Generation of agent simulation models by using process mining methods on the example of E-learning process

Mikhail Dorrer and Alexandra Dorrer
Reshetnev Siberian State University of Science and Technology, Krasnoyarsk, prospect “Krasnoyarsky rabochy” 31, Russia

Abstract. The main objective of this work is to develop a method for the formation of agent simulation models based on the actual business processes implementation data. To solve this problem, we used the educational process management system Moodle, ProM process mining system, and AnyLogic simulation system. The initial data of the educational process obtained from the Moodle system were processed using Process Mining methods, and a heuristic miner based on the Weijters algorithm was used. Business process models were described in the eEPC notation. The prognostic ability of simulation models constructed as a result of the application of the proposed method turned out to be quite high. The error in predicting the total execution time of the process and the total complexity of the process did not exceed 7%.

1. Introduction
Most information systems used to support the implementation of business processes belong to the class of factographic, having the functionality of registering facts of operations in an organization. Such systems keep logs containing data on cases that occurred in the organization (i.e., process instances, options for their execution), the time at which the operations were performed, the executors (people or systems) who performed these operations, and other types of data. These logs are the starting point for the analysis of processes, and they are usually called ‘event logs’ [1] and are used for process mining.

In the authors’ view, the idea of combining model building and its analysis as part of a common approach seems attractive. Such a solution would reduce the complexity of the forecast of the developed model, as well as reduce the risk of errors when separately building an activity model and a simulation model. However, currently there are published works only on the joint use of technology for the intellectual analysis of Process mining processes and discrete-event simulation using colored Petri nets (implemented in CPN Tools). Agent simulation models are not used in this area, although they are more universal and more flexible than discrete-event ones.

2. Methods
The purpose of this work was to develop and verify a method for developing agent-based simulation models based on formal models of business processes. The construction of models of business processes should be carried out using algorithms for the intellectual analysis of processes (Process Mining). When building models in accordance with the proposed method, the following steps must be performed (figure 1):

- Creation of an event log based on data in the information system.
- Restoring the structure of the formal model of the business process in a notation to describe the operational level of business processes (in our case, eEPC). Defining the relationship between organizational structure and business functions.
- Clarification of the business process formal model by experts and its addition / change if necessary. Necessary for error analysis.
- Creation of an agent simulation model according to the previously proposed translation technique (discussed in detail in [2]).
- Improving the adequacy of the simulation model to the required level.

3. Results
As an object of study, the authors chose the processes of distance learning that occur at the Reshetnev Siberian State University of Science and Technology. Automation of distance learning processes at the university is provided using the Moodle server [3]. As an application, data on students taking discrete mathematics courses were chosen, since this course has been going on for quite a long time and representative statistics have been accumulated on it in the system.

Discipline "Discrete mathematics" is taught in 1 year. The number of hours allocated to the study of the discipline according to the curriculum - 180.

The course group consisted of 50 students.

As part of the study of the discipline, individual settlement tasks and a course project are envisaged. The course ends with an exam.

The distance course contains 3 topics: 1) Sets. Relationships on sets; 2) Counts; 3) Algorithms on graphs. Each topic of the course contains lecture and presentation material, tests.

Information about all student actions when working with the course is stored in the database of the distance learning system.

Student learning process data were obtained by an SQL query to the Moodle database (using MySQL Workbench utility). The data returned by the request is written to a text file in CSV format (Comma-Separated Values) for further processing.

To create an event log in MXML format, the XESame software product is used as follows:

- Since the time data in the Moodle database tables are stored in the Unix timestamp format (which means the number of seconds elapsed since the first of January 1970), it was necessary to convert the time to the format “MM-DD-YYYY hh:mm”. This can be done, for example, in MS Excel using formulas;
To use XESame, ODBC data source to a text file that contains the event log must be created in the local Windows system;
Next, XESame configures the connection to the previously created ODBC data source;
After the connection is configured, transformation rules are set up between the fields of the ODBC data source and the event log elements in MXML format;
After setting up the transformation rules, it is necessary to start the procedure for generating the event log in MXML format.

