Software Requirements Specification

for

Automated CPN Model Generation

Version 1.0

Prepared by: Mehmood Ahmed, Younes Müller, Shubham Balyan, Sonam Chugh, Deekshith Shetty

Created on: 22.11.2020

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
SRS_Automated_CPN_Model_Generation	22.11.2020	First version	1.0

1. Introduction

1.1 Purpose

In this document the requirement analysis of the “CPN Model” project is documented. The project’s goal is to automatically generate colored petri net (CPN) models out of event logs.
The deliverable lists requirements to the software and describes, which technologies will be used for the project.
The project mainly aims at two types of stakeholders:

1. Data Scientists in the area of process science that want to use colored petri nets for analysis.
2. Researchers that want to enhance the capabilities of pm4py.

1.2 Product Scope

The goal is to analyse a process log and to develop a model for the underlying process, which is then represented by a colored petri net (CPN). A CPN is a petri net, where the tokens have the ability to convey data and transitions only allow certain tokens to pass.
In our case the petri net, representing the process should contain information on the execution time and waiting time of the activities, and at decision points the empirical probability of each decision option. Further information like the distribution of roles and decision criteria for the decision points are also possible. This CPN model then can be used to simulate a process for an amount of time, hopefully yielding an event log similar to the original one.

The user can upload an event log in XES or CSV file format. The software discovers a process model in the form of a petri net, using a suitable algorithm like the Inductive Miner.
Then it enhances the model with performance information and decision distributions. The CPN is presented to the user, who then has the opportunity to change the additional information, such as waiting time and execution time. In order to work with the CPN model, it can be exported to a .cpn file, which follows the format of the CPN Tools software. The software should feature a graphical user interface, which presents the cpn to the user and allows them to perform each described step of the process.

1.3 Project Management

The project management of “CPN Model” project is carried out in a Trello board. The dashboard contains all the user stories as product backlogs and will be moved to sprint backlogs as the sprints progresses. The source code management will be handled on GitHub.

Trello: https://trello.com/b/SEjiZD71/virtual-office

Github: https://github.com/Younesmueller/CPN-Model-Process-Discovery-WS-20/

1.4 References

1. Description of the discovery and simulation of cpns:
    a. Discovering simulation models
    - A. Rozinat, R.S. Mans, M. Song, and W.M.P. van der AalstInternational Journal on Software Tools for Technology Transfer (STTT), Volume 10, Number 1, Pages 57-74
2. CPN Tools Website:
    - http://cpntools.org/
3. CPN Tools .cpn file description
    - http://cpntools.org/2018/01/09/dtd-for-net-files/
4. PM4Py Documentation
    - https://pm4py.fit.fraunhofer.de/docs

2. Overall Description

2.1 Product Perspective

As per the market research done by International Data Corporation (IDC), most companies are completely unaware of their processes not performing at the maximum capacity and are losing 20-30% of their revenue. Process mining provides a way to detect such problems early and allows companies to use it for monitoring and optimizing their process. Providing a tool to do computer simulations is very useful and versatile and allows us to gain insight into the operation of systems.

The project allows a user to create colored petri nets called CPN models from a real event log that can reflect the real process as precisely as possible and follows a business process management approach. Information systems deployed in any industry usually logs the events happening in a business process, now these event logs can be transaction logs or audit logs. Further analysis of these event logs can reveal important domain knowledge and patterns that can help the organizations to improve the quality of their business process, fulfill customer expectations, and increase profits. As an added advantage, the pain points and the delays within organizational processes can also be identified in this way, and hence the allocation of resources to various tasks can be performed efficiently. This project aims to fulfill the business requirement of creating a simulation model for a business process, with some additional information to have a better understanding of the process and its various aspects.

In the process of creating the simulation models as an end result, users will also be able to have the option to change various parameters that would reflect in the final result. This simulation model can be further used to create simulated event logs.

