Eindhoven University of Technology

MASTER

Patient appointment coordination using a multi agent system : a case study at the AMC

Blaauw, R.

Award date:
2009

Link to publication

Disclaimer
This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain

Patient Appointment Coordination

Using a Multi Agent System

A case study at the AMC

Robin Blaauw

July 7, 2009

Technische Universiteit Eindhoven
Department Mathematics and Computer Science

Study: Business Information Systems
Research Group: Information Systems

Supervisor: Prof.dr.ir. J.A. La Poutré, CWI & TU/e
Daily supervisor: Dr. ir. I.B. Vermeulen, CWI
Advisor: Dr. ir. S.G. Elkhuizen, AMC
Advisor: Prof. dr. P. J.M. Bakker , AMC.

Centrum Wiskunde & Informatica
Kruislaan 413,
1098 SJ Amsterdam

Academisch Medisch Centrum,
Department: Kwaliteit en Proces Innovatie (KPI)
Meibergdreef 9,
1105 AZ Amsterdam

Abstract

Today's hospitals face the challenge of ever increasing demand for better patient service and resource capacity. This master thesis specifically aims to a certain aspect of patient service which appeared to be one of the most desired ones namely combining appointments on the same day. Many patients that arrive at the hospital have to undertake a series of test all at different departments on a different day. Scheduling these different appointments on the same day seems to be rather complicated, till today most of these departments act autonomously with closed agendas, practically making it very hard to coordinate these appointments. The reason for this appears to be the local specialism knowledge and the continuous pressure for efficiency due to shortage of resource capacity.

In our work we will discuss and analyze the departments at one of the largest hospitals in the Netherlands focusing specifically on combination appointments. Interviews and observations at several departments will give us some insights in possible bottlenecks regarding the coordination of appointments in their scheduling process.
We will make a first approach on building a simulation application that uses real hospital data to simulate the coordination of combination appointments using a multi agent system. Multi agents are very suited to deal with this kind of coordination problems and fit more in the organizational structure of today's hospitals than the currently used non dynamic scheduling systems. A simulation with real hospital data with a timeframe of a year will give us some insights on how multi agents would perform in a real life situation and if the departments can maintain their efficiency.

Acknowledgements

I would like to thank especially Dr.ir. I.B. Vermeulen, my daily supervisor, for giving me the guidance, support and motivation to carry out my master thesis. The KPI department at the AMC hospital for their support and the daily cup of tea that kept me going. Prof.dr.ir. J.A. La Poutré, my supervisor, who proposed this study to me and supported me in writing my master thesis. Prof.dr. P. J.M. Bakker and Dr. ir. S.G. Elkhuizen, my advisors, for their feedback and support during studies at the AMC hospital.

Table of content

1 Introduction

1.1 Introduction

This master thesis has been carried out at the AMC hospital one of the largest hospitals in the Netherlands. Patient service is becoming more important these days; one of the most desired patient services is the possibility to have all their appointments on the same day. With the current scheduling process, coordinating appointments is hardly done because the current scheduling process is not suited for appointment coordination. The departments all have their own agendas which in most cases no other departments have access to, making appointment coordination difficult.

1.2 Objective and research question

Providing better service for the patient is a large opportunity. Not only will it be possible to schedule several appointments on the same day but also taking into account other kinds of preferences from the patient. The biggest challenge however is on how to implement this kind of service without decreasing the department's performance and also take in to account the department preferences.

Therefore we define our research question:
"Is it possible to implement a scheduling system at the AMC hospital that will enable us to automatically coordinate different appointments of a patient on the same day without any impact on the departments' performance? "

Automated scheduling, scheduling at the outpatient department without direct approval of the diagnostic departments, will be necessary in order to enable the coordination of appointments. By giving the departments control over the automated scheduling we hope it will be possible to maintain the same performance of the department.

The goal of the project is to analyze the current practice at the AMC for scheduling combination appointments. A case study at several outpatient departments as well as auxiliary departments like Radiology and Endoscopy is conducted in order to analyze their scheduling method and to classify their constraints and preferences. Patient's attributes that have influence on scheduling need to be specified. By means of a multi agent system we will model a system that will: take into account patient preferences, automate scheduling and finally will enable the coordination of combination appointments. Finally we will perform some simulations using real hospital data from the AMC to show how a multi agent system could perform within a certain environment.

1.3 Related work

This master thesis is mainly based on previous work[1,2] both written by Ivan Vermeulen, my daily supervisor. These papers [1,2] use multi agents to schedule and coordinate appointments for patients within a hospital.

"Multi-agent Pareto appointment exchanging in hospital patient scheduling" [1] discusses a distributed exchange mechanism where agents exchange appointments with a pareto effect, meaning exchanging appointments by every agent involved will benefit from this exchange. This

is possible since patients have multiple appointments in their treatment plan and only the last appointment determines the total duration of the treatment. He conducted several simulation experiments showing that this dynamic way of scheduling can perform very well and is more suited in a hospital environment.

The second paper “Adaptive Resource Allocation for Efficient Patient Scheduling ” also uses a hospital setting in combination with multi agents to schedule patients appointments . The main focus in this paper[2] is the flexible allocation of resource capacity per patient group. As we will also see in this master thesis classifying patients to the right patient group and the flexibility in resource allocation per patient group is a key element in resource efficiency.

1.4 Outline

Chapter 2: This chapter describes the current scheduling process of diagnostic departments and outpatient departments.

Chapter 3: We discuss the theoretical background of combination appointments. Efficiency and patient classification are key elements to enable the coordination of appointments. We will introduce a multi agent model for the coordination task.

Chapter 4: A simulation application has been implemented. This application will simulate the multi agent system we have defined in chapter 3. Several simulations with different settings using a First Come First Serve scheduling algorithm have been conducted.

Chapter 5: The results of the simulation are analyzed and compared to the original hospital data.

Chapter 6: During this study a lot of knowledge was gained about the current scheduling process at the AMC. Based on the simulation results and the analysis of the scheduling process we make recommendations to hospital that should be considered when implementing scheduling software.

2 Scheduling in the hospital in the present situation

Scheduling patients is a huge task for a hospital. Every day, depending on the size of the hospital, several hundred appointments are made. These scheduling activities take a lot of time, therefore every department in the hospital has their own specialized schedulers accepting appointments, communicating with doctors and other departments. The quality of their work is important to the hospital, since inefficient scheduling impacts directly on waiting times, patient satisfaction, staff utility and finally the overall profit of the hospital.
In order to make improvements to the scheduling process we analyzed the current scheduling process. For this study we made observations of scheduling processes at the AMC during several weeks. These findings will be discussed in this chapter and will show why we think that especially combining appointments could be a great opportunity to the hospital.
For our study we have chosen surgery and urology as outpatient departments and for the auxiliary departments we selected long function, radiology, endoscopy and anesthesiology.
The reason we selected surgery and urology is that these departments are making appointments at several auxiliary departments on a regular basis, which is exactly what we are interested in.

2.1 The outpatient Department

The outpatient department is often the starting point of every patient coming to the hospital. At the outpatient department new patients will attend their first consult with the doctor. During a week time we interviewed and observed the nursing staff off the outpatient department surgery and urology.
The nursing staff is responsible for all activities that are needed to make an appointment for the patient. After the patient has been seen by the doctor at the outpatient department the doctor will hand over the patients' "status", "form with current appointments" and other forms needed to perform certain scans or procedures (tests), to the nursing staff. The nursing staff has the responsibility to check whether these forms were filled out correctly and ask the respective doctor to complete the needed forms if required.

After the doctor finished the consult he hands over these files in person and will also communicate to the nurse when the next consult should be made. Once the nurse has all the needed forms they will try to schedule the needed appointments together with the patient that is present.

2.1.1 Scheduling a patient

Scheduling a patient starts at the office of the nursing staff. The nurse has all necessary forms that where provided by the doctor and knows when the next consult should be planned.
In most cases planning the next consult of the patient is not a problem because they will be scheduled far in the future. However it can happen that the schedule of the doctor is filled or the patient needs to been seen on short time due to severity of the disease. The nursing staff can't schedule the patient within the requested timeframe instructed by the doctor, in this case there are several options:

- Scheduling a patient within a timeslot not for this kind of patients. They often use timeslots for new patients (NP) and **controlling patients (CP)** in the doctor's schedule**.** New patients have an initially high priority since they have to be diagnosed as quickly as possible in order to classify a patient as urgent or not. If no other controlling patient's

slots are available anymore “new patient timeslots” that are not used within three days can be scheduled for “controlling patients” if they have a higher urgency.

- The doctor can decide to do an overbooking. Overbooking always has to be confirmed by the doctor himself. Overbooking a patient means planning two patients on the same timeslot. This procedure is a last resort procedure which will be avoided as much as possible. Overbooking a patient means overtime for the doctor and a longer waiting time for all the other patients scheduled on that day. During our observations overbooking was common practice resulting in longer waiting times and unsatisfied patients.
- Scheduling the patient after the instructed timeframe. This is only possible with approval of the doctor and indicates a severe shortage of available resources and will lead to overtime for doctors and nurses.

Once the patients “controlling appointment” has been planned, the procedures and tests requested by the doctor have to be scheduled. Depending on the kind of test, several actions have to be performed by the nursing staff. There are three ways how an appointment for these tests can be scheduled:

- For certain tests it is possible to do a simple phone call requesting the test. This is the case for most echo’s. A scheduler defines the timeframe and gets an appointment from the scheduling software within this timeframe. Since the patient is present when this phone call is conducted the patient can be asked directly whether this appointments is convenient. This way of planning an appointment can only be done with resources that don’t require a pre assessment by the responsible staff conducting the tests.
- MRI, CT and PET-scans for example can never be planned directly. These scans are the scare resources of the hospital. For this kind of tests forms have to be filled in by the requesting doctor. On this form the date of the planned controlling consult is written by hand. These forms are being sent to the department where a medical doctor first has to check these form on completeness and will try to classify the patient. This classification can only be done by doctors with the right expertise. The patient is instructed by the outpatient department to wait for a phone call of the auxiliary department where he will be informed when his test will be conducted. Usually the department will try to plan this test before the controlling appointment which has been manually written on the request form. In some cases however it is possible that the test cannot be planed before this scheduled controlling appointment. In this case the patients have to contact the outpatient department themselves in orders to reschedule the controlling appointment.
- Planning an appointment directly within the schedule of the requested department. For some procedures certain arrangements have been made with other departments. The outpatient department is allowed for a certain kind of test to schedule their patients at that department. The outpatient department surgery has such an arrangement with the endoscopy department. For some endoscopic procedures the outpatient department surgery and urology are allowed to plan in the local schedule of endoscopy.

2.2 Auxiliary departments

Most combination appointments are made with the auxiliary departments. Auxiliary departments are al diagnostic departments that provide tests and expertise that give more insight to the disease or complaints of the patient. After the first consult one or several appointments are made at these auxiliary departments. We have chosen Radiology,

Anesthesiology, Lung function and Endoscopy departments to analyze the current work process especially regarding combination appointments. The following sections will describe the findings of each department.

2.2.1 Lung Function

The lung function department performs breathing tests to see how well the lungs work. These tests will show how much air the lungs can hold, how quickly one can move air in and out of the lungs, and how well the lungs add oxygen and remove carbon dioxide from the blood. The results are used to diagnose lung diseases and give insights into the severity of the lung problems.

One of the things we observed, was that the lung function department at the AMC was not crowded at all, it turned out that the lung function department doesn't have any resource capacity problems like most other departments.

Several interviews were made with Erica Dijkers, head nurse of the lung function department, to give us some insights in the scheduling process of the lung function department.

As mentioned before this department doesn't have any resource capacity problems and the patients are being sent by the outpatient departments directly without any prior patient planning. If an outpatient department wants to schedule an appointment it has to digitally fill in a form, only if the whole form is filled in correctly an appointment can be scheduled without any interaction of the lung function department. About 80% of the appointments at the lung function are scheduled this way. The schedule of the lung function department is completely open, anyone can make appointments to their needs. The only constraint is the completeness of the digital form.

