POLITECNICO
MILANO 1863

CLup project Roberto Buratti, Hrvoje Hrvoj, Ozan Incesulu

Requirement Analysis and Specification Document

Deliverable:	RASD
Title:	Requirement Analysis and Verification Document
Authors:	Roberto Buratti, Hrvoje Hrvoj, Ozan Incesulu
Version:	1.0
Date:	5-12-2020
Download page:	https://github.com/Furcanzo/BurattiIncesuluHrvoj
Copyright:	Copyright © 2020, Roberto Buratti, Hrvoje Hrvoj, Ozan Incesulu – All rights reserved

Contents

List of Figures

List of Tables

1 Introduction

1.1 Purpose

1.1.1 Description of the Proposed System

The project C̈Lup - Customer Line upïs a line spot reservation system that is planned to be used by managers, clerks and customers of many local vendors and chains. The system aims to come up with a handy solution for the ongoing issue of proper social distancing management particularly in the matter of grocery shopping, by providing assistance to cope with the customer load for managers and helping customers access to products in a safe and controlled manner.

In particular, users will be able to see the locations, get a line number and book in advance for the grocery stores they would like to visit. Once assigned a line number, the customer will be able to track the estimated time of arrival of the line as well as will wait for the notification that informs about his or her line's forthcoming arrival, hence waiting time in the line in the crowd is minimum. Also, C̈Lupp̈rovides uniquely generated QR codes per the line number, which can be utilized by the store managers as a proper monitoring tool in the entrances and exits of the locations. All in all, the general purpose of the product is to keep the congestion levels in lines of the locations at minimum via providing useful features for all the users.

1.1.2 Goals

- G_1 Customers can issue a line number for a location.
- G_2 Customers can issue line numbers for their future visits.
- G_3 Customers can detail their visits by category and/or product.
- G_4 Customers can plan their visit to the store.
- G_5 Customers may prefer to use alternative time slots or partner stores for their visit.
- G_6 Managers can prevent customers from issuing line numbers.
- G_7 Managers can customize the system to allow optimizations for increased granularity, flow control and time slot forecasting.
-
- G_9 Customers can obtain printed line number tickets.

1.2 Scope

1.2.1 Targeted Users

CLup aims to resolve the problem of Customers queueing up in front of a location, without any control over the availability of place in the location and further contact tracing by the managers.

Customer:

Customers will be able to obtain specific line numbers for various locations using CLup, which they can track the estimated time available with and also view the location on a map application to plan their visit. Customers can further obtain line numbers for future visits, based on the info provided by the system about the availability of free spots on the specific time intervals. They may prefer to visit a different branch of the same chain. Furthermore, they can provide specific products they intend to purchase or set

an estimated time for their visit to allow finer granularity. Some customers may also prefer to obtain a line number upon visiting the store physically. To plan their visit in a time slot where the location will be less crowded, the users can see the occupancy of the location based on already taken line numbers and forecasts provided by the system.

Clerk:
The clerk (which can be a shopping assistant or a security detail) can monitor the flow of customers and manually intervene in case of missing line numbers via performing manual checkout for a specific customer, or by printing line numbers physically for some customers.

Manager:

The location manager (which could be an actual manager, or someone responsible for handling customer management) can provide details regarding the availability of products and the location in general, by setting the opening hours, maximum allowed customers in shop, in-shop location of different products and categories, maximum amount of reservations that can be made per customer, line number timeout and the location. Also, for chains and relevant stores, the manager can add chain members for the location.

1.2.2 Relevant Phenomena

Phenomenon	World / Machine	Shared
Line Number	Machine	Yes
Line Number Ticket	World	No
Product	World	No
Product Category	World	No
In-store Location	World	Yes
Occupancy Forecast	Machine	No
Store	World	Yes
Time Slot	Machine	Yes
Ticket Printer	World	Yes
Line Number Timer	Machine	No
Customer Scheduling Algorithm	Machine	No

Table 1: Relevant Phenomena of CLup

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

- *Location*: the physical location of the business that operates the line reservation system
- *Manager*: the user in charge of executive action within the location
- *Customer*: the user with the goal of making a visit to the location
- *Clerk*: the user in charge of handling the entrance and exit of customers
- *Hard-coded Super User*: a single user that is used to create managers of locations administered by the software vendor
- *Visit Time*: the time interval in which a customer performs a visit to the location