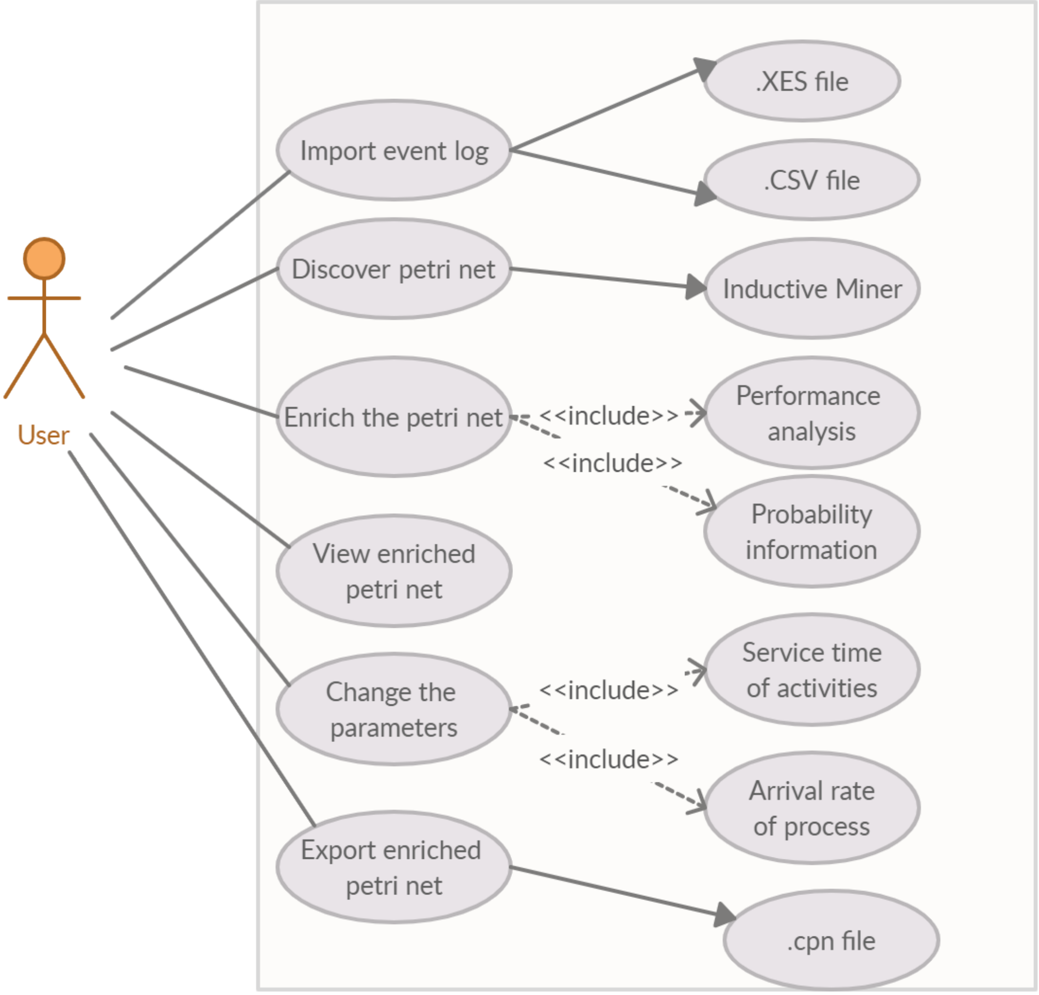
2.2 Product Functions

The user can perform different functions as part of the project and the functions that can be performed from a user perspective are as follows:

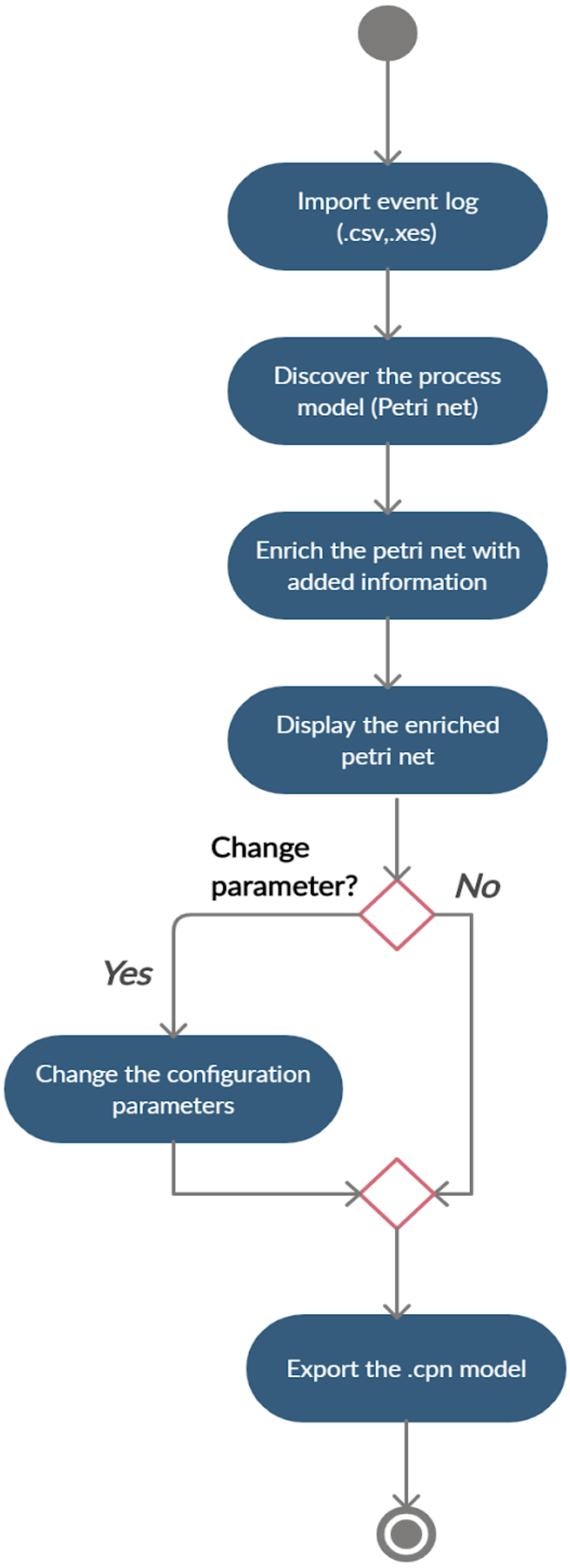
- Import an event log (XES or CSV format)
- Discover a process model in the form of a petri net
- Enrich the discovered petri net with performance analysis and probability information
- View the enriched petri net
- Change the parameters for CPN model generation like service time of activities, arrival rate of the process
- Export the enriched petri net as a CPN model