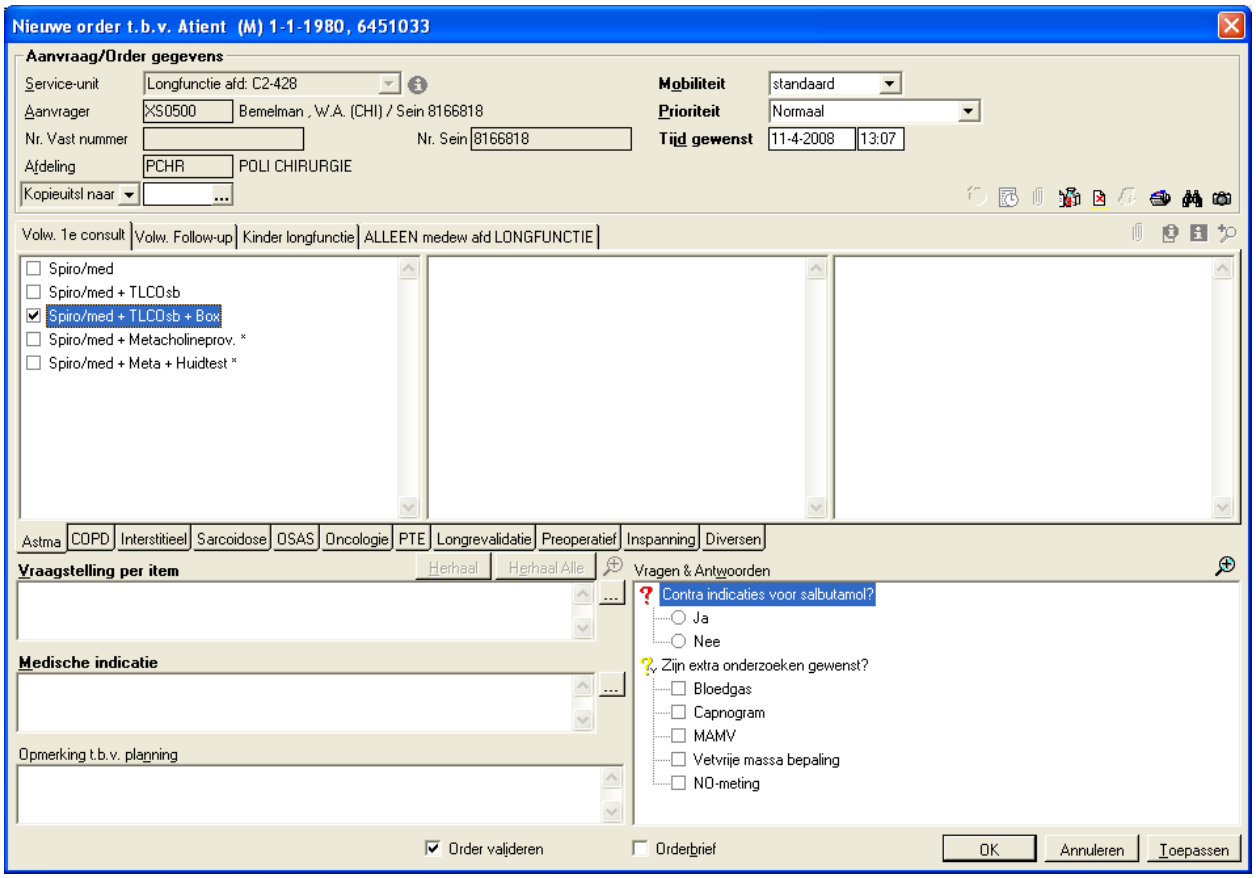


Figure 2.2-1 Digital Form of the lung function department that enables direct scheduling by other departments without interaction of the lung function department.

Once this form (Figure 2.2-1) has been fully completed by the consulting medical doctor a record(Figure2. 2-2) named “to be scheduled” is generated by the scheduling software which indicates that the patient can be scheduled. The patient will be send to the nursing staff, the nursing staff can then select this “to be scheduled” record in order to finally schedule the appointment.

As we can see in Figure 2.2-1 this record represents a type of procedure in our case it is “Spiro/med+TLCOsb+Box”, this is the result of the questions that were answered in the digital form by medical doctor. This procedure represents a combination of appointments which is indicated by “Combi” checkbox which is checked in our case. The lung function department has a very special way to coordinate the schedule of their medical doctors and their resources. At the lung function every appointment is a combination appointment which represents a combination between the schedule of the medical doctor and the resource necessary to conduct the procedure.

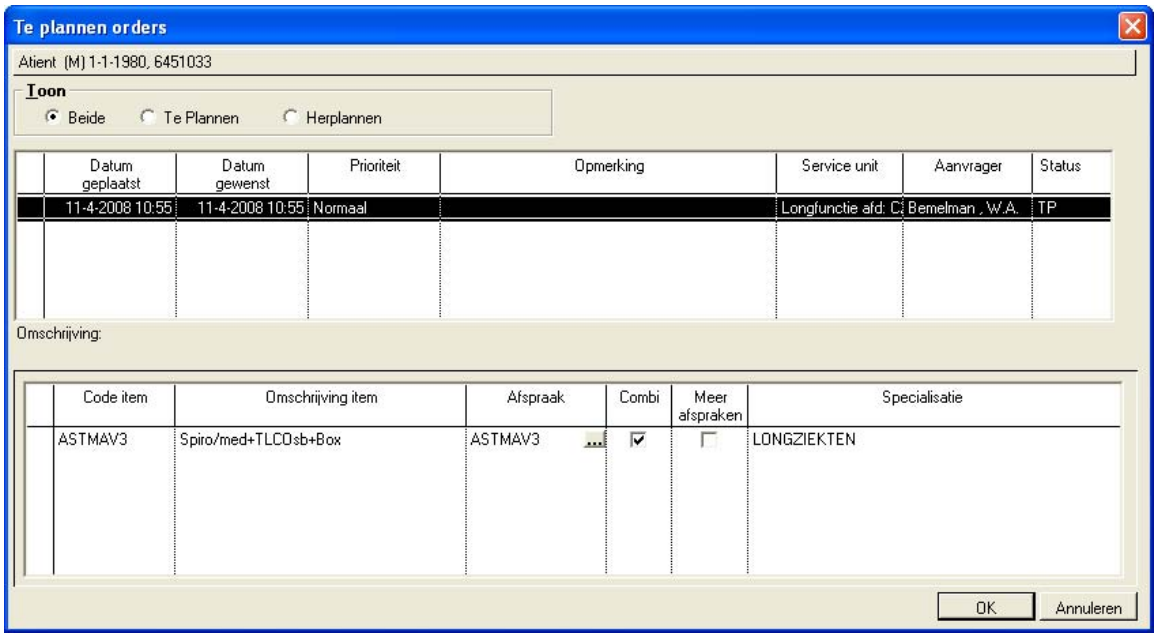


Figure 2.2-2 "To be scheduled" records for a patient, only if this record is shown a patient can be scheduled directly at the longfunction department.

This kind of combination is a completley different kind of combination appointments which we will discuss in the next chapters where a combination consists of apointments between different departements in a sequential matter rather than two appointments in parralel at the same time. Figure 2-3 shows the scheduling of this combiantion appointment, the upper appointment is the resourse and the lower appointment is the medical doctor.

Normally a department wouldn't be able to schedule a resource room and a resouce medical doctor as a combination. This is verry convenient for the medical doctor because their agenda can be filled nicely but it is not efficient for the schedule of the resources. It seems that the lung function has enough resource capacity to that allows them to schedule this way which results in inefficient and not fully utilised resources. Analyzing the agendas at the lungfunction department it shows us a nice fully filled agenda of the medical doctor and a not so nice planned agenda of the resources with appointments scheduled at any time of the day with a lot of unused timeslots. The most valuable resources at the longfunction appears to be the staff itself, the rooms and instruments don't seem to be to costly.

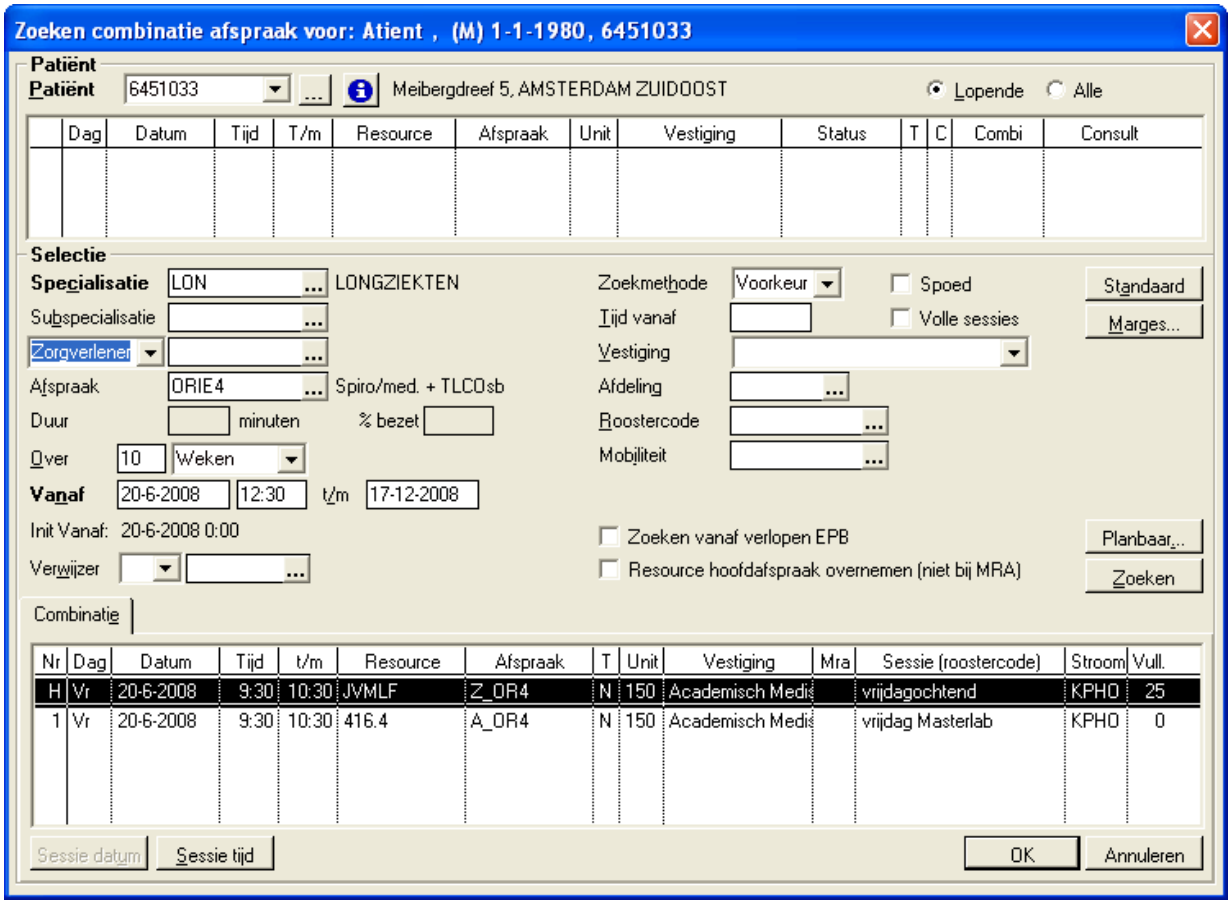


Figure 2.2-3 Scheduling a combination appointment at the lungfunction. A combination between a resource and a specialst.

The lung function department, together with the blood laboratory, are the only departments that are using this kind of open agenda. For our experiment the lung function department provides an ideal situation that can be seamlessly integrated to our experiment model described in the next chapters without any adjustments to their current situation.

2.2.2 Anesthesiology

Anesthesiologists drug the patient during surgery. Only with the right combination of drugs and dosages, a safe procedure can be performed. Prior to the surgery the patient has to be interviewed to determine the overall health of the patient, sometimes additional medical tests like an ECG or Blood test are required. Most important factors that are of interest to the Anesthesiologists are: weight, age, medical history and ongoing medications.

Anesthesiology is one of the most interesting departments for our study since it is always part of a combination appointment. In this situation a combination appointment is defined as several sequential appointments that are required for a single patient at different departments. Currently anesthesiology has completely revoked other department's rights to schedule appointments. The revoked access apparently has something to do with inefficient scheduling of the other departments in the past and possible lack of expertise. The Team Innovation and Processmanagement(TIP), now known as Quality and Process Innovation (Dutch = Kwaliteit en Process Innovatie KPI) at the AMC, had performed a former project on this department.

The anesthesiology had some serious resource capacity problems and apparently they were not able to cope with these problems by their own. Therefore the TIP department was asked to perform investigations at the department of anesthesiology and to quickly solve their capacity problems. One of the main issues was that the other departments didn't have the right knowledge, expertise or were not precise enough in scheduling the patient for the right timeslot. This practice resulted in patients not being scheduled at the right medical doctor and the durations of the consults where not representing the actual time needed.

To quickly solve these problems at the department of anesthesiology department, TIP advised the anesthesiology to allow local scheduling only, other departments were not allowed to schedule patients at the anesthesiology by their own anymore. The patients had to make the appointments themselves at the anesthesiology front desk. Furthermore TIP introduced the ASA physical classification procedure to classify patients for a certain timeslot. Depending on whether the patient is ASA class I till IV more or less time will be reserved for the patient. This ASA classification is based on a questionnaire the patient has to fill out. The nursing staff then will apply the ASA classification based on this questionnaire.

The next section will describe our observations that were made at the anesthesiology scheduling desk during several observation sessions. It will give us some insights and better understanding how the department is currently scheduling the patients.

Observations at the department of anesthesiology

The main goal of the observations was to determine on how patients are classified by the schedulers. Every patient has to fill in a questionnaire before making an appointment. This questionnaire can be send by mail to the anesthesiology or in most cases the patient can be send to the anesthesiology department right away to make an appointment. Once the patient is at the anesthesiology the patient is classified by several factors:

- Age of the patient
- Mobility of the patient
- Distance from home address to the AMC

- Urgency
- Kind of operative procedure and the corresponding waiting list for that procedure or estimated date and time of the operative procedure.
- Complexity of procedure

Based on these factors patient can be scheduled into one of the following timeslots:

- B20, normal timeslot with a duration of 20 minutes
- B30, normal timeslot with a duration of 30 minutes
- B20S, urgent timeslot that comes available one week in advance, duration 20 minutes
- B30S, urgent timeslot that comes available one week in advance, duration 30 minutes
- XSPOED, urgent timeslot comes available 2 days in advance, duration 20 minutes
- INLOOP, reserved timeslot for patients, available 10 hours in advance:
    - Bad mobility
    - High Age
    - Home address is a long distance from the AMC
    - Sometimes misused for making combination appointments

Medical doctors available:

- Junior assistant, mostly 20 min. normal consults
- Senior assistant, all kind of procedures
- Chief of medical staff, difficult procedures, children, patients with high age, patients that have a uncertain ASA classification

A full agenda of the anesthesiology is shown in attachment 3. The agenda shows all the different timeslots and the three medical doctors: young, senior assistant and the chief of medical staff.