- *Line Number*: A number that indicates the ordering of a specific customer in the line
- *Line Number Ticket*: A physical ticket printed that features the line number and the QR code.
- *Time Slot*: Specific intervals of time that are determined by the opening hours and average visit time per customer.
- *Partner Store*: A different location that is included in the same beneficiary chain of command (such as another member of the franchise or store chain) or in a mutual agreement with the specific location
- *Product*: Any item, items, service or services demanded by the customer, and provided by the location to the customer.
- *In-store Location*: A location of a specific product or a product category inside the location.
- *Working hours*: The time intervals that the store is open during each day.
- *Maps API*: Google Maps API that is used as a mapping service

1.3.2 Acronyms

- **RASD**: Requirement Analysis and Specification Document
- **QR Code**: Quick Response Code
- *API*: Application Programming Interface

1.3.3 Abbreviations

- G_n: n^{th} goal
- D_n: n^{th} domain assumption
- R_n: n^{th} functional requirement

1.4 Revision history

1.5 Reference Documents

- Specification Document: R&DD Assignment AY 2020-2021
- IEEE Std 830-1998: IEEE Recommended Practice for Software Requirements Specifications
- ISO/IEC 18004:2015 QR Code bar code symbology specification
- Maps API documentation

1.6 Document Structure

This document is composed of six sections, each with the purpose described below:

- **Introduction**: This section provides an introduction of the problem, the scope of the project with details regarding the goals, target users and phenomena. The goals of the project are formulated in accordance with the description of actions and actors in the Specification Document. Within the project's scope, properties and duties of different users and user groups are described in Target Users section. Furthermore, under the scope of the project, the relevant phenomena of the project is presented through their relevance to the world and the machine.

- **Overall Description**: This section builds upon the introduction section by providing expansion of the scope and relevant functionalities of the system. The basic perspective is provided considering the shared phenomena between the world and the system and the integrations of the system with other parties and products. The overall actions that can be conducted over the system is described using a detailed class diagram, portraying all required components of the system, their inner state and provided functionalities for other components to facilitate their purpose. All core features of the system is listed and explained using user scenarios to provide a better understanding of the real-world condition on which the use case should apply to, a state chart for demonstrating the relationship between various states the system can be in during the execution of the provided core feature scenario. Next, the requirements for the system to archive the aforementioned goals and allow the execution of the provided scenarios. The user characteristics, based on their roles in the system is further evaluated in this section, through the means of the needs that they have, how the system archives those needs via its requirements and goals. No system can be designed without a broad idea on what domain it exists in and how much the environment limits its ability to perform its features. Ergo, we provide the expected assumptions of the domain, the dependencies of the system to external components and the limits imposed by the world on the system.

- **Specific Requirements**: This section is the main section of the document. The external interfaces that the system requires to function correctly is presented with details and additional mockups if needed. The interfaces section is mainly concentrated on different views of user interfaces as the user-facing part is the main way to integrate with the system. In this section, functional requirements of the system are presented in detail, a common use case diagram is provided, and each use-case of the system is analyzed with details on a use case description table and a sequence diagram. The performance requirements of the system, with a focus on The design constraints of the system, with a focus on different standard compliances, hardware limitations of the system to function and other constraints, like GDPR are also elaborated on this section. The vertical aspects that needs consideration are evaluated under the Software System Attributes subsection of this document, with an emphasis on Reliability, Availability, Security, Maintainability and Portability.

- **Formal Analysis Using Alloy**: This section features various models built and hypothesis verified using Alloy. This section demonstrates that some aspects of the requirements document can be formally proven to be correct and provides additional information about the proposed system to the engineers.

- **Effort Spent**: This section features the effort table, in which all team members provide a rough estimation on the time spent on creation of the various sections of the document.

- **References**: This section features different reference materials referred inside this document.

2 Overall Description

2.1 Product Perspective

2.2 Product functions

2.3 User characteristics

2.4 Assumptions,dependencies and constraints

2.4.1 Domain Assumptions

- D_1 %80 of the customers and all clerks and managers have basic ICT skills, has an email address that they are willing to use to authenticate to the system and has a smart phone or equivalent device that can connect to the Internet, have a browser that supports UTF-8, display QR codes and has a mapping application.
- D_2 Locations will not be visited by no more than 1000 people in any time slot.
- D_3 %98 of the customers will arrive at the given location either without a ticket or with a ticket that has not timed out.
- D_4 E-mail addresses are not shared by multiple users of the system.
- D_5 Clerks' mobile devices are equipped with at least one camera that the system can use.
- D_6 All users have a basic understanding of how the line numbering system works and respects the ordering provided by the system.
- D_7 Managers have an estimate for the amount of reservations that their location can at most have.
- D_8 Managers' device has location services that has a location acquisition error for no more than 20 meters.
- D_9 Clerks are constantly monitoring the locations entrances and exits.
- D_{10} Locations have printing equipments that are in 5 meters range of all the entrances that can print QR codes and line numbers.
- D_{12} At least one manager is available in the location during the working hours.
- D_{13} The customer's entry and exit to the store is determined by whether the clerks have checked them in and out.
- D_{14} The customer has their line number or line number ticket available with them through their visit, including their exit from the store

