Requirement Specification

Streaming event data compliance checking in Python

Jingjing Huo	376323
Sabya Shaikh	384606
Zheqi Lyu	378653

Content

1 Introduction

1.1 Purpose

The purpose of the document is to convert all assorted ideas about our system to formal definition, including its requirements with respect to consumers, project initiator etc., and provide a detailed overview of our system, its parameters and goals.

1.2 Scope

Process mining is widely used to visualize processes, analyze processes performance based on event logs. Unfortunately, the majority techniques handle the offline events data. Moreover, some techniques are applied on live event streams under the assumption, that the input stream has no spurious events which leads to the degradation of results quality on the real data. In order to release the assumption, we build probabilistic automata to filter out these spurious events.
Our project will be able to effectively check event compliance, i.e. when anomalous pattern is detected on the streaming data, alerts will be triggered and spurious events will be filtered out using probabilistic automata with different prefix length. Our system will handle 1000,000 streaming events in 1 hour. Eventually, the user must be able to access web-service via the client using http post request to provide the streaming events in order to check event compliance.

1.3 Definitions

CaseID: It refers the CaseID in Event log.
In the below example, there will be two cases, with CaseID 1 and 2.

Window: It contains the last k events of one case in our memory, where k is the prefix size.
In below example, if prefix size is 2 and window content is “BC” with CaseID=1 and an event “D” with same CaseID occurs, then the current window will become “CD”.

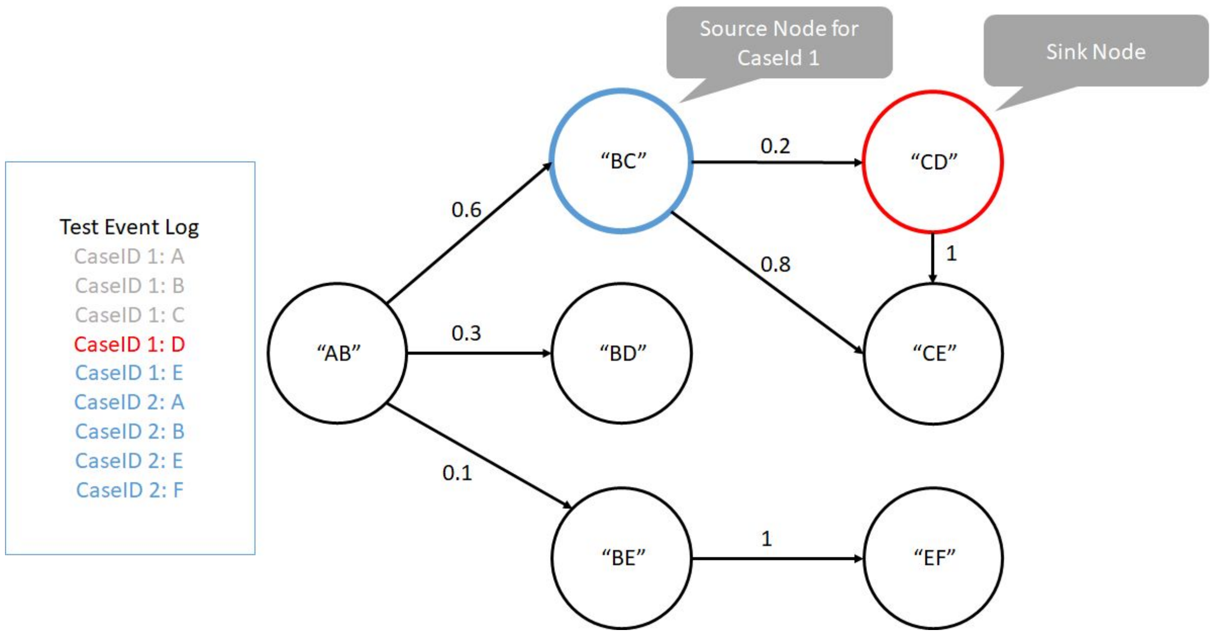
Sink_Node: It has the same content as that of the window.
In the below example, “CD” is supposed to be the Sink_Node.

Source_Node: It has the same content as the previous window with same CaseID as that of the input event.
In below example, the Source_Node is “BC”.

Connections: The occurrence of events from “Source_Node” to “Sink_Node”.

Test Event Log: Event log file used for testing.

Training Event Log: Event log used for training.



1.4 Stakeholder

Business: The alert is triggered only when business asks for it.
For example, in our system, the deviations can occur in two cases :
Case 1: The incoming event is never supposed to occur after a particular event;
Case 2: The incoming events probability to occur after a particular event is lesser than the threshold probability.
“Triggering alerts only when deviations have occured” saves time and CPU cycles for the server.