2.3 Behaviour Models

2.3.1 Use Case Diagram



2.3.2 Activity Diagram


Import event log (.csv,.xes)
Discover the process model (Petri net)
Enrich the petri net with added information
Display the enriched petri net
Change parameter?
No
Yes
Change the configuration parameters
Export the .cpn model

2.4 Operational Requirements

The project will make use of the open source tools and technologies to achieve the objective of the project. The various technologies and the tools used are as follows:

- Python 3.7
- pm4py library: process mining package for Python
- PyCharm: the Python IDE
- Microsoft Visual Studio Code: source code editor
- CPN tools software
- Trello: project management application
- GitHub: source code management software

2.5 Design and Implementation Constraints

The design and implementation constraints of the project are listed as follows:

- The format of the event log can only be either of these two: XES or CSV.
- Implementation has to be done in Python using the pm4py library.
- Use existing pm4py functionalities as extensively as possible, already existing tools in pm4py should not be reinvented.
- Agile project management approach should be followed throughout the process.
- The implementation code should be of high quality - it should be readable, modular, testable and should follow object oriented programming style.
- The end product should be a modular design where every feature is wrapped into a separate module and the modules depend on each other through well-written APIs.
- The various deadlines as mentioned in the project initiation document should be met in time.

2.6 Assumptions and Dependencies

Assumptions on the provided event log:

- is in CSV or XES format.
- has valid syntax based on the format.

Dependencies

- The pm4py library and its implementations of algorithms.

3. System Features and API Specification

3.1 Import an event log

3.1.1 Description

The event log (in XES or CSV format) is imported to the server/backend by the user. The event logs must record the scheduling, the start, and optionally the completion of an activity along with the time stamp, a performer, and some additional data.
The user can specify the event log file to upload using file path in curl command. The file is then stored in an appropriate location in the server.

3.1.2 Priority:

This user feature is of high priority.

3.1.3 Functional Requirements

- The event logs must record the scheduling, the start, and optionally the completion of an activity along with the time stamp, a performer, and some additional data.
- The user specifies the event log file to upload using file path in curl command.
- The event log is validated in the server, i.e, the file type and the content is checked.
- If the file is invalid, it returns a message "The event log file is not valid".
- If the file is valid, the file is stored in an appropriate location in the server for further processing and a success message is displayed in the GUI along with a unique identifier for the event log.
- **Acceptance Criteria**: the event log is successfully imported without any errors.

3.1.4 API Specification

Method	Resource	Description
POST	/eventlog	Input: Event log file (Filetype: XES or CSV) Output: HTTP Response 200 OK { "event_log_id": <EVENT_LOG_UNIQUE_IDENTIFIER>

		"message": "Event log uploaded successfully"
		}
		
		HTTP Response 400 Bad Request
		{
		"message": "The event log file is not valid" + errormessage
		}

3.2 Discover a process model in the form of a petri net

3.2.1 Description

From the event log imported (referred to by its unique identifier), a control-flow discovery algorithm is applied to automatically create a process model that reflects the causal relations between the examination activities in the log. This process model is in the form of petri net.