The event log initially contained 80 process instances and 1,658 events. Next, the event log was filtered and expanded in ProM as follows (using special ProM add-ons):

- The log is filtered in such a way that only sequences that end with the specified operation remain in it. In our case, all sequences should end with the operation “Final Test”, i.e. in future, incomplete instances of events will not be analyzed. The add-on “Filter Log” was used.
- The start and end operations, which play the role of a single entry point into the process and a single exit point, are added to the event log. Some process intelligence algorithms require such a single entry / exit point. The add-ons “Artifical start task” and “Artifical end task” were used.

After filtering, the event log contains 55 instances of processes (cases) and 645 events (events). The filtered and extended event log was further analyzed in ProM. To restore the process model, the Weijters heuristic algorithm [5] (Heuristics Miner add-on for ProM) was used, since it yields a fairly accurate process model, and the results can be converted to an eEPC model.

The Weijters heuristic algorithm is based on the α-algorithm and is intended to build a model of the work flow (operations) from the event log. For the algorithm to work, it is necessary that the event log contain at least 3 fields: process instance identifier, time, operation identifiers. Time is used to restore the order of operations. In the absence of time information, it is assumed that the sequence of operations corresponds to their sequence in the event log [1].

The parameters of the heuristic analyzer were established based on the principle of increasing the fit of the resulting model to the event log (fitness). Fitness indicator - compliance of the obtained model with the event log, demonstrates how the restored model allows the reproduction of all process instances in the event log. For each reconstructed model, a simulation model was created, for which experiments were carried out and indicators for labor intensity and total course time were calculated. This was done to increase the adequacy of the simulation model. The heuristic analyzer settings are listed in table 1 (parameters in bold italics are those that are changed relative to the default settings).

| Parameter \ Model                  | H1   | H2   | H3   | H4   |
|-----------------------------------|------|------|------|------|
| Relative-to-best threshold        | 0.5  | 0.85 | 0.85 | 0.85 |
| Positive observations             | 10   | 10   | 3    | 2    |
| Dependency threshold              | 0.9  | 0.4  | 0.6  | 0.4  |
| Length-one-loops threshold        | 0.9  | 0.9  | 0.8  | 0.4  |
| Length-two-loops threshold        | 0.9  | 0.9  | 0.8  | 0.4  |
| Long distance threshold           | 0.9  | 0.9  | 0.8  | 0.4  |
| Dependency divisor                | 1    | 1    | 1    | 1    |
| AND threshold                     | 0.1  | 0.1  | 5    | 5    |
| Use all-events-connected-heuristic| Yes  | Yes  | Yes  | Yes  |
| Use long distance dependency heuristics | No | No  | No  | No  |
Here “H1” is the heuristic model obtained with the default settings, “H2” is the heuristic model obtained at the second iteration, “H3” is the heuristic model obtained at the third iteration, “H4” is the heuristic model obtained at the fourth iteration.

For all reconstructed models, we determine the correspondence to the event log - the “fitness” indicator (table 2).

| Model | H1  | H2  | H3  | H4  |
|-------|-----|-----|-----|-----|
| Fitness | 84.9 | 87  | 87.3| 91  |

The model of the learning process obtained at the fourth iteration (“H4”) as the most relevant to the event log is shown in figure 2.

Next, it was necessary to extract information about the actors for operations from the event log using the add-on to analyze the organizational structure of the Role Hierarchy Miner for ProM. This miner schematically indicates which actor (rectangles “Student” and Artificial (ProM)) in which operation (ellipse with the name of the operation) is involved (figure 3). Also, this add-on provides a detailed table with an analysis of the execution of operations by the actors.

**Figure 2.** The restored model in the C-Net format, the fourth iteration (“H4”).
Figure 3. Actors Data for Event Log Operations.

Figure 3 shows that all operations, except for the fictitious “ArtificialStartTask” and “ArtificialEndTask” (artificial operations added by ProM), are performed by the same actor - the student.

Based on the model of the e-learning course developed using the heuristic algorithm, an agent simulation model AnyLogic was created using the techniques described previously in [2]. Figure 4 shows the main elements of the resulting ASM. In general, AnyLogic ASM, based on event-based eEPC models, consists of active objects (agents) that each have their own rules of behavior (defined by a state diagram in UML) and interacting with the external environment and each other by sending messages (the message is a specially implemented software Java class).

Figure 4, a) shows the tree of model objects, figure 4, b) displays the composition of objects nested in the “Main” class. The environment (environment) is modeled by the active “Main” object (contains variables, a timer, tables) into which other active objects can be nested (in this case, only one active “student” object is nested, the composition of which is shown in figure 5).