2.4.2 Dependencies

2.4.3 Constraints

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

3.1.2 Hardware Interfaces

3.1.3 Software Interfaces

3.1.4 Communication Interfaces

3.2 Functional Requirements

Name	**Book future line number**
Actors	Customer
Entry conditions	The customer is logged in the web app and wants to book a visit to the store.
Event flows	• The customer clicks on the "book a visit" button in the web app • The web app asks the time slot and the estimeted time of the visit presenting as default value the average of the previous times of visit of the same user • The customer sets the time slot and the estimated time of the visit • The web app asks what category of products the custumer wants to buy • The customer set the products categories • The web app generate the line number contacting the server • The web app generate the QR code on the informations retrived from the server
Exit conditions	The customer have booked a visit to the store
Exceptions	• The server cannot retrieve the line number since the time slot is full

Table 2: Use Case: Book future line number

Name	**Grant access**
Actors	Clerk, Customer
Entry conditions	The customer has already obtained the QR code and he is going to access the store.
Event flows	• The Clerk scan the QR code from the customer smartphone or printed ticket using the clerk web app • The Clerk web app analyze the QR code and contact the Server • The server decides if the custumer can enter based on the informations recived that were stored in the QR code • The server communicate to the clerk app its decision • The clerk let the customer enter the store
Exit conditions	The customer enter the store
Exceptions	• The server communicate to the clerk that the customer cannot enter because his/her line number is not reached yet

Table 3: Use Case: Grant access

Name	**Initialize**
Actors	Manager
Entry conditions	The manager of the store needs to set the basic informations of the store in order to start the service
Event flows	• The manager clicks on the initialize button • The web app request to the server the initialization form • The initialization form is returned to the manager • The manager fills the form and send it back to the server through the web app • The server stores the informations and return an ack
Exit conditions	The system is initialized and can offer all its functions
Exceptions	• Some mandatory part of the form is not filled.

Table 4: Use Case: Initialize

Name	**Monitor state**
Actors	Manager
Entry conditions	The manager wants to monitor the number of customers in the store in real time
Event flows	• The manager clicks on the "monitor" button • The web app sends the request to the server • The server return the actual number of customers in the store • The web app repeats the process periodically until the manager clicks the "stop monitor" button
Exit conditions	The manager is informed on the number of the customers in the store in real time
Exceptions	•

Table 5: Use Case: Monitor state

Name	**Print guest ticket**
Actors	Clerk, Customer
Entry conditions	The customer is at the store to buy a physical ticket since he doesn't have a smartphone or doesn't want to use it
Event flows	• The customer ask the clerk to have a ticket • The clerk generate a ticket using the clerk web app • The clerk web app communicate to the server the intention to generate a ticket • The server retrieve a line number and generate a ticket • The server sends the ticket back to the web app • The clerk print the ticket • The clerk gives the ticket to the customer
Exit conditions	The customer have a ticket
Exceptions	• The server cannot retrieve the line number since the time slot is full.

Table 6: Use Case: Print guest ticket

Name	**Print guest ticket**
Actors	Clerk, Customer
Entry conditions	The customer is at the store to buy a physical ticket since he doesn't have a smartphone or doesn't want to use it
Event flows	• The customer ask the clerk to have a ticket • The clerk generate a ticket using the clerk web app • The clerk web app communicate to the server the intention to generate a ticket • The server retrieve a line number and generate a ticket • The server sends the ticket back to the web app • The clerk print the ticket • The clerk gives the ticket to the customer
Exit conditions	The customer have a ticket
Exceptions	• The server cannot retrieve the line number since the time slot is full.

