Bachelor Thesis

Bachelor Industrial Engineering and Management

Developing a Digital Serious Game for Healthcare Logistics: Appointment Scheduling of Elective Surgical Patients in the Operating Room

Author

Noor Mansour

Supervisors

Prof. Dr. ir. E.W. Hans (University of Twente)

Ir. Rob Vromans (Rhythm B.V.)

Dr. ir. A.G. Leeftink (University of Twente)

Management Summary

Motivation & Objective

The increasing demand on the healthcare system to deliver high-quality care to more people with less available resources forces decision-makers to think about ways to improve the efficiency of their operations. CHOIR (Center for Healthcare Operations Improvement and Research) is a research center of the University of Twente that aims to help healthcare practitioners understand and deal with complex operations management challenges. Although solutions to these complex challenges exist, decision-makers in healthcare hesitate to implement them in practice. One reason for this hesitation is the lack of knowledge healthcare practitioners and decision-makers have in the field of operations management. To close this knowledge gap, the design and development of a serious game are proposed. This research deals with the gap in knowledge about the tactical capacity allocation of elective surgical patients to the operating room. More specifically, it aims to create a serious game that teaches healthcare practitioners about the effects of different scheduling policies of elective surgical patients, the effects of fully static schedules in a variable environment, and the impact of the operating room schedule on downstream departments. The objective is not only to create a serious game but also to evaluate and discuss insights gained during the design and development process.

Approach

First, literature research forms the basis for the needed knowledge about serious gaming and the scheduling of elective patients in the operating room. The literature research aims to answer how educators use serious games for operations management topics especially for healthcare logistics and how an educator could construct such a serious game. Additionally, it explores how the scheduling of elective patients at the tactical level in surgical services work. Second, based on the literature research, we construct a conceptual design of the serious game by following a conceptual modeling framework for simulation-based serious games. Third, we transform the conceptual design into an interactive web-based application, written in the general-purpose programming language R with help of its libraries R Shiny and R Simmer. Last, we create a game script for the deployment of the serious game in professional education and publish the serious game online with an openly accessible source code.

Results

One part of the results of this research is the programming language-independent conceptual design and the implementation of the two-player serious game that is openly accessible to anyone with an internet connection[1]. The serious game can be used to teach the effect of different appointment scheduling strategies to students and healthcare professionals in an easily accessible way without the need for any prior knowledge in operations management and/or mathematics. But this research also provides insights into the development process of a serious game. The development environment, which consists of the programming language R, the web application library R Shiny and the library for discrete event simulation R Simmer, is suitable for developing interactive dashboards. Although it lacks an easy implementation for more advanced gamification techniques such as guided game turns or a role-based game structure. Also, while the R simmer package provides a quick implementation of simple simulations, it lacks functionality for the implementation of more complex simulation models. The literature research shows that serious games about operations management in healthcare settings are rarer than serious games in manufacturing settings. Additionally, there is a lack of openly and easily accessible serious games for operations management. The conceptual modeling framework for simulation-based serious games proved to be useful in creating a simulation model fitting for the

[1] Link to game: https://noormansour.shinyapps.io/Appointment_Scheduling_Simulation_Game/

purpose of teaching. However, it failed to integrate the application of gamification techniques that would make the serious game more fun and entertaining to play.

Conclusion and Outlook

The development of the serious game highlights that the use of serious gaming is an interesting approach to teach operations management topics in an easy-to-understand manner. It also shows that a simple serious game can be developed without a large investment in human and monetary resources as this game was developed by one student developer with limited prior experience in programming. We propose that a project involving multiple and more experienced people with a bigger time frame could yield an engaging serious game that is beneficial for bridging the gap in knowledge for students and healthcare professionals. As the serious game and its source code are publicly available and the serious game was developed with extensibility in mind, continuous development of extensions is possible. Extensions related to the gamification techniques could include guided in-game turns, dynamically changing graphs/visualizations, and multiple players with different interacting roles. Also, extensions to the scope of the topics in the game can be made by adding more details to the simulation model like changeover times. In addition, including more components to the model such as a preoperative screening would increase the possible decisions. Further research into the topic of serious gaming in operations management for healthcare is encouraged. Moreover, we advise for the development of a serious game with a bigger scope in terms of resources available, topics addressed and gamification techniques used. Also gathering empirical evidence on the effectiveness of the serious game in teaching the learning objectives can be a step for future research.

Table of Contents

1 Chapter 1 – Introduction

This chapter introduces the reader to the context and problem description of this thesis. Section 1.1 describes the research motivation and goal. Section 1.2 identifies the core problem through a problem cluster and specifies the exact core problem to be solved in more detail. Section 1.3 presents the research questions and links these research questions to the following chapters of the thesis.

1.1 Background & Research Motivation

1.1.1 Context

The importance of the healthcare system to society is recently highlighted by the impacts of the coronavirus pandemic. The healthcare system is not only under stress in the short term due to the pandemic but also in the long term. The aging society leads to an increased demand for healthcare while it also decreases the possible supply by decreasing the size of the workforce (Britnell, 2019). Therefore, the efficient usage of resources is essential to providing high-quality care in the long term. Hospitals face many logistical challenges hindering them to increase their efficiency and performance.

CHOIR (Center for Healthcare Operations Improvement and Research) is a research group at the University of Twente that deals with Operations Management in healthcare and aims to solve the logistical challenges hospitals face. The CHOIR spinoff Rhythm implements these gained theoretical insights into practice, aiming to improve the logistical performance of hospitals. Surprisingly, a large portion of Rhythm's time is not spent on implementing the proven solutions to these problems but on so-called 'change management'. Change management includes convincing and explaining to decision-makers the underlying operations management principles that govern and influence the hospital as a system. Both Rhythm and CHOIR experience that healthcare professionals such as healthcare managers, administrators, and clinicians have little education in this domain. This lack of knowledge negatively affects the decision-making in hospitals, resulting in a lower performance like longer waiting times for patients.

1.1.2 Research Goal

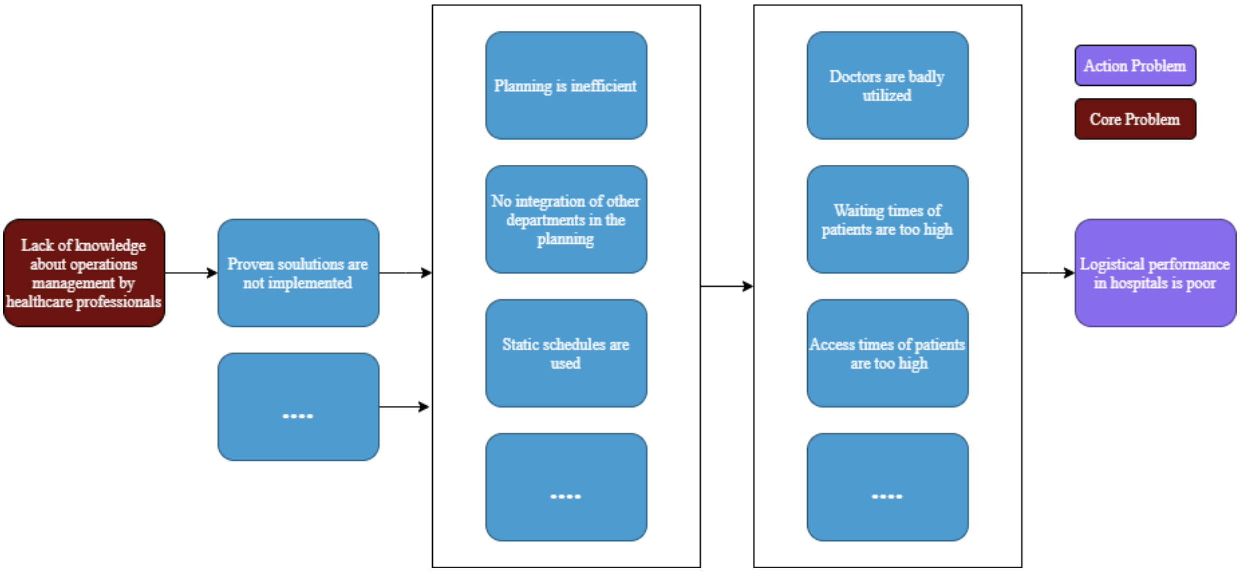
The goal of this thesis is to design a serious game that helps in filling this gap in knowledge of healthcare professionals and students about the operations management view of a hospital and important behaviors of the system. We will take a special look at the appointment scheduling of the operating room (OR), the hospital's largest cost and revenue center (Denton et al., 2007). More specifically we want to teach healthcare professionals and students the effect that different Master Surgical Schedules (MSS) with different scheduling policies for elective surgical patients have on, not only the performance of the OR but also on the subsequent Wards. Master surgical schedules are cyclic schedules that define on which day a surgical specialty can operate in which OR, see section 3.2.1 for more details. By designing a serious game we aim to facilitate teachers and consultants to fill this gap in operations management knowledge such that decision-makers will have a better understanding of how hospitals work in the view of an Industrial Engineer and which interventions might lead to better performance. An increased understanding will hopefully lead to a more informed decision-making process and in turn better overall performance for the hospital. Additionally, this research aims to provide helpful insights into the process of creating such a serious game for university lecturers and healthcare consultants.

1.2 Problem Identification

1.2.1 Problem Cluster

Figure 1 depicts the problem cluster that maps the main problems and relates them in a causal chain. The main action problem that we face is the poor logistical performance hospitals experience. This poor performance can be caused by many problems like the bad utilization of doctors or high patient waiting/access times. The shown causes for the action problem are by no means exhaustive and can be extended. But for the purpose of this research, only a small selection of problems is highlighted. All of these problems are also caused by many other problems like inefficient planning, missing integration of departments, and static schedules. The missing implementation of already existing and proven solutions can lead to these bad practices. The most important core problem that causes the missing implementation of these solutions was agreed upon to be the lack of knowledge about operations management by healthcare professionals.

Figure 1-1: Problem Cluster



1.2.2 Identification of Core Problem

The core problem is defined as the lack of knowledge of healthcare professionals about operations management. This problem satisfies the necessary criteria to be classified as a core problem (Heerkens & van Winden, 2017). Not only is it relevant to all stakeholders but it is also directly influenceable and even simple solutions can have a large impact on solving the core problem. The core problem itself is only a cause and not a direct consequence in our problem cluster and is agreed upon by all stakeholders involved to be the core problem to solve.

1.2.3 Learning Objectives

Several learning objectives are introduced to specify which knowledge should be conveyed precisely. For more structure the learning goals are grouped into general learning goals, learning goals related to the effect of the MSS on downstream resources, and learning goals related to the different scheduling policies. The measurement of norm and reality is defined as follows:

- REALITY:
  - Healthcare professionals do not have knowledge about … *LEARNING GOAL.*
- NORM:
  - Healthcare professionals do have knowledge about … *LEARNING GOAL.*

The *LEARNING GOALS* are defined and grouped in the following way:

General perspective on the hospital

- … the perspective of an Industrial Engineer on the elective patient care path as a connected supply chain consisting of different departments such as the operating room and the ward.

- … the negative impact of variability in arrival rates of the patients and service rates of the surgeons on the overall performance of the OR and the wards.

Open vs Block vs Modified Block Scheduling

- … the differences between open, block, and modified block schedules.
- … the positive impact of adopting a modified block schedule instead of a static block schedule on utilization and access times due to its more flexible capacity allocation to actual current demand.

Impact of the MSS Schedule on downstream resources (especially the wards)

- … the fact that the MSS has an impact on departments beyond the operating room such as the wards in which patients after surgery need to recover.
- … the fact that the MSS can be changed to level the downstream resource usage (bed occupancy in wards).

The variable knowledge in this case is a binary variable. Either the learning goal is attained, or it is not. A more granular specification of attained knowledge could be considered in the future, but the specific and simple learning goals make it possible to use the binary definition.

1.3 Research Questions

To achieve our research goal, it is necessary to answer the following research questions. The research questions are closely linked to the game design research methodology (Greenblat, 1988) and the conceptual design methodology for simulation-based serious games (van der Zee et al., 2012), see section 2.4 for a detailed discussion. Additionally, the research questions provide the outline for the thesis and are defined as follows:

1. **How can serious games be used to convey learning goals related to operations management? (Ch.2)**
   1.1. What are serious games and what kind of serious games exist? (Ch. 2.1)
   1.2. What serious games exist for teaching operations research principles especially related to logistics in healthcare/appointment scheduling? (Ch. 2.2)
   1.3. How are game techniques used in serious games to convey learning goals? (Ch. 2.3)
   1.4. What are possible methodologies one could follow to design a serious game that uses a simulation to model a system? (Ch. 2.4)
2. **How does appointment scheduling for the OR work in the case of elective patients and what impact does it have on the performance of the OR? (Ch. 3)**
   2.1. How is appointment scheduling of elective patients at the tactical level done in hospitals and what decisions must be made? (Ch. 3.1)
   2.2. What is a Master Surgical Schedule? (Ch. 3.2)
   2.3. What impact does the Master Surgical Schedule have on downstream departments such as the Ward? (Ch. 3.2.1)
   2.4. What are open, block, and modified block schedule policies and what kind of impact do they have on the operating room's performance? (Ch. 3.3)
3. **What is a possible conceptual design for a serious game for appointment scheduling? (Ch. 4)**
   3.1. What is the learning environment? (Ch. 4.2)
   3.2. Which concrete objectives does the game have? (Ch. 4.3)
   3.3. What outputs should the game create to convey the learning goals? (Ch. 4.4)
   3.4. What inputs can be used by the operator of the game to set up the model? (Ch. 4.5)
   3.5. How large is the model scope and what decisions can the player make? (Ch. 4.6)
   3.6. How can gamification techniques be used to enhance the learning experience? (Ch. 4.7)
   3.7. How can we assess if the conceptual design is appropriate? (Ch. 4.8)

4. **How can the game be developed? (Ch. 5)**
   4.1. Which tools should be used for the development of the tool? (Ch. 5.1)
   4.2. How should the game be implemented in R and R shiny? (Ch. 5.2-5.4)
   4.3. How can we verify an implemented simulation model? (Ch. 5.5)
   4.4. How can we validate an implemented simulation model? (Ch. 5.6)
5. **How can the game be used in practice? (Ch. 6)**
   5.1. In which cases can the game be used? (Ch. 6.1)
   5.2. How can we validate if the game conveys the specified learning goals? (Ch. 6.3)
6. **What insights have been obtained and what are possibilities for future research? (Ch. 7)**

2 Chapter 2 – Serious Gaming

This section aims to answer the research questions about serious gaming through literature research. Section 2.1 aims to give a definition of serious gaming and an overview of the field of serious gaming by using a taxonomy and providing an initial classification of our serious game. Section 2.2 aims to find inspiration in already existing serious games by exploring two serious games related to appointment scheduling in operating rooms and general operations research concepts respectively. Section 2.3 discusses relevant gamification techniques that help to convey learning goals and how they might be implemented into the game. Section 2.4 ends with a discussion on the chosen methodology for creating the conceptual design of the serious game.

2.1 Positioning within Serious Gaming Literature

2.1.1 Definition of Digital Serious Gaming

Serious games distinguish themselves from other games by their main purpose. Djaouti et al. (2011) define digital serious games broadly as: "any piece of software that merges a non-entertaining purpose (serious) with a video game structure (game)" (p. 2). Many definitions stress that even though serious games aim to deliver serious content, they should still entertain the user with game mechanics that ideally facilitate the acquisition of learning objectives (de Lope & Medina-Medina, 2016; Zyda, 2005).

2.1.2 A Serious Gaming Taxonomy

To create an overview of the big field of serious gaming and understand their application area and methods, a taxonomy is chosen to provide the required information. Although the field of serious gaming in academia and industry is continuously growing (Laamarti et al., 2014) an accepted standard taxonomy is still missing (de Lope & Medina-Medina, 2016). Not all taxonomies aim to categorize any serious game but might focus on specific application domains or specific purposes.

Laamarti et al. (2014) provide a general taxonomy, shown in table 1 that classifies serious games based on the following five criteria: Activity, Modality, Interaction style, Environment, and Application Area. The first criterion is about the type of activity performed by the player like physiological, physical or mental. The modality criterion defines how information is perceived by the player for example through visual, auditory, or haptic modality. Also, the interaction style with the game is a criterion and encompasses keyboard/mouse but also more sophisticated means of interaction like brain interfaces. The environment is a multi-criterion value that consists of several dimensions like 2D or 3D, online or offline, virtual or mixed reality, mobility and location awareness.

Table 1 Taxonomy of Serious Games

Application Area	**Activity**	**Modality**	**Interaction Style**	**Environment**
Education	Physical exertion	Visual	Keyboard/mouse	Social presence

Well-being	Physiological	Auditory	Movement tracking	Mixed reality
Training	Mental	Haptic	Tangible interfaces	Virtual environment
Advertisement		Smell	Brain interface	2D/3D
Interpersonal communication		Others	Eye gaze	Location awareness
Health care			Joystick	Mobility
Others			Others	Online

Furthermore, Riedel & Hauge (2011) provide a classification framework that is more specific for serious games that simulate real-life environments. Serious games are classified based on the simulation level and the skills mediated. The simulation level describes to which extent the real world is simulated in the game. The hierarchy starts with the Universe/World/Civilization level and goes to industry, inter-organizational, business, intra-organizational, team, or techniques level. Logistical skills, risk management, knowledge management, or product manufacturing are some examples of overarching terms for skills that can be mediated through serious games that simulate real-life environments.