An allocated timeslot with the duration of 20 or 30 minutes is based on the questionnaire. This questionnaire is again used by the anesthesiologist to determine the patients ASA physical status I till IV. The questionnaire alone is not enough to determine the ASA status, therefore scheduling the patient on the 20 or 30 min timeslots is an estimation of the patients ASA status. Estimated patients of ASA status I and II are scheduled for the 20 min timeslots and ASA status III till IV are scheduled for the 30 min. timeslots.

The urgency of the patient is determined by the OR (Operative Room) request. This request is either digitally available at the “AMC zorg desktop” or the patient has a paper copy of this request. On the request an estimated or a scheduled date is provided. In the case there is no OR request the appointment cannot be scheduled, the requesting department has to be contacted to get this missing planned OR date. For the elective procedures the scheduler has to know about the status of the waiting list for the specific procedure. The status of the waiting list is available for the day care centre but not for the clinical OR. The scheduler has to have knowledge about the estimated waiting time.

Once the patient has been classified for a 20 min or 30 min appointment and the timeframe where the patient has to be scheduled is known an appointment will be made.

If the patient is more than 60 years of age they will have to do an ECG and take a blood test. This can be done either right after the appointment is made or on the date of the appointment. Most

patients with ASA status III or IV are older people who will be scheduled in the 30 min timeslots instead of the 20 minutes ones. Timeslots are scheduled in the morning so they can do an ECG and Blood test prior to the consult, the results will be available to the anesthesiologist. These tests can be done right away there is no appointment needed, the patient will get forms for the ECG and the blood test to hand in at the LAB.

For every appointment that is made the scheduler looks if the patients has other appointments at the hospital within the same timeframe to make a possible combination appointment. In other cases a combination appointment is necessary with the requesting department. The requesting department has to gather information and do some test that are needed for the consult with the anesthesiologist. In order to make these combination appointments possible, sometimes reserved urgent timeslots like INLOOP are misused for this purpose. The appointment at the requesting department is made before the patient makes an appointment at the anesthesiology.

Patients that have had a consult at the anesthesiology within the past 6 month and no changes regarding their health occurred they will only have to fill in an “extended clearance”. The medical doctor then will decide if a consult for this patient is necessary.

Combination appointments

The anesthesiology is often part of whole combination of appointments at different departments and medical doctor. Especially patients classified at ASA 3 or 4 often have to do an additional ECG and a blood test. Then again after these tests, the patient has to be seen by the medical doctor before the OR procedures can be performed. So a patient can have for example four appointments which could theoretically take place on the same date. In practice however it is unlikely that you will get these appointments on the same day since radiology in our example has long waiting list and there is no space for any special request. The ECG and Blood test are a different story they can be done at the same time as the appointment at the anesthesiology since there is no waiting list there. Figur 2.2-4 shows the different appointments of such a patient.

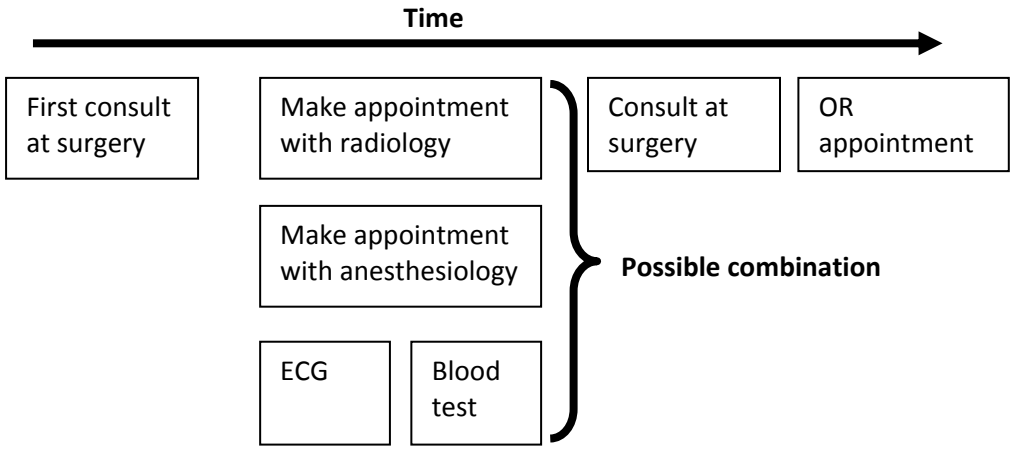


Figure 2.2-4 *Appointments of a typical ASA 3 or 4 classified patient*

Current practice on combination appointment is that either the anesthesiology or the radiology in our example calls the other department by phone to see if it is possible to schedule an appointment on a single day. One can imagine that doing so by human interaction regarding the schedule of the anesthesiology, the schedule of the radiology and the schedule of the patient self, it becomes extremely difficult to find an appropriate appointment that is acceptable by all parties.
As a result the patient isn't asked at all in most cases but these appointments are made without regarding the patients personal agenda, the patient will be informed by a letter. In case the patient is not able to come at this date it has to inform all departments them self and schedule every single appointment themselves again.

Allow remote scheduling again
Our aim is to open the agenda of the anesthesiology again. Although we know they had some bad experience in the past, it is the only way to enable a more coordinated scheduling of combination appointment and thus providing a more service to the patient. Conversations with the head nurse of anesthesiology, responsible for all scheduling taking place, made it clear that it will be hard to convince the staff in taking such an action. The TIP department that closed their agenda is now recommending opening it again; this could be confusing for the staff. However due to the new ASA classification procedure it has become easier to classify patients, it should be possible to let other departments classify patients using this classification questionnaire. Together with the head nurse we discussed the possibility to install small terminals at the departments. Patients that for example had a consult at the surgery could immediately fill in the questionnaire digitally at such a terminal which will classify the patient for a certain ASA physical status without the need to personally go to the anesthesiology and schedule the appointment there. Another advantage would be that this will allow us to perform a coordinated combination appointment since the patient can be scheduled directly. This is one of the main goals of this paper and the next chapters we will discuss in detail what opportunities there are and how such a solution would look like. The head nurse was positive for this idea and would be willing to cooperate in such a project.

2.2.3 Endoscopy
Endoscopy is a minimal invasive diagnostic medical procedure that is performed on a patient to see inside of the patient's body and organs by using a small tube. This tube can be a rigid or flexible and provides visual images, it can also be used to take samples or remove foreign objects from inside the body. During such a procedure the patient may receive a conscious sedation so they are not aware of any discomfort that may take place. Most endoscopic procedures are rather painless.

At the time observations and interviews were made, the endoscopy had very long waiting lists reaching up to 15 weeks. Just like the anesthesiology department, endoscopy had withdrawn remote scheduling by other departments in their agenda, only local scheduling by their own personnel was allowed. It seems that taking back control over scheduling seems to be the first response whenever a department faces severe capacity problems. However capacity problems where so severe at the endoscopy department, all kinds of lobbying were taking place by the medical doctors to get their patient seen first. As a counter measure only two assigned schedulers were allowed to do the scheduling to prevent any unfair situation. Another

consequence that resulted from the long waiting lists was that the endoscopy had introduced three different urgent classification types which is a mix of English and Dutch terms “dag urgent” (day urgent), “spoed” (urgent) and “urgent” (also urgent but less urgent then “spoed”). These are all urgent patients but since there is too less capacity even urgent patients have been sub classified in these three urgent types. Of course this is an undesirable situation and this extra classification is not really solving the problem. Finally all requests for an endoscopic procedure have to be confirmed first by a medical doctor and then again by the staff medical doctor. This procedure is necessary to prevent wasting any scare capacity.

The KPI (formerly TIP) department had been performing a project on the endoscopy department to investigate their problems and to provide solutions. By the end of my observations these new ideas had been presented to the department. The main recommendations where:

- Central scheduling, all scheduling of staff, patients and rooms should be done central desk only to get better control.
- Only allowing patients form within a certain region nearby the Hospital, other patients have to go to a different hospital.
- Reducing authority of the schedulers to a limited extend needed to perform their duty.
- Requests that are incomplete will be rejected.

There were a lot more measures taken, but they relate to the daily working routine and don’t affect the scheduling part.

Some special arrangements are made with surgery, urology and lung function departments. Surgery has a on every Thursday morning. The surgery department is allowed to schedule and perform the procedures themselves during this session. Urology has two permanent rooms available, just like surgery they schedule and perform the procedures themselves. Lung function also has a room available to perform their procedures, however scheduling the appointment is done by the endoscopy department.

Endoscopy knows many different procedures; they all differentiate in difficulty, complexity, duration and needed instruments to perform the procedure. Each room has their own special instruments that is applicable to a certain set of procedures. Some procedures have longer waiting lists because they require scare resources like medical doctor with certain expertise and available instruments that are needed for the procedure. Other procedures however are rather simple and don’t require scare resources therefore their waiting lists are shorter and their applications for such a procedure doesn’t necessary require such a strict verification by the medical doctor and the staff medical doctor.

Endoscopic procedures that are relatively easy are: SIGMO, OGD, OES they together form the category MDL (Dutch: Maag Darm Lever; English: Stomach, Intestine, Liver). In the near future COLO(Endoscopic procedure of the large intestine) will also be added to this category. This MDL category can theoretically be scheduled without further confirmation from a medical doctor at the endoscopy department.

Because we are interested whether it would be possible to let other outpatient departments schedule these MDL procedures we made some interviews with the responsible staff at the endoscopy. If we would manage to let outpatient department schedule these procedures it would clear the way for us implementing coordinated combination appointments at the outpatient department together with the patient without the patient having to make these

appointments themselves. Apparently the departments really like to maintain control over their agenda, the only way they would allow the outpatient departments to schedule appointments, would be if they could be assured that every appointment made complies to the rules and prerequisites they define.

The endoscopic procedures require a certain amount of expertise in order to be able to assign the right patient to the right procedure, this is why the staff of the endoscopic department is not convinced that other departments would be able to schedule appointments without their support. However there are some doctors especially from other departments who think they should be capable of a correct patient classification for the MDL procedures.

Other procedures at the endoscopic department are unfortunately not so easy to perform and schedule. These procedures will always require a medical doctor expertise before any appointment can be made. These procedures are ERCP(endoscopic procedure of the biliary ducts), EUS(ultrasound imaging of the stomach, gullet, pancreas and anus), PROCTO(proctoscopic procedure of the anus), PEG(a probe that is attached to the stomach which can be used to apply medicine and nutrition). For every single appointment request the medical doctor has to estimate:

- Whether it is the right procedure
- Total duration of the procedure
- The urgency of the patient

Currently complex procedures require the request form to be fully completed. Only with a fully completed request form and the feedback with the medical doctor on duty, he will fill in the necessary documents for such a procedure which then again have to be confirmed by the head medical doctor. After it has been confirmed by the head medical doctor it can be scheduled by the scheduler of the endoscopic department. These forms are all on paper, in the near future they would like to switch to a digitalized version of these forms.

We highly support digitalization of these forms. It brings many benefits and opportunities for us. Not only will it be much easier to analyze these digital forms for any future research, it will also bring guided automated classification of the patient one step closer. Guided automated classification is a key element of automates scheduling of combination appointments as we will see later on.

2.2.4 Radiology

Radiology is one of the most important diagnostic and therefore the most referred department at the hospital. Radiology provides a whole range of diagnostic imaging techniques like X-Ray, Echo, MRI and CT scans. These scans again can be divided in a whole range of scans applicable to a certain part of the body. Nearly every department depends on the diagnostic services that radiology provides. This is why the radiology is so interesting for our study; radiology has the highest probability in being part of a combination appointment with any other department.

It was interesting to observe that also this department has closed their agenda to other departments due to the same reasons: not enough resource capacity and the believe that other departments are not able to correctly classify the patient for the correct diagnostic procedure.

The radiology has three to four schedulers permanently accepting phone calls from other departments trying to schedule an appointment.
The kind of diagnostic procedure differentiates a lot from each other. The echo's and X-ray's are rather simple procedures that don't require expensive and scare resources like the CT and MRI scanners. Still every appointment has to be made by the schedulers at the radiology. The most common practice is that the radiology is called for an appointment; if possible an appointment will be made immediately. For some requests the radiologist first has to determine the urgency of the patient and he will also have to check whether the requested scan is correct, before the patient can be scheduled. After the request has been accepted by the radiologist the patient will be informed by the radiology, he will receive a letter or will be contacted by telephone within short notice. The radiology always requires completed request forms for every diagnostic request for all MRI and CT scan. The radiologist will examine the request and will decide if the requested scan matches the patient's complaints and the hypothesis stated by the requesting medical doctor.