Table 7: Use Case: Print guest ticket

Name	**Schedule a stop**
Actors	Manager
Entry conditions	The manager wants to schedule a period of time in which the store will be closed and so the system shouldn't allow users to book a visit in that time
Event flows "schedule a stop" button	• The manager click on the • The web app returns a form for requesting the time of the stop • The manager fills tha form and sends it to the server through the web app • The server stores the information in the database and return an ack
Exit conditions	The system as a scheduled stop stored in its database and will use it to prevent customers from booking a visit in that time period
Exceptions	• There is already a planned stop in that period

Table 8: Use Case: Schedule a stop

Name	**See amount of customers in the store**
Actors	Customer
Entry conditions	The customer is logged in the web app and wants to know how much customers are in the store in order to decide if book a visit or retrive a line number
Event flows	• The customer clicks on the "customers in store" button • The web app forward the request to the server • The server returns a table containing the amount of customers in the store for each time slot
Exit conditions	The customer know the amount of other customers in the store and can plan his/her visit
Exceptions	

Table 9: Use Case: See amount of customers in the store

Name	**See store location**
Actors	Customer, Maps API
Entry conditions	The customer needs to know where the store is located and the time needed for going there
Event flows	• The customer clicks on the "store location" button • The web app contact an external API that provides a Map service sending the information of the store location stored in the web app • The external Maps API returns a map from the customer position to the store with a time estimation
Exit conditions	The customer know how to go to the store and the time needed for going there
Exceptions	• The customer's smartphone doesn't have a GPS • The customer didn't provide the authorization to the web app for accessing the GPS information

Table 10: Use Case: See store location

Name	**Sign up**
Actors	Customer
Entry conditions	The customer opened the web app and it's not registered yet, but want to register in order to have access to the system functionalities
Event flows	• The customer click on the "sign up" button • The customer inserts his/her credentials • The web app sends the information to the server • The server store the data relative to that user in the database • The server returns an ack to the web app
Exit conditions	The customer is now registered and can use all the functionalities of the web app
Exceptions	• The credentials inserted in the registration form are already used (e.g. the email) • The credentials inserted in the registration form are wrong (e.g. not a valid email)

Table 11: Use Case: Sign up

Name	**Sign up**
Actors	Customer
Entry conditions	The customer opened the web app and it's not registered yet, but want to register in order to have access to the system functionalities
Event flows	• The customer click on the "sign up" button • The customer inserts his/her credentials • The web app sends the information to the server • The server store the data relative to that user in the database • The server returns an ack to the web app
Exit conditions	The customer is now registered and can use all the functionalities of the web app
Exceptions	• The credentials inserted in the registration form are already used (e.g. the email) • The credentials inserted in the registration form are wrong (e.g. not a valid email)

Table 12: Use Case: Sign up

Name	**Stop for emergency**
Actors	Manager
Entry conditions	An emergency occurred and the manager wants to immediately stop the system to distribute line numbers
Event flows /her choice to stop the system	• The manager click on the "Emergency stop" button • The web app ask for confirm to the manager • The manager confirms his • The web app contact the server • The server stops the service and return an ack
Exit conditions	The system has interrupted the service
Exceptions	• The manager abort the operation instead of confirming it

Table 13: Use Case: Stop for emergency

3.2.1 Requirements

- R_1 The system must allow users to authenticate using their e-mail address and password.
- R_2 The system must allow customers to register using their e-mail address, their name, surname, phone number and a new password.
- R_3 The system must provide a hard-coded super user to allow addition of locations and managers of locations.
- R_4 Managers must be able to add additional managers and clerks as users.
- R_5 Managers must be able to set and update location specific information, that are maximum number of customers in the location at any given time, opening and closing hours of the store per each day, line number timeout, limit of reservation per customer on a predetermined time interval that is one of month, week or day, and location of the place
- R_6 Managers can add any other location as a partner store.
- R_7 Managers can stop the system from issuing any more tickets for a given day
- R_8 Managers can schedule the system stop for a future time.
- R_9 Managers can set the in-shop locations for different categories and product items.
- R_{10} In case of a system stop, no further line numbers can be issued for the given time slots.
- R_{11} In case of a system stop, all line numbers in the stop time slots has to be cancelled.
- R_{12} The system must cancel those line numbers that the customer didn't arrive to the location for more than the set timeout interval.
- R_{13} In case of a ticket cancel, customer must be notified with an e-mail notification.
- R_{14} Clerks must register the entrance and exit of customers via scanning the QR code for their line number.
- R_{15} Clerks must be able to generate line number tickets in a printer compatible format.

