles, i.e. the role diagrams of the actors can also capture information about group membership of actors. However, in contrast to roles it is important to represent the overall dynamic of the groups. This dynamic can only implicitly be regarded by group-memberships of single actors. Therefore, an extension of the modeling approach which at any time explicitly represents the learning groups would be interesting. This can for instance be achieved by a respective grouping of the state charts in the pool of actors.

3 Collaboration Flow Mining

In this section we introduce an approach to mine a learnflow model given a log file containing recorded learning activities. We construct a learnflow model which reflects a learning process performed by the students (maybe unknown to the teacher) or if the log file is filtered by the teacher even a desired learning process.

If an information system supports an actor while performing a learning activity this event can be recorded and gathered in protocol instances. Each recorded event contains information about the respective process instance, the name of the activity, the time of its occurrence and the involved actors. First, the events are ordered by their respective process and process instance. Second, they are ordered by the time of their occurrence within each process instance. Thus, each process instance yields a sequence of activities together with the respective actors. In the following we use these sequences as an input to a mining algorithm creating a process model. This process model can be used for verification and analysis issues or even as an input for information systems controlling the learnflow.

The tool Freestyler records the activities of the students. Figure 4 shows a part of a Freestyler log file of the considered learning process. It also shows a sequence of activities together with the corresponding actors resulting from that log file. The teacher may filter the log file by adding additional learning sequences or by removing unwanted learning sequences according to certain criteria such as subsequently measured learning achievements. In this case the mining yields a model of the desired learning process, while without filtering a model of the actual behaviour of the students is generated.

In the following we assume the log file to be complete for the given learning process, i.e. each possible learning sequence of the learning process is recorded in the log, where we distinguish learning sequences in the form shown in the last table of Figure 4. After abstracting from the set of actors in the learning sequences well know process mining algorithms [1, 3, 4] can be used to automatically construct a model of the control flow of the learning process. Since we consider a complete log file, we highly recommend to use a precise mining algorithm. Such algorithms exactly reproduce the sequences

Log file

Process	Process instance	Action	Student	Time
Photosynthesis	Group A	Introduction	Andi, Basti, Robin	10:03:12
Photosynthesis	Group A	Question	Robin	10:06:43
Photosynthesis	Group B	Introduction	Bert, Caro, Hans	10:07:33
...	...	...	...	...

Learning sequences

Group A (Introduction; Andi,Basti,Robin), (Question; Robin), (Planning; Andi,Basti), (Modeling; Andi,Basti), (Hypothesis; Andi,Basti,Robin), (Experiment1; Andi), (Experiment2; Robin), (Data; Basti), (Analysis; Andi,Basti,Robin), (Presentation; Andi,Robin)

...

Projection of learning sequences onto single students

Introduction, Planning, Modeling, Hypothesis, Experiment1, Analysis, Presentation
Introduction, Planning, Modeling, Hypothesis, Data, Analysis
Introduction, Question, Hypothesis, Experiment2, Analysis, Presentation
...

Learning sequences for role annotations

Group A (Introduction; -,-,-), (Question; In), (Planning; In,In), (Modeling; InPl,InPl), (Hypothesis; InPlMo,InPlMo,InQu), (Experiment1; InPlMoHy), (Experiment2; InQuHy), (Data; InPlMoHy), (Analysis; InPlMoHyE1,InPlMoHyDa,InQuHyE2), (Presentation; InPlMoHyE1An,InQuHyE2An)

...

Fig. 4. Example log file

of a log file if this is possible. Precise mining algorithms for Petri nets are base on the so called theory of regions. In [4] we propose to apply regions of languages for mining, since the sequences of a log file in a natural way determine a language. We have also implemented a respective algorithm. From the log file in Figure 3 this algorithm generates the Petri net model shown in Figure 1 yet without role annotations. In order to generate these role annotations and the state chart describing the dynamic roles of the students we introduce an additional mining method.

For every learning sequence and each occurring student within the learning sequence we consider the sequence of activities the student performs. Figure 4 shows this projection of the learning sequences onto the students for the considered example. All these sequences are integrated into a deterministic state chart in the form of a tree. Its states are determined by the history of previously performed activities (also regarding the order of the activities) and are named accordingly. In our example the resulting state chart is given in Figure 5 yielding a first model of roles for our learning process. In order to consistently use these roles as annotations in the Petri net model, in every learning sequence each actor must be renamed by the role describing the activities performed by the actor in the history of this learning sequence (see Figure 4 lower part). The role annotation of an activity in the Petri net is determined by all roles or combinations of roles in the case of a collaborative activity that occur together with this activity in any such learning sequence.