To measure the correctness of the alerted event we will need the below listed information in the alerts send to user:

- Automaton of which window size rejected it.
- Current Source_Node in automata.
- The reason the alert was triggered:
  case 1 : Due to lower probability of occurrence of Sink_Node than threshold;
  case 2: There is no path between Souce_Node and Sink_Node in Automata.
- Alert Count - To show the overall statistics of alerts for a client (only after the entire streaming event log has been processed).

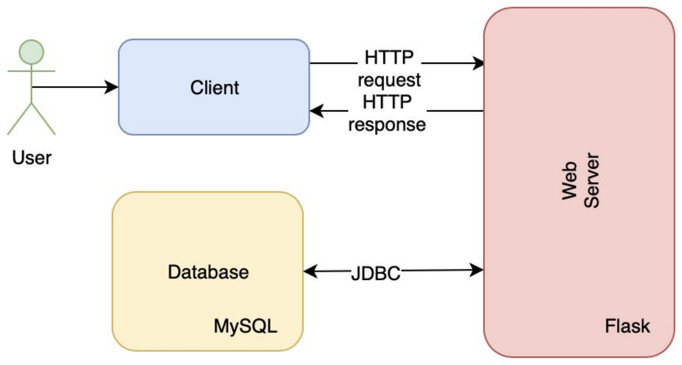
User: The user can easily get the alerts for spurious events in the streaming data. And the response time is limited to 1s for 300 events as the requests shall be processed parallely using multi -threading environment of server.

1.5 Reference

[1]https://sourceforge.net/software/compare/Modern-Requirements-Suite-vs-Visure-Requirements-vs-IRIS-Business-Architect-vs-Accompa/
[2]https://www.modernrequirements.com/modern-requirements-suite4tfs-3/
[3]https://visuresolutions.com/requirements-management-tool/
[4]https://web.accompa.com/features/
[5]https://biz-architect.com/products/

2 Overview

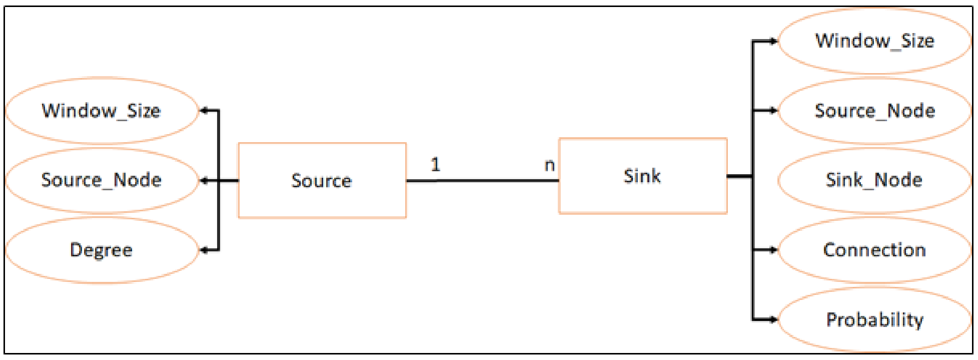
2.1 System Model



The user provides the events logs via client. The client uploads this event logs to Web server using the HTTP POST-request. The data is processed and checked for conformance and an alert via HTTP-response is triggered when a deviation is observed. The database is used to store the automata built.

2.2 Product feature

We use MySQL Database to store the Automaton and the Alert information.
Automata:



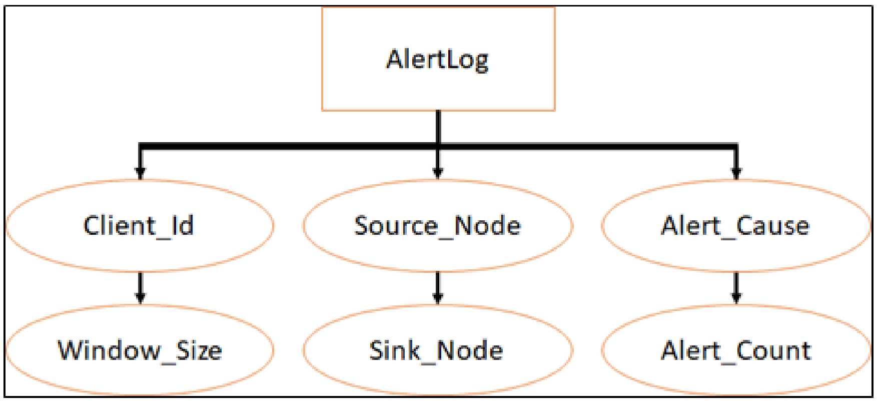
The field descriptions of **Source** entity are as below:
Window_Size: Type: Integer. It defines the length of the prefix.
Source_Node: Type: String. It is the event names (based on prefix length).
Degree: Type: Integer. It is the number of outgoing edges from a Source_node, i.e, the number of events that occur after Source_Node event.

The field descriptions of **Sink** entity are as below:
Window_Size: Type: Integer. It defines the length of the prefix.
Source_Node: Type: String, It is the event names (based on prefix length).

Sink_Node: Type: String. It is the event names (based on prefix length).
Connection: Type: Integer. It defines the number of times Sink_Node event occurs after a Source_Node event.
Probability: Type: Double. It defines the probability of a particular Sink_Node event occuring after a Source_Node event.

One Source node can have multiple Sink nodes.

AlertLog



The field descriptions of **AlertLog** entity are as below:
Client_Id: The client that requested the service for a particular event
Window_Size: Type: Integer. Same as Automata
Source_Node: Type: String. Same as Automata
Sink_Node: Type: String.

Case 1: If the event can occur after the current Source_Node but probability of this event to occur is lower than threshold then the Sink_Node is that particular event with reference to Sink Entity;

Case 2: If the event should never occur after the current Source_Node then then the Sink_Node is that particular event with no reference to existing Automata.

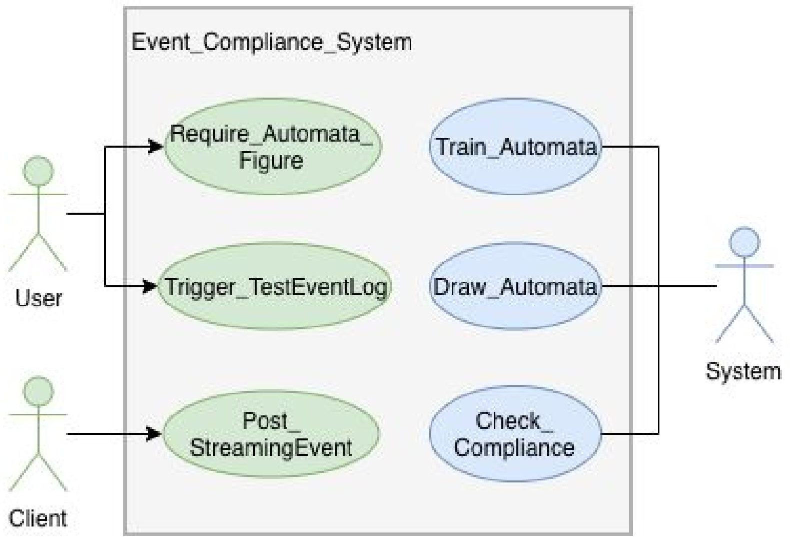
Alert_Cause: Type: Char: 'M' or 'T'.

- 'T'(Threshold): When the alert is due to lower probability of occurrence of Sink_Node than threshold.
- 'M'(Missing Sink_Node from Automata): When there is no path between Source_Node and Sink_Node in Automata.

Alert_Count: Type: Integer. The number of times the alert was triggered for a particular path from Souce_Node to Sink_Node.

2.3 User Classes and Characters

The following **Use case Diagram** describes the interactions among the elements of our event compliance system, in particular, user, client and system:



2.4 Operating Environment

Operating environment for the system is as listed below:

- **Database:** MySQL
- **Server system:** Python with Flask
- **Operating system:** Mac OS and Windows

2.5 Constraints

- The system can handle only one input event log to train at a particular instance, hence the user can't build another automata.
- Due to the constraints of our single input file for training, we can only test online data from the same scenario.
- Our automata is static once the training phase ends, we don't update it using online data.

2.6 Assumptions

The order of event arrival of a trace corresponds to the order of execution.

- The event is spurious, if any of the automata alerts that event to be spurious.
- The memory is large enough to keep all the temporary cases.
- The streaming event traces are completed.

3 System Features

3.1 External Interface Requirement

3.1.1 User Interface:
The feedback will be showed in Command Line, where the user runs the client service.

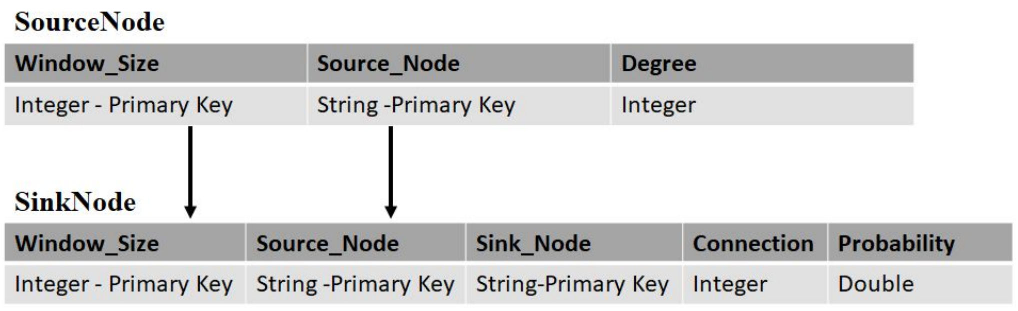
3.1.2 Hardware Interface:

RAM: 8GB
OS: Windows 64 Bit, Mac OS 64 Bit
Processor Name: Intel Core i5
Number of Cores: 2
Processor base frequency: 2 GHz- 2.5 GHz

3.1.3 Software Interface

- **Database**: The structure of Automata and Alert logs are stored in MySQL database.

Automata:



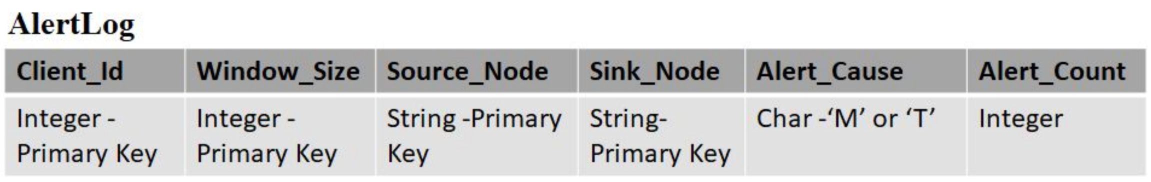
SourceNode

Window_Size	Source_Node	Degree
Integer - Primary Key	String -Primary Key	Integer

SinkNode

Window_Size	Source_Node	Sink_Node	Connection	Probability
Integer - Primary Key	String -Primary Key	String-Primary Key	Integer	Double

Alert Logs:



AlertLog

Client_Id	Window_Size	Source_Node	Sink_Node	Alert_Cause	Alert_Count
Integer - Primary Key	Integer - Primary Key	String -Primary Key	String- Primary Key	Char -'M' or 'T'	Integer

- JDBC driver will be used for the connection between webserver and database.

3.1.4 Communication Interface

- HTTP Protocol will be used for communication between the client and the webserver.
- uWSGI will be used for hosting services.

3.2 Functional Requirements

3.2.1 Use case for User

Name	**Tigger_TestEventLog**
ID	UC01
Description	The user runs the client service
Actors	User
Organisational Benefits	This use case helps to run the client service that provides the streaming data to the webservice
Frequency of Use	Multiple times - whenever the user needs to test the data
Triggers	None
Preconditions	None
Postconditions	The client service has been run
Main Course	1. The user runs the client service 2. The user will get feedbacks if some events are alerted
Alternate Course	To 1:: a) The user interupts the service
Exceptions	To 1::a) The client service is not avalible.

Name	**Request_AutomataFigure**
ID	UC02
Description	The user requests automata figure via client using PULL-request to access webservice
Actors	User
Organisational Benefits	This use case helps to request a figure of the automata created by using the training event log file and labels the alerts by using the testing event log file
Frequency of Use	Everytime a user needs to see the automata
Triggers	None
Preconditions	The system is available and automata PDF file is created
Postconditions	None
Main Course	1. The user runs the client service

	2. The user gets the PDF containing the automata and alerts
Alternate Course	None
Exceptions	To 2:: If the system is busy in building the automata, then trigger an alert of “The system is working hard to build a automata! Please try it later!”

3.2.2 Use case for Client

Name	**POST_StreamingEvent**
ID	UC03
Description	The client reads the file in local and posts each event line by line using POST-request to webservice, this acts as streaming event
Actors	Client
Organisational Benefits	This use case helps to provide the streaming data to the webservice to check its compliance
Frequency of Use	Multiple times whenever the user needs to test the data
Triggers	UC01 is executed
Preconditions	Event log file is available in local directory
Postconditions	The streaming event is sent to webserver
Main Course	1. The client reads the test event log line by line from local and sorts the data according to the timestamp 2. The client sends that each event using the HTTP POST-request to web server 3. Waits for the corresponding response from the service
Alternate Course	To 2:: a) The client service is interupted by user
Exceptions	To 1::a) The file is not available b) False format of the test event log file

3.2.3 Use case for Server

Name	**Train_Automata**
ID	UC04
Description	The system uses the training event log file and stores it in database appropriately to build Probabilistic Automata

Actors	System
Organisational Benefits	The system uses the training event log to train and build Probabilistic Automata.
Frequency of Use	Only once, when the system is initialized
Triggers	None
Preconditions	Event log file is available in server local directory
Postconditions	Draw_Automata will be automaticly executed
Main Course	1. Read the training event log line(event) by line(event) 3. Update the automata information in database 4. Repeat 1-2 until end of the file and close the file 5. Update the probability attribute to complete the automata information
Alternate Course	To 1,2,3,4,5:: If a request of showing automata is received, then trigger an alert of “The system is working hard to build a automata! Please try it later!”
Exceptions	To 2,3,4:: If interrupted, then restart to train the event log To 5:: If interrupted, then repeat 5

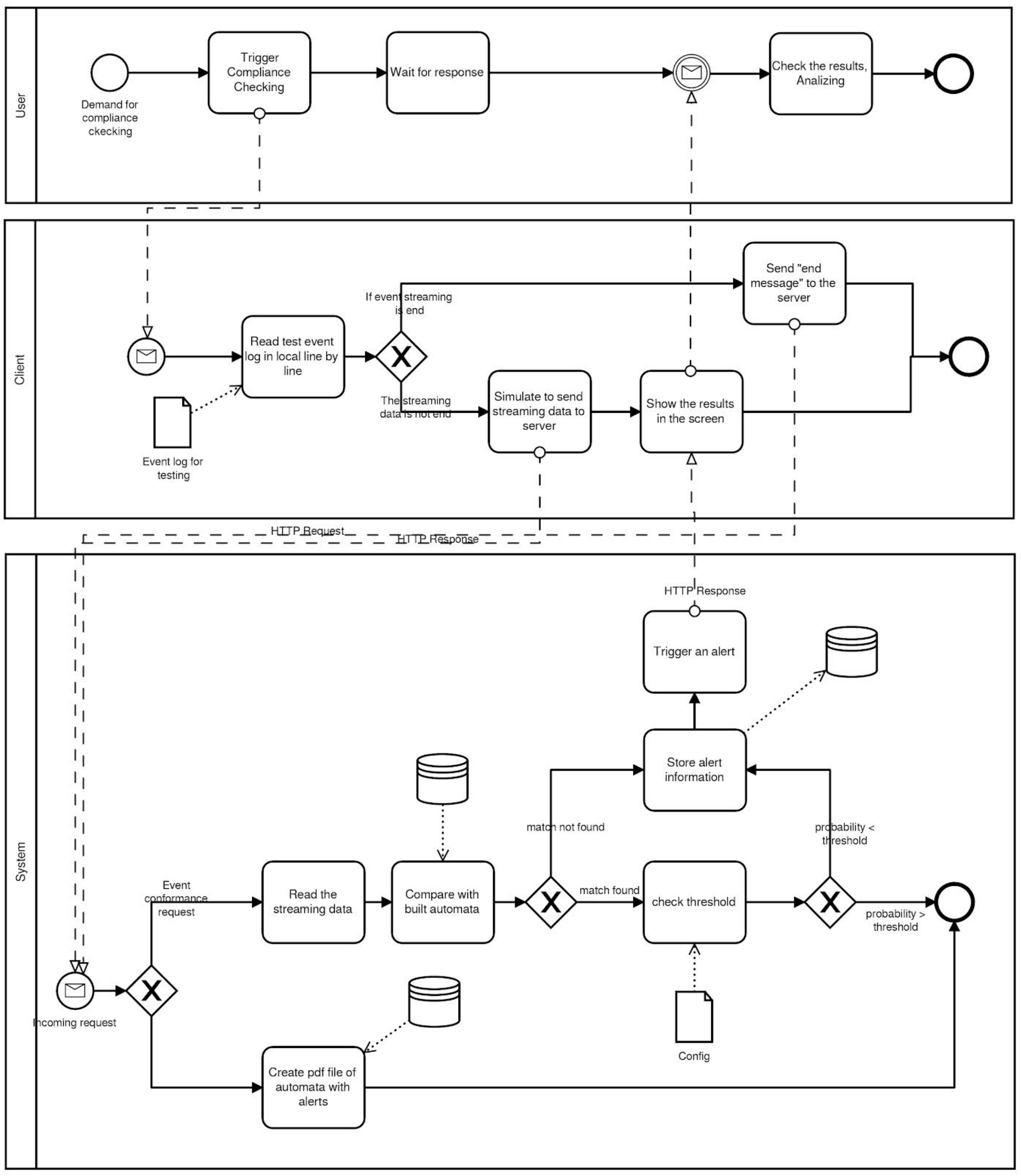
Name	**Check_Compliance**
ID	UC05
Description	The system compares the received event with the automata stored in the database and based on the probability of its occurrence it alerts the user
Actors	System
Organisational Benefits	The system uses Probabilistic Automata to check compliance of the streaming event
Frequency of Use	Everytime a event is sent to webserver to check compliance
Triggers	POST request with a streaming event from the client
Preconditions	The server is avalible and probabilistic Automata is built
Postconditions	Alert is sent in case deviations from normal behavior has occured
Main Course	1. Wait for the events to occur 2. Read the event and compare this event to the

	events sequence in automata entity in database 3.a. If match found in the table, compare this events probability with the threshold probability. 3.a.i If probability is lower than threshold alert the user, and store the error to Alert log in database 3.b If match not found alert the user, and store the error to Alert log in database
Alternate Course	4. Repeat 1, until get the “END” message from client
Exceptions	To 3,4 :: If interrupted, then restart this user case

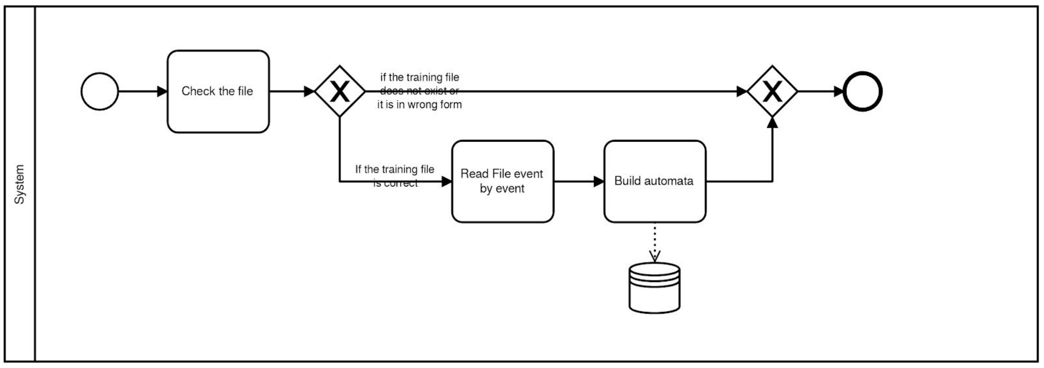
Name	**Draw_Automata**
ID	UC06
Description	The system uses the information of automata and error log in database to draw the figure in a PDF file
Actors	System
Organisational Benefits	The system uses the data in database to draw Probabilistic Automata
Frequency of Use	Everytime a new request of showing automata
Triggers	UC05 is executed
Preconditions	The service gets the “END” message from client
Postconditions	None
Main Course	1. Create a PDF file locally 2. Draw automata in the file 3. Store it locally
Alternate Course	None
Exceptions	None

3.2.4 Function model (BPMN)

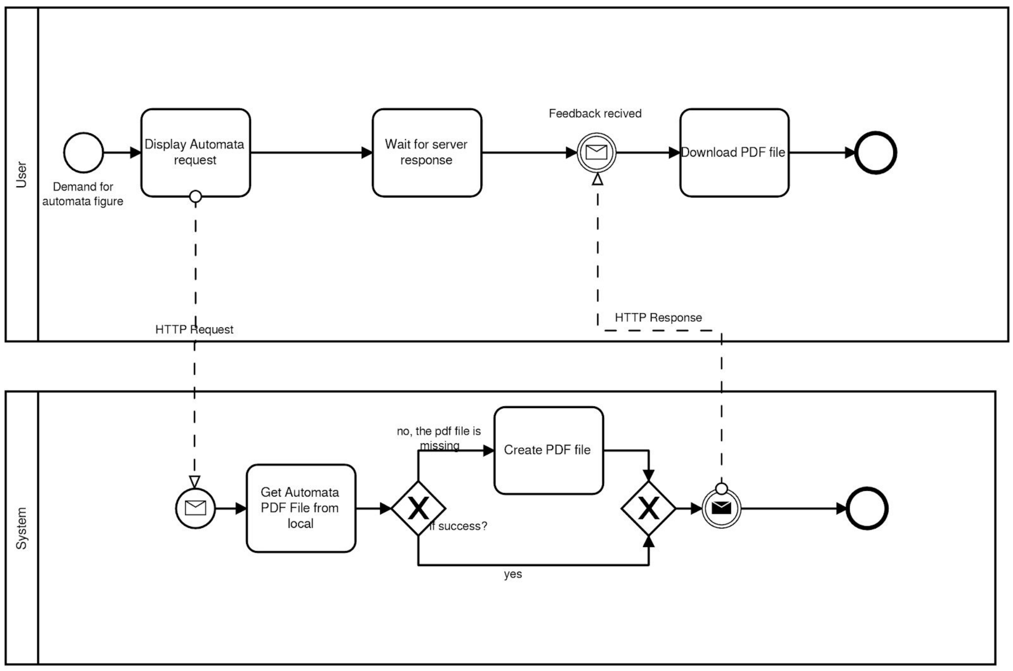
Check Compliance:


User
Demand for compliance ckecking
Trigger Compliance Checking
Wait for response
Check the results, Analizing
Client
Read test event log in local line by line
Event log for testing
If event streaming is end
Send "end message" to the server
The streaming data is not end
Simulate to send streaming data to server
Show the results in the screen
HTTP Request
HTTP Response
System
HTTP Response
Trigger an alert
Store alert information
match not found
probability < threshold
Event conformance request
Read the streaming data
Compare with built automata
match found
check threshold
probability > threshold
Incoming request
Config
Create pdf file of automata with alerts

Building Automata:


System
Check the file
if the training file does not exist or it is in wrong form
If the training file is correct
Read File event by event
Build automata

Display Automata:



3.3 Non Functional Requirements

3.3.1 Software Quality Attributes

- Performance
  - 1,000,000 events will be processed in 1 hour. Multi streaming data come in at the same time.
  - Generation of alert in case of deviation of a single event shall take less than 4 ms seconds for 95% of the cases.
  - Rendering the pdf document of Automata to user shall take less than 1s in 99% of the cases.

- Reliability
  - Failure of compliance checking is when an event is spurious but alert wasn't triggered.The failure rate of our system shall be less than 0.01%. This means while processing 300 events in one second, a failure can occur in every 0.5min. This failure depends on the quality of our automata and the training event log.
  - The failure that our system can not build automata or generate the automata figure shall be 0.01%. If this failure occurs we need to restart our training phase to ensure that our automata is created successfully.

- Availability
  - The services shall achieve 99.9% uptime.

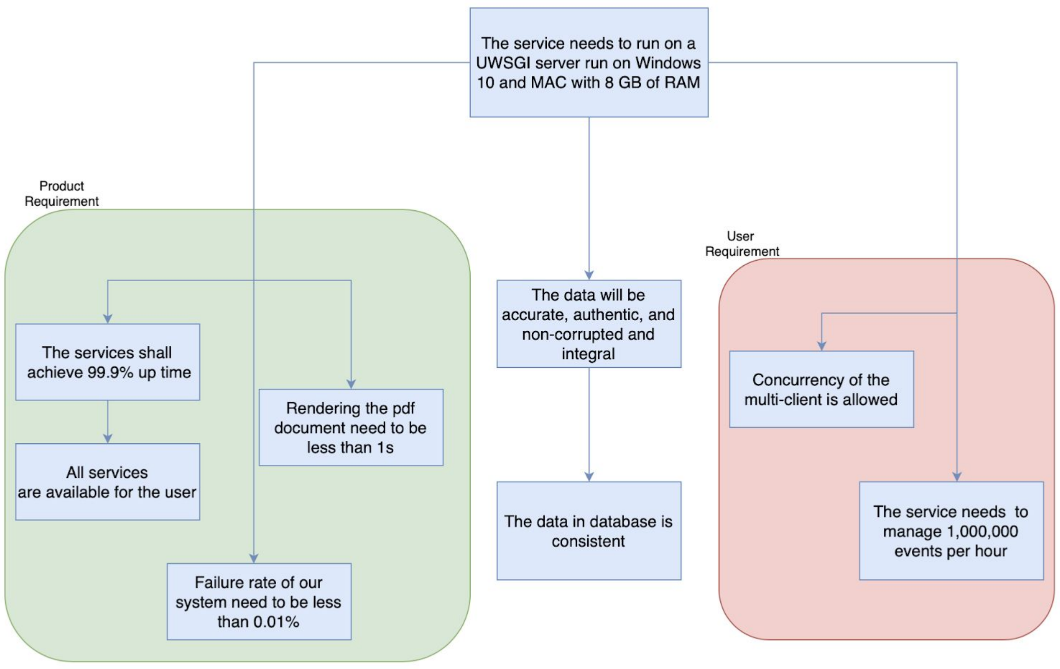
In case the server is busy building automata, the user shall be notified with busy status- This would happen rarely as we consider training the automata only once initially.

- Integrity
    - The data shall be 100% accurate.

    The data maintained in the system will be accurate, authentic, and non-corrupted at any point of time because the training data is build only once with high supervision. Thereafter, no changes are allowed on automata, hence, maintaining the consistency.

- Accessibility
    - All the services (request for automata figure, trigger conformance checking) will be available to users 24*7 if the user has connection to internet.

3.3.2 Allocation and Flow-Down of Requirements


The service needs to run on a UWSGI server run on Windows 10 and MAC with 8 GB of RAM
Product Requirement
The services shall achieve 99.9% up time
All services are available for the user
Rendering the pdf document need to be less than 1s
Failure rate of our system need to be less than 0.01%
The data will be accurate, authentic, and non-corrupted and integral
The data in database is consistent
User Requirement
Concurrency of the multi-client is allowed
The service needs to manage 1,000,000 events per hour

4 Requirements Management Tools Comparison

In the following table we provide a comparison about four different requirements management tools. They have own benefits and can be used to easily manage our project requirements. [1]

	Modern Requirements Suite	**Visure Requirement**	**Accompa**	**IRIS Business Architect**
Platform	Windows/ SaaS	Windows	Windows/ Mac/ Saas/Web	Windows/ SaaS/ Cloud/ Mac/ Web
On-Premise Version	Available	Available	Not available	Available
Collaboration Feature	Available	Available	Available	Available
History Tracking Feature	Available	Available	Available	Available
AI based Business Analyst BOT	Available	-	-	-
Automated Requirements Elicitation	Alice BOT captures requirements using text-to-voice or voice-to-text interactions with your device and chats	Can capture requirements from MS Excel, MS Word and Outlook	“SmartForms and “SmartEmails” are used to capture the requirements from web and mails.	-
Description	See Below			

Description:

Modern Requirements Suite: It contains diagramming, document and report creation, trace analysis, Use Case and Mockup creation, Automatic Test Case and User Story generation.[2]

Visure Requirement: It is a state-of-the-art Software specifically designed to provide an integral support to the complete Requirement process. It integrates in the same environment support for other processes such as risk management, test management and issue and defect tracking.[3]

Accompa: It is a leading cloud-based requirements management software that helps us capture, track and manage requirements for our products & projects. It is delivered as a hosted, cloud-based software - this means you don't have to download or install any software! In addition, one more attractive is that Accompa is highly customizable.[4]

IRIS Business Architect: It has more functions than the other. It enables key business stakeholders and business/enterprise architects to plan business models and execute initiatives in alignment with their corporation's strategies. From business strategies, decision making, customer journeys, value streams and gap analysis down into business capabilities and their organizational, technology and information enablers, IRIS Business Architect provides rich management functionality to continually and dynamically reflect relevant changes within a corporation's business architecture.[5]

5 Phase Review

Zheqi Lyu:

About team: As last week we worked well. We always sat together and discussed some details about our project, especially for requirement analyse. In my opinion, this is a effective and pleasant way and I enjoy this.

About this document: This document helps us to have a clear view about what we need to do. Although we had some doubts during this phase, with the help of tutor we overcame and had a deeper understanding.

About me: I'm keeping to improve my english and Python programing skill, so that I can do well in the coming phase.

Jingjing Huo:

Requirement analysis document is one of the most important docs during the software developing, since it decides how many works we exactly need to do and how we do it. Figuring out the requirements is the start of a successful software development.

So far I like the way we worked. We set some fixed time for group meeting and discuss some details together and gather some questions to ask tutor. This is great because there are always some teammates in the discussion who can ask questions you didn't expect, and some wrong ideas can be corrected. Although the time is limited in the next stage, since there are a lot work to do, we must do some tasks separately, I don't think this tradition will change. At the end I appreciate the help from tutor and my teammates.

Sabya Shaikh:

The introduction of Requirement analysis phase was really a good idea. It was of great help to understand the project well. We had tons of doubts with the functionalities and everything was cleared by the end! We can now easily focus on designing and implementing these functionalities.
Working in my current group essentially helps us to brainstorm ideas and clear any doubts within our team.Team-mates are really understanding and it is easy to schedule meeting as per everyone’s need. I enjoy working with this team.
Arranging meetings with tutor to gain his insights on our document is definitely required to be in same page with him and to make sure we are progressing in the right direction and we could easily achieve this by communicating with him via mails and scheduling personal meeting.