- R_{17} Customers must be able to obtain a line number, except when the system is stopped or the store is full.
- R_{18} Customers must be able to obtain line numbers for different time slots in the future.
- R_{19} Customers can not obtain line numbers that exceed the quantity per time interval limits.
- R_{20} Customers can not obtain line numbers for time intervals that the system is stopped by a manager.
- R_{21} Customers must be able to see the estimated time available for their line number.
- R_{22} Customers must be able to set or update their phone number, password, name and surname.
- R_{23} Customers can select specific product and/or product categories they plan to visit in the location while obtaining a line number.
- R_{24} Customers can set an estimated time for their visit while obtaining a line number.
- R_{25} Customers must be able to view the shop location
- R_{26} Customers can view the occupation forecasts for the location at different time slots.
- R_{27} Customers can see the alternative suggestions for time slots while obtaining a line number for the future.
- R_{28} Customers can view the occupancy for the partner stores, if preferred time slot is not available while obtaining a line number.
- R_{29} Customers can view their line numbers with the number and the QR code.
- R_{30} The system must be able to provide a forecast for the occupancy of each location for any given time based on past visits.

3.3 Performance Requirements

3.4 Design Constraints

3.4.1 Standards compliance

3.4.2 Hardware limitations

3.4.3 Any other constraint

3.5 Software System Attributes

3.5.1 Reliability

3.5.2 Availability

3.5.3 Security

3.5.4 Maintainability

3.5.5 Portability

4 Formal Analysis Using Alloy

4.1 Alloy introduction

In this section is presented the Alloy representation of some critical aspects the system must present. It is worth highlighting that below only stable states are considered, in which the system may be found. As a consequence, no edge cases are treated.
In particular, the following features are modelled:
- There can't never be more customers in a section then the maximum capacity of that section, in any given moment.
- A booking is possible only by a registered customer.
- Every visit to the store is granted by an access title.

4.2 Alloy code

```
// Signatures ------------------------------------------------------------------

abstract sig Store {
sections : some Section
}

one sig PrimaryStore extends Store{
partnerStores : set PartnerStore
}

sig PartnerStore extends Store {}

sig Section {
maxCapacity: one Int
}{
maxCapacity > 0
}

sig Email {}

sig Password {}

abstract sig User {
email : one Email,
password : one Password
}

sig Customer extends User {}

some sig Manager extends User {
store : one Store
}

some sig Clerk extends User {
manager : one Manager,
```



```
store : one Store
}{
manager.store = store
}

abstract sig AccessTitle{
customer : lone Customer,
sections: some Section,
estimatedEnterTime : one Time,
estimatedExitTime: one Time,
}{
estimatedEnterTime.timestamp < estimatedExitTime.timestamp

}

sig Ticket extends AccessTitle{
lineNumber : one ActualLineNumber
}

sig Booking extends AccessTitle{
lineNumber : one BookedLineNumber
}{
estimatedEnterTime.timestamp > lineNumber.timeSlot.start.timestamp
estimatedEnterTime.timestamp < lineNumber.timeSlot.end.timestamp
#customer = 1
}

sig Visit{
accessTitle : one AccessTitle,
enterTime : one Time,
exitTime : lone Time
}{
enterTime.timestamp < exitTime.timestamp
}

//UTC standard format: number of seconds from 1970/01/01 00:00:00
sig Time {