Type of scan	**Request from**	**Scheduling procedure**
X-ray	General practitioner	Can make an appointment for the next day by phone.
	Clinic	Can just walk in for an x-ray.
	Out patient department	Can just walk in for an x-ray.
Echo	Out patient department	The outpatient department makes an appointment by phone for most echo's. Some echo's require a radiologist expertise and therefore a form needs to be submitted.
MRI	Out patient department	Scheduling only possible by the submission of a request form after which the medical doctor will schedule the patient
CT	Out patient department	Scheduling only possible by the submission of a request form after which the medical doctor will schedule the patient

Figure 2.2-5 *Scans and their scheduling procedures at the Radiology*

Radiologists have specific knowledge and they will use their expertise and experience to judge whether the request is valid. This form holds information about:

1) Relevant clinical background from the patient
2) Hypothesis
3) The requested scan

Only if all of these items match, meaning that based on the relevant clinical background and the hypothesis the requested scan is the correct scan for this scenario, the radiologist will allow this scan to be scheduled. If this is not the case the radiologist will either contact the medical doctor for further details or he will send the request back with comments why the request wasn't accepted.

The following items are also determined by the radiologists based on the information on the request form:

- The kind of scan, specifies exactly what needs to be scanned in order to get a good picture of the problem.
- Which scanner will be used, the four scanners are not the same some are better for certain scans.
- The estimated duration of the scan,
- If the patient needs to come early

For every kind of scan average durations of such an procedure is known, however there are factors that can influence the normal durations of the procedure:

- A doctor has to be present during the procedure. In case of uncertainty the doctor could decide to perform some additional scans.
- Extra series of scans can be needed in case of uncertainty.
- Whether the patient has to be injected with contrast media prior to the scan.
- If the patient is afraid, it takes more time because the patient is not allowed to move during a scan.
- Children can be afraid of small spaces, if this is the case the will have to be sedated to remain calm during the scans.
- For certain scans different spoiles are needed, they have to be changed this takes time.
- If the patient is moving, it will take more time to complete the scans.

All these items are important for scheduling the patient, the whole efficiency of the radiology departments depends on whether the scan is successful (provides the information needed, no extra scans needed) and the durations of the scan has been estimated correctly.

As the radiology already struggles with all kind of resource problems and has put in place a system that in their point of view will most likely bring the best possible efficiency it may be difficult to propose any changes to their way of scheduling. Opening up their agenda to make it possible for other departments to schedule appointments directly and thus enabling combination appointment will be quite difficult.

For the echo's and X-ray's it would be possible without much effort. A digital form that can check on consistency of the request would suffice. However for MRI and CT a normal digital form will not be sufficient. We mustn't forget that if we were to open up the agenda MRI and CT to other departments to directly schedule an appointment, we would have to transfer the knowledge and experience of the radiologist to such an digital form request. Based on interviews with the radiologists this is nearly impossible, on the other end, the medical doctors from the outpatient department think they would very well be able to request the right scan with some guidance by a digital form. In the next chapter "Automated scheduling at the MRI"

we will see what it would take to enable scheduling by other departments without direct intervention of the radiology.

2.3 Evaluation of the current combination appointment process

As we can see in the way appointments are being scheduled at other departments it seems to be quite difficult to combine appointments in order to give better service to the patient in terms of faster treatment and less traveling to the hospital. In practice, they will try to combine appointments for patients that have a long traveling distance where ever it is possible.
So why is it not possible to make combination appointments in the way appointments are being scheduled? Let's take a look at the first kind of appointment scheduling. Scheduling an appointment by telephone has the advantage that the patient can agree on this appointment and the patient isn't left in uncertainty. However if a patient needs more than one test at different departments it becomes more difficult to coordinate. Since all involved parties are not aware of each other's schedules, scheduling a combination appointment becomes rather complex. In practice the test from which it is known to have the longest waiting time is scheduled first and after that the second test is scheduled with the request to schedule it on the same day as the other test. This way of scheduling combination appointments will not always result in actually being able to schedule these test on the same day.
Tests that need forms to be handed in and experts to classify the patient are even more difficult to combine with other appointments. In practice combination appointments that require a prior assessment of the department where the test is being performed, they will try to communicate with the other department for the second appointment. For example a patient needs a MRI or a CT scan before a planned operation. The radiology department will contact the anesthesiology department to schedule the test and the appointment at the anesthesiology. The request for combining these appointments is manually written on the request form, there are no standard procedures for these situations.

Currently combination appointments are not scheduled in the scheduling software as a combination appointment. The software is not aware of the fact that appointments depend on each other. This causes the problem that in case a single appointment is canceled by a patient the software does not know it should also inform the other departments where the other appointments are scheduled. The result is that the other departments will get a no show and their efficiency decreases.

Department	Already applied	Should be solved
Radiology	- Fast mamma diagnosis: predefined combination with radiology, schedulable by other departments.	- Internal patient classification - Closed agenda - Paper request forms - Requests by phone
Lung function	- External patient classification supported by x/care - Digital request forms - Open agenda	
Anesthesiology		- Internal patient classification - Closed agenda
Endoscopy	- Some timeslots are reserved and can be scheduled by other departments	- Internal patient classification - Partially closed agenda

Figure 2.3-1 Overview of the current state of the diagnostic departments at the AMC hospital

The current scheduling workflow model can be seen in *Figure 2.3*-2. The model represents the workflow that is applicable to most outpatient department. After the consult at the outpatient department the will have to wait for his appointments to be scheduled. Some appointments at other departments require either a confirmation by phone or a request form. Request forms are necessary for test and or procedures where patients need to be classified and verified by the medical doctor of the department. The medical doctor can either approve or deny such a request. In case of a rejection of the request the medical doctor will consult the outpatient department to gather more information and finally classify the patient for the correct procedure. In case the patient needs several appointments at different departments this process is repeated for every single appointment.

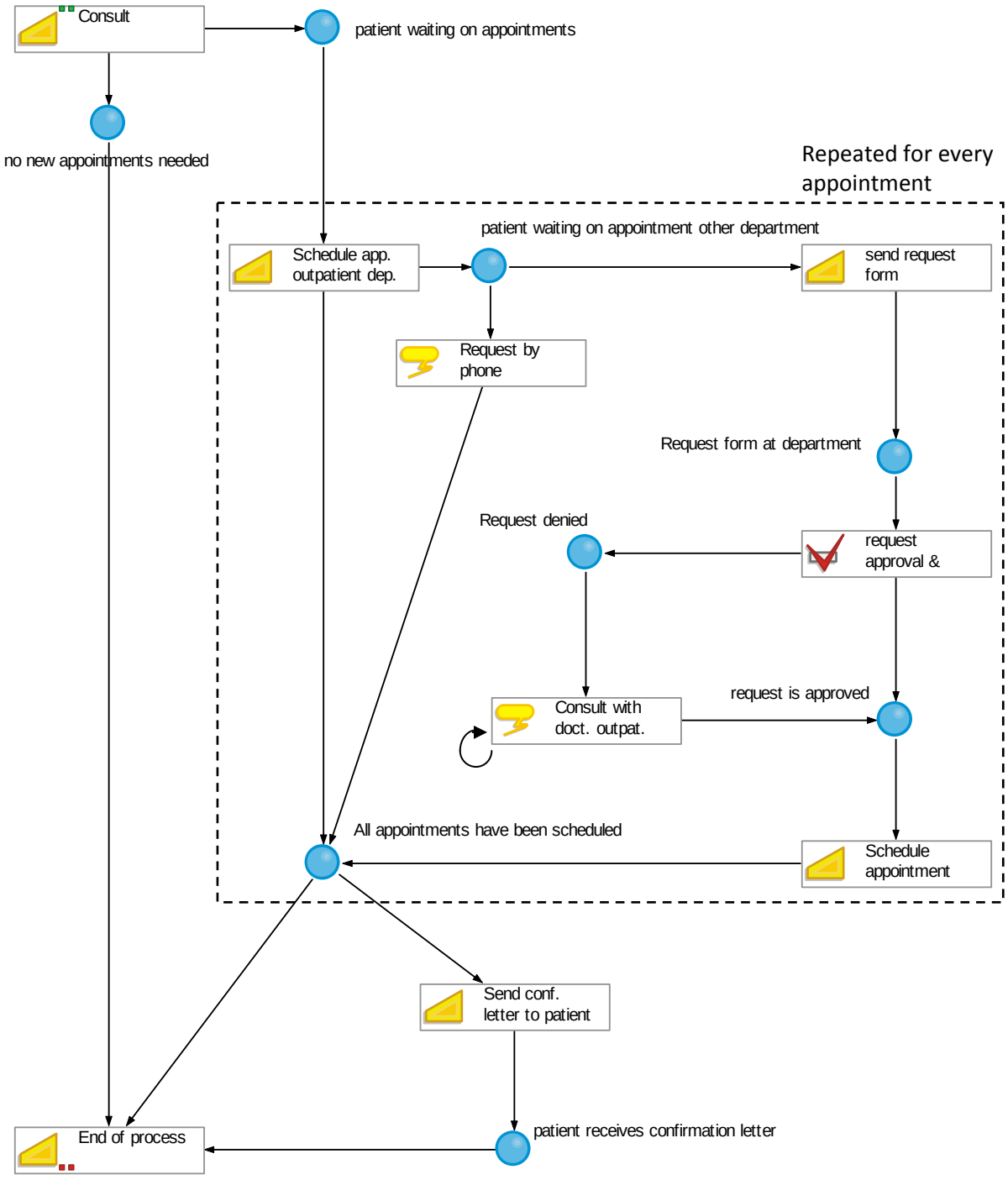


Figure 2.3-2 Workflow model of the current scheduling process

2.4 X/Care

Scheduling appointments is always supported by some kind of software. There are many different scheduling software packages available to hospitals and it is a very important decision to choose the right software fitting their needs. The AMC has chosen X/CARE software package to handle all their scheduling.

During our research we had the opportunity to use X/CARE with a patient Database that contained all patient data till January 2008. It took some time to figure out how exactly X/CARE works because it is not so intuitive to use and there seems to be so many exceptions and special methods implemented that there is not a single good way of using it. This was also noticed during the observations at the different departments; every department seems to be using X/CARE in a different way. There was no strict guideline on how to use X/CARE applicable to the whole hospital. It seems that X/CARE is modified for many special situations applicable to every departments needs.

There was continuously need for adding new functionality to the software. We could tell from the structure of the Database and they way in which these functionalities where implemented that the software was not initially designed to be adjusted this much. During our observations of the staff using the X/Care we noticed that the system was not so responsive.
For the implementation of a simulation application in chapter 4.3 we had a good insight in the database structure of X/Care, we noticed that there where many tables and columns added for new functionality and at the same time old redundant tables or columns where not removed. We were confronted with the complexity of this rather large database when we tried to collect data from this database for our research. In the chapter 4 “simulation” we will discuss why we used a simpler database structure for our simulation purposes.

X/CARE has a lot of functionality, unfortunately it is not always being used at its full extend. We observed that the staff just knew enough to be able to perform their tasks but did not use X/Care at its full potential. Apparently the hospital is aware of this fact and studies have been made to exactly address this problem.

For our research where are interested in how X/CARE handles combination appointments. First let us define again what we mean with a combination appointment: coordinating a set of sequential appointments of the same patient on the same day. X/CARE does indeed provide functionality of combination appointments, however as it appears there are different definitions of combination appointments, X/CARE combination appointments can be:

- A predefined combination of test/procedures within the same department. There are for example special combinations appointments predefined in X/CARE for a quick breast cancer diagnostic that consist of a series of test all performed at the radiology conducted all at once. See Figure 2.4-1
- Combination of a resource and a medical doctor. The lung function for example uses this kind of combination to schedule a combination appointment with a medical doctor and an available room. See figure 2.2-3

- Single appointments at different departments that are in fact combination appointments. X/CARE only supports combination appointments if they have been predefined first. So if a patient needs a series of appointments that is not defined, you will have to coordinate these appointments manually.