2.1.3 Classification of our Serious Game

Applying the taxonomy to position our initial idea of the game in the literature reveals that our application area is mainly education but also health care, the activity type of the player will be mental, the game will communicate via the visual modality and interactions with the game will be done with the keyboard/mouse. The tool will be a 2D game that is accessible on the internet and playable solo or with 2 players locally on one machine. There is no location awareness of the player nor is the game engaged in virtual or mixed reality environments. Our tool is aiming to convey logistical skills in appointment scheduling at the technique level. According to Riedel & Hauge's (2011) classification, our game joins the likes of the beer game and the JIT game, which both convey logistical learning goals like the management of a supply chain and just-in-time production respectively. For a more detailed discussion about the design of the game, see chapter 4.

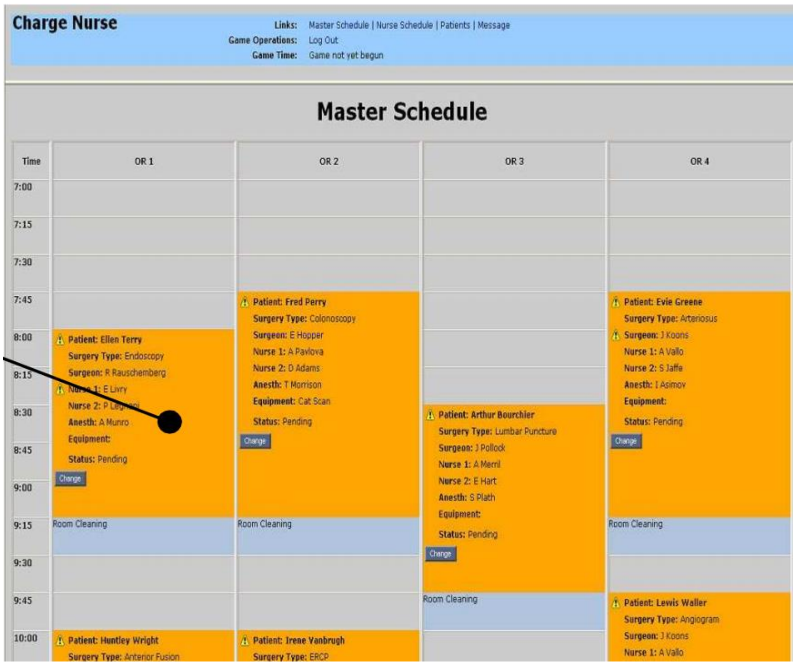
2.2 Serious Games in Operations Management for Healthcare

2.2.1 A Serious Game for the Management of the Master Schedule of an Operating Room

An interesting serious game to look at is the web-based role-playing application of the management of the master schedule of an operating room (Mattarelli et al., 2006). In this game, three players take on the roles of the charge nurse, the anesthesiologist in charge, and a surgeon coordinator that manage the master schedule of the operating room. All roles also have different individual responsibilities that might conflict with each other or the general objective of completing all scheduled surgeries on time and safely. Players receive interrupting notifications of events and problems that have to be dealt with in addition to the other responsibilities.

Although the game's purpose is not to educate but rather to experiment with how these interrupting notifications affect the performance of the players, the conceptual design of the serious game with the incorporation of multiple players and competing objectives is still interesting to consider as a possible game technique. The game provides an engaging and dynamic environment that represents a simplification of a real-life situation. It seems to be a sophisticated game that is implemented with a team of more than 3 professional developers. Figure 2 shows the implementation of the Master Surgical Schedule.

Figure 2-1 Master Schedule of Serious Game



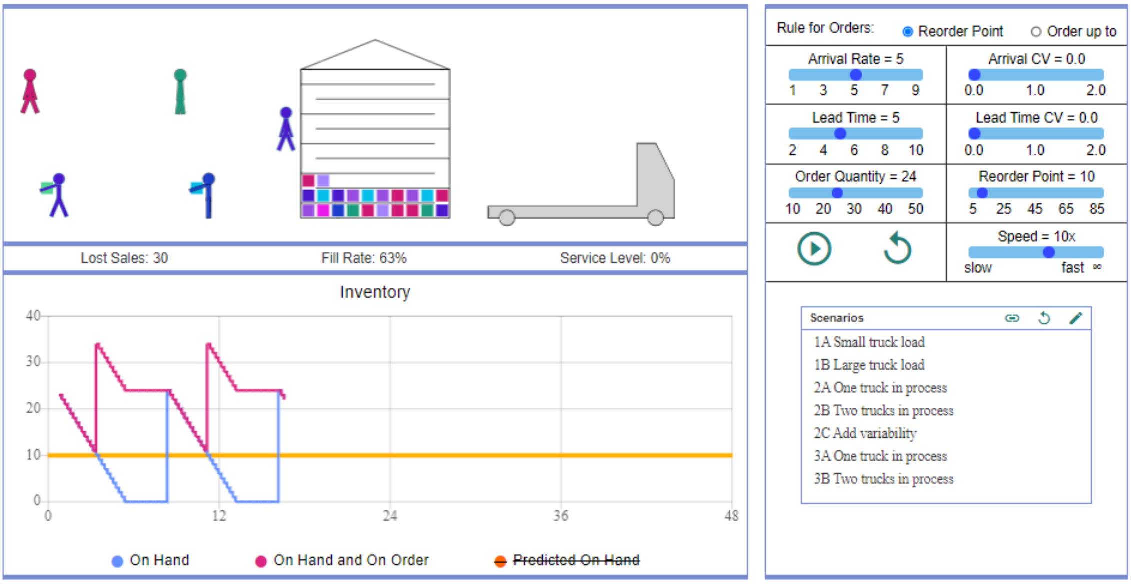
Note: Adapted from *Design of a Role-Playing Game to Study the Trajectories of Health Care Workers in an Operating Room* by (Mattarelli et al., 2006)

2.2.2 Interactive Web-Based Simulation for Operations Research Concepts

Dobson & Shumsky (2006) present a web-based simulation that teaches operations management topics like the economic order quantity or littles law. Although it is not defined as a serious game, the web-based simulation called Tiox (https://tiox.org/stable/) represents the theory behind the economic order quantity by animating the process and showing a graph that is continuously updated and synchronized with the activities of the animation. The user can change input settings like the arrival rate, the order quantity, and the reorder point. Changes in these inputs have an immediate response on the animation and the graph, providing immediate feedback to the user. Additionally, the ability to create scenarios can help in highlighting the impact of different situations.

As a standalone Tiox is missing some guidance since the plethora of inputs/scenarios can be overwhelming for the user and the learning objectives are not directly clear even though the results of the changes in inputs are immediate. In conjunction with a guiding operator, the tool teaches operations research concepts visually and intuitively.

Figure 2-2 Tiox User Interface for the Economic Order Quantity



While the operating room management game provides a more sophisticated system with multiple players with different roles and game phases, the Tiox tool uses simpler techniques like facilitating visualizations and instant responses to player inputs. These gamification techniques enable to convey the learning goals to the player in an entertaining way and it is necessary to take a closer look at what other techniques exist and what effect they can have on the player.

2.3 Gamification Techniques

The literature gives us insight into what game mechanics are used in serious gaming and for which purpose they are implemented. Especially scoring mechanisms seem to be crucial for the successful implementation of a serious game in healthcare (Zhang et al., 2021). Many game mechanics are mapped to their pedagogical aspects by Suttie et al. (2012) but a closer look will be taken at those who seem to be technically doable and useful for our specific tool:

- Game Turns
- Realism
- Progression
- Responsiveness
- Tutorial
- Questions and answers

Game Turns facilitate the player to evaluate. It helps the user to assess the situation, reflect on his actions and possibly discuss issues with other users or an operator/instructor (Uskov & Sekar, 2015). Also, it may increase engagement if the next turn is linked with a prior hypothesis that is made and that the player wants to explore (Proulx et al., 2016). As the aim of our game is to convey certain learning goals, evaluation is important for the user to understand the logistical effects in their context. Game turns could be implemented by dividing them into learning goals. For example, each group of learning goals (as defined in section 1.2.3) is addressed in specific game phases. Given a certain scheduling policy and a certain allocation in the MSS, the first game phase could only incorporate the change from deterministic to variable arrival/service rates and hence show how this would affect the performance of the system. Subsequent game phases could then address the implications of the MSS in relation to the wards etc.

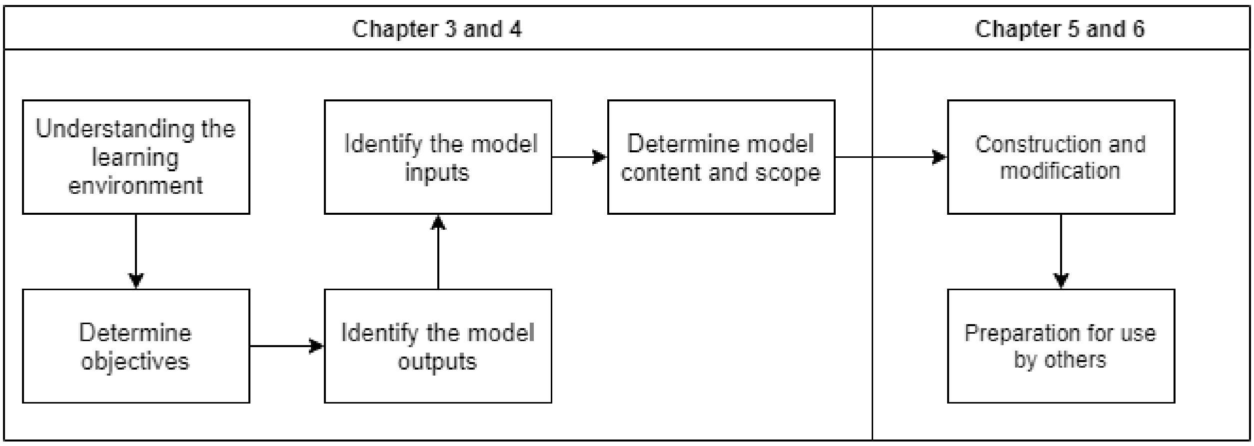
Progression and responsiveness are both essential aspects for a serious game to convey the learning goals (Proulx et al., 2016; Suttie et al., 2012; Zhang et al., 2021). It is considered beneficial for serious games that are based on simulation if every action has an immediate response that promotes the evaluation and experimentation of the player (Jackson et al., 2020; Kulkarni et al., 2019). Progression can be linked with game turns as the player may experience an increase in his achievements as the game phases progress. This might boost motivation to complete the game through the end and explore it.

Tutorials are used to help the user understand the initial situation and the possible interventions he can make. A game should be easy to use and therefore a tutorial at the beginning, showing how the tool works and guiding the user through the first game turns, is desirable. It avoids overwhelming the user with information and possibilities to do and would be adequate for target group consisting of students and healthcare professionals that have no to limited technical knowledge (D'Amours et al., 2017; Jackson et al., 2020; Marín-Vega et al., 2019). Question and answers are also used to deepen the understanding and engage the player in reflection (Proulx et al., 2016). A question after each game turn can make sure that the player attained the learning goals or clear up any misconceptions that were made regarding these learning goals.

2.4 Methodological Approaches to Game Design

A successful serious game project needs to follow a structured and proven methodological approach. By examining the literature on methodologies about serious game design and development, a methodology specifically constructed for the design and creation of simulation-based serious games emerged. To guide the process of designing a serious game specifically based on simulation, van der Zee et al. (2012) modify the modeling framework for simulations by Robinson (2008) and extend the highly cited game design process of Greenblat (1988). We will use this proposed framework for the development of the conceptual model because it fits the simulation modeling approach we aim for while redefining it for pedagogical purposes. The detailed steps and the activities of the conceptual modeling framework are shown in Appendix A. This Framework will serve as the main guide to create the design of the serious game. Also, the remainder of the thesis is structured corresponding to the steps in the framework as shown in Figure 4. Chapter 3 is part of the first step called "understanding the learning environment" which is concerned with understanding the subject matter. Chapter 4 is divided into subsections that are directly related to the steps of the framework, while chapters 5 and 6 correspond to the Construction and modification as well as the preparation for use by others step of the game design process (Greenblat, 1988).

Figure 2-3 Steps of the Methodology related to the Chapters



3 Chapter 3 – Appointment Scheduling in OR

Following the methodology from section 2, the goal of this chapter is to enhance the understanding of the subject matter of the learning goals: tactical appointment scheduling of elective surgical cases for the OR. This is done through literature research.

Section 3.1 defines which exact capacity decisions are to be taught and understood by using a taxonomy. Section 3.2 discusses the concept of a Master Surgical Schedule (MSS) and what factors may hinder successful implementation. Section 3.3 answers the question of what impact the MSS can have on other departments in the hospital especially the wards. Section 3.4 shows the different scheduling policies open, block, and mixed block scheduling and discusses what they imply for the performance of the OR.

3.1 Tactical OR Capacity Allocation of Elective Surgeries

To understand the learning environment of the serious game and define specific learning goals, a thorough understanding of the underlying concept and decisions is necessary. We focus on the managerial area of resource capacity planning. Due to a large number of capacity decisions in healthcare, a taxonomy is chosen to help get an overview of the relevant field and specify our learning environment.

Hulshof et al. (2012) provide a taxonomy for resource capacity planning and control decisions in health care. They classify these decisions based on the service in health care and its hierarchal level. Healthcare services are defined as ambulatory care services, emergency care services, surgical care services, inpatient care services, home care services, and residential care services. The hierarchal levels used are strategic, tactical, offline operational, and online operational. We are especially interested in the OR scheduling of the hospital and therefore our interest lies in the surgical care services and the tactical decision hierarchy.

For the operating room, we can find the planning decision of capacity allocation. OR capacity allocation contains three steps. In the first step, patient groups are defined based on their medical subspecialty, medical urgency, diagnosis, or resource requirements. Secondly OR time is subdivided for the prior defined patient groups. The third step is to allocate specific blocks of OR time to these patient groups or specific surgeons/surgery groups (Guerriero & Guido, 2011; Hulshof et al., 2012). One of the objectives of capacity allocation at this tactical level is to trade off the utilization of the surgical resources and patient access time (Hulshof et al., 2012).

This research focuses on the third step also known as block scheduling. Blocks scheduling is performed at the tactical hierarchy (Guerriero & Guido, 2011). More specifically we are interested in the block scheduling of elective outpatients. The surgeries of elective patients can be planned in advance in contrast to urgent or emergency cases that require surgery with a short waiting time or an immediate procedure. These schedules can have a cyclic nature, i.e. they are repeated periodically and are then termed Master Surgical Schedules (MSS).

3.2 Master Surgical Schedules

Cyclic block schedules are also called Master Surgical Schedules (van Oostrum et al., 2008). MSS define for each day a surgical specialty or surgical procedures to an operating room and a time block. Table 2 provides an adapted example from a case study at a European children's hospital (M'Hallah & Visintin, 2019). This MSS has a planning horizon of 2 weeks indicated by the Day columns and is not open on weekends (Day = 6 & 7). For every day 2 sessions are scheduled in which surgical specialties (e.g. ORL = Otorhinolaryngology, CHPED = Pediatric surgery) are allocated. Specific surgeries are then scheduled by the responsible surgeons on the assigned time slots.

Table 2 Example of a cyclic MSS

Session	**DAY 1**	**DAY 2**	**DAY 3**	**DAY 4**	**DAY 5**
1	CHPED	CHNEO	MAX	CHNEO	CHPED
	ORL	EGDS	OCU	ODONTO	ORTONC
	TRAUMA	ORL	ORL	ORTO	OCU
	URO	URO	URO	URO	URO
2	ORL	CHMAN	CHPED	CHNEO	CHNEO
	ORTO	CHNEO	MAX	OCU	CHPED
	TRAUMA	ODONTO	ORL	ORTO	ORTONC
	URO	EGDS	URO	TRAUMA	URO
	DAY 8	**DAY 9**	**DAY 10**	**DAY 11**	**DAY 12**
1	CHPED	CHNEO	CHPED	CHNEO	CHMAN
	TRAUMA	EGDS	MAX	ORL	CHNEO
	ORL	OCU	ORL	ORTO	CHPED
	URO	URO	URO	URO	URO
2	ODONTO	CHNEO	CHPED	OCU	CHMAN
	ORTO	ODONTO	MAX	ORL	CHNEO
	TRAUMA	EGDS	ORL	ORTO	OCU
	URO	ORL	URO	TRAUMA	URO

Note: Adapted from *A stochastic model for scheduling elective surgeries in a cyclic Master Surgical Schedule*, by (M'Hallah & Visintin, 2019)

Hospitals use the MSS because its cyclic nature provides the ability of early coordination of resources like personnel, minimized efforts compared to creating a new schedule each period, and gives surgeons the possibility to remain in charge for scheduling patients to specific time slots (van Oostrum et al., 2010). Nevertheless, there needs to be a consideration of certain factors that have an impact on the successful implementation of an MSS.