Objects placed in the environment (class "Main") mean: student - an agent that simulates a student; fileOutput - an object of the “text file” type for recording output information about modeling into a text file; fileEventLog - an object of the “text file” type for generating an event log in a text file; allTime - a variable for storing the total runtime calculated by the simulation model; allTimeTrud - a variable for storing the complexity of the process calculated by the simulation model; caseID is a variable for storing the process instance number during the multiple run of the simulation model.

The agent in the simulation model is a student for whom the process of passing the e-learning course is simulated. Figure 5 shows the tree of objects of the Student class, and the composition of objects nested in this class is also displayed.
All transitions between processes are carried out according to the logic of exclusive OR (XOR) and occur sequentially.

After the agent simulation model was created, one million experimental runs were carried out and the mathematical expectation and variance were calculated from the resulting data for the complexity and total time the student took the course. The results of calculations of the mathematical expectation (MO) and variance for simulation models in comparison with similar characteristics (row “ID” in the tables, meaning “initial data”) characterizing a selection of experimental data on the process (the size of the experimental sample is 55 cases of the process by the number of students who have completed training) are presented in tables 3-4.

Table 3. The results of the simulation model: an assessment of the complexity after 1 million runs.

| Model | Mathematical expectation, min | Deviation, min | Deviation, % | Dispersion, min^2 | Deviation, min^2 | Deviation, % |
|-------|-------------------------------|----------------|--------------|-------------------|-----------------|--------------|
| ID    | 280.94                        | -              | -            | 47 966.09         | -               | -            |
| H1    | 261.06                        | 19.88          | 7            | 59 158.53         | 11 192.43       | 23           |
| H2    | 248.76                        | 32.19          | 11.5         | 51 970.72         | 4 004.62        | 8            |
| H3    | 233.33                        | 42.62          | 17           | 40 975.72         | -6 990.37       | -14.6        |
| H4    | 216.7                         | 64.25          | 23           | 34 670.51         | -13 295.58      | -27.7        |

Table 4. Simulation Model Results: Estimated Total Run Time after 1 Million Runs

| Model | Mathematical expectation, min | Deviation, min | Deviation, % | Dispersion, min^2 | Deviation, min^2 | Deviation, % |
|-------|-------------------------------|----------------|--------------|-------------------|-----------------|--------------|
| ID    | 1,921.17                      | -              | -            | 5 940             | 602.01          | -            |
| H1    | 2024.45                       | -103.27        | -5.38        | 4 777             | -1 162          | -19.58       |
| H2    | 1854.35                       | 66.82          | 3.48         | 4 247             | -1 692          | -28.5        |

Figure 5. Composition of objects of class "Student" describing agent "Student".
4. Discussion
Thus, despite the rather complex (high dispersion) experimental data, which is usually characteristic of real organizational and production systems, it can be concluded that the reconstructed model in eEPC notation (using the “H4” and “H3” models as an example) to the agent simulation model AnyLogic is correct.

It can also be seen that a high fitness index for the event log does not guarantee high adequacy of the simulation models created on their basis, as shown by the examples of the “H4” and “H3” models. According to the authors, this is due to the presence of significant noise in the event log. For example, a significant number of testing attempts were interrupted by test subjects before the test was completed, but without disconnecting from the system, which led to a disconnection from the system only by waiting time, and, therefore, an instance of the testing process with an abnormally high execution time appeared in the event log.

In addition, a small number of instances of events in the log could also adversely affect data representativeness. Therefore, when creating simulation models based on models reconstructed by methods of intellectual analysis of processes, especially when the noise level in the event log is large, it is necessary to do several iterations to obtain an adequate simulation model.

References
[1] Van der Aalst W M P 2011 Process Mining: Discovery, Conformance and Enhancement of Business Processes (Berlin: Springer-Verlag)
[2] Lantsev E A and Dorrer M G 2013 Agent-based simulation of business processes in EPC notation Scientific and Technical Bulletin of Information Technologies, Mechanics and Optics 3 86-92
[3] Moodle.org. https://docs.moodle.org/36/en/Main_page (2019)
[4] Weijters A J M M and Van der Aalst W M P, Alves de Medeiros A K 2006 Process Mining with the Heuristics Miner-algorithm BETA Working Paper Series 166 16