3.2.2 Priority:

This user feature is of high priority.

3.2.3 Functional Requirements

- Process model is discovered on the event log referred by its unique identifier.
- If the event log is not found, return a message "The event log file doesn't exist for the provided identifier".
- If found, a control-flow discovery algorithm is applied on the event log.
- **Acceptance Criteria**: Process model is outputted as a petri net.

3.2.4 API Specification

Method	Resource	Description
GET	/processmodel?eventlog=<EVENT_LOG_UNIQUE_	Input: Unique identifier of the event log file on which process model is to be discovered Output: HTTP Response 200 OK

	IDENTIFIER>	{
		"event_log_id": <EVENT_LOG_UNIQUE_IDENTIFIER>,
		"process_model_img": <PROCESS_MODEL_IMAGE>
		}
		
		HTTP Response 404 Not Found
		{
		"message": "The event log file doesn't exist for the provided identifier" + errormessage
		}

3.3 Enrich petri net with performance information

3.3.1 Description

We would like to gain more insight into the performance perspective of the process by enhancing the process model with information about arrival rate of process, activity service time (execution times and waiting times) and the probabilities for taking alternative paths.

The arrival rate of a process is defined as the number of new cases arriving per time unit (on average) at the process. The execution time is the time between the start and the completion of the activity. The waiting time is the time between the point at which the last activity that is a direct predecessor of this activity was completed and the moment at which the execution of the activity itself is started.

3.3.2 Priority:

This user feature is of high priority.

3.3.3 Functional Requirements

- Process model is discovered and enriched on the event log referred by its unique identifier.
- If the event log is not found, return a message "The event log file doesn't exist for the provided identifier".
- If found, a control-flow discovery algorithm is applied on the event log to generate a process model.

- Enhance the process model with information about the arrival rate of cases. This can be derived from the start times of the first activity in each process instance.
- Enhance the process model with information about execution times and waiting times for the activities. Extract this information using the timestamp from ‘start’ event and ‘complete’ event (if available) in the event log. In case the activity completion timestamp is not available, the completion time is set using the start time of next activity.
- Enhance the process model with information about the probabilities of alternative paths based on how often each path was followed during log replay.
- Specify the execution and waiting times in terms of a normal distribution, by calculating their mean and variance values for each activity.
- All the mentioned above values are measured in minutes.
- **Acceptance Criteria**: Enriched process model is outputted as a petri net.

3.3.4 API Specification

Method	Resource	Description

GET	/processmodel/enriched?eventlog=<EVENT_LOG_UNIQUE_IDENTIFIER>	Input: Unique identifier of the event log file on which process model is to be discovered and enriched
		Output:
		HTTP Response 200 OK
		{
		"event_log_id": <EVENT_LOG_UNIQUE_IDENTIFIER>,
		"process_model_img": <PROCESS_MODEL_IMAGE>
		}
		
		HTTP Response 404 Not Found
		{
		"message": "The event log file doesn't exist for the provided identifier" + errormessage
		}

3.4 View the enriched petri net

3.4.1 Description

After enriching the discovered process model in the form of a petri net with performance analysis and probability information, we would like to display the petri net to the user to have some kind of visual understanding of how the final simulation model will look like.

3.4.2 Priority:

This user feature is of high priority.

3.4.3 Functional Requirements

- The enriched process model can also be referred to by the event log’s unique identifier.
- If the process model is not found, return a message "The process model doesn't exist for the provided identifier".
- If found, return the enriched process model with performance analysis and probability information.
- **Acceptance Criteria**: Enriched process model is outputted as a petri net.

3.4.4 API Specification

Method	Resource	Description

GET	/processmodel/enriched/view?eventlog=<EVENT_LOG_UNIQUE_IDENTIFIER>	Input: Unique identifier of the event log file for which enriched process model has been discovered.
		Output:
		HTTP Response 200 OK
		{
		"event_log_id": <EVENT_LOG_UNIQUE_IDENTIFIER>,
		"process_model_img": <PROCESS_MODEL_IMAGE>
		}
		HTTP Response 404 Not Found
		{
		"message": "The process model doesn't exist for the provided identifier" + errormessage
		}

3.5 Change the parameters for CPN model generation

3.5.1 Description

After the petri net is enriched with the performance information, users can change the following parameters for CPN model generation - service time of activities and arrival rate of the process.

3.5.2 Priority:

This user feature is of high priority.