Many planning approaches try to maximize the utilization of the OR. Van Oostrum et al. (2010) argue that it is not enough to only see the utilization of the OR as the main performance measure. Utilization should always be considered in conjunction with robustness against disruptions (like overtime, resource unavailability, etc.) and robustness against cheating (e.g. a surgeon requesting more OR time than he needs).

Additionally, the OR planning should consider its impact on other departments and their planning and resources. Other departments such as the wards clinic are affected by the planning of the OR. Not considering the impact of MSS on the wards can lead to bad utilization and patient cancelations as is discussed in the following section (van Oostrum et al., 2010).

3.2.1 Impact of MSS on Downstream Departments

The hospital managers need to incorporate the related departments into the development of the MSS because the MSS impacts the demand for resources throughout the hospital (Beliën et al., 2006; M'Hallah & Visintin, 2019; Vanberkel et al., 2011). For example, some surgeries require the patient after the surgery to inhibit a bed for recovery or do a preceding blood test (Beliën et al., 2006). Naturally, if many surgeries that need the patient to visit a bed after surgery are scheduled the more beds are needed for this period. Vanberkel et al. (2011) provide a model that describes the workload of downstream departments as a function of the MSS. The ward occupancy distribution, the patient admission/discharge distributions, and the distributions for ongoing interventions can be computed using their model. By applying the model in a hospital in the Netherlands, Vanberkel et al. (2011) show that through the use of their model it is possible to level demand for the downstream departments.

3.3 Open Scheduling vs Block Scheduling vs Modified Block Scheduling

Another interesting decision to be made is about the implementation of a block schedule or a modified block schedule. Compared to normal block schedules, modified block schedules reserve a fraction of capacity that is to be scheduled not at the beginning of the creation of the MSS but at a later date. Scheduling at a later date enables a more flexible adaption of capacity to actual demand because information about demand becomes more reliable as time progresses (Hulshof et al., 2012; Patterson, 1996). For example, a tactical planning meeting can be defined one week in advance to schedule the flexible fraction of OR time to the actual current demand. For this research, we focus purely on the scheduling of elective patients. So, the flexible fraction of the modified block schedule is not considering emergency patients for which oftentimes unscheduled OR capacity is reserved due to their uncertain and urgent nature. One could assume that emergency patients are treated in specific ORs that are not considered in this game.

Kamran et al. (2019) use a stochastic mixed-integer linear programming model to create an MSS schedule that incorporates the modified block scheduling policy and a reserved slack policy, which takes care of emergency patient arrivals. Numerical experiments with real-life data from the general surgery department of Radboud University Medical Center in Nijmegen show that the MSS with the modified block scheduling policy uses resources more efficiently and can plan more surgeries in a week while still being feasible.

4 Chapter 4 – Conceptual Design of the Educational Game

This chapter presents the conceptual design of the serious game that aims to teach the prior defined learning goals by following the methodology of van der Zee et al. (2012). Section 4.1 provides a summary table of the conceptual model to give an overview and first impression. Subsequent sections explain the decisions made for the conceptual model in detail. Section 4.2 is about the definition of the learning environment of the tool. Section 4.3 presents the model objective and explains some characteristics of the game. Section 4.4 identifies and elaborates upon the outputs of the model while section 4.5 addresses the operator inputs used to set up the game. Section 4.6 addresses the scope of the model and defines the components and their relationship. Section 4.7 explains the gamification techniques applied to the game and section 4.6 assesses the model based on criteria defined by the methodology.

4.1 Overview

Table 3 provides a summary table and an initial overview of the conceptual design of the model according to van der Zee et al. (2012). Every relevant decision is mentioned and grouped according to their activities. The summary table can be used as a point of reference and detailed explanations and justifications can be found in the subsequent sections.

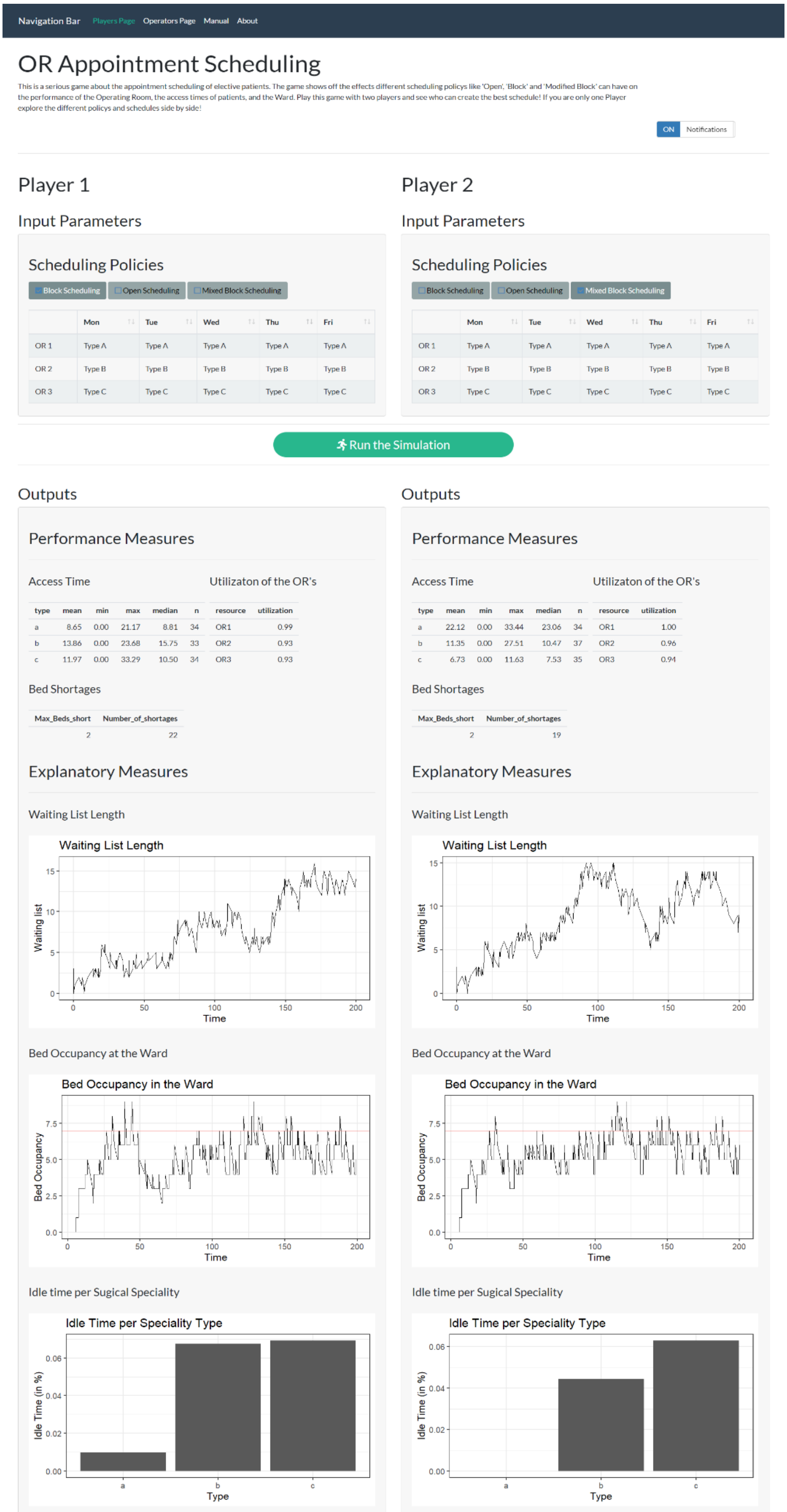
Table 3 Overview of Conceptual Design

Activity	Details
1. Understanding the learning environment	• *Clients*: Educators at University, healthcare logistics consultants • *Subject matter experts*: Professor at University, healthcare logistics consultant • *Subject matter*: Appointment scheduling of elective patients for the operating rooms at the tactical decision level • *Players*: students, healthcare professionals • *Operators*: University Teachers, health care consultants • *Context of use*: Lecture of Operations Management, a workshop for healthcare professionals (Game played to support an explanation given by the operator) • *Appropriateness of a computer-based game format*: confirmed
2. Determine objectives – Modeling objectives	• *Pedagogic purposes*: educate students/healthcare professionals by fostering their awareness and insights on the way appointment scheduling (policies) of surgical cases in the operating room have an impact on not only the performance of the operating room but also on other departments such as the ward. • *Modeling objectives*: Facilitate the players learning on improving access time, staff utilization of the OR, and the bed occupancy at the ward by employing an Operating Room schedule and schedule policy particularly deciding between an open, block, or modified block scheduling system
– General project objectives	• *Project requirements*: Time frame of 10 -15 weeks; limited technical implementation skills • *Model nature*: ○ *Visualization*: patient access time, utilization of OR, bed shortages as (color-coded) numerical KPIs; Waiting list, bed occupancy at the ward, arrival rate of patients as explanatory time-series graphs; Simulation model visualized

<table>
<tr><td></td><td>
in a picture indicating components of the simulation and flow items
Player interaction: simple menus buildup of buttons, input boxes and slider bars (operator), slider bars (players); simple menu buildup of a modifiable schedule (player), input boxes selecting the policy (player), slider bars (operator), buttons (operator)
Responsiveness: Immediate
Model/component re-use: Yes, it is possible by changing operator inputs
</td></tr>
<tr><td>3. Identify the model outputs</td><td>
Performance measures (player achievements): patient access time, OR utilization, ward bed shortages
Explanatory measures: Number of patients operated (throughput), length of the waiting list, bed occupancy, arrival rate, staff overtime, staff idle time
Format: Time series graphs for explanatory measures, numerical KPIs for performance measures
</td></tr>
<tr><td>4. Identify the model inputs</td><td>
Operator Inputs:
External factors
Arrival rates and variability
Service rates and variability
Resource Capacity
Number of surgical specialties to be scheduled
Number of operating rooms available
The bed capacity of the ward
MSS development prerequisites
Number of surgical specialties (Step 1)
Subdivision of operating room time per specialty (Step 2)
</td></tr>
<tr><td>5. Determine model content and scope</td><td>
Scope: Table 4, Figure 7
</td></tr>
</table>

Additionally, Figure 5 shows the user interface of the player page from the game to give an initial first impression. The subsequent sections explain the decisions made and the nature of the game in detail.

Figure 4-1 User Interface of the Game



4.2 Understanding the Learning Environment

The learning environment of a game is defined by the subject matter, interest of the client, educational background, the interest of the players, and the context of use.

The **subject matter of the game** is about hospital logistics in general. More specifically it is about the capacity allocation of the OR schedule at a tactical level and concerns learning goals related to the MSS, its effect on other departments, and the effect of different scheduling policies, see chapter 3. The goal of the game is to convey the learning goals as defined in section 1.4. Because attaining these learning goals can lead to a more informed decision process for hospital staff and a fun and easy achievement of learning goals for students, the clients are interested in the success of this project.

The **educational background** of the healthcare professionals and the students is medical and business/mathematics respectively. While students are in the process of acquiring the education, healthcare professionals are already bound by the systems and the mindset of medical education. The challenge is to define a game that convinces these healthcare professionals to change their routines while maintaining simplicity such that even with a lack of the mathematics behind the concept, the situation can be understood, and the learning goals easily acquired.

The **interest of players** lies in learning about concepts that can improve the flow of patients, access times, and the underutilization of resources in hospitals. For students, it is interesting to play the game since it provides a fun and interactive way to learn how to schedule appointments for the OR.

The **context of use** is a student course about operations research/management or healthcare logistics and workshops with hospital staff, which are both lead by operators experienced in the topic. It should also be usable as a publicly available online game with the use of appropriate guides enabling the player to play the game on his own without an operator.

4.3 Modeling Objectives

The definition of the learning environment gives the game its outline. To properly place the simulation model in the game context, the simulation modeling objectives have to be defined. Simulation modeling objectives highlight the utility of the model for players learning (van der Zee et al., 2012). The modeling objectives can be described as the player's achievement attained by mastering his decision-making skills. They are formulated for our case as follows:

- The simulation model is to facilitate the players learning on improving access time by choosing between the open, block, and mixed block scheduling policies while trading off associated changes in the OR utilization and bed occupancy at the ward.
- The simulation model is to facilitate the players learning on improving the access time and OR utilization by adapting a mixed block scheduling approach compared to a static block scheduling approach.
- The simulation model is to facilitate the players learning on decreasing bed shortages by taking into account the dependency of the ward from the MSS when allocating surgical specialties on the MSS.

4.3.1 Model Nature

Because the learning environment is the education of students and healthcare professionals, a simpler fictitious model, which does not represent a specific real system but is still plausible, is better suited than a more real and complex model that might be necessary for training healthcare professionals (Klabbers, 2003). Therefore, the model scope and detail are limited to isolate the learning goals and create a clear link between the decisions of the player and its effects on the system.

The goal for a simple but clear and easy-to-use game and the scope of the project necessitate the game environment to be a 2D game with visualizations of graphs showing the performance and explanatory measures of the underlying system. Section 4.4 goes into more detail about the outputs and how the game presents them. The User Interface needs to be intuitive and simple without presenting many objects to the player or mathematical calculations, see Figure 5 in section 4.1 for a mockup of the User Interface. The user interacts with the model via slider inputs, checkboxes, number input boxes, and an interactive schedule. For every input/decision opportunity a little Information guide will tell the user what this input/decision is about.

The model has to be responsive to highlight the impact of the changes done by the user. The longer the model takes, the less attention will be paid to the connection between the impacts and the changes of the user. Due to the scope and learning outcomes of the game, the model should only take a short waiting time to be done with the calculations.

With the goal in mind to promote further research and possibly to build an even more complex game, the model should be designed for extendibility. This implies the ability of the model to increase its scope and incorporate more learning goals related to capacity decisions in healthcare. For that, a general and programming language independent conceptual model is provided. Additionally, the programming language used to develop the game needs to have much flexibility and be extendable in its functionality, see section 5.1.

4.4 Identifying Model Outputs

4.4.1 Relevant Performance Measures

The relevant performance measures can be extracted from the modeling objectives that relate to the pedagogical purpose (van der Zee et al., 2012). For that reason, the performance measures are:

- the patient access time,
- the utilization of the OR and
- the bed shortage in the wards.

Patient access time is defined as the time from the moment that the patient requests an appointment until the planned surgery day. This measure is chosen because almost all of the learning goals relate to the improvement of patient access times as it is highly influenced by the MSS. The same reasoning can be applied to the utilization of the OR as it is directly influenced by changes in the MSS. As one of the learning goals for the player is to realize that the MSS does not only influence the OR but many departments beyond that, also the bed shortages of the ward are included, see section 3.2.1. Additionally, these performance measures are incorporated to introduce a trade-off between access time and utilization of resources that is prominent in capacity allocation decisions (Hulshof et al., 2012).

4.4.2 Explanatory Measures

The performance measures are accompanied by explanatory measures that give the player insight into why and how the performance measures changed. The following explanatory measures are defined:

- Length of the waiting list
- Idle time of doctors per surgical specialty
- Ward bed occupancy per surgical specialty

To explain the patient access time the length of the waiting list is used. To explain the utilization of the OR, the idle time of doctors per specialty is used. And to explain the bed shortages, the total bed occupancy as well as the bed occupancy per speciality type is used. For example, if the access time of one player is higher than the other, the length of the waiting list can indicate why that is the case. The player could also recognize that a certain specialty has a very high idle time due to the way they are scheduled on the MSS. Additionally, the bed occupancy per surgical specialty provides insights into how the MSS might impact the distribution of patient types arriving at the ward.

4.4.3 Format of Data

The format of the data is of importance especially related to the fact that the effect of decisions should ideally be instantly recognizable. To ensure this instant recognition of changes and the opportunity to compare values more easily, the main performance measures are represented as numerical values. The patient access time will include the mean, maximum, minimum, and median. The utilization of the operating room will be represented as a percentage. The bed shortages will be represented by the number of times the maximum capacity of the ward was exceeded and how high the highest need for beds was above the capacity. These numerical values might be color-coded based on a specified threshold or based on the results of the second player of the game, see section 4.7 for the split-screen idea.

The explanatory measures are expressed graphically. The length of the waiting list and ward occupancy will be shown as a time-series graph, while the idle time of doctors per surgical specialty is represented with a bar graph. The explanatory measures are shown as time-series graphs to promote the recognition of patterns in the data (e.g. longer waiting list on Mondays) and to more easily be able to compare and highlight differences between different schedule strategies.