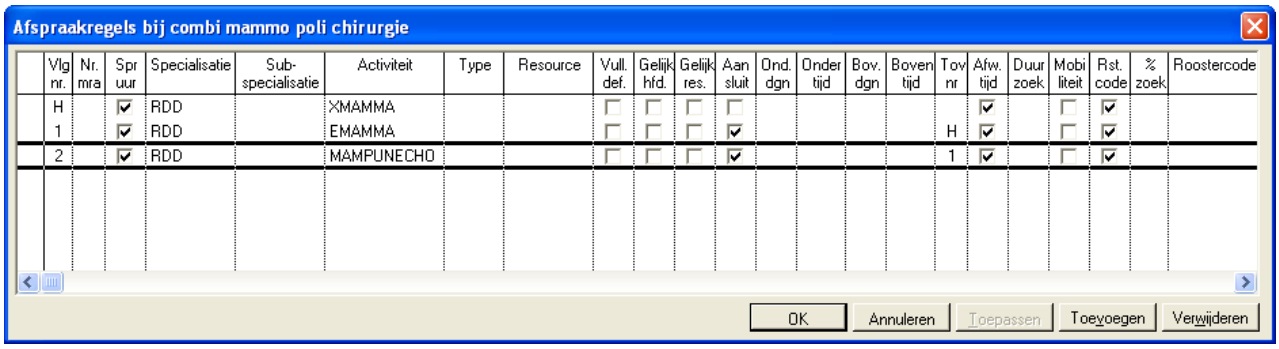
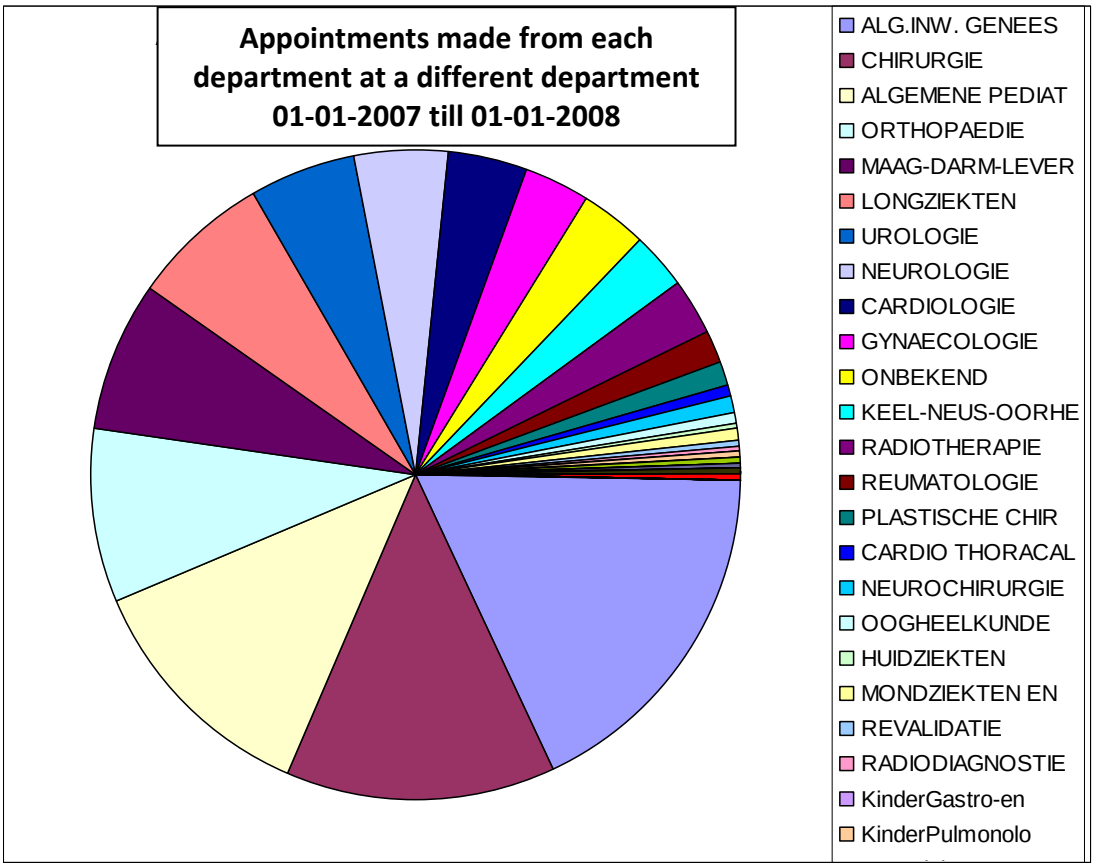


Figure 2.4-1 Predefined combination rules called "mammo" for breast cancer diagnostics

One of the main problems of X/CARE and maybe many other scheduling software is that they don't store any information about how different appointments relate to each other.

2.5 Scope

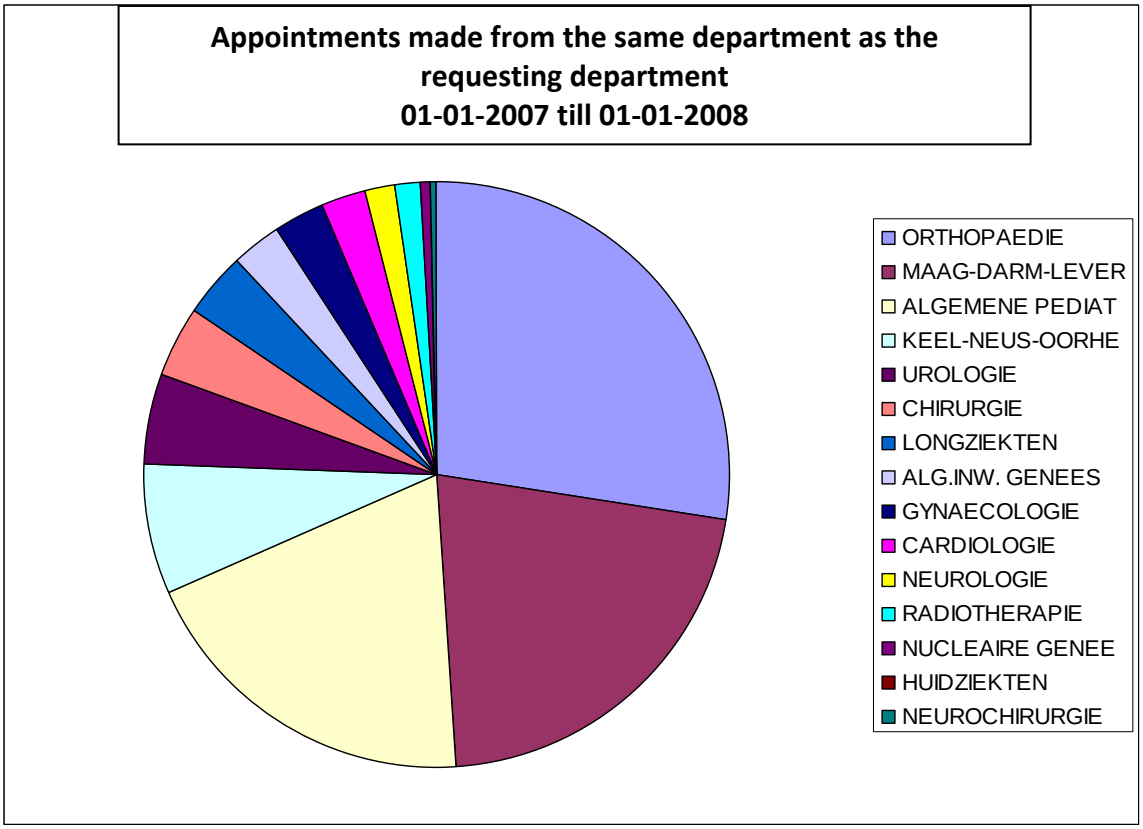
To get an idea of the scope of the project we did some analysis on the x/care database which is used to schedule all the appointments at the AMC. Because combination appointments are not scheduled as these, it is very difficult to extract this information from the database. We first made a query which represents the amount of appointments made that where referred to by the internal medical doctor. As can bee seen on the pie chart below the department that is referring the most is the internal medicine and surgical department. The internal medicine department makes 20.023 appointments in the year 2007 at other departments.



ALG.INW. GENEES	20023	REVALIDATIE	378	RADIOLOGIE	37
CHIRURGIE	15096	RADIODIAGNOSTIE	248	Spoed hulp	36
ALGEMENE PEDIAT	13664	KinderGastro-en	198	Infectie Ziekte	34
ORTHOPAEDIE	9602	KinderPulmonolo	194	VERLOSKUNDE	21
MAAG-DARM-LEVER	8364	Vasculaire gen	116	Vaatchirurgie	21
LONGZIEKTEN	7923	KinderChirurgie	99	BACTERIOLOGIE	18
UROLOGIE	6014	KinderOncologie	82	Kindergenetica	18
NEUROLOGIE	5287	Hematologie	78	Neonatologie	16
CARDIOLOGIE	4219	Nierziekten	78	KinderCardiolog	15
GYNAECOLOGIE	3860	PSYCHIATRIE	73	KinderMetabole	13
ONBEKEND	3859	Traumatologie	71	Tropen gen	12
KEEL-NEUS-OORHE	3130	KLINISCHE GENET	62	Inw. Gen. APT	11
RADIOTHERAPIE	2877	KinderEndocrino	61	GERIATRIE	4

REUMATOLOGIE	1943	Oncotherapie	58	ALLERGOLOGIE	3
PLASTISCHE CHIR	1195	KinderHem./-Imm	55	Dietetiek	1
CARDIO THORACAL	839	KinderNefrologi	53	PATHOLOGIE	1
NEUROCHIRURGIE	792	Endocrinologie	47	KinderPlast. Re	1
OOGHEELKUNDE	579	KinderNeurologi	43		
HUIDZIEKTEN	512	NUCLEAIRE GENEE	43		
MONDZIEKTEN EN	393	ANESTHESIOLOGIE	40		

Although this data doesn't say anything on the amount of combination appointments it does give insight into the amount of scheduling appointments at other departments.
The following query uses the login information of the x/care users to discover if the appointment was made by an employee of the same requesting department, which would indicate that the department has privileges to make appointments them self at other departments.



ORTHOPAEDIE	153
MAAG-DARM-LEVER	118
ALGEMENE PEDIAT	108
KEEL-NEUS-OORHE	40
UROLOGIE	28
CHIRURGIE	21
LONGZIEKTEN	20
ALG.INW. GENEES	15
GYNAECOLOGIE	15
CARDIOLOGIE	15
NEUROLOGIE	8
RADIOTHERAPIE	8
NUCLEAIRE GENEE	3
HUIDZIEKTEN	1
NEUROCHIRURGIE	1

The data shows that this amount is significant less then the data of the previous chart.
We have to note this may be not a real representation of the actual process because the login account is not always provided with the specialism the user is working for.
Apparently very few appointments are made from the department them self at other departments. The difference in these numbers form the first chart show the potential improvement that can be made in allowing departments to schedule at other departments.
Further queries will concentrate on actual combination appointments, but these queries are more difficult.

3 Combination appointments

In the previous chapter we discussed the scheduling processes at the different departments that are currently applied at the hospital. A possible combination appointment can be defined as a series of appointments at different departments that may be scheduled on the same day. The problem with the coordination of combining several appointments is that neither the current scheduling process nor the software that is used, supports the combination appointment process to the desired level.
Nearly every department has resource capacity problems resulting in waiting lists which forces them to operate as efficient as possible. Revoking the access to their agenda to other departments seemed to be the first response in achieving a higher efficiency. If we would like to implement a system that will support coordination of appointments we have to make the agendas accessible to other departments.
Efficiency of a department is one of the key elements. If we can open the agenda to the other departments without a decrease in efficiency it would enable us to coordinate appointments. As we will see the efficiency is depending on correct patient classification.
The next key element would be automated scheduling. Most requests for diagnostic tests are currently done on paper. Automated scheduling will fully digitalize every paper request and will give us the possibility to directly schedule an appointment, which again is necessary for the coordination of appointments.
We will model an agent based architecture that will enable automated scheduling. Only with the support of automated scheduling we will be able to coordinate combination appointments.

Our model will describe a system that will:

- Enable direct scheduling of appointments at other departments (automated scheduling).
- Provide the medical doctor a control mechanism to guarantee the correct classification of the patient.
- Guarantee high efficiency using an optimal automated scheduling method.

3.1 Efficiency

As we have seen the need for local efficiency at the departments is the reason why nearly every diagnostic department have closed their agenda to other departments. The efficiency of the resources, departments and the whole hospital is determined by the scheduling processes.
A way to analyze efficiency of a department is to describe what causes inefficiency. In practice we can say that every time a resource is at idle the resource is not used efficient. Optimal efficiency of a resource means 100% utilization. This means that the medical doctor or any other kind of resource that is being scheduled theoretically should never be without work. We list the following reasons for resources not being used:

- Wrong estimation of the duration of procedures
- Assigning timeslots for every procedure which results in a unequal distribution of the procedures which can cause empty timeslots.
- Wrong estimation of the number of urgent timeslots per resource. Correct estimation of urgent timeslot is crucial; claiming to many urgent timeslots will cause empty timeslots.

- Wrong patient classification. Patients need to be assigned to the correct timeslot. The correct timeslot represents estimated duration, urgency and the right resource. If any of these is estimated wrong, efficiency will decrease.

In practice one could wonder if it is desirable to use a medical doctor at 100% capacity, a medical doctor could have other duties in between like paperwork that is not documented in the schedule of the doctor.

In short we can conclude that all reasons for inefficiency have something to do with patient classification and timeslot allocation.

3.1.1 Timeslots

There are several theories [11] about waiting queues, they all deal with the question whether it is more efficient to use a separate waiting queue or not. The reason to introduce this separate waiting queue is to specialize the service that is provided to the costumer or patient, thus serving the patient better and quicker.
In practice specific timeslots are scheduled at a department where a respective medical doctor only sees specific kind of patients. These reserved timeslots are often repeated every week at the same time.
Another reason to define different timeslots is the urgency of the patient. Based on previous experience an estimated amount of timeslots is scheduled for these more urgent patients.

If we take a look at the schedule for Anesthesiology and the Radiology MRI schedule, we see that the schedule of the Anesthesiology is totally fragmented into dozens of tiny timeslots. This in contrast to the schedule of the MRI, they have defined only one big slot meaning appointments can be scheduled anywhere.