Although we have not formally proven this yet, it can be shown that using this approach and given a complete log file, i.e. a complete set of learning sequences, a model is calculated which has the same behaviour as show in the log file (if such a model exists). That means, the mined model and the log define the same learning sequences in the form depicted in the last table of Figure 4. In our example the generated model has equivalent behaviour to the learning process model shown in Figure 1. Yet the model has a much lager set of roles. The reason for this is that the role model encodes the complete history of each actor.

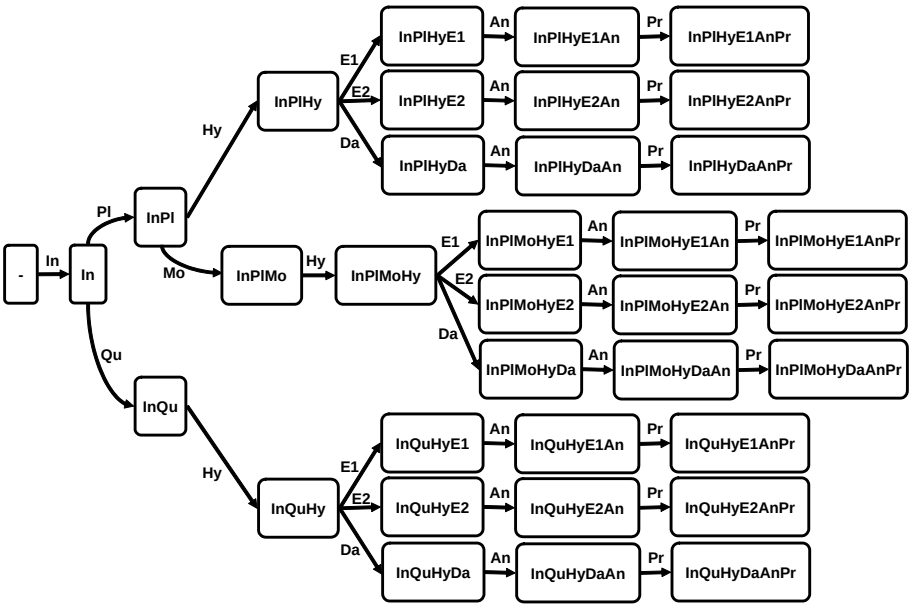

Fig. 5. Role diagram with finest granularity

In the following our goal is to simplify the model of roles by combining roles. Therefore, we developed the following rules. Remark that the annotations in the Petri net model have to be renamed consistently.

- Role transitions triggered by actions performed collaboratively by all actors belonging to one process instance can be neglected.
- Roles having the same postset of roles (respectively an empty postset) and the same outgoing activities can be fused, if for each outgoing activity the roles occur together with the same roles over all learning sequences. Thereby, outgoing activities between the roles to be fused can be neglected.
- Only as a last step roles having an empty postset can be neglected.

It can be proven, that given a complete log file, applying these rules to the previously mined role diagram again yields a behaviour equivalent learnflow model. In our example with these three rules we get a model of roles which is isomorphic to the model shown in Figure 2. Apart from the role names that are not present in the log files anyway we were in this case able to recreate the original learning process model from a complete log file. The role names have to be assigned by the teacher afterwards.

Finally, in practical settings typically we have to assume that not all possible learning sequences have been recorded in a log file. For such log files the presented approach has to be adapted. Heuristics can help to infer the missing sequences and to integrate them into the process model. For the control flow perspective existing methods from the area of process mining can be used in this context [3]. For the role diagrams we plan

to use methods from the theories of structural equivalence and generalized block modelling [7] where missing information is penalized and a solution with minimal penalties can be used for the generalized role model.

References

1. Bergenthum, R., Desel, J., Harrer, A., Mauser, S.: Learnflow mining. In: DeLFI, LNI 132, GI (2008) 269–280
2. Aalst, W., Hee, K.: Workflow Management: Models, Methods, and Systems. MIT Press (2002)
3. Aalst, W.: Finding Structure in Unstructured Processes: The Case for Process Mining. In: ACSD 2007, IEEE (2007) 3–12
4. Bergenthum, R., Desel, J., Lorenz, R., Mauser, S.: Process Mining Based on Regions of Languages. In: BPM 2007, Springer (2007) 375 – 383
5. Song, M., Aalst, W.: Towards comprehensive support for organizational mining. Decision Support Systems **46**(1) (2008) 300–317
6. Jensen, K.: Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use. Volume 1-3 of Monographs in Theoretical Computer Science. Springer (1992, 1994, 1997)
7. Doreian, P., Batagelj, V., Ferligoj, A.: Generalized Blockmodeling. Volume 25 of Structural Analysis in the Social Sciences. Cambridge University Press, Cambridge (2005)