4.5 Identifying Model Inputs

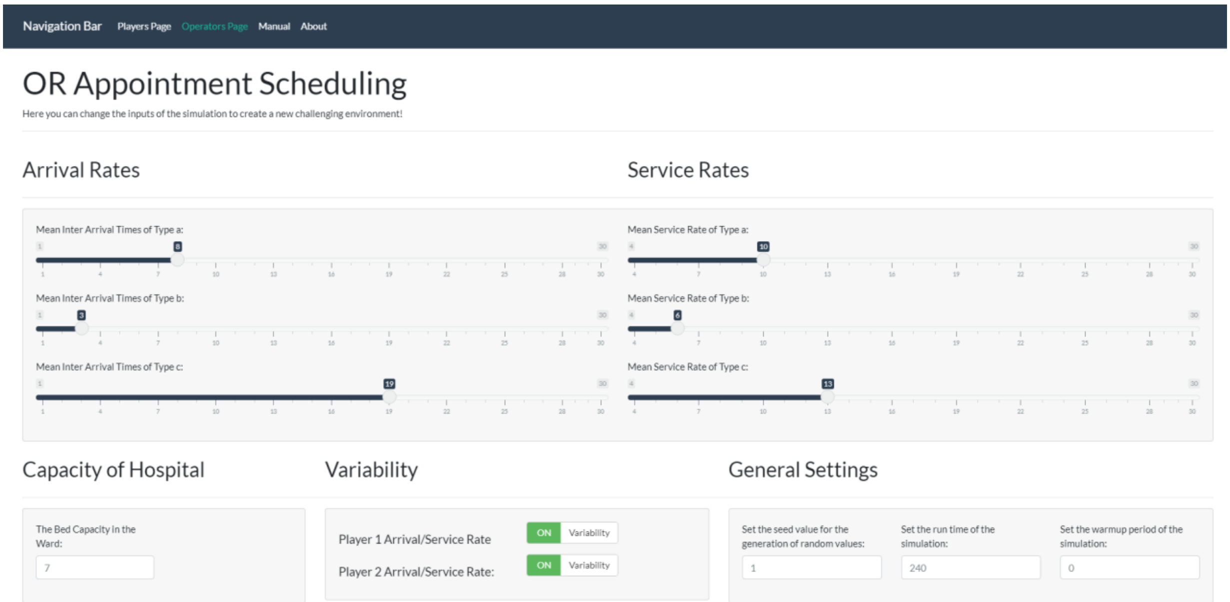
The operator uses model inputs to change the initial configuration of the system. These inputs are also called operator inputs. The operator inputs are different from the game inputs that the player decides on, which are elaborated upon in section 4.6 (defined as Jobs). With operator inputs, the operator can change the status of the system to see how the player behaves to different configurations like increased variability in arrivals. It hence acts like the experimental factors a researcher would change to observe changes in the dependent variables. The operator is also able to introduce different game phases and situations to promote the consecutive acquisition of learning goals, see section 6.1 for more detail. The operator inputs were chosen with the following question in mind: With which inputs can the operator challenge the player such that the learning goals are highlighted? The following inputs were identified:

- Mean arrival rate and coefficient of variation, per specialty
- Mean service rate and coefficient of variation, per specialty
- The maximum bed capacity of the ward
- Run time of the simulation
- Warmup period
- Random number seed

To teach the impact of variability on the performance of the system, the operator needs the possibility to change the arrival/service rate from deterministic to variable. This is achieved by giving the operator the freedom to change the mean and the coefficient of variation of these rates. To increase complexity and highlight the effects of the capacity allocation decisions, the operator's ability to change the capacity of the wards can be useful. For example, the operator might challenge the player to create a working schedule when the bed capacity of the wards is lower than the round before, pressuring him into making an informed decision that levels the workload on the ward. Moreover, the run time of the simulation can give the operator the possibility to extend a game turn to show off how a schedule might perform better in the longer term than in the short term. Because the goal of the simulation is to teach and compare different schedules with each other, the initialization bias does not play a major role. Since the simulation is not based on a specific real system, one could argue that the hospital is empty at the beginning and hence justify the initialization bias. Nevertheless, a warmup period is added to give the operator the possibility to eliminate the initialization bias at the cost of longer waiting times for the game to run the simulation. In addition, the random number seed provides the operator with the possibility to create different environments with different arrival and service patterns. By changing the random number seed, the operator can highlight the impact of variability on the system and challenge the player to react to this uncertain environment.

Figure 6 provides a look into the user interface of the operator's page in the game. Splitting the game into a player and an operator page is important to avoid overwhelming the user with input options. In this way, the player has a clear and simple UI while the experienced operator can change the many input settings on a different page without confusing the player.

Figure 4-2 The User Interface of the Operators Page of the Game



4.6 Determining Model Scope

Van der Zee et al. (2012) suggest an alternative format to determine the model scope that, different from usual formats, incorporates the player's interaction with the model. The component types that make up a model are agents, flow items, and jobs. Agents are defined as the nonmovable, intelligent infrastructure of the operations system. Flow items are defined as movable items and are distinguished into 4 subtypes: goods (e.g. material parts), resources (e.g. workers, tools, vehicles), data (e.g. monitoring data), and job definitions. Job definitions are the messages that control the movement of goods, resources, and data. For example, pricing decisions would be a job definition since they contain messages that influence the movement of the goods (a price change influences how many products are sold). Jobs are activities that link flow items with agents like the activity of selling a product that would link the flow item product with the agent shop. The model scope can be specified by using this definition that facilitates the incorporation of the player of the game via the agents' components. Additionally, the framework defines the main inputs of the player in the Job definitions category. The main inputs of the player are selecting the scheduling policy and allocate surgical specialties to operating rooms and days.

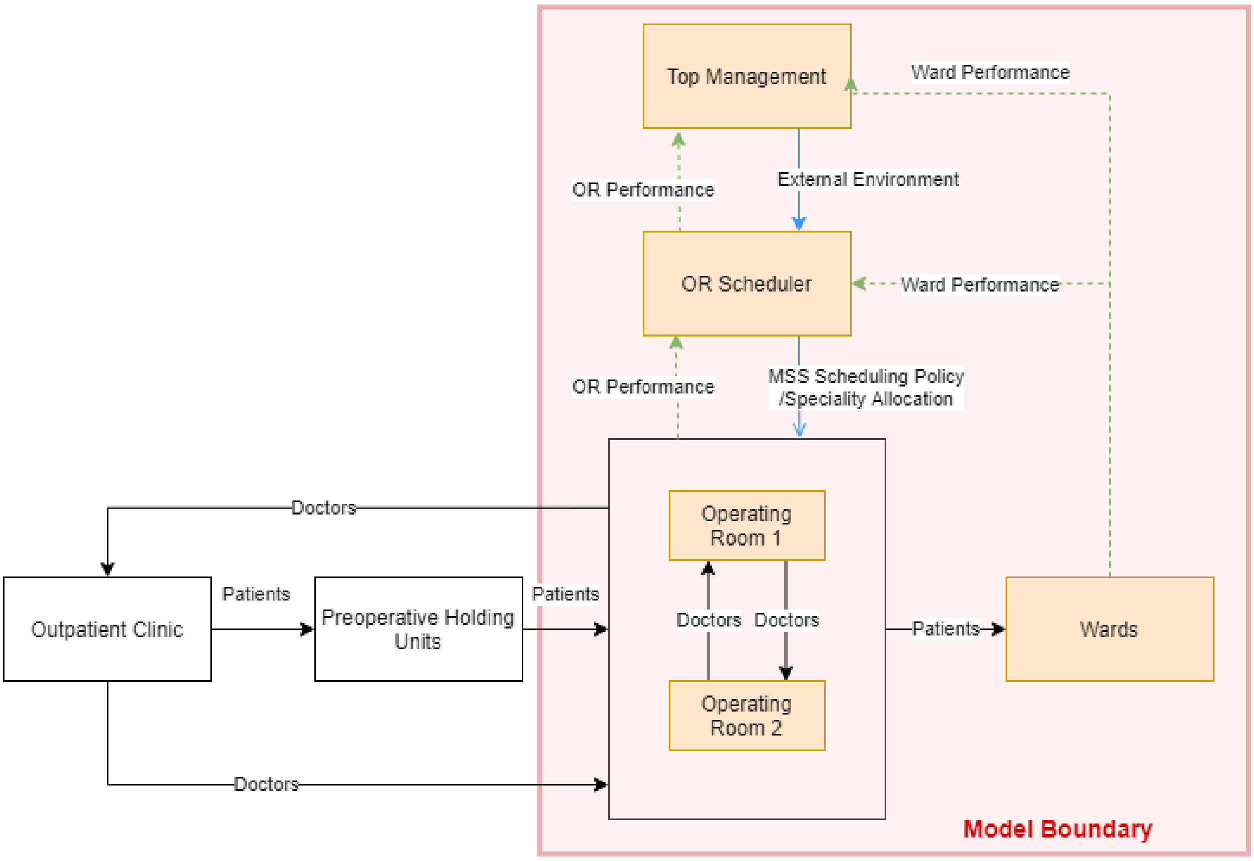
The model scope is represented in Table 4. It explains which components are included in the model and why, while Figure 7 shows how the different model components relate to each other and highlights the system boundary. Also, assumptions and simplifications are highlighted in Table 4.

Table 4 Model Scope

Component	In/exclude	Justification
Agents		
OR Management	Include	Game operator's role; assumption: also controls natural variability factors like arrival variability/can make changes to capacity dimensions
OR Scheduler	Included	Player's role; key influence on system performance
Operating Room	Included	Operations system under study; key influence on system performance;

		Assumption: No changeover time is needed between surgeries. Operating Room is open and available all the time during the simulation if no other patient is in it.
Preoperative holding units	Excluded	Assumption: all preoperative tasks are done beforehand because they do not add direct value to the learning goals
Intensive Care units	Excluded	Simplification: To ensure a simple model, only the ward will yield as an example for a postoperative unit
Ward	Included	Essential to show the impact of the MSS on other units and the need for leveling resources.
Outpatient clinic	Excluded	Assumption: Outpatient clinic is not valuable for the defined learning goals as it would add more complexity and other learning goals
Flow items		
Goods		
Elective Patients	Included	Represents the goods that flow through the system; key influence on system performance
Emergency patients	Excluded	Assumption: Out of scope for the learning goals and defined learning environment; would add another layer of complexity; possible extension
Doctor	Included	Represents the main resource of the system that can process the surgical patients and is moved/used according to the schedule provided by the player
Data		
OR performance	Included	Feedback for player and game operator; see model outputs
Ward Performance	Included	Feedback for player and game operator; see model outputs
MSS Schedule	Included	Assumption: Every day only one specialty can be scheduled to ensure simplicity: Weekly schedule excluding the weekends
Job definitions		
External environment	Included	Every factor that can be controlled by the game operator but are not changeable in reality like arrival rates and variability; see Model inputs
MSS scheduling policy decision	Included	A key influence on system performance; determines how the MSS schedule can be built (Open, block, modified block schedule)
MSS specialty allocation decision	Included	A key influence on system performance; determines how patient types are allocated to the operating room weekly.
Jobs		
Select the scheduling policy	Included	Player's main activity; key influence on system performance; impacts how the player can allocate surgical specialties to operating rooms and days
Allocate surgical specialties to operating rooms and days	Included	Player’s main activity; key influence on system performance; defines in which order patient types are operated and at which time.
Operate patients	Included	Doctor’s activity that processes patients in OR
Consult patients	Excluded	Doctor’s activity at the OC and time the patient is scheduled for OR
Patients recover	Included	Relates to the recovery process of patients at the ward

Figure 4-3 Relation of Model Components and Scope



*Note***.** Black solid lines represent flows of goods, green dotted lines represent the flow of data and blue solid lines represent the flow of job definitions.

4.7 Applied Gamification Techniques

To ensure that the learning goals are acquired in a successful and fun way, some gamification techniques as discussed in chapter 2 are applied in the design of the game.

4.7.1 Two Player Split Screen

The decision to make the game a two-player split-screen game is driven by the aim to ensure that different settings and decisions can be easily compared to each other while simultaneously providing a fun competitive element to this game. The split-screen still enables one to play the game with one player, facilitating an easy comparison of different decisions that can be made. If the game is played with two players the competitive aspect provides a fun learning experience that additionally incorporates the goal to achieve better performance measures than the opponent, promoting the exploration of different decisions to come to the optimal schedule and schedule policy. To utilize the instant recognition of the differences in the numerical performance measures, color coding could be used in relation to the performance of the second player. For example, if the first player has a lower average patient access time than the second player, the better patient access time would be highlighted in green while the worse one is in red, see Figure 5. Also, the multiple player aspect can facilitate discussion and reflection among the players and the operator. One example could be that the first player wonders why his access time is lower than the second player even though he implemented an open schedule policy that according to the first player's expectation should yield a lower access time. This could be followed by discussions and the clear-up of misconception and/or the highlighting of the underlying concepts at play.

4.7.2 Tutorial Technique

Although the game is purposely designed to be simple in its learning goals and user interface, the tutorial technique would still prove useful to introduce the user to the game. The Tutorial of the game is implemented via the operator. Through the model inputs, the operator has influence and control over the environment the decisions are made in. The operator can guide the user with verbal explanations on how to play the game, give explanations to what decisions had which impact, and propose/guide the player to different phases that may contain different learning goals. An operator's manual can help the operator on how the game might be played, how the game might be guided, and which decisions can have which impact. Also, a user guide that is provided in the game itself explaining the different policies and decision options, as well as the outputs and how they can be related to each other, will help the understanding and the introduction into the game.

4.8 Model Assessment

To determine whether the designed conceptual model is appropriate, a set of requirements must be considered not only at the end but also in the process of creating such a conceptual model. The framework of van der Zee et al. (2012) identifies four requirements that the model should be assessed on:

- *Validity*: "a perception, on behalf of the modeler, that the conceptual model can be developed into a computer model that is sufficiently accurate for the purpose at hand".
- *Credibility*: "a perception, on behalf of the clients, that the conceptual model can be developed into a computer model that is sufficiently accurate for the purpose at hand".
- *Utility*: "a perception, on behalf of the modeler and the clients, that the conceptual model can be developed into a computer model that is useful as an aid to the users' education, given a specified learning context".
- *Feasibility*: "a perception, on behalf of the modeler and the clients, that the conceptual model can be developed into a computer model with the time, resource and data available". (p.39)

Due to the aimed simple nature of the serious game and its purpose to educate people unfamiliar with the learning environment, the model ought to be of simple nature as well. That is why this model, although very simple, is perceived to provide enough accuracy to show the effects of appointment scheduling policies and allocation on the performance of the operating rooms and wards.

The model promises a high utility for use in the learning context of lectures or professional workshops. Due to its simple nature, not much time has to be spent on playing the game. Nevertheless, it provides the user with an intuitive introduction to some concepts of appointment scheduling of operating rooms.

The model does not require a very detailed or sophisticated simulation model and hence is expected to be a feasible conceptual model to implement. Although the feasibility depends largely on the skills of the developer not only regarding simulation modeling but also programming for the whole software system that wraps the simulation model into a game.

5 Chapter 5 – Development of the Educational Game

This chapter gives recommendations on how to approach the technical implementation of the game. Section 5.1 starts with the selection of the development tool and recommends using R and its shiny environment. Section 5.2 then describes how the simulation model could be implemented with the simulation library Simmer. Section 5.3 presents the libraries that could be used to create the outputs of the game. Section 5.4 elaborates how the reactive environment in shiny works and proposes a way to model the reactivity. Section 5.5 and 5.6 explain how the model could be verified and validated respectively to increase the confidence in the simulation model.

The source code of the implementation is openly accessible on the web via GitHub: https://github.com/NoorMansour1/Surgical-Chain-R-Shiny.

5.1 Selection of the Development Tool

The programming language R is suitable for the development of the conceptual model into a web-based educational tool. R has the advantage of providing an extensive library for visualizing (ggplot2) and the functionality to program a simple discrete event simulation with the R Simmer library. Another important factor that leads to the decision of choosing R is the relatively easy creation of reactive applications with a compelling user interface without prior knowledge of web technologies. R shiny provides the possibilities to create interactive user interfaces that are easily published on the internet, creating a web page without the need to know web technologies like JavaScript or HTML. For these reasons, R was chosen as the most promising candidate that fits the requirements.

Python and its version of shiny called Dash were also considered as development tools. The main argument against Python is that Dash requires more time to learn and is considered to be more complex, requiring more code despite not providing more functionality than R shiny (Dario Radečić, 2020).

Microsoft Excel was also considered as the development tool. Excel's main drawbacks compared to R is the lack of functionality related to publishing a developed tool on a web page/make it accessible through the browser as well as the limited possibilities to develop more sophisticated user interfaces. Also, the lack of functionality that Excel and VBA have compared to R and its many libraries has to be taken into account. Another drawback compared to R and Python is that versioning and collaboration on the code, through for example the use of GitHub, is not provided with Excel and VBA.

5.2 Simulation Model in Simmer

As the model can be defined as a system of queues in which patients flow through the hospital, a discrete event simulation will be used to realize the conceptual model into a computer program. A discrete event simulation is suitable because the system's status changes based on events like the arrival of a patient or the completion of surgery. The time in between these events is not interesting to observe and hence a continuous simulation is not suitable.