The MRI has an urgent timeslot planed at the end of the day. This has the advantage that it causes no empty spaces in the schedule. The fragmented schedule of the anesthesiology is not desirable because this results in five different waiting queues for each specific type of timeslot. A wrong estimation of patient arrival in one of these queues will cause inefficiency for the whole resource. The more queues we define, in our case type of timeslots or patient group we define, the bigger is the change of an unequal distribution among these queues, which could cause inefficiency. However if a patient is waiting in the wrong queue it will also cause inefficiency [11].
The schedule we see in figure 3.1-3 is the schedule of the junior assistant anesthesiologist. All the different timeslots allocated for this anesthesiologist are performed within the same room with the same equipment. There is no reason why the anesthesiology would need to define all these different timeslots. If the anesthesiology would apply one large timeslot just like the MRI it would give them more flexibility, since for every patient that can't be scheduled within the available timeslots for that patient group an authorized person has to override the system in order to schedule the patient on a different timeslot not ment for this patient group.

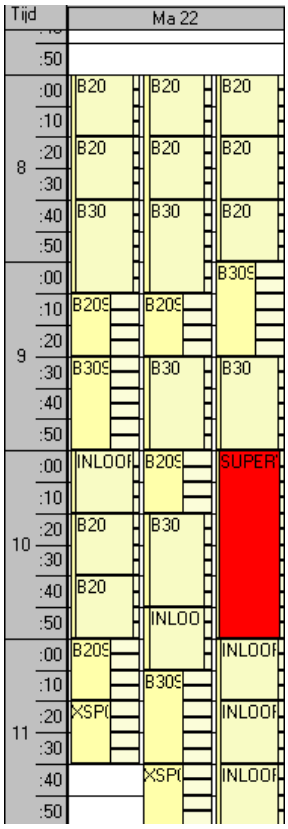


Figure 3.1-2 Part of Anesthesiology schedule, showing an empty schedule

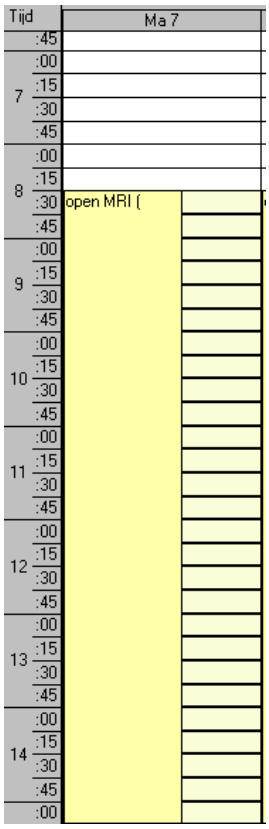


Figure 3.1-1 Part of MRI schedule, showing an empty schedule

Different timeslots at the Anesthesiology:		**Different timeslots at the MRI (Radiology)**	
B20	= timeslot of 20 min.	Open MRI	=normal
B30	= timeslot of 30 min.	SpoedBLK	=urgent
B20S	= urgent timeslot of 20 min.		
B30S	= urgent timeslot of 30 min.		
INLOOP	= without appointment (20 min)		
XSPOED	= urgent (20 min)		

Figure 3.1-3 Different timeslots of Anesthesiology and the MRI

The motive of the anesthesiology to apply all these different timeslots was their believe that the X/Care software could not schedule a patient for example the urgent patients into a timeslot that hasn't explicitly been reserved for this kind of patient.

Reducing the number of timeslots reduces the number of waiting queues will increase the scheduling flexibility for the anesthesiology. There is however another very convenient advantage in regard to combination appointments. Currently most departments still apply many different timeslots for every type of patient. One can image that coordinating several appointments on a single day within a certain timeframe becomes difficult if we are only able to schedule the appointment on a specific day or a certain time frame of the day. The flexibility of a single timeslot a day is great advantage to coordinated scheduling.

If this approach has only advantages one could ask why hospitals still apply all these different timeslots. First we have to note that in practice it is not always possible to implement a single timeframe a day because in some cases the timeslot is defined by certain resources that are only available during a certain period of time.
Second scheduling appointments in the middle of the day without predefined slots may seem inefficient because it could cause empty spaces when it is not know what kind of patients will be scheduled and what their duration is. To prevent this we have applied an algorithm that only leaves empty spaces between timeslots at the size of the minimum possible timeslot for that resource which turned out to be very efficient. We will discuss this with our simulation results in the next chapters.
Third and last reason against this approach is that the number of urgent timeslots is not predefined anymore, so in order to still be able to handle urgent request with a single timeslot a day approach one would have to reserve a certain amount of time a day. The MRI solved this by handling all the urgent request at the end of the day.
In the work of I. Vermeulen [2] the dynamic resource allocation can even dynamically reassign capacity to the urgent, clinical en normal patients based on the demand and future expectations.

3.1.2 Shared resources

In our search to further optimize the efficiency and to improve the coordination of appointments we have researched the possibility to combine resources with the same capabilities we will call shared resources from now on. This would balance unequal distribution over resources capable of doing the same procedure or diagnosis and it would again greatly improve our chance in coordinating appointments of different departments which was our first aim when introducing this approach.
If we look at the anesthesiology for example they have three different resources called junior, senior assistant and the chief of medical staff. They are all capable of handling the same kind of patients with the difference that the more complicated patients should be seen by the senior assistant or the chief of medical staff. So rather than to define a certain amount of specific timeslots to each resource it would be more efficient to define the capabilities of each resource. Figure 3.1.2-1 shows an example on how the requests are distributed over all capable resources that can handle certain requests. Because some timeslots at the anesthesiology are defined by their urgency one would have to define the certain amount of time that is reserved for these timeslots. How these urgent patients or more complicated patients are assigned to the right resources is defined by the scheduling method which can individually be implemented for every single department.

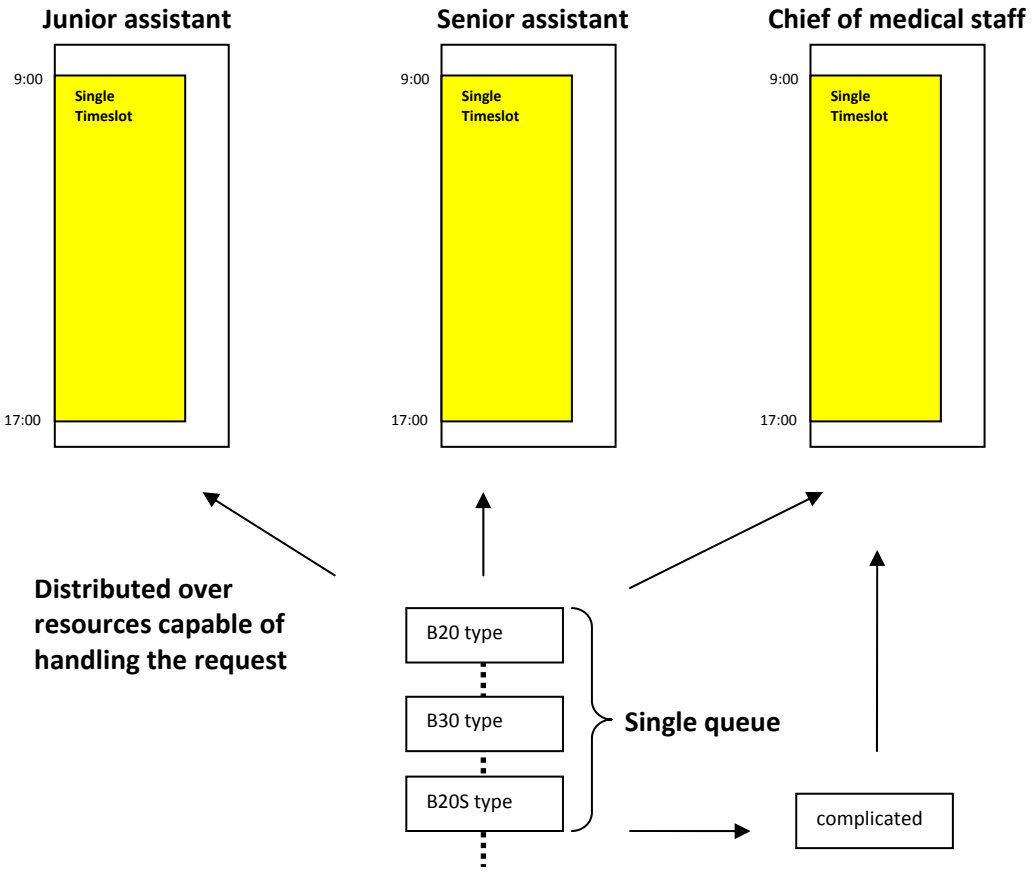


Figure 3.1-4 Shared resources example Anesthesiology

For our simulations we applied this approach only partially. We simply combined resources that were nearly equal but we did not define for every resource what their capabilities where. We have to note that for complicated cases they can only be seen by the chief of medical staff all other cases can equally distributed over all resources. The theoretically equal distribution over these resources does not mean that we can't differentiate among these resources; we could still define rules that send more complicated (children, older patients) to the senior assistant or chief of medical staff without having any impact on the equal distribution over these resources.

3.1.3 Hospital organization

The current operational, tactical and strategically way the AMC is coordinating the scheduling of staff, resources and patients does not allow much coordination and cooperation.
The main problem is the local authority and the desire to keep in charge of the local schedule of the departments. Departments believe that in order to fully utilize their staff and resources it is necessary to keep in control of their own scheduling. The departments and the outpatient

departments have to realize that they together have the same goal: serving the patient as well and quickly as possible. Departments and outpatient department should not only look at their own local efficiency but at the efficiency of the hospital.

3.2 Patient classification

This section will discuss one of the most important aspects that impact on the hospitals efficiency. Patient classification, the process where the patient is classified in to a certain patient group based on complaints the patient has and on the diagnosis that has been performed by a doctor. All throughout the hospital and at every level this classification is performed. The nursing staff classifies new patient based on the diagnosis from the general practitioner and on the complaints of the patient. Then again if the patient needs an appointment at another department, the patient is classified again. This can be done by the department where the patient has his appointment or the outpatient department that refers the patient. The classification of a patient represents the urgency, the kind and or complexity of procedure or diagnosis the patient needs and the health state of the patient. Urgency and health state do not necessarily have to relate, a patient can be not urgent but due to his health state it is expected the appointment will take longer for example. For coordinating appointments we need immediate classification while the patient is still present so patient has the possibility to directly choose one of the best combinations that match their preferences. The other alternative would be that the patient can browse online at home to see if his classification has been completed, after completion he can choose a desired combination.

3.2.1 Better classification increases efficiency

For the diagnostic departments in most cases the patient classification is done by the medical doctor of the department where the patient has an appointment. At the AMC the main reason for this is the lack of resources and long waiting times resulting in a strong need for high efficient use of their resources. Classifying the patient correctly has proven to be the solution for increasing this efficiency. By classifying the patient, a better estimation of the needed timeslots can be made. Eliminating variable factor in the schedule as far as possible reduces waiting times for other patients in case an appointment takes longer then scheduled but also reduces the idle time of doctors when they are finished earlier than scheduled. There is many [11] literature and case studies on this topic available; eliminating the variable factor increases the efficiency. The departments at the AMC are responsible for their own local efficiency. This is could be the reason why the departments have cautious attitude towards other departments scheduling in their agendas.

3.2.2 Specialist classification versus outpatient classification

For every procedure, diagnosis or appointment the patient will be classified by every single department the patient is sent to. The medical doctors at the departments believe that they are far more capable of classifying a patient than the doctor of the outpatient department from where the patient is sent. They provide a service to the outpatient department, using their knowledge and expertise. This service however is mandatory, so in fact it is a control mechanism of the department. They simply want to be in control and don't have enough trust in a possible classification by the outpatient department.
Several interviews with doctors at the departments and the outpatient departments have been made to address this issue. It is remarkable that nearly every doctor at the outpatient departments state that they are capable in determining the right patient classification

themselves just as well or even better. Their arguments are that they have personal contact with the patient, better knowledge of the patient's history and that with passing information to the departments information is lost or misunderstood.
These seem strong arguments. We reviewed request forms that were submitted to the diagnostic department, most of them provided little information limited to only the mandatory fields on the request form. Besides the little information on the request, the requests are difficult to read sometimes due to the requests not being digital. Some medical doctors at the outpatient department believe that they are just as capable of determining the patient's classification as the radiologists who receive little information on the request form and had no direct contact with the patient. This of course is not applicable to all medical doctors but to fully investigate the possibility of medical doctors not from the radiology to classify patients for the radiology more studies need to be done.

3.2.3 Eliminating local patient classification

Local patient classification is defined as the classification performed by medical doctor of the department where the patient is being seen. Our interest is to eliminate local patient classification.
In our desired situation, the patient will receive his coordinated appointments immediately after the consult at the outpatient department. This is only possible if we can directly classify the patient and have the authority to schedule the patient at the department using this classification. Some alternatives are that even the patient hasn't been fully classified he is still scheduled for an appointment; the correctness of this appointment is evaluated later on by the department. If the false classification ratio is not to high this could be a compromise. Further more the patient could wait until he has been classified by al departments and is then informed by phone or email that he can choose a combination online. Once the patients is online he will only receive appointments that are still available real time, the timeslots are not reserved.