3.5.3 Functional Requirements

- The enriched process model can also be referred to by the event log’s unique identifier.
- If the process model is not found, return a message "The process model doesn't exist for the provided identifier".
- If found, return the enriched process model with performance analysis and probability information.
- Enriched process model is outputted as a petri net.
- **Acceptance Criteria**: User should be able to change the parameters - service time of activities, and arrival rate of process for the final CPN model.

3.5.4 API Specification

Method	Resource	Description

POST	/processmodel/enriched/changeparameter?eventlog=<EVENT_LOG_UNIQUE_IDENTIFIER>	Input: JSON Request
		{
		“service_time”:100,
		“arrival_rate”:0.5
		}
		Output:
		HTTP Response 200 OK
		{
		“message”: ”The process model has been updated”
		}

		HTTP Response 400 Bad Request
		{
		"message": "Something went wrong while updating the model." + errormessage
		}

3.6 Export the enriched petri net to CPN model

3.6.1 Description

After the petri net is enriched with the performance information, users can export the petri net as a CPN model in the form of a .cpn file.

3.6.2 Priority:

This user feature is of high priority.

3.6.3 Functional Requirements

- The enriched process model can also be referred to by the event log’s unique identifier.
- If the process model is not found, return a message "The process model doesn't exist for the provided identifier".
- If found, export the enriched process model with performance analysis and probability information to a CPN model.
- **Acceptance Criteria**: The output file should have .cpn extension.

3.6.4 API Specification

Method	Resource	Description
GET	/processmodel/enriched/export?eventlog=<EVENT_LOG_UNIQUE_IDENTIFIER>	Input: Unique identifier of the event log file for which enriched process model has been discovered. Output: HTTP Response 200 OK Content-Disposition: attachment; filename="model.cpn"

		HTTP Response 400 Bad Request
		{
		message: “Unable to generate .cpn file” + errormessage
		}

3.7 GUI for importing logs

3.7.1 Description

Improve the user experience by providing a web UI to allow users to upload the event log for which the process model is to be discovered. The user uploads the event log file by selecting the file in a file explorer dialog.

3.7.2 Priority:

This user feature is of medium priority.

3.7.3 Functional Requirements

- On clicking the upload button, a file explorer dialog opens where the user can select the event log file.
- Show progress indicator when the file is being uploaded to the server.
- On success, display the server APIs success message and the computed process model petri net on the UI.
- On failure, display the failure message on the UI.
- **Acceptance Criteria**: The user should be able to upload event log successfully without any errors.

3.8 GUI for changing the parameters

3.8.1 Description

Improve the user experience by providing a web UI to allow users to change the parameters - service time of activities, and arrival rate of process for the final CPN model. The user enters the parameter values using dropdowns and textbox.

3.8.2 Priority:

This user feature is of medium priority.

3.8.3 Functional Requirements

- Users can enter the parameter values by selecting the dropdowns and textbox.
- User submits the request by clicking on the submit button.
- Show progress indicator when the computation is performed on the server.
- On success, display the server APIs success message and the enriched process model petri net on the UI.
- On failure, display the failure message on the UI.
- **Acceptance Criteria**: The user should be able to change the parameters for the final CPN model generation successfully without any errors.

4. Other Nonfunctional Requirements

4.1 Performance Requirements

In any process mining application, the most expensive task, performance-wise, is mainly discovering the process model. The response time to obtain the model depends on the size of the event log, average trace length and the discovery algorithm. As the event log will be provided by the user and the chosen algorithm Inductive Miner is not the fastest of the discovery algorithms, an appropriate message will be displayed to the user during processing. Also, a timeout will be set in order to terminate the HTTP request if the process takes way too long.

4.2 Safety Requirements

In any case, the application should not modify the original event log. Also, in case of any error, the error message will be sent to the client side to enable users and the developers to identify the problem.

4.3 Security Requirements

The application does not have any identity authentication requirements. Any type of user having access to the application can use it without any additional privileges.

The user’s event log, the generated petri net, CPN model and any other generated data by the application will not be made accessible or sold to any third parties without user’s consent.

4.4 Software Quality Attributes

In addition to the unit testing by the developers, the application will be tested thoroughly by the Quality Engineer before deploying the application, to identify any bugs. Also, to ensure the correctness of the generated CPN model, the model will be verified using CPN tools and will be cross validated with the event log.

The application is required to be flexible with the 2 common formats of the event data i.e XES and CSV. Also, it should handle files with both start time, completion time of an activity and files with start time only.

Due to the well designed and easy to use interface, the application will be used without any mandatory training. However, a user manual will be provided to the user describing all the features and usage of the application.