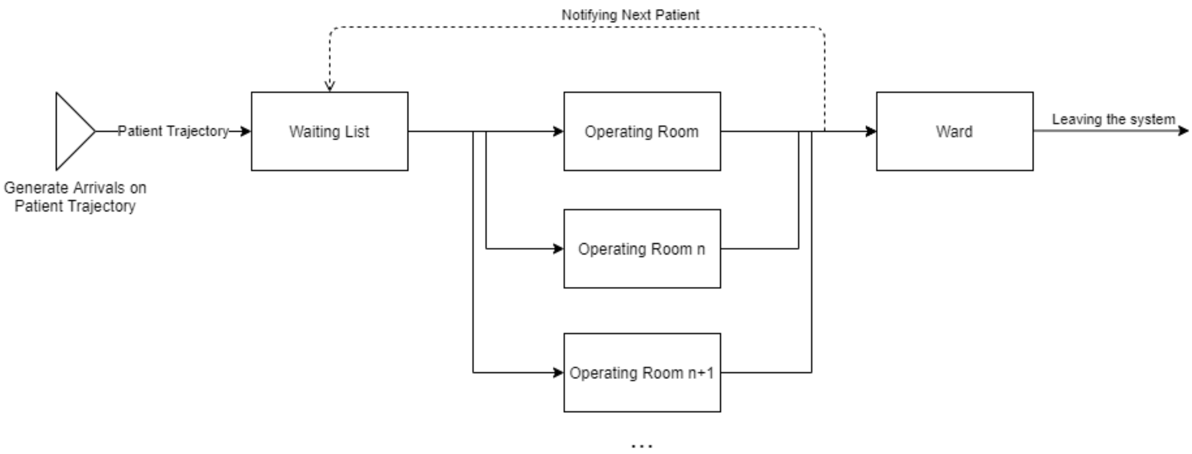
The Simmer package provides a library for creating discrete-event simulations in R (https://r-simmer.org/). Simmer is a generic framework that utilizes the concept of trajectories, which are common paths for simulation entities of the same type. It provides a fast implementation of simple systems and is fast in calculation speeds. Although, it lacks functionality to implement more complex models that for example implement a patient schedule or different priority rules.

In the Simmer environment, three types of resources are defined for the model: Waiting list, Operating Room, and the Ward. Patients can seize the waiting list until they can move to the operating room and then after receiving surgery they move to the ward. When seizing the waiting list, the patient is waiting for a notification that informs him when and to which Operating Room to go to. The patient that is done with the surgery has the responsibility to send this notification and inform the patients waiting on the waiting list who is next for that specific operating room. The selection of the next suitable patient is based on the scheduling policy and the schedule the player provides as input. For example, if operating room 1 is assigned for surgical specialty A on Monday in a block scheduling policy, then the first Patient to go to the operating room will be the first patient of type A that entered the waiting list.

Once he finishes he will notify the next patient according to the scheduling policy. Let's say this patient of type A finishes on the second day Tuesday which is scheduled to be only for patients of type B. Then the patient will notify the patient that is waiting the longest in the waiting list of type B. In some cases, no patients are waiting and then the waiting patient handler is called, who jumps to the next event in the simulation to check if a suitable next patient can be found and he repeats that until someone is found. In the open scheduling policy, the longest waiting patient independent of their type is chosen. The mixed block scheduling policy combines the block and open schedule policy, more precisely operating room 1 follows the open scheduling policy while the others follow the block scheduling policy.

Figure 8 shows a visual representation with the different components of the Simmer environment namely a generator (the triangle on the left side), resources and trajectories as well as a special notification flow. The patient trajectory is described with the solid lines while the dotted line shows the flow of the notification from the exiting patient to the ones waiting. As mentioned before the next suitable patient is selected based on the scheduling policy and the schedule.

Figure 5-1 Graphical Representation of Implemented Simulation Model



5.3 Implementation of Visualization

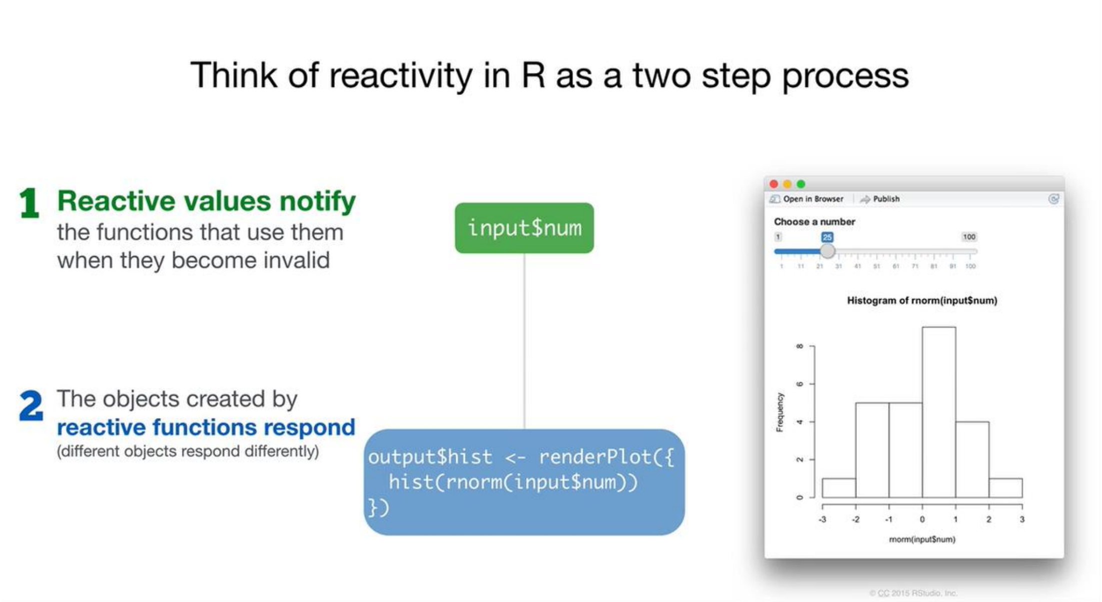
To create appealing graphs from the data the simulation provides, the popular ggplot2 library of R can be used (https://ggplot2.tidyverse.org/). The ggplot2 library provides extensive functionality for creating all different kinds of visualizations that are easily extendible. Additionally, the plotly library can be used to create interactive graphs that allow for more advanced options like selecting and highlighting certain data points or animating the progress of data over time (https://plotly.com/r/).

5.4 Implementation of the Simulation Model in Shiny

R shiny provides the software framework that enables the tool to work interactively and be published and played only with a browser online. The advantage of combining shiny with R is the fast loading speeds and the abstraction from web technologies/web programming languages like JavaScript, CSS, or HTML. Developing a web application without these technologies would not be possible but shiny allows the user to build basic web applications only using R.

Probably the most important and noteworthy concept of shiny is reactivity. Figure 9 provides a short explanation slide of the basic concept of reactivity. To create a reactive application like the right one in Figure 9, where a change in the input slider for the number of bins should instantly change the histogram shown in the application, reactivity needs to be considered carefully. Shiny provides the ability to create reactive values that when changed notify reactive functions that they are invalid and have to rerun the code. When the reactive function reruns its code, it fetches the new changed reactive value for its calculations hence providing a new histogram with the updated number of bins for this example. Reactivity allows the game to not only work by simulating with a restart for each run but also interactively changing input parameters while the simulation is still running.

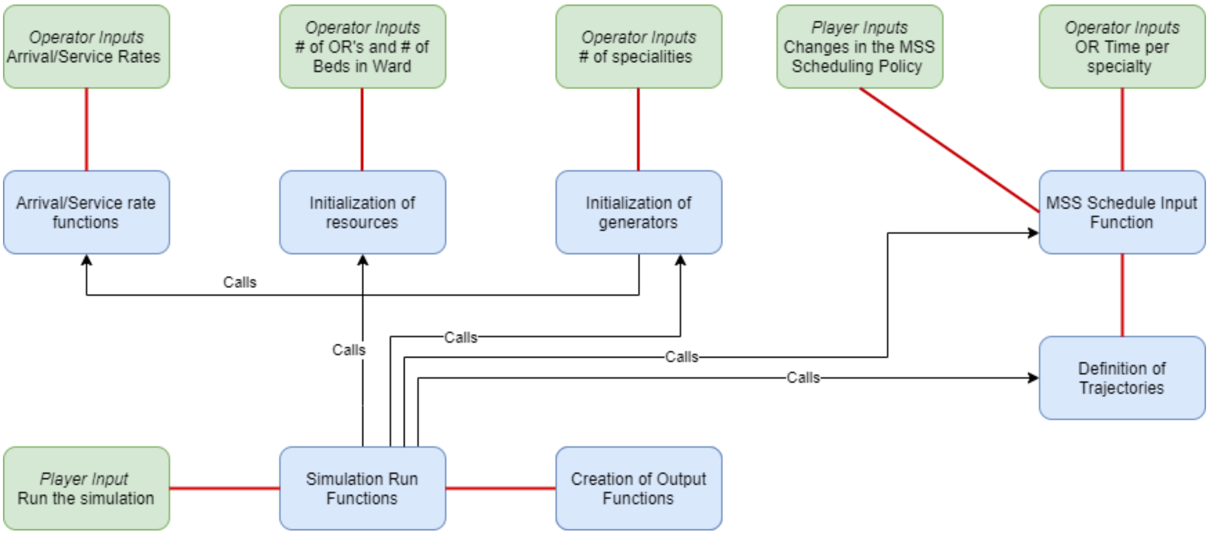
Figure 5-2 Explanation of Simple Reactivity



Note. From *How to Start Shiny (Complete)* by RStudio inc. (https://shiny.rstudio.com/tutorial/#written-tutorials)

Considering reactivity is important to ensure that the data used for the simulation model is corresponding to what the user inputs and the data displayed in the graphs are updated on the related data. Also, reactivity can lead to many reruns of unnecessary code blocks that slow down the processing speed of the tool immensely. Figure 10 shows how reactivity could be implemented in the game. The green boxes represent reactive values that are changed by either the operator or the player of the game. The blue boxes represent reactive functions that get notified by changes in the reactive values they are linked with (illustrated with the red line). These reactive functions rerun once the reactive value notifies them. Not all functions are related to each other reactively. Some functions call other functions in a non-reactive way (illustrated by the black arrow). That means the calling function is not responding to changes in the called function. For example, the function Initialization of generators is calling the arrival/service rate function non-reactively. So, if the operator changes arrival rates, the arrival function reruns and reacts. But the Initialization of the generators function will not rerun only because the arrival functions changed. It will call the arrival functions only when it needs to.

Figure 5-3 Reactivity of the Game



5.5 Verification

Verification of a simulation model is about assessing whether the conceptual model has translated into a computer model appropriately (Robinson, 2004). Verification is a continuous process that is performed during model coding by checking the code, inspecting the performance, and examining the output reports.

The simulation package Simmer of R provides the functionality to output log messages to the command file that can be used to trace every action made in the system and analyze how the entities behave in the simulation model. Additionally, it is possible to stop the simulation from running once a certain specified event occurred to facilitate the analysis of the causes of this event. With these methods bugs in the code can be found. It is important to check if the components defined in the conceptual model are represented in the computer model also taking the assumptions and simplifications into consideration. So, checking the behavior of single components is part of the continuous effort to verify the computer model. Boundary cases with unusual but possible input settings should also be checked. As an example, Appendix B shows some documentation of tests made with the help of log messages that print the events of the simulation to the console.

5.6 Model Validation

Validation of a model deals with the question of whether the model is sufficiently accurate for the purpose at hand (Robinson, 2004). Similar to verification, validation is a continuous process that occurs at almost all phases of the simulation study. Different from verification, validation compares the conceptual/computer models with the real-life system and tries to determine if the achieved accuracy is satisfactory to achieve the defined purpose.

Since this research is not based on a specific real-life hospital and the purpose is not to study interventions for a specific scenario but to educate players on a simple but still plausible system, a very accurate model is not necessary. So, the sufficient accuracy for our model is relatively low compared to simulation study's that have the analysis of capacity decisions as their purpose. Setting expectations on what decisions should lead to which results based on literature is a way of validating the model. Then these decisions are entered into the model to compare if the expected results match the actual results. Because the simulation model is not related to a real system, it is possible to validate it by comparing it with another model (Robinson, 2004). For example, one could compare the outputs of the simulation model for the ward bed occupancy with the mathematical model of Vanberkel et al. (2011) that predicts the ward occupancy based on a given MSS.

6 Chapter 6 – Preparation for Use

This chapter aims to illustrate in which scenarios the serious game can be used and how one could approach the validation of the serious game's pedogeological aim. Section 6.1 discusses the learning environments in which the use of the tool would be beneficial and gives some examples to highlight it. A game script describing how to play the game to reach the learning goals is provided as well. Section 6.2 closes this chapter with a recommendation on how to validate if the serious game actually conveys the learning goals to the user and hence fulfills its purpose.

6.1 Use Cases

The game is suitable for use cases in which the operator wants to convey the learning goals in a simple way to an audience that is unfamiliar with operations research topics. The player of the game will see the impact of his decisions immediately without the need to present mathematical formulas and will be engaged to compete and reflect on his decisions with the second player. It provides an initial intuitive introduction to appointment scheduling at a tactical level without overwhelming the user with abstract mathematical explanations.

As can be read from section 4.2, the use cases for the tool will mainly be in guided sessions where an operator that is experienced with the topic leads the player through the game. Guided sessions include lectures at schools or universities as well as professional workshops. Due to the limited scope of the game in terms of learning goals and input options, playing the game is not expected to take longer than half an hour. Because of its short playtime, the game can be used more easily in different scenarios.

For example, a lecture about appointment scheduling might use the tool as a short introduction to present the learning goal that modified block schedules perform better in terms of access time since they allocate some time dynamically to better adapt to changing customer demand. Two students could play the game in front of the class and showcase different scheduling policies and their impacts. Guided by the teacher also the other learning goals such as the impact of variability or the impact of the MSS on the wards can be explored interactively. This setting might also be possible in professional workshops in which healthcare professionals are educated about capacity allocation from an operations research perspective. Two professionals could under the guidance of the operator try to achieve better results based on their assumptions on the best policy. Different situations can be explored involving variability and tradeoffs might be highlighted. Appendix D provides a game script, in which the learning goals are listed. The game script is a guide for the operator/player on how to achieve the specified learning goals. It describes which settings and environments to choose and how the outputs might be interpreted.

6.2 Validation of Learning Objectives

To prove that the serious game successfully fulfills its purpose, it is necessary to validate that the tool conveys the learning goals to the player. This validation can be empirically achieved by setting up a questionnaire in which players can give their opinion on the attainment of the learning goals after playing the game. Additionally, players could be asked via a questionnaire about their knowledge of the presented topic before playing the game and after playing the game. If it can be observed that the players improved in their understanding of the topic, the effect of the game on the attainment of the learning goals would be shown.

7 Chapter 7 – Discussion and Conclusion

7.1 Conclusion

This study aims to address the gap of knowledge that healthcare professionals have in operations management concepts applied to the context of healthcare systems. This thesis proposes a digital serious game as a solution that incorporates an active and accessible learning experience. This simulation-based serious game is designed to specifically teach healthcare professionals and students about the effects of decisions made related to the tactical capacity allocation of the Master Surgical Schedule of an operating room. A programming language-independent conceptual model is provided that presents how a serious game that can be able to help teachers and consultants fill this gap of knowledge might look like.

It also shows how a simulation study that applies the conceptual modeling framework for simulation-based serious games of van der Zee et al. (2012) with the objective of teaching in conjunction with gamification techniques looks like. The modeling framework presents itself as a suitable guide to conceptualize a simulation with teaching as its goal. But it misses the incorporation of gamification aspects that make the serious game fun and engaging to play.

Recommendations for the development of the serious game are given and the serious game is developed in the programming language R and R Shiny. The general programming R is the foundation of the development environment for the serious game. The development environment also consists of the R Shiny package, to create an accessible web page and user interface, as well as the R Simmer package, to program the simulation model. The development environment is suitable for the creation of an accessible web page with an interactive simulation as its backend in a short time frame and with limited prior knowledge of the R language. However, more complicated simulation models are increasingly hard to implement with the R simmer package, interactive animations of the simulation model are only doable with the use of other programming languages like Javascript, and the implementation of gamification techniques such as game rounds or an interactive tutorial requires a profound knowledge of the R Shiny package and the programming language Javascript. Hence the development environment is suitable for the implementation of simple interactive dashboards but not for the creation of a more sophisticated interactive serious game that wants to make use of gamification techniques.

The literature research indicates that there is a lack of openly accessible serious games for operations research in healthcare settings. Therefore the thesis also aims to provide an openly accessible serious game with an openly accessible source code, to ensure continuous collaboration and transparency. In addition, practical use cases of the serious game are highlighted and techniques for verification and validation are elaborated. To access the serious game only an internet connection is needed and the following link:

https://noormansour.shinyapps.io/Appointment_Scheduling_Simulation_Game/

The learning goals mentioned in section 1.2.3 are addressed by the game. Appendix D provides a game script that explains how to reach the learning goals, that were specified at the beginning of the thesis. The game facilitates the use of scenarios that preset the input values such that the learning goals are highlighted by the output.

The obvious limitation of the study pertains to the limited ability to comment on the effectiveness of the designed serious game. Future research can incorporate the assessment of the effectiveness of the serious game through a survey with participating students and doctors. Although the results provide a serious game design with limited scope and a limited set of learning goals, it promotes further research into the use of serious gaming, especially for healthcare logistics, as an effective tool to convey otherwise complex and mathematical learning goals intuitively. Also, it proposes that the creation of serious games does not necessarily need too much strain on human or capital resources as this game is developed by a bachelor student with no prior knowledge of R and its libraries, limited programming knowledge in general, and in a limited time frame. Rather it promotes the simplicity of serious games to facilitate easily recognizable and isolated learning goals.

7.2 Possible Future Improvements and Extensions

One of the biggest strengths of this research is its potential for extensions and improvements. Extensions can not only be made on the scope and detail of the conceptual model but also on the gamification techniques used. The proposed conceptual model can be extended in scope by incorporating more components and/or adding more detail to the model. For example, the outpatient clinic might be incorporated into the model, adding a layer of possible decisions and learning goals related to capacity allocation decisions for outpatient clinics and the coordination between the outpatient clinic, operating room, and ward. Or by adding changeover times between each surgery in the model, the detail of the model would increase, providing a more accurate presentation of reality.