We will discuss a approach that will enable us to immediately classify a patient by the outpatient department but at the same time the diagnostic departments remain in control over the classification process and the scheduling process. We will use the knowledge and expertise of the medical doctor to build a medical decision tree. This decision tree is continuously monitored and reevaluated by the medical doctor of the regarding department. We can use the decision tree to let the requesting outpatient departments classify the patients. This way we will use the expertise and knowledge of the medical doctor while enabling immediate patient classification. The classification system will use the following data for patient classification:

1) Previous classification score of the requesting doctor based on history records. Some doctors just have more experience and therefore their classification is more reliable then assistants for example. In case a doctor requests a procedure or diagnosis where in the past he made to many wrong classifications the classification is marked and will be manually evaluated by the medical doctor. However the appointment is still booked, in most cases a wrong classification can still be performed on the same resource but just needs more or less time which doesn't impact date and time of the original appointment.
2) The patient's medical history.
3) The patient's current health state and all medications the patients the used or still uses.
4) The urgency of the patient.

5) Diagnosis of the doctor using the medical decision tree. The doctor will be guided through the decision tree where he will be asked to provide information about symptoms and complaints of the patient.

The advantages are that the system can take into account any kind of information that is available from the patient but also of the doctor requesting the appointment. The main complaint of the medical doctors was that not enough information was provided, using the decision tree the requesting doctor is forces to provide certain information and complete the process otherwise he can't make the appointment. This will increase the quality of the request.

3.2.4 Automatic classification with the MRI

In the current situation the MRI just like the CT en some echo's will have to be classified by the medical doctor first. The MRI is a difficult and complex scenario because the medical doctors use all kinds of knowledge and expertise to estimate whether the requested scan is the appropriate one. The Radiology wants to ensure that they can deliver the same service quality to all of their patients. We will not build a knowledge base for the MRI or a decision tree but we will define the variables that are needed to schedule a patient(based on interviews with Rachel Snoeks, a MRI laboratory technician) :

1) Which scanner:
   The Radiology has four MRI scanners. The scanners differentiate on several aspects. Not all scans can be performed an all scanners. The first aspect is strength. M4 has 1 Tesla, MR2 and MR3 have 1,5 Tesla and MR1 has 3 Tesla. A stronger scanner does not necessarily have to provide better pictures because it also creates more noise, it depends on the kind of scans. Furthermore scanner MR4 is an open Scanner that is used for special patients that are not able to lay still, don't fit in the tube or a scared in narrow places. MR1 is mainly used for head scans. MR2 and MR3 have a lot of accessories (coils) available to perform a variety of scans. The radiology has a list where for each kind of scan the duration of such a scan, whether or not contrast media has to be used and on which scanner the scan can be performed is defined. Using this list it should not be too difficult to determine on which scanner the scan has to be performed once it is know what kind of scan the patient actually needs.
2) Extra series. In some cases there is a certain amount of uncertainty about what the problem of the patient is. For these patients there is a high possibility that more scans are required. A radiologist has to be present and extra time needs to be scheduled for this kind of patient.
3) Cardio patients. In case a hart film is made this has be timed exactly with the hart beat rate of the patient in order to get a scan of each state of the hart.
4) Contrast media is needed. If contrast media is needed there first will be a series of scans without the contrast media and then with the contrast media. Finally the scans can be compared.
5) Narcosis is needed. Children that can't lay still or patient that are scared of narrow places will have to be sedated.
6) Patients with bad mobility or in a wheelchair. If possible they will be scanned in MR4 this is an open scanner.

The most difficult is the classification of the patient to a certain kind of scan, however for scheduling purposes we only need to know the variables we just stated. If the patient is classified for the wrong kind of scan in most cases the correct scan is still performed on the same scanner with the same duration and the other variables still apply. This is because most scans of a certain area for example the spine are all performed on the same scanner. If the patient is classified for the wrong type of spine scan it does not necessarily impact the scheduling if the scanner and duration could remain the same. Classification would not need to be that precise for out scheduling purposes.

We think that with a carefully designed decision tree by the medical doctor patients can be classified for the correct procedure in most cases. Than in case the patient is classified wrong the impact to the scheduling is still could be limited. Patients will be scheduled for a certain scanner but the radiologists can still evaluate the request after the appointments have been booked. This will enable us to immediately provide possible appointments and coordinate appointments. The radiologists can evaluate the request they don't trust depending on their own reasons. These reasons can be that the requests come from an inexperienced doctor or that the kind of scan has a high percentage of wrong classification. All these parameters can be defined by the radiology department giving them full control. The performance of the classification can be monitored real-time so incase the classification is very bad they can decide to check every request of a certain doctor or scan.

3.2.5 Determine the expertise of the non-specialist

The automatic classification for the MRI for example in theory sounds very reasonable. The automatic classification would make it possible to automate the coordination of appointments. In order to estimate the feasibility of this study we would have to investigate the feasibility of automatic classification in a real hospital. Our idea was to implement a digital request form, to collect data about the expertise of the non-specialist. Digitalizing request forms would provide us with valuable information about the current value of correct requests. Another step would be to implement the decision tree and evaluate how well this would perform in real life. A medical decision tree requires a lot of effort for each possible procedure that could take many months and is not in the scope of this project. However digitalizing the request for a certain department would have been feasible. A simple optional web form would have been enough to collect this valuable information. Due to lack of time and resources at the outpatient department we were not given the opportunity to implement this. In the near future the hospital will fully converge to digital requests. Future research will then be able to measure to current level of correct classification. Once the request forms have been digitalized it will also be possible to gradually introduce automatic classification using decision trees. The lung function department already applies this in a certain extend.

3.3 Multi Agents

The scheduling process in the hospital is much decentralized. In our case study at the AMC we observed that all the departments act as autonomous entities which all have their own goals and responsibility over their own scheduling process. Furthermore the hospital has a very dynamic environment which requires to continuously adjust the schedules of the departments. These characteristics overlap with the domain of software agents, multi agent systems and a multi agent coordinated planning. An agent can be defined as [8]:
"An agent is a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives"
A multi agent system simply consists of multiple agents that interact with each other in order to achieve their own personal goals or a common goal. If we model a hospital as a multi agent system we would define every resource and department as an agent. They all are autonomous objects that have their own goals and need each other to achieve their goals.
Then there is the domain of multi agent coordinated planning, we give the following definition of a multi agent planning problem [9]:
"Given a description of the initial state, a set of global goals, a set of (at least two) agents, and for each agent a set of its capabilities and its private goals, find a plan for each agent that achieves its private goals, such that these plans together are coordinated and the global goals are met as well."
The set of agent in our case would be the patients, the resources and the departments. The global goal of the hospital is to work as efficient as possible and to provide the best service given the medical constraints to the patient. All individual agents have their own goals but they also have to meet the global goals as good as possible. The individual goal of the patient's agents is to improve their health state as quickly as possible. The department's goal is to meet certain norms which include patient service level and efficiency of their department. The resources on their behalf will try to run as efficient as possible.

3.3.1 Opportunities using multi agents in an hospital

The biggest strength of agents is that they are autonomous. They can perform actions without the interaction of a human being. In health care this is a very desirable characteristic since the environment is in constant change and actions need to be taken to react, which is exactly what agents do. We would like to mention opportunities that multi agents could provide:

- Performing complex coordination tasks.
  For our study we are mainly interested in the possibilities of multi agents coordinated planning. The coordination of appointments is a complex task. The currently used scheduling software at the AMC is capable of finding appointments of different resources and provides possible combinations for some certain predefined combinations. However the scheduling software is not capable of taking into account the goals of each entity.

- Patient's agent can interact with the patient directly in both ways. Although the patient agent will be created by the outpatient department, the human patient could be provided with some kind control over his agent. In case the patient agent finds an appointment that that is more convenient to the human patient, an appointment is cancelled or test results are available it can inform the patient by sms, phone or email.

The human patient can control the agent through a web interface. This web interface could be merged with Electronic Patient Dosier (EPD) that will be introduced in the Netherlands very soon. They will be able to see their appointments and if necessary can change these appointments. The agent will then try to make new appointments. Because the patient is already classified for the original appointment, booking the new appointment won't require any further approval.

- Real-time monitoring and alerting of all resources. The efficiency, utilization and the expected shortages could be monitored real-time. In case one of the goals of the agent are not met or are expected to be endangered in near future, the agent will inform the responsible staff of this problem.

- The scheduling algorithm could be adjusted by the resource agent itself. In case the agent detects that a certain patient group, urgent patients for example, are not served well, it could adjust its scheduling algorithm to meet its goals better.

We will not include all these opportunities in our simulation, we just want to emphasize what benefits multi agents can have comparing to the static currently implemented software at most hospitals. For our simulation we will only focus on the complex coordination task.

3.4 Workflow model new scheduling process

The Model focuses primarily on the auxiliary departments. The auxiliary departments are each represented by an agent that is specifically designed to serve the department as good as possible. Instead that the outpatient department gets full access of the auxiliary department the outpatient department can do a request.

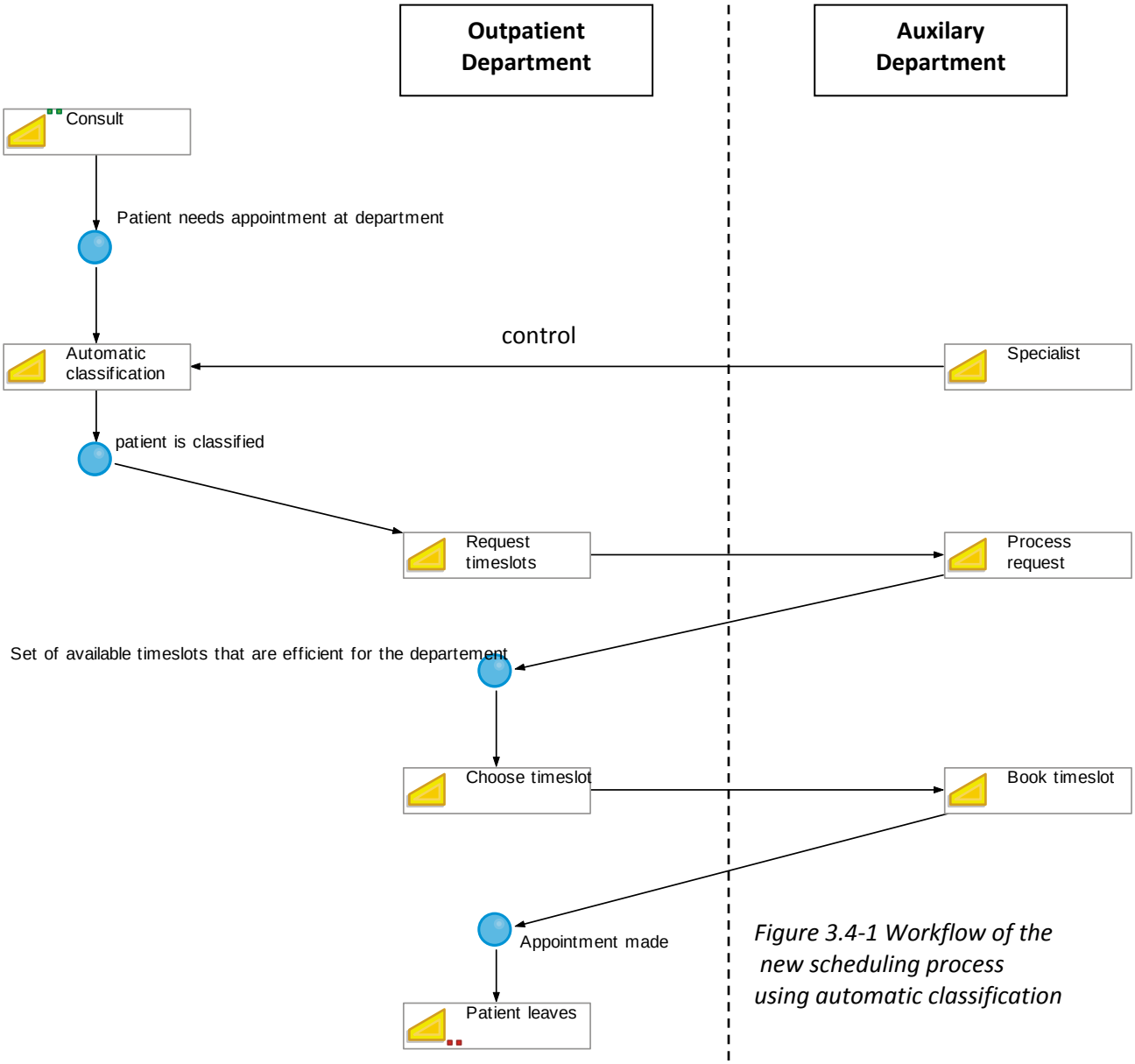


Figure 3.4-1 Workflow of the new scheduling process using automatic classification

For certain requests a prior classification of the patient is needed, the doctor at the outpatient department will classify the patient using the automatic classification module provided by the medical doctor of the requested department. *Figure 3.4-1* shows the workflow of this new process.

This automatic classification module uses the knowledge of the medical doctor and enforces the requesting doctor provide essential data about the patient. With this module we don't need any interaction or approval of the auxiliary department anymore.

Once the patient is classified we will request timeslots at the different departments. These timeslots all have a certain value representing the costs of such a timeslot. Depending on the patient's classification and the scheduling algorithm of the department and resources, the cost for each timeslot is calculated. In our model we define cost as parameter that represents how much a timeslots is worth to the hospital. The patient agent will always provide the cheapest solutions first to the patient since they are most efficient for the hospital. Cheap or low cost of a timeslot can have different meanings depending on the implementation of the scheduling method. The patient chooses one of the options provided and the appointment will be booked.

3.5 Patient, Department and Resource Agents

A complete overview of the agents and their communications can be seen in *Figure* 3.5-1. It shows all the participating agents (Patient, Department and Resource Agent) and their communication.

3.5.1 Patient Agent

The main purpose of the patient agent is to schedule and coordinate appointment to improve the patient's health state as quickly as possible. First the agent is created by the outpatient department. The agent is aware of the current health situation, any medical records such as current medications, current treatments, peacemaker or prosthesis. The agent also knows personal data such as age, weight, length, home address and distance to hospital.

After the consult the patient agent is instructed to request available timeslots for a certain procedure or test as requested by the doctor. The agent will contact the department agent, in case a patient classification is necessary for this request the patient agent is will be instructed complete a patient classification form first. The patient agent informs the medical doctor at the requesting department that he has to complete the classification form first. Once the classification form has been submitted and approved by the digitally request form the patient agent is able to request available timeslots for the desired procedure. The patient agent will receive a result set containing all timeslots that are applicable to the patient. Each timeslots has a cost value. The cost represents the value it has to the department and resource. They agent will try to spend as less as possible but will also take into account the patient preferences into a certain extend. In case several appointments need to be made, the patient agent will receive timeslots from all of these departments and will make a coordinated appointment list that is sorted ascending on total costs of the combination.
The patient agent can be controlled by the patient through a web interface. The patient can consult its appointments and can instruct the patient agent to find another alternative. The patient can also instruct the patient agent to inform the patient if certain events occur, for

example if an appointment becomes available on a short notice or certain test results become available.

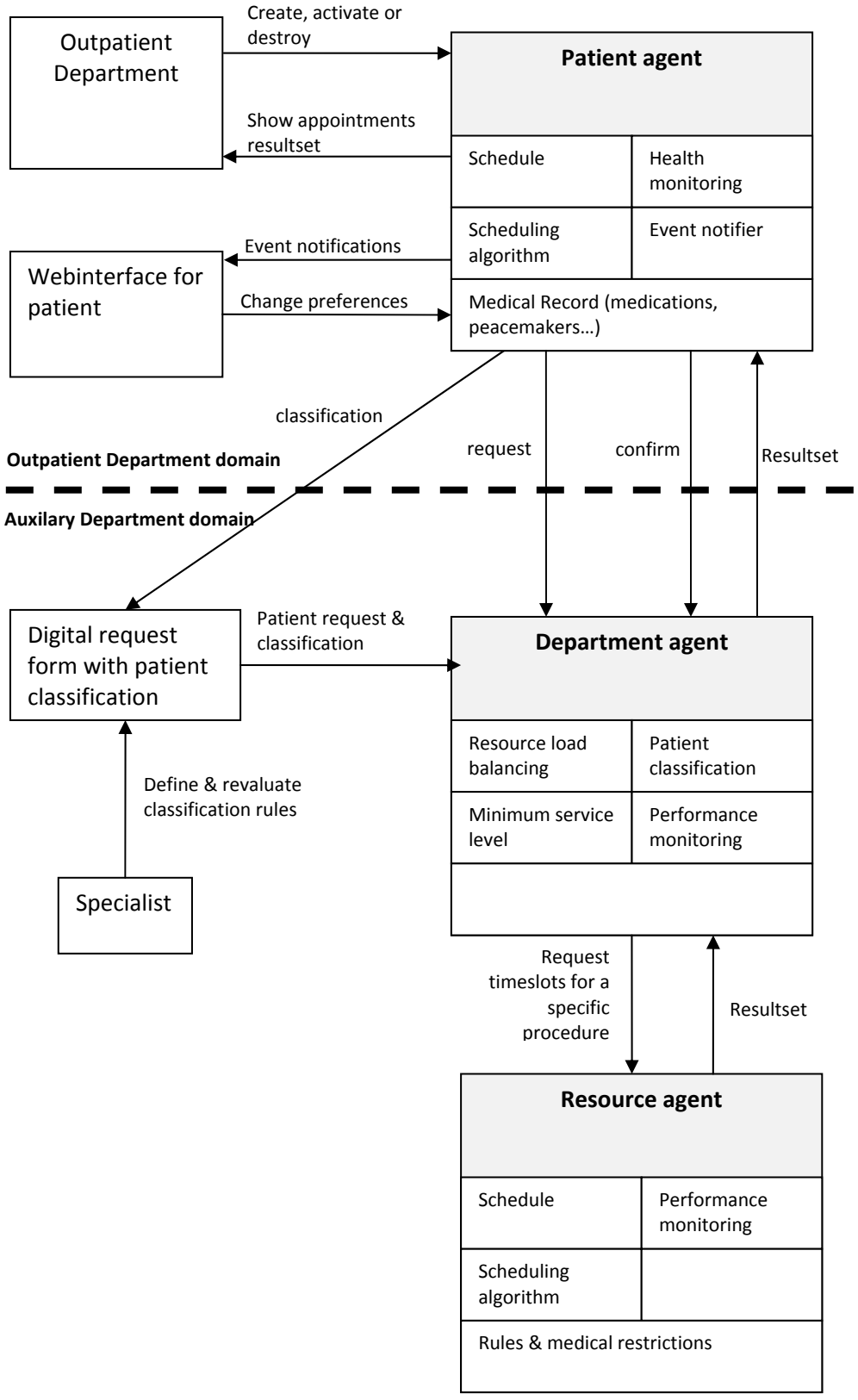


Figure 3.5-1 Communication between agents and other objects

3.5.2 Department agent

The department agent interacts as an intermediate between resource agents and patient agents. Timeslots for resources cannot be requested at the resource agent directly. The department agent acts as a control mechanism which has several purposes:

- Ensure the patient has been classified correctly using the digital request form which is maintained and reevaluated by the medical doctors at the request department. The digitally request form will provide the requested department with the urgency, medical status and the preferences of the patient.
- To maintain and monitor a minimum service level for:
    - The department.
    - Its resources.
    - Procedures.
- Load balance between different resources which can perform the same procedures. In case a procedure can be performed on several resources the department agent will request timeslots of every resource and will make a combined list of all of these timeslots.
- To maintain and monitor the efficiency of:
    - The department.
    - Its resources.
- Check the requests if they meet rules and restrictions that are applicable to the resources or procedures

3.5.3 Resource agent

Resource agents respond to requests for timeslots of their department. Every resource agent has its own specialized scheduling algorithm. Each algorithm can have a different purpose targeted at different kind of patient groups. The algorithm will calculate depending on its purpose the cost of each timeslot.
A possible algorithm we will also use in our simulation is a First Come First Serve (FCFS) algorithm. This algorithm is only suitable for resources that don't differentiate between patient urgency and therefore most likely have enough resource capacity. The cost of an early timeslot will be cheap and the cost of a late timeslot will be more expensive. Because the patient agent overall object is to provide the cheapest timeslots to the patient the patient will shown the earliest timeslots available.
However in most cases resources will apply different costs for different patient groups, an urgent patient will get earlier timeslots cheaper than a less urgent patient. There are many strategies[2] that can be applied here; we don't want to specify every possible algorithm this is for further research. But we have defined a certain level of abstraction that allows us to experiment with many different scheduling algorithm s within the same environment using costs to differentiate between timeslots.

3.6 Locking timeslots

One of the problems that will occur if several departments will try to schedule appointments at an auxiliary department at the same time is that they will most likely receive the same timeslots. Because the timeslots are not scheduled immediately and depend on the preferences of the patient, we have to ensure that the provided timeslots are still available once the patient has decided for an option. This is one of disadvantages of decentralized planning.

This problem can be solved if we lock the timeslots for certain amount of time. Once the patient has decided which timeslots to book, the remaining timeslots will be available again. Because we don't want to lock every single available timeslot of a certain resource we will only lock the most likely timeslots to be chosen. The most likely chosen timeslot will be the timeslots that are part of the top 3 combination sorted on earliest available combinations.

The amount of timeslots to be locked should be determined carefully since locking too many timeslots will prevent other departments to retrieve desired timeslots.

If the patient chooses a timeslot that is not within the locked timeslot set, the patient agent has to check first on availability prior to booking the timeslot. Especially for combination appointment we would like to lock the timeslots that are part for the most likely chosen combinations.
Figure 3.6-1 shows a sequence diagram when requesting timeslots from the department agent. The most likely timeslots are locked before the available options are reported back to the patient.

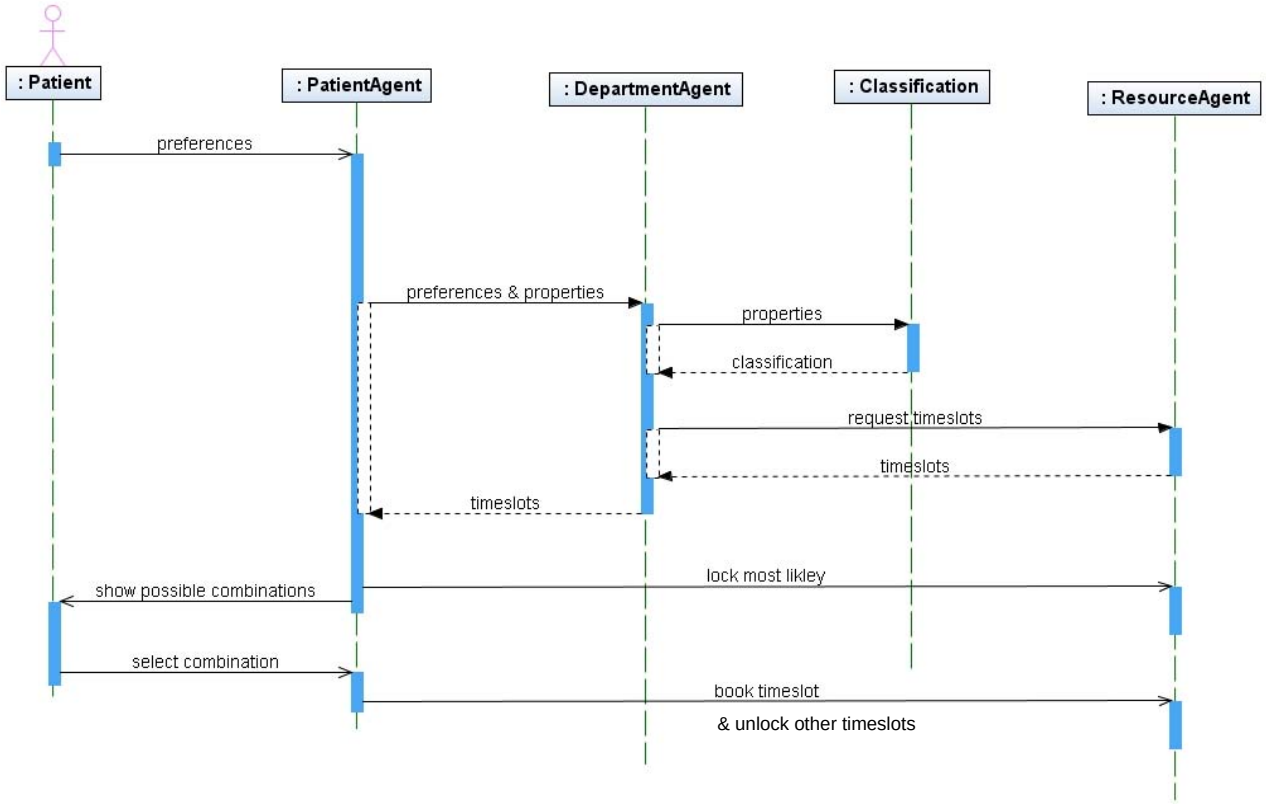


Figure 3.6-1 Sequence diagram requesting timeslots

4 Simulation

Based on the previous discussed models we have programmed a simulation application that enables us to analyze our theories on coordinated multi agent planning. We will use real hospital data and generate patient agents for every single patient that has made an appointment during the year 2007 at the AMC at one of the auxiliary departments. Several simulations with different parameters and scheduling algorithms will be performed.

Before we discuss the simulations and the results of the simulations we would like to mention that we haven't included patient urgency in our scheduling method. Because of this the we can't make any conclusions on how well the system would perform in a more realistic environment with patient urgency. These findings will serve as preliminary investigations and further research has to be done.

4.1 The collection of data for the simulation

Most simulations that have been conducted in the field of multi agent scheduling in hospitals used a poison distribution for its patient arrival. The poison process is stochastic process in which events occur continuously and independently of one another [10]. For most simulations this is a nice way to simulate a patient arrival especially since it is possible to perform several simulations with different arrival patterns. However we have chosen to use real hospital data for our simulations. The advantage of using real hospital data is that we include possible arrival patterns that are not included in a poison distribution. The disadvantage is that we only have a single arrival pattern which makes it difficult to make assumptions for all possible scenarios. Using this hospital data we hope that our simulations will give us insights whether it is feasible or not to implement any of our proposed theories in real hospital setting.

The scheduling software at the AMC uses a huge and complex oracle database. This database has been expended over several years due to constant modification of its structure. The result is a nearly unreadable database which documentation consists of several thousand pages. We experienced the database to be extremely slow. This was also noticed during our observations at the departments when using X/care. For our simulation we decided to implement a new and simpler version of the database, which would allow us to perform simulations much quicker.

With the help of the database administrators at the AMC we managed to extract the information we needed about all the patients, departments and resources we were interested in. Attachment 9 