timestamp : one Int
}{
timestamp>=0
}

sig TimeSlot {
start : one Time,
end : one Time
}{
start.timestamp < end.timestamp
}

abstract sig LineNumber {
```



```
timeSlot : one TimeSlot,
number : one Int
}{
number >= 0
}

sig ActualLineNumber extends LineNumber{}

sig BookedLineNumber extends LineNumber{}

//Facts------------------------------------------------------------------

fact credentialInUsers {
all mail : Email | mail in User.email
all pw : Password | pw in User.password
no mail : Email, disj user1, user2 : User | mail in user1.email && mail in u
no pw : Password, disj user1, user2 : User | pw in user1.password && pw in u
}

fact everySectionInAStore {
all section : Section | section in Store.sections
no section : Section, disj store1, store2 : Store | section in store1.sectio
}

fact everyLineNumberInAnAccessTitle {
all ln : LineNumber | ln in (Ticket.lineNumber + Booking.lineNumber)
no ln: LineNumber, disj t1, t2 : Ticket | ln in t1.lineNumber && ln in t2.lin
no ln: LineNumber, disj t1, t2 : Booking | ln in t1.lineNumber && ln in t2.li
all disj ln1, ln2 :LineNumber | ln1.number = ln2.number => ln1.timeSlot != l
}

fact AVisitForEachAccessTitle {
no at: AccessTitle, disj visit1, visit2 : Visit | at in visit1.accessTitle &
}

fact everyPartnerSoreInAPrimaryStore {
all partnerStore : PartnerStore | partnerStore in PrimaryStore.partnerStores
}

fact bookOnlyAStore {
all accessTitle : AccessTitle | one store : Store | accessTitle.sections in
}

fact consistenQueue {
all disj at1, at2 : AccessTitle |
((at1<:Ticket).lineNumber.number < (at2<:Ticket).lineNumber.number && (at1<:
at1.estimatedEnterTime.timestamp =< at2.estimatedEnterTime.timestamp

all disj at1, at2 : AccessTitle |
((at1<:Booking).lineNumber.number < (at2<:Booking).lineNumber.number && (at1
```



```
at1.estimatedEnterTime.timestamp =< at2.estimatedEnterTime.timestamp
}

fact noOverBooking {
all  time : Time, section : Section |
#{at : AccessTitle | at in
{visit : Visit |
((visit.enterTime.timestamp =< time.timestamp && visit.exitTime.timestamp>=
(visit.enterTime.timestamp =< time.timestamp && visit.accessTitle.estimatedE
(visit.accessTitle.estimatedEnterTime.timestamp =< time.timestamp && visit.a
section in visit.accessTitle.sections
}.accessTitle
} =< section.maxCapacity
}

fact equallyLongTimeSlots{
all ts1, ts2 : TimeSlot | ts1.end-ts1.start = ts2.end-ts2.start
}

//Assertions------------------------------------------------------------

assert noMoreVisitsThanAccessTitle {
#Visit =< #AccessTitle
}
check noMoreVisitsThanAccessTitle

assert atLeastACustomerForHavingABooking {
#Customer = 0 => #Booking=0
}
check atLeastACustomerForHavingABooking

assert neverMoreCustomerThanMaxCapacity {
all  time : Time, section : Section |
#{c : Customer | c in
{visit : Visit |
((visit.enterTime.timestamp =< time.timestamp && visit.exitTime.timestamp>=
(visit.enterTime.timestamp =< time.timestamp && visit.accessTitle.estimatedE
(visit.accessTitle.estimatedEnterTime.timestamp =< time.timestamp && visit.a
section in visit.accessTitle.sections
}.accessTitle.customer
} =< section.maxCapacity
}
check neverMoreCustomerThanMaxCapacity

//Worlds----------------------------------------------------------------

//Normal Condition
pred world1{
#PartnerStore=1
#Section=3
```



```
#User=5
#Customer=3
#AccessTitle=5
#Visit=3
#TimeSlot=5
}

//No Customers
pred world2{
#Customer=0
}

//Saturated Store
pred world3{
#PartnerStore=0
#Section=1
Section.maxCapacity=3
#Visit=3
no cus : Customer | cus not in AccessTitle.customer
#TimeSlot=1
#Time = 2
}

run world1 for 10
run world2 for 10
run world3 for 10
```

4.3 World generation

In the following paragraph are displayed examples of worlds generated from the predicates featured in the Alloy implementation.

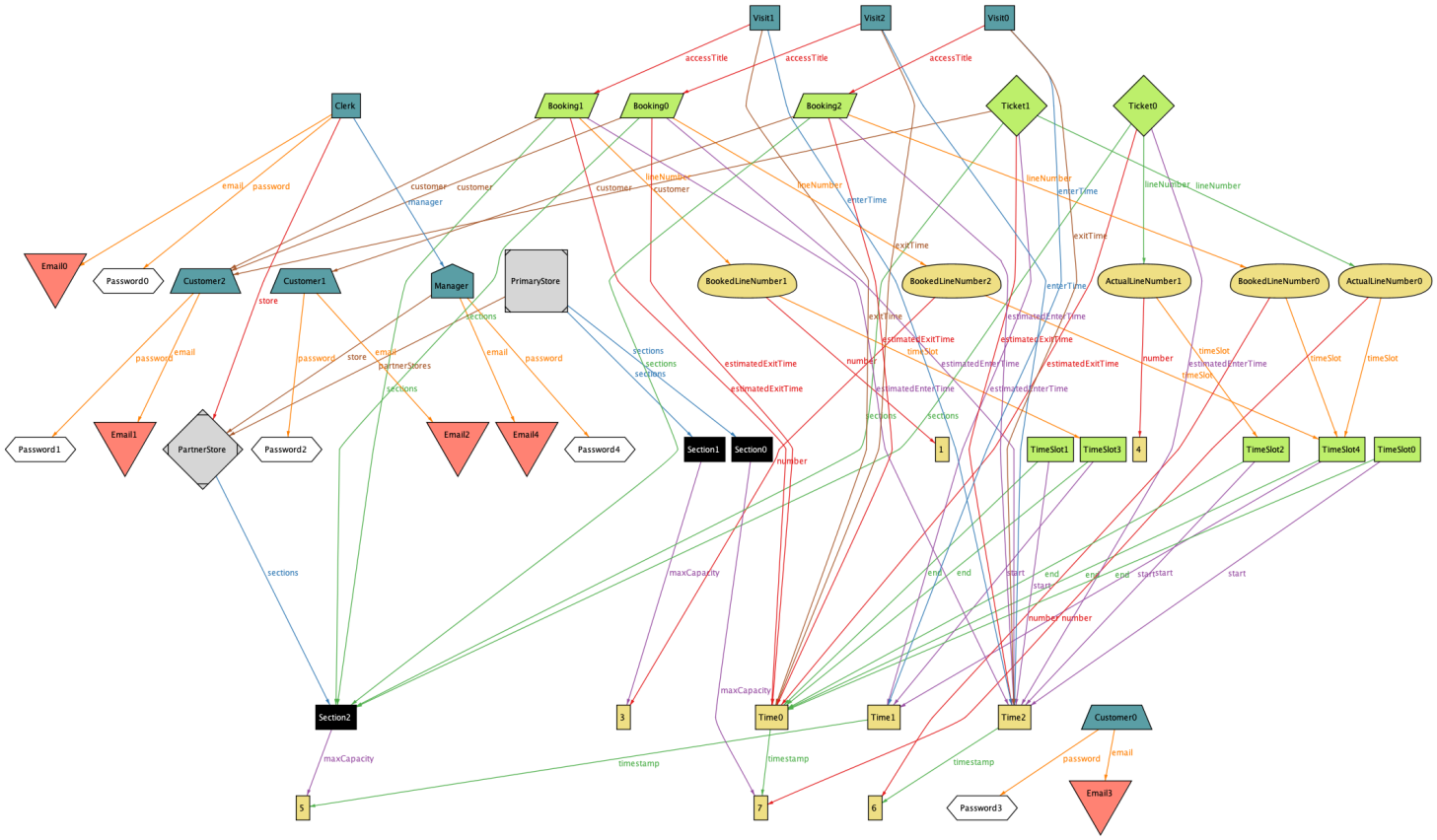

Figure 1: World 1

World 1 describes the model in normal conditions, its only purpose is to show an instance of the model without checking any particular property.

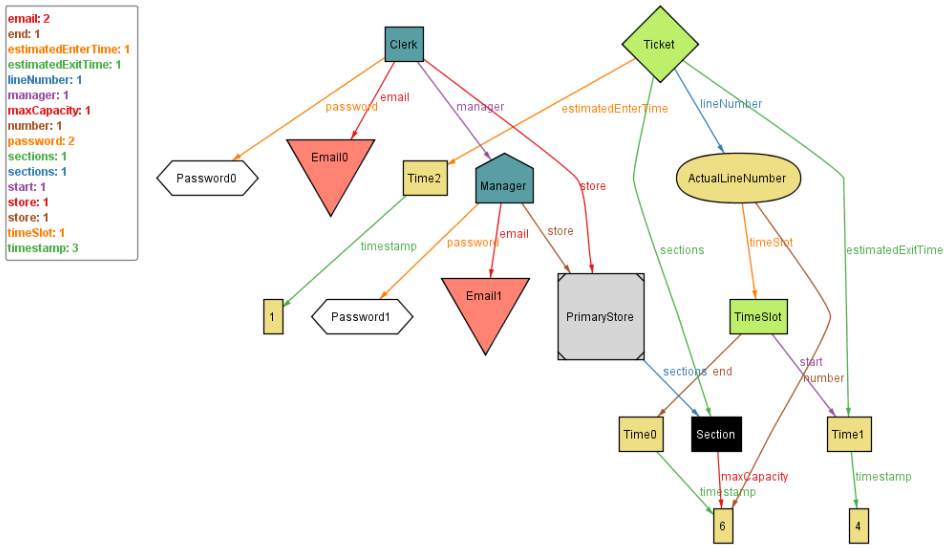

Figure 2: World 2

World 2 describes the model when there are no registered customers, so that there can not be bookings and all the tickets are made as guest going to the clerk in the store.

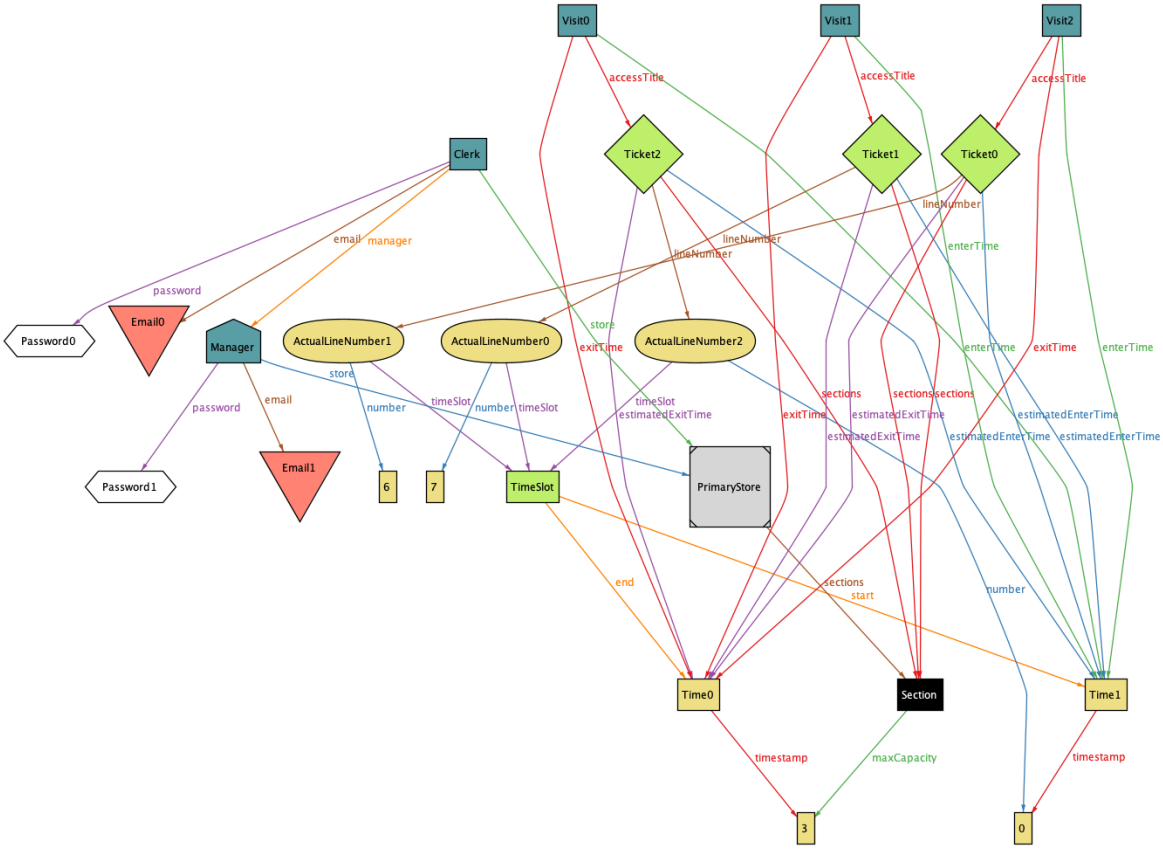

Figure 3: World 3

World 3 describes the model when there is only one section and it has the same number of customers as the maximum capacity.

4.4 Alloy Analyzer results

6 commands were executed.
The results are:
#1: No counterexample found. noMoreVisitsThanAccessTitle may be valid.
#2: No counterexample found. atLeastACustomerForHavingABooking may be valid.
#3: No counterexample found. neverMoreCustomerThanMaxCapacity may be valid.
#4: Instance found.world1 is consistent.
#5: Instance found.world2 is consistent.
#6: Instance found.world3 is consistent.

5 Effort Spent

Date:	Person:	Part:	Time (in hours):	Description:
18/10/2020	Ozan Incesulu	General Structure	0.75	Imported and built the general document structure, switched LF -> CRLF for Windows, replaced template parts with names, year and project title
18/10/2020	Ozan Incesulu	Introduction	1.5	Start writing introduction by adding comments for draft goals, adding further subsections, writing basic definition of the system, some abbreviations, definitions, acronyms and references
24/10/2020	Ozan Incesulu	Document Structure	1.5	Write the document structure of the introduction, provide some comments and assumptions regarding how other parts of the document shall be structured.
26/10/2020	Ozan Incesulu	Scope	1.25	Write the Scope section, by defining different users of the system and define different phenomena with the categories they belong. Also add additional definitions.
26/10/2020	Aydin Javadov	Scope	1	Extend the goal test to provide details about system function
29/10/2020	Ozan Incesulu	Introduction	0.25	Add the clerk role to the system instead of using door automation
31/10/2020	Ozan Incesulu	Domain Assumptions	1	Write domain assumptions for the system, considering different users and phenomena
31/10/2020	Ozan Incesulu	Requirements	1.5	Write system requirements for the system based on agreed goals of the system.
06/11/2020	Ozan Incesulu	Goals	1	Write system goals, merge changes from previous team member, some extra housekeeping tasks.