Many extensions and improvements are possible in regards to the gamification techniques used. An interesting extension would be to incorporate animated real-time graphs that visualize the player how the patients flow through the system and how his decisions can impact that flow, yielding an even more intuitive approach to attain the learning goals. Also, the addition of an automated in-game tutorial that would introduce the player to the game and guide him through multiple game phases would yield a valuable extension. It could take on all the responsibilities the operator takes in the current design and enable players to independently play the game without the restrictions of needing a human operator. Another possible interesting extension would be the incorporation of multiple players with different roles as used by (Mattarelli et al., 2006). A multi-player game could include the OR Scheduler but also a player that represents the surgeons and another player that represents the manager of the ward. All of them have different responsibilities and actions to perform in the game and their objectives might not be aligned for every decision. Although a complex endeavor, this extension would provide additional soft skill learning goals like communication and coordination while exploring the different objectives of each actor in the creation of an MSS.

The most important next step for this research would be the empirical validation of the effectiveness of the serious game in conveying the specified learning goals to students and healthcare practitioners. A survey or an interview study might achieve to answer the question of how effective the serious game is in conveying the learning goals.

Bibliography

Beliën, J., Demeulemeester, E., & Cardoen, B. (2006). Visualizing the Demand for Various Resources as a Function of the Master Surgery Schedule: A Case Study. *Journal of Medical Systems*, *30*(5), 343. https://doi.org/10.1007/s10916-006-9012-5

Britnell, M. (2019). *Human : solving the global workforce crisis in healthcare*.

D'Amours, S., Marier, P., Rönnqvist, M., Azouzi, R., & Fjeld, D. (2017). The Online Wood Supply Game. *INFORMS Transactions on Education*, *18*. https://doi.org/10.1287/ited.2017.0174

Dario Radečić. (2020, December 9). *Python and Dash vs. R and Shiny*.

de Lope, R. P., & Medina-Medina, N. (2016). A Comprehensive Taxonomy for Serious Games. *Journal of Educational Computing Research*, *55*(5), 629–672. https://doi.org/10.1177/0735633116681301

Denton, B., Viapiano, J., & Vogl, A. (2007). Optimization of surgery sequencing and scheduling decisions under uncertainty. *Health Care Management Science*, *10*(1), 13–24.

Djaouti, D., Alvarez, J., & Jessel, J. (2011). *Classifying Serious Games: The G/P/S Model*.

Dobson, G., & Shumsky, R. (2006). Web-based Simulations for Teaching Queueing, Little's Law, and Inventory Management. *INFORMS Transactions on Education*, *7*(1), 106–124. https://doi.org/10.1287/ited.7.1.106

Greenblat, C. S. (1988). *Designing games and simulations: An illustrated handbook*. SAGE Publications, Incorporated.

Guerriero, F., & Guido, R. (2011). Operational research in the management of the operating theatre: a survey. *Health Care Management Science*, *14*(1), 89–114. https://doi.org/10.1007/s10729-010-9143-6

Heerkens, H., & van Winden, A. (2017). *Solving Managerial Problems Systematically*. Noordhoff Uitgevers.

Hidayatno, A., Zulkarnain, Hasibuan, R. G., Wardana Nimpuno, G. C., & Destyanto, A. R. (2019). Designing a Serious Simulation Game as a Learning Media of Sustainable Supply Chain Management for Biofuel Production. *Energy Procedia*, *156*, 43–47. https://doi.org/https://doi.org/10.1016/j.egypro.2018.11.083

Hulshof, P. J. H., Kortbeek, N., Boucherie, R. J., Hans, E. W., & Bakker, P. J. M. (2012). Taxonomic classification of planning decisions in health care: a structured review of the state of the art in OR/MS. *Health Systems*, *1*(2). https://doi.org/10.1057/hs.2012.18

Jackson, J., Iacovides, J., Duncan, M., Alders, M., Maben, J., & Anderson, J. (2020). Operationalizing resilient healthcare concepts through a serious video game for clinicians. *Applied Ergonomics*, *87*, 103112. https://doi.org/https://doi.org/10.1016/j.apergo.2020.103112

Kamran, M. A., Karimi, B., Dellaert, N., & Demeulemeester, E. (2019). Adaptive operating rooms planning and scheduling: A rolling horizon approach. *Operations Research for Health Care*, *22*, 100200. https://doi.org/https://doi.org/10.1016/j.orhc.2019.100200

Klabbers, J. H. G. (2003). The gaming landscape: a taxonomy for classifying games and simulations. *DIGRA Conference*, 4–6.

Kulkarni, S. S., Mai, B., Amirkiaee, S. Y., & Tarakci, H. (2019). Dynamic Interactive Visualizations: Implications of Seeing, Doing, and Playing for Quantitative Analysis Pedagogy. *INFORMS Transactions on Education*, *19*(3), 121–142. https://doi.org/10.1287/ited.2018.0203

Laamarti, F., Eid, M., & el Saddik, A. (2014). An Overview of Serious Games. *International Journal of Computer Games Technology*, *2014*, 358152. https://doi.org/10.1155/2014/358152

Laksmi, M. P., & Ardi, R. (2020). Serious Simulation Gaming as Learning Media for Plastic Waste Recycling Management System in Indonesia: A Conceptual Model. *Proceedings of the 3rd Asia Pacific Conference on Research in Industrial and Systems Engineering 2020*, 187–192. https://doi.org/10.1145/3400934.3400969

Marín-Vega, H., Alor-Hernández, G., Colombo-Mendoza, L., Sanchez-Ramirez, C., García-Alcaraz, J., & Avelar, L. (2019). Zeus – A tool for generating rule-based serious games with gamification techniques. *IET Software*, *14*. https://doi.org/10.1049/iet-sen.2019.0028

Mattarelli, E., Fadel, K. J., & Weisband, S. P. (2006). Design of a Role-Playing Game to Study the Trajectories of Health Care Workers in an Operating Room. *CHI '06 Extended Abstracts on Human Factors in Computing Systems*, 1091–1096. https://doi.org/10.1145/1125451.1125658

M'Hallah, R., & Visintin, F. (2019). A stochastic model for scheduling elective surgeries in a cyclic Master Surgical Schedule. *Computers & Industrial Engineering*, *129*, 156–168. https://doi.org/https://doi.org/10.1016/j.cie.2019.01.030

Patterson, P. (1996). What makes a well-oiled scheduling system? *OR Manager*, *12 9*, 19–23.

Proulx, J.-N., Romero, M., & Arnab, S. (2016). Learning Mechanics and Game Mechanics Under the Perspective of Self-Determination Theory to Foster Motivation in Digital Game Based Learning. *Simulation & Gaming*, *48*(1), 81–97. https://doi.org/10.1177/1046878116674399

Riedel, J. C. K. H., & Hauge, J. B. (2011). State of the art of serious games for business and industry. *2011 17th International Conference on Concurrent Enterprising*, 1–8.

Robinson, S. (2004). *Simulation: the practice of model development and use* (Vol. 50). Wiley Chichester.

Robinson, S. (2008). Conceptual Modelling for Simulation Part II: A Framework for Conceptual Modelling. *The Journal of the Operational Research Society*, *59*(3), 291–304. http://www.jstor.org.ezproxy2.utwente.nl/stable/30132750

Suttie, N., Louchart, S., Lim, T., Macvean, A., Westera, W., Brown, D., & Djaouti, D. (2012). Introducing the "Serious Games Mechanics" A Theoretical Framework to Analyse Relationships Between "Game" and"Pedagogical Aspects" of Serious Games. *Procedia Computer Science*, *15*, 314–315. https://doi.org/https://doi.org/10.1016/j.procs.2012.10.091

Uskov, A., & Sekar, B. (2015). Smart Gamification and Smart Serious Games. In D. Sharma, M. Favorskaya, L. C. Jain, & R. J. Howlett (Eds.), *Fusion of Smart, Multimedia and Computer Gaming Technologies: Research, Systems and Perspectives* (pp. 7–36). Springer International Publishing. https://doi.org/10.1007/978-3-319-14645-4_2

van der Zee, D.-J., Holkenborg, B., & Robinson, S. (2012). Conceptual modeling for simulation-based serious gaming. *Decision Support Systems*, *54*(1), 33–45. https://doi.org/https://doi.org/10.1016/j.dss.2012.03.006

van Oostrum, J. M., Bredenhoff, E., & Hans, E. W. (2010). Suitability and managerial implications of a master surgical scheduling approach. *Annals of Operations Research*, *178*(1), 91–104.

van Oostrum, J. M., van Houdenhoven, M., Hurink, J. L., Hans, E. W., Wullink, G., & Kazemier, G. (2008). A master surgical scheduling approach for cyclic scheduling in operating room departments. *OR Spectrum*, *30*(2), 355–374. https://doi.org/10.1007/s00291-006-0068-x

Vanberkel, P. T., Boucherie, R. J., Hans, E. W., Hurink, J. L., van Lent, W. A. M., & van Harten, W. H. (2011). An exact approach for relating recovering surgical patient workload to the master surgical schedule. *The Journal of the Operational Research Society*, *62*(10), 1851–1860. http://www.jstor.org.ezproxy2.utwente.nl/stable/23412418

Zhang, C., Hauge, J. B., Härenstam, K. P., & Meijer, S. (2021). Game experience and learning effects of a scoring-based mechanic for logistical aspects of pediatric emergency medicine: Development and feasibility study. *JMIR Serious Games*, *9*(1). https://doi.org/10.2196/21988

Zyda, M. (2005). From visual simulation to virtual reality to games. *Computer*, *38*(9), 25–32. https://doi.org/10.1109/MC.2005.297

Appendix A: Conceptual Modeling Framework and Game Design Process

The chosen methodology of van der Zee et al. (2012) is cited 40 times by most relevant articles for this research. Papers that aim to design a simulation-based serious game for operations research/ supply chain refer to and use the methodology (Hidayatno et al., 2019; Laksmi & Ardi, 2020). Furthermore, it perfectly fits into the requirements and objectives of this assignment namely to develop a serious game that teaches operations research/management concepts to students and professionals from different fields via the use of simulation models and relates directly to the highly cited framework for simulation modeling from Robinson (2008). Table A.1 provides a detailed overview of the steps and the activities performed in them.

Table A.1: A modified conceptual modeling framework for simulation-based serious gaming – detailing activities.

Activity	Details
1. Understanding the learning environment	• Understand the subject matter, context of use, and likely players/operators, preferably by interviewing clients and subject matter experts • Explore learning needs, given the environment, i.e., student education or professional training • Decide on the appropriateness of a computer-based game format
2. Determine objectives – Modeling objectives	• Identify the game's pedagogic purposes • Express modeling objectives in terms of players' achievements in mastering their decision-making skills

– General project objectives	• Establish and assess project requirements on resource use
	• Clarify the nature of the model and its use concerning:
	    o Visualization
	    o Player interaction
	    o Responsiveness
	    o Model/component re-use

3. Identify the model outputs	• Check modeling objectives for relevant performance measures, indicating player achievements • Establish model outputs helping to identify potential bottlenecks in systems operations and explain player achievements • Determine format for representing responses
4. Identify the model inputs	• Select quantitative and qualitative data that can be changed, to represent alternative system configurations appealing to (alternative) groups of players • Determine the range over which model inputs may be varied

5. Determine model content: scope and level of detail	• Determine model scope:
	    o Identify the system boundary
	    o Identify all components in the real system that lie within the model boundary (include player roles)
	    o Assess whether to include components
	• Determine model detail (attributes) for all components included
	• Identify assumptions and simplifications concerning model scope and detail, and assess their impact on model responses

Note. Adapted from *Conceptual modeling for simulation-based serious gaming* by (van der Zee et al., 2012)

Appendix B: Verification Test with Log messages

Figure B.1 shows screenshots of the log messages that are printed to the console by the patients that go through the hospital. The left number of every row indicates the simulation time the action takes place, followed by the patient identification given a type of patient (A, B, C) and his number. With this, the patient flow can be checked for any unwanted behavior. The verification with the conceptual model is possible with this technique.

Figure B.1 Console Log Messages of Simulation Run

```
4.08971: patient_C0: Begin OC
10.1301: patient_C1: Begin OC
17.7759: patient_A0: Begin OC
18.3019: patient_B0: Begin OC
18.6721: patient_C0: End OC, Waiting at 'home' for OR
18.6721: patient_C0: Beginning Surgery
19.5927: patient_C2: Begin OC
20.9637: patient_C3: Begin OC
26.6842: patient_C2: End OC, Waiting at 'home' for OR
27.0184: patient_B0: End OC, Waiting at 'home' for OR
27.0184: patient_B0: Beginning Surgery
27.9006: patient_A1: Begin OC
27.9034: patient_B1: Begin OC
31.6618: patient_A0: End OC, Waiting at 'home' for OR
31.6618: patient_A0: Beginning Surgery
32.752: patient_A1: End OC, Waiting at 'home' for OR
33.6455: patient_B2: Begin OC
35.6311: patient_C3: End OC, Waiting at 'home' for OR
36.2521: patient_B1: End OC, Waiting at 'home' for OR
36.3742: patient_A2: Begin OC
39.9105: patient_C4: Begin OC
39.9446: patient_C0: Surgery done, going to ward
39.9446: patient_C2: Beginning Surgery
40.2205: patient_A0: Surgery done, going to ward
40.2205: patient_A1: Beginning Surgery
44.7478: patient_B2: End OC, Waiting at 'home' for OR
45.2891: patient_C1: End OC, Waiting at 'home' for OR
45.3166: patient_C4: End OC, Waiting at 'home' for OR
45.8289: patient_C2: Surgery done, going to ward
45.8289: patient_C3: Beginning Surgery
45.8412: patient_B3: Begin OC
50.3923: patient_C3: Surgery done, going to ward
50.3923: patient_C1: Beginning Surgery
50.4267: patient_A0: Treatement done
50.4267: patient_A0: Revisit going to the OC again
50.994: patient_B0: Surgery done, going to ward
50.994: patient_B1: Beginning Surgery
51.8455: patient_A1: Surgery done, going to ward
52.7862: patient_C0: Treatement done
54.7299: patient_A2: End OC, Waiting at 'home' for OR
54.7299: patient_A2: Beginning Surgery
54.7299: patient_A2: Beginning Surgery
55.9461: patient_A0: Revisit Treatement done
56.5197: patient_B3: End OC, Waiting at 'home' for OR
56.8805: patient_B1: Surgery done, going to ward
56.8805: patient_B2: Beginning Surgery
57.0989: patient_C3: Treatement done
57.3987: patient_B0: Treatement done
58.4192: patient_A3: Begin OC
63.509: patient_A1: Treatement done
65.759: patient_C1: Surgery done, going to ward
65.759: patient_C4: Beginning Surgery
66.9513: patient_C2: Treatement done
67.7334: patient_B1: Treatement done
68.0787: patient_A4: Begin OC
70.3941: patient_A3: End OC, Waiting at 'home' for OR
71.8999: patient_B2: Surgery done, going to ward
71.8999: patient_B3: Beginning Surgery
74.9334: patient_A5: Begin OC
75.4567: patient_A6: Begin OC
75.8434: patient_B3: Surgery done, going to ward
78.3086: patient_C4: Surgery done, going to ward
79.7488: patient_B4: Begin OC
81.1683: patient_C1: Treatement done
81.1683: patient_C1: Revisit going to the OC again
81.3863: patient_B2: Treatement done
83.4193: patient_A5: End OC, Waiting at 'home' for OR
85.2972: patient_A4: End OC, Waiting at 'home' for OR
87.6056: patient_A7: Begin OC
88.5552: patient_B4: End OC, Waiting at 'home' for OR
88.5552: patient_B4: Beginning Surgery
88.6746: patient_B3: Treatement done
88.6746: patient_B3: Revisit going to the OC again
90.7768: patient_B5: Begin OC
92.2357: patient_B6: Begin OC
92.9136: patient_C5: Begin OC
93.3928: patient_C4: Treatement done
93.4489: patient_C1: Revisit Treatement done
94.4265: patient_C6: Begin OC
95.9504: patient_A7: End OC, Waiting at 'home' for OR
96.9098: patient_B3: Revisit Treatement done
99.2773: patient_A2: Surgery done, going to ward
```

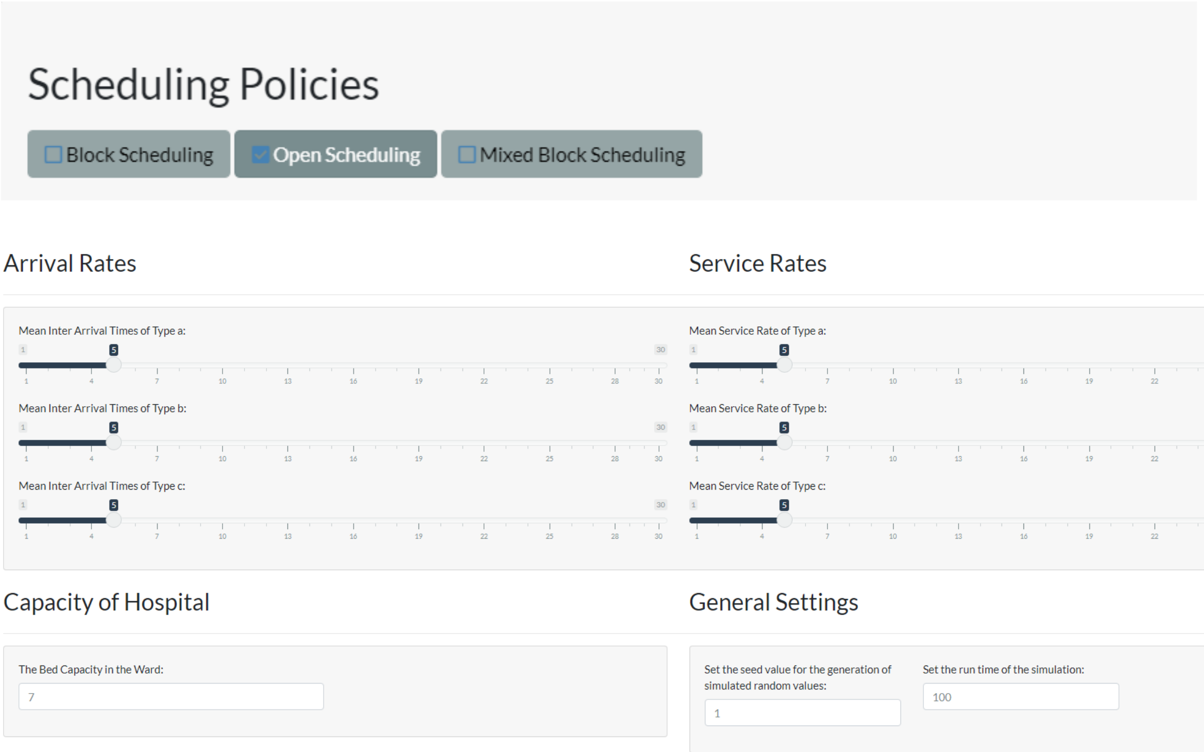
The following part is a test of the functionality of the simulation divided by the 3 scheduling policies. Thereby the test is first formulated as a question, followed by the input settings used to set up the simulation, and finally, the outputted log messages are shown and the results are discussed.

1. Open Scheduling Policy

Question: Does the open schedule policy work (messaging the one that is waiting for the longest in the queue regardless of their patient type)?

Set-Up:

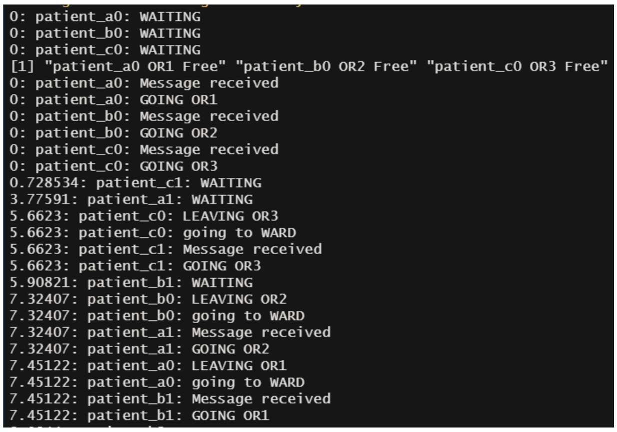
Figure B.2 Input Settings for Open Policy Test



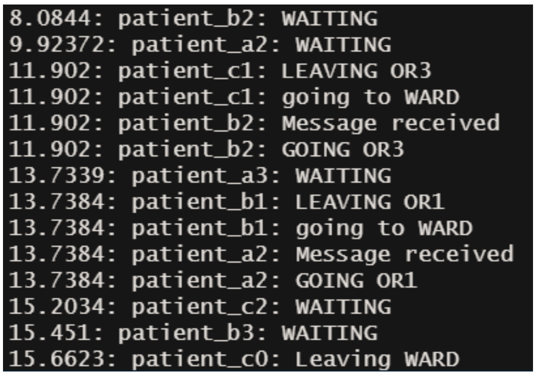
Results:

The output shows that the open policy is successfully implemented. After the initial opening by the patients a0, b0, and c0, the next ones waiting are c1,a1, and b1 in that order. And the first one to leave the OR is patient c0 who calls c1, the second to leave is patient b0 who calls a1, and lastly patient a0 who calls b1. So the order of waiting people was respected independently of the patient type they belong to. This can also be seen in the further chain of messages.

Figure B.3 Console Log Messages of open scheduling policy



```
0: patient_a0: WAITING
0: patient_b0: WAITING
0: patient_c0: WAITING
[1] "patient_a0 OR1 Free" "patient_b0 OR2 Free" "patient_c0 OR3 Free"
0: patient_a0: Message received
0: patient_a0: GOING OR1
0: patient_b0: Message received
0: patient_b0: GOING OR2
0: patient_c0: Message received
0: patient_c0: GOING OR3
0.728534: patient_c1: WAITING
3.77591: patient_a1: WAITING
5.6623: patient_c0: LEAVING OR3
5.6623: patient_c0: going to WARD
5.6623: patient_c1: Message received
5.6623: patient_c1: GOING OR3
5.90821: patient_b1: WAITING
7.32407: patient_b0: LEAVING OR2
7.32407: patient_b0: going to WARD
7.32407: patient_a1: Message received
7.32407: patient_a1: GOING OR2
7.45122: patient_a0: LEAVING OR1
7.45122: patient_a0: going to WARD
7.45122: patient_b1: Message received
7.45122: patient_b1: GOING OR1
```



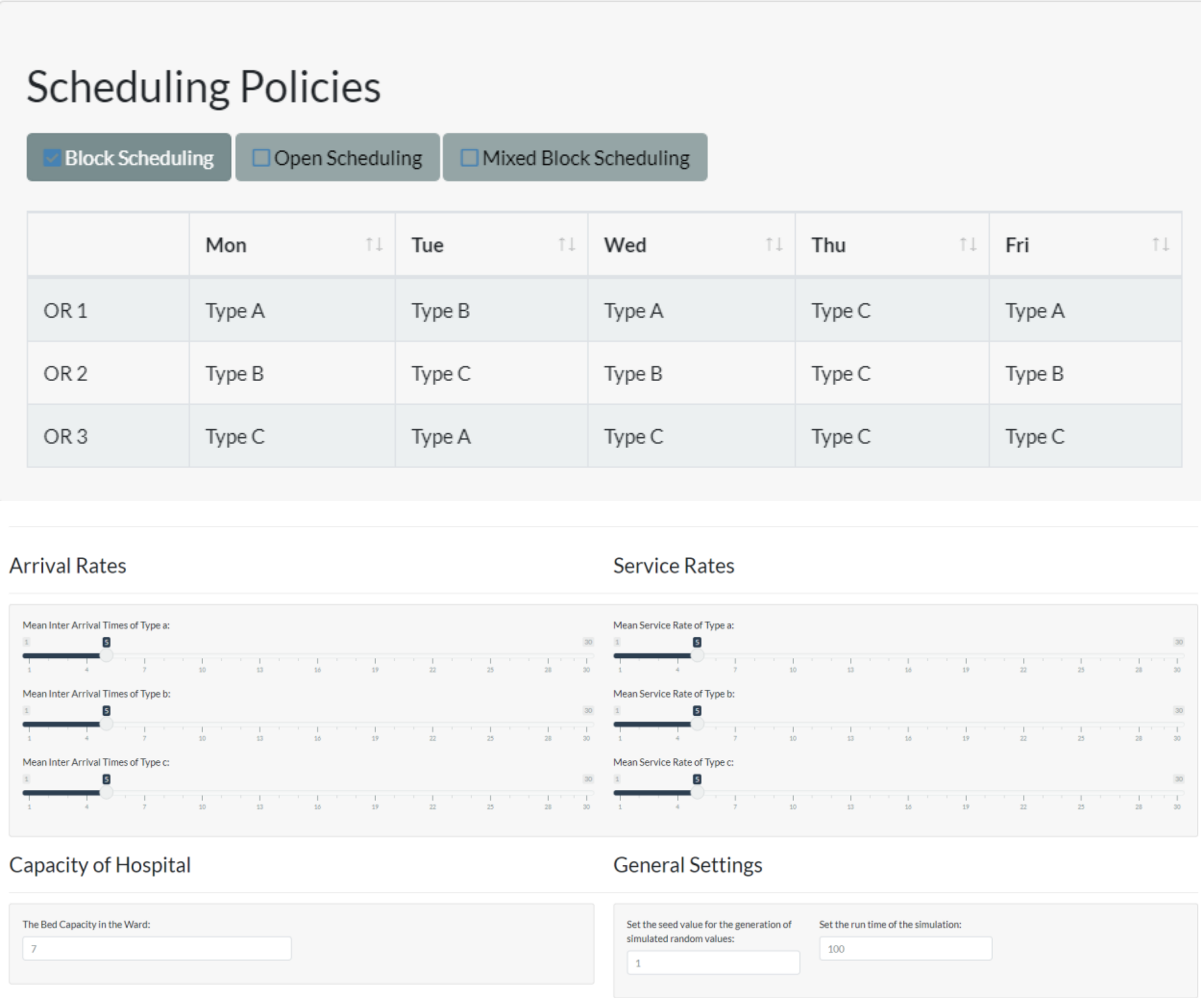
```
8.0844: patient_b2: WAITING
9.92372: patient_a2: WAITING
11.902: patient_c1: LEAVING OR3
11.902: patient_c1: going to WARD
11.902: patient_b2: Message received
11.902: patient_b2: GOING OR3
13.7339: patient_a3: WAITING
13.7384: patient_b1: LEAVING OR1
13.7384: patient_b1: going to WARD
13.7384: patient_a2: Message received
13.7384: patient_a2: GOING OR1
15.2034: patient_c2: WAITING
15.451: patient_b3: WAITING
15.6623: patient_c0: Leaving WARD
```

2. Block Scheduling Policy

Question: Is the block scheduling policy implemented correctly also with the day block shifts?

Set-Up:

Figure B.4 Input Settings for Block Scheduling Test



<u>Results:</u>

Figure B.5 Console Log Messages of initial selection of patients

```
0: patient_a0: WAITING
0: patient_b0: WAITING
0: patient_c0: WAITING
[1] "patient_a0 OR1 Free" "patient_b0 OR2 Free" "patient_c0 OR3 Free"
0: patient_a0: Message received
0: patient_a0: GOING OR1
0: patient_b0: Message received
0: patient_b0: GOING OR2
0: patient_c0: Message received
0: patient_c0: GOING OR3
0.728534: patient_c1: WAITING
3.77591: patient_a1: WAITING
5.6623: patient_c0: LEAVING OR3
5.6623: patient_c0: going to WARD
5.6623: patient_c1: Message received
5.6623: patient_c1: GOING OR3
5.90821: patient_b1: WAITING
7.32407: patient_b0: LEAVING OR2
7.32407: patient_b0: going to WARD
7.32407: patient_b1: Message received
7.32407: patient_b1: GOING OR2
7.45122: patient_a0: LEAVING OR1
7.45122: patient_a0: going to WARD
7.45122: patient_a1: Message received
7.45122: patient_a1: GOING OR1
```

The output shows that the block scheduling policy is successfully implemented. After the initial opening by the patients a0, b0, and c0, the next ones waiting are c1,a1, and b1 in that order. And the first one to leave the OR is patient c0 who calls c1, the second to leave is patient b0 who calls b1, and lastly patient a0 who calls a1. So only the patient types scheduled for that block and operating room are messaged for entry.

Figure B.6 Console log messages for the functionality of wait handler

```
8.0844: patient_b2: WAITING
9.92372: patient_a2: WAITING
11.902: patient_c1: LEAVING OR3
11.902: patient_c1: going to WARD
11.902: wait_handler_OR30: Time now: 11.9020210675411
11.902: wait_handler_OR30: Time to the next event: 0.001
11.903: wait_handler_OR30: Time now: 11.9030210675411
11.903: wait_handler_OR30: Time to the next event: 1.83184764252548
13.7339: patient_a3: WAITING
13.7349: wait_handler_OR30: Time now: 13.7348687100666
13.7349: wait_handler_OR30: Time to the next event: 0.00455146729337719
13.7384: patient_a1: LEAVING OR1
13.7384: patient_a1: going to WARD
13.7384: patient_a2: Message received
13.7384: patient_a2: GOING OR1
13.7394: wait_handler_OR30: Time now: 13.7394201773599
13.7394: wait_handler_OR30: Time to the next event: 1.4649561434794
15.2034: patient_c2: WAITING
15.2044: patient_c2: Message received
15.2044: patient_c2: GOING OR3
```

Also, the functioning of the waiting handler is demonstrated by the output. As patient c1 leaves the OR3, no other patient of type C is currently waiting on the waiting list. Only when the next patient c2 is arriving at the waiting list, he gets a message from the waiting handler.

Figure B.7 Console log messages for Patient Type Switch between Days

```
22.2853: patient_b2: LEAVING OR2
22.2853: patient_b2: going to WARD
22.2853: patient_b3: Message received
22.2853: patient_b3: GOING OR2
23.7384: patient_a1: Leaving WARD
24.48: patient_a3: LEAVING OR1
24.48: patient_a3: going to WARD
24.48: patient_b4: Message received
24.48: patient_b4: GOING OR1
25.6862: patient_b3: LEAVING OR2
25.6862: patient_b3: going to WARD
25.6862: patient_c4: Message received
25.6862: patient_c4: GOING OR2
25.7139: patient_a5: WAITING
25.9187: patient_c5: WAITING
26.5177: patient_c3: LEAVING OR3
26.5177: patient_c3: going to WARD
26.5177: patient_a4: Message received
26.5177: patient_a4: GOING OR3
27.6106: patient_b1: Leaving WARD
28.1577: patient_a2: Leaving WARD
```

The switch of patient types according to the input schedule is also successfully implemented. One can observe that the transition between Monday to Tuesday at time 24 implies that the patient type called for the OR1 now has to be switched from type A to type B. The following output showcases this successful switch as patient a3 leaves the OR1 in the new block and instead of calling type A, he calls the next available patient of type B. Also, the switch of OR2 from B to C and the switch from OR3 from C to A is presented by the output.

Figure B.8 Console log messages for Scheduling only one patient type on a day

```
72.1068: patient_c10: Leaving WARD
72.1457: patient_c12: LEAVING OR3
72.1457: patient_c12: going to WARD
72.1457: patient_c13: Message received
72.1457: patient_c13: GOING OR3
73.5453: patient_b12: WAITING
73.8214: patient_a12: LEAVING OR1
73.8214: patient_a12: going to WARD
73.8214: wait_handler_OR10: Time now: 73.8213799740605
73.8214: wait_handler_OR10: Time to the next event: 0.001
73.8224: wait_handler_OR10: Time now: 73.8223799740605
73.8224: wait_handler_OR10: Time to the next event: 0.270563523641622
74.0919: patient_b10: Leaving WARD
74.0929: wait_handler_OR10: Time now: 74.0929434977021
74.0929: wait_handler_OR10: Time to the next event: 0.550768029311167
74.6427: patient_a10: Leaving WARD
74.6437: wait_handler_OR10: Time now: 74.6437115270133
74.6437: wait_handler_OR10: Time to the next event: 1.07571840808118
75.7184: patient_b11: LEAVING OR2
75.7184: patient_b11: going to WARD
75.7184: wait_handler_OR20: Time now: 75.7184299350945
75.7184: wait_handler_OR20: Time to the next event: 0.001
75.7194: wait_handler_OR10: Time now: 75.7194299350945
75.7194: wait_handler_OR10: Time to the next event: 0.709354325669376
75.7194: wait_handler_OR20: Time now: 75.7194299350945
75.7194: wait_handler_OR20: Time to the next event: 0.709354325669376
```

Additionally, the rather extreme input case of scheduling only one patient type on a day is shown in the following output. On Thursday only patients of types C are scheduled and hence from time 72 onwards one can see the waiting handler of OR1 and OR2 being active due to no patients of type C being available. They also successfully skip any arrival of other patient types.

3. Modified Blocks Scheduling Policy

Question: Is the modified block scheduling policy implemented correctly? Also with regards to the fact that operating room 1 supports an open scheduling policy while the other 2 follow the block scheduling policy? Do day shifts work as well?

Set-Up:

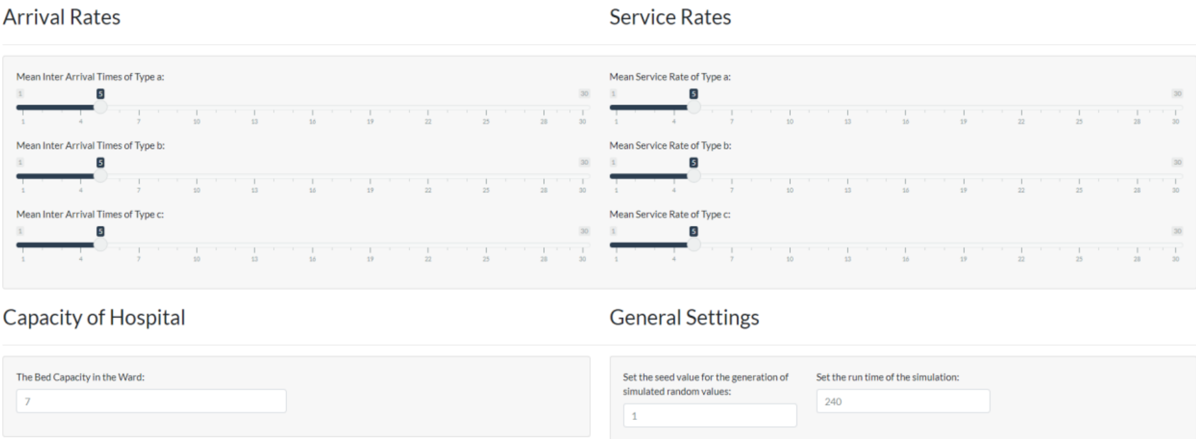
Figure B.9 Input Settings for Modified Block Scheduling Test

Input Parameters

Scheduling Policies

☐ Block Scheduling ☐ Open Scheduling ☑ Mixed Block Scheduling

	Mon	Tue	Wed	Thu	Fri
OR 1	Type A	Type A	Type A	Type A	Type A
OR 2	Type B	Type B	Type B	Type B	Type B
OR 3	Type C	Type C	Type C	Type C	Type C



<u>Results:</u>

Figure B.10 Console Log Messages of initial selection of patients in modified block scheduling policy

```
0: patient_a0: WAITING
0: patient_b0: WAITING
0: patient_c0: WAITING
[1] "patient_b0 OR1 Free" "patient_a0
0: patient_b0: Message received
0: patient_b0: GOING OR1
0: patient_a0: Message received
0: patient_a0: GOING OR2
0: patient_c0: Message received
0: patient_c0: GOING OR3
0.728534: patient_c1: WAITING
3.77591: patient_a1: WAITING
5.90202: patient_c0: LEAVING OR3
5.90202: patient_c0: going to WARD
5.90202: patient_c1: Message received
5.90202: patient_c1: GOING OR3
5.90821: patient_b1: WAITING
8.0844: patient_b2: WAITING
8.3179: patient_a0: LEAVING OR2
8.3179: patient_a0: going to WARD
8.3179: patient_a1: Message received
8.3179: patient_a1: GOING OR2
8.51098: patient_b0: LEAVING OR1
8.51098: patient_b0: going to WARD
8.51098: patient_b1: Message received
8.51098: patient_b1: GOING OR1
```

The output shows that the initialization is successful. The schedule is correctly implemented as Patient B goes to OR1, Patient C goes to OR 3 and Patient A goes to OR 2. After the initial opening, the next ones waiting are c1,a1, and b1 in that order. And the first one to leave the OR is patient c0 who calls c1, the second to leave is patient b0 who calls b1, and lastly patient a0 who calls a1. As OR3 and 2 follow a block scheduling policy and Patient b1 is the last one to arrive as OR1 is the last one to finish, the behavior is in line with the modified block scheduling policy.

Figure B.6 Console log messages for the functionality of wait handler

```
9.4015: patient_b3: WAITING
9.92372: patient_a2: WAITING
11.6704: patient_b1: LEAVING OR1
11.6704: patient_b1: going to WARD
11.6704: patient_b2: Message received
11.6704: patient_b2: GOING OR1
11.9501: patient_a1: LEAVING OR2
11.9501: patient_a1: going to WARD
11.9501: patient_a2: Message received
11.9501: patient_a2: GOING OR2
12.6199: patient_c1: LEAVING OR3
12.6199: patient_c1: going to WARD
12.6199: wait_handler_OR30: Time now: 12.6198664689265
12.6199: wait_handler_OR30: Time to the next event: 0.001
```

Also, the functioning of the waiting handler and the block scheduling policy for operating room 3 is demonstrated by the output. As patient c1 leaves the OR3, no other patient of type C is currently waiting on the waiting list. Only when the next patient c2 is arriving at the waiting list, he gets a message from the waiting handler.

Figure B.7 Console log messages for Open policy in OR 1

```
34.0277: patient_c5: LEAVING OR3
34.0277: patient_c5: going to WARD
34.0277: patient_c6: Message received
34.0277: patient_c6: GOING OR3
34.1019: patient_c9: WAITING
35.2581: patient_b6: LEAVING OR1
35.2581: patient_b6: going to WARD
35.2581: patient_c7: Message received
35.2581: patient_c7: GOING OR1
35.812: patient_a3: LEAVING OR2
35.812: patient_a3: going to WARD
35.812: wait_handler_OR20: Time now: 35.8119967590858
35.812: wait_handler_OR20: Time to the next event: 0.001
```

Here patient b6 out of OR1 invites the patient that is waiting for the longest independent of his type namely patient c7. This shows the implementation of the open scheduling policy while OR 3 still only accepts patients of type C and OR 2 only patients of type A.

Figure B.8 Console log messages for Day Shift

```
48.2211: patient_c10: LEAVING OR3
48.2211: patient_c10: going to WARD
48.2211: patient_c11: Message received
48.2211: patient_c11: GOING OR3
49.4979: patient_a6: WAITING
49.5389: patient_b12: WAITING
49.7942: patient_a7: WAITING
50.0552: patient_c14: WAITING
51.7555: patient_a5: LEAVING OR2
51.7555: patient_a5: going to WARD
51.7555: patient_b8: Message received
51.7555: patient_b8: GOING OR2
51.8891: patient_b7: LEAVING OR1
51.8891: patient_b7: going to WARD
51.8891: patient_b9: Message received
51.8891: patient_b9: GOING OR1
```

Also, the day shift is successfully implemented. This can be seen from the change of the patient types for OR2 as the last patient a5 calls the next patient b8. OR 3 still follows the block policy for patients of type C.

Appendix C: Guide for Future Students

The following section will answer interesting questions for future students, that might want to work on a similar project themselves or extend this project. The purpose of this guide is to help the students get access to the code, guide them to useful resources, and giving them the ability to understand and change it.

Where do I start?

Ideally, you should read or skim through the thesis or at least the management summary to get an idea and overview of the project and the conceptual design. Now get started with the technical intricacies of the project requires you to:

- Get at least a bit familiar with R (recommended as it makes understanding a lot easier)
- Install RStudio, the main IDE for the R language (optional)
- Take a look at the source code of my project from GitHub and fork it
- Analyze the code and
- Experiment with the code
- Try to publish your own very simple application

Once you have done this you should get a good overview of the technical implementation of this project and ideally a good feeling of what is possible or how you could add extensions.

How do I get to the source code of the tool?

Assuming that you got a bit familiar with R and installed an IDE

The source code of the tool can be found under my GitHub repository with the following link: https://github.com/NoorMansour1/Surgical-Chain-R-Shiny. You can take a look at the code in GitHub itself or fork this repository so that you can use it and change it locally on your computer. Depending on the IDE you want to use there are different methods to fork a GitHub repository, here is an example if you use RStudio: https://community.rstudio.com/t/how-to-fork-a-github-repo-to-rstudio-on-windows-10/51969.

Is there any help to understand the code in general?

The GitHub includes a README file that explains the file structure and the comments in the code itself provide the main explanation. Also the Manual and the About page explain a lot about the higher-level structure of the tool as well as chapter 5 of this thesis.

What is R shiny and how is the general structure of the code?

R shiny is a library that provides the functionality to create a web page and publish it online. For more information visit https://shiny.rstudio.com/ for many useful resources and examples of R Shiny applications. In short R Shiny’s structure is divided into the user interface and the server. The server usually is the brain of the application and handles the calculations by receiving inputs and sending outputs to the user interface. The user interface gives the user the possibility to input data and showcases the outputs. One of the strong points of R shiny is that it abstracts the use of front-end technologies like JavaScript and CSS that would otherwise be necessary to build a web application. Also, another important concept of R Shiny is reactivity that is explained in section 5.4 of this thesis.

Is there any help to understand the coded simulation model?

The comments in the code explain most of what is happening even if you are unfamiliar with R or the libraries. Although it would be useful to take a look at the discrete event simulation package R Simmer to understand how the simulation works as it uses different concepts than other more common simulation software's: https://r-simmer.org/. In short, simmer mainly works with trajectories which are paths the patients in the simulation follow. These paths define which resources a patient may take and how long he blocks or seizes these resources. The simulation described in this paper works has another special technique used by me namely sending and receiving messages. Patients can receive and send messages and this fact is used to enable the creation of a schedule that controls how patients flow through the system. Patients that finish at the operating room send a message only to the patients waiting in the waiting list that are suitable as next candidates defined as the select_next_patients() function. This function takes as input the appointment schedule and the current time. Also getting the current time is not as straightforward as it seems. The current day is extracted from another resource that implements a schedule, which returns the days Monday to Friday in a repeating cycle based on days that are 24-time units long. Other little details are explained in the comments of the code.

How can I build and publish an application on my own?

Please follow the tutorial given on https://shiny.rstudio.com/ for a nice and easy tutorial on how to build a simple shiny application. To publish the tool just press the publish button you can find in the upper right-hand corner of the text editor window in RStudio. Then sign up for free and publish your application almost instantly.

Appendix D: Game Script

This is a game script that describes how to play the game to reach certain learning goals. It first presents the learning goal, then gives instructions on how to set up the game or which scenario to use, and last describes how one can interpret the output.

Learning goal 1: Impact of variability

Showcase the negative impact of variability in arrival rates of the patients and service rates of the surgeons on the overall performance of the OR and the wards.

How

Choose the scenario "Impact of variability".

The input settings change automatically to deterministic arrival/service rates for player 1 and variable arrival/service rates for player 2. Also, both players will follow the same block scheduling policy and the seed is set to 5 while the run time is 300 and the warmup period is 100. All these settings can be changed as the player/operator see fit.

Alternatively, go to the operator's page and turn off the variability for one or both players. This immediately shows the effects of variability on the performance. The deterministic environment only works with the block scheduling policy.

What do we see

In the **Deterministic Environment**, we can see that there is no access time, 100% utilization, and no bed shortages. The waiting list graph shows that patients arrive exactly with 5 minutes inter-arrival time and leave immediately from the waiting list. Also, the ward behaves in the same way and stays at a constant level with patients immediately arriving when the others leave the ward. In addition, the distribution of patient types shows the same pattern with no patient type having more patients present in the ward than another. This is a perfect system, in which everything works as planned and is punctual.

In the **Stochastic Environment**, we can see the existence of different access times, different utilizations under 100%, and some bed shortages. The waiting list graphs present a randomly growing waiting list.

Also, the ward occupancy graph shows randomness with different high and low points. The idle time shows different idle times for different patient types. The performance of the system is subject to the randomness of the patient's arrival and service times. The performance compared to the perfect planned system is worse but closer to reality.

Learning goal 2: Block Scheduling vs Open Scheduling

Showcase the differences between the open and block scheduling approach.

The open scheduling policy assigns a waiting patient the first operating room that is free independent of his surgery type based on a First In First Out rule. The block scheduling policy assigns each operating room a surgery type per day. On this day and in this operating room time only the specific patient type can be scheduled.

The open scheduling policy is very close to the actual demand of the patients and hence has very high utilization as well as very low access times for patients. But this policy is usually not realistic to implement into practice. The block scheduling policy is often used in practice but provides longer access time and lower utilization compared to the open policy. Although it levels the occupancy at the ward more.

How

Choose the scenario “Block vs Open Scheduling”.

The input settings change automatically to a block scheduling policy for player 1 and an open scheduling policy for player 2. The seed is set to 5, the run time to 300, and the warmup period to 100. The settings can be changed as the player/operator sees fit.

Alternatively, select the block policy for player 1 and the open policy for player 2 on the player's page. It is possible to try out different runs with different seeds that return different results and again highlight the impact of variability and randomness on the performance but also showcases how these policies behave in different situations.

What do we see

The open scheduling policy has generally lower **access times** that are spread evenly on the three patient types, while the block scheduling policy has higher access times that are not evenly spread across the patient types as is evident from Type B and its high access time (in the scenario example).

Also, the **utilization** of the open policy is higher (almost 100%) while the block scheduling policy does not utilize the operating room fully (in the scenario: OR1 has 82%).

The **waiting list length** showcases the higher flexibility of the open scheduling policy to actual demand, as the waiting list for the block scheduling policy steadily increases, while the open policy holds its much lower waiting list length level.

The **idle time per specialty type** highlights the much higher idle time of the block schedule and the great utilization of the open schedule.

Learning goal 3: Block Scheduling vs Mixed Block Scheduling

Showcase the differences between the block and mixed block scheduling approaches.

The mixed block scheduling policy combines the normal block scheduling policy with the open scheduling policy. In the mixed block scheduling policy, operating room 1 is adapting the open scheduling policy while operating rooms 2 and 3 adopt a block scheduling policy.

The goal is to show that allocating part of the capacity dynamically to demand (in this case operating room 1) yields improvements to the access time and utilization. The mixed block scheduling is presented as a solution that combines the practicality of the block schedule with the better performance measures of the open scheduling policy.

How

Choose the scenario "Block vs Mixed Block Scheduling".

The input settings change automatically to a block scheduling policy for player 1 and a Mixed Block scheduling policy for player 2. The seed is set to 5, the run time to 400, and the warmup period to 100. Additionally, the schedule is changed such that it entails all patient types equally as the operating room 1 row of the schedule is not considered anymore. Also to make a comparison possible both players have the same schedule. The settings can be changed as the player/operator sees fit.

Alternatively, select the block policy for player 1 and the Mixed Block policy for player 2 manually on the player's page. It is possible to try out different runs with different seeds that return different results and again highlight the impact of variability and randomness on the performance but also showcases how these policies behave in different situations.

What do we see

The mixed block scheduling policy has generally lower **access times** that are spread evenly on the three patient types, while the block scheduling policy has higher access times that are not evenly spread across the patient types as is evident from Type C and its high access time (in the scenario example). Also the mixed block scheduling policy has overall lower maximum values for the access time of the patients, highlighting the strength of variable capacity allocation.

Also, the **utilization** of the Mixed block policy is higher while the block scheduling policy has an overall lower utilization.

The **waiting list length** showcases the higher flexibility of the mixed block scheduling policy to hold the waiting list at a lower level than the static block scheduling policy even at the occurrences of spikes in arrivals.

The **idle time per specialty type** provides the opportunity for the player to modify the schedule to eliminate the high idle time for a certain speciality and adjust the schedule more to actual demand.

Learning goal 4: Open vs Mixed Block Scheduling

Showcase the differences between the open and mixed block scheduling approach.

The goal is to show that even though the open scheduling policy is better in performance, it is not much better than the mixed block scheduling approach and combines the practicality of the block scheduling approach with the improvements of the open scheduling approach in terms of performance due to the flexibly allocated capacity.

How

Choose the scenario "Open vs Mixed Block Scheduling".

The input settings change automatically to an open scheduling policy for player 1 and a mixed block scheduling policy for player 2. The seed is set to 5, the run time to 400, and the warmup period to 100. Additionally, the schedule is changed such that it entails all patient types equally as the operating room 1 row of the schedule is not considered anymore. The settings can be changed as the player/operator sees fit.

Alternatively, select the block policy for player 1 and the Mixed Block policy for player 2 manually on the player's page. It is possible to try out different runs with different seeds that return different results and again highlight the impact of variability and randomness on the performance but also showcases how these policies behave in different situations.

What do we see

The open scheduling policy has generally lower **access times**. The maximum waiting time, as well as the median for the mixed policy, is higher than the open policy. But the performance differences are not significant.

Nevertheless, the **utilization** of the open policy is higher while the mixed block scheduling policy has an overall lower utilization.

The **waiting list length** shows that the open scheduling policy deals better with fluctuations in patients arrivals. In the scenario example, this is apparent due to the strong increase in the waiting list length of the mixed policy from time unit 250 onwards, which the open policy can adapt to and maintain a constant and lower level in the waiting list.

The **ward occupancy** shows that the mixed block schedule can level the occupancy of the ward to reduce the number of exceeded capacity. In the scenario example, the bed occupancy of the open schedule is generally higher and not leveled compared to the mixed block schedule.

The **idle time per specialty type** highlights that the mixed policy has much higher idle time than the open scheduling policy.

Learning goal 5: Impact on the ward

Showcase that a different policy/schedule can have significant changes on the ward.

The goal is to show that the schedule of the operating room does have an impact not only on the operating room but also on the bed occupancy of the ward.

How

Choose the scenario “Impact on the ward”.

The input settings change automatically to block policies for both players, the seed is set to 5, the run time to 300, and the warmup period to 100. Additionally, the schedules are changed such that each day entails only one patient type except for the first day. For operating room 1, the sequence is Type B, C, A. And for operating room 2, the sequence is Type A, B, C.

Alternatively, manual manipulation of the schedule given a block scheduling policy can highlight the impact the different schedules have on the ward. Ideally, many of the same patient types are scheduled on the same day to showcase the distribution of different types in the output.

What do we see

The **Bed occupancy per specialty type** shows the impact the two schedules have on the ward. The schedules have a visible impact on the pattern of bed occupancy. The first schedule follows the sequence Type B, C, A, and also the ward shows with some time delay the same pattern in its occupation. The bed occupancy of the ward for the second player also follows a clear pattern defined by its schedule. The differences between these performances and their clear links to the schedule, showcase that the schedule has an impact on the distribution of patient types on the ward and that one could predict the bed occupancy based on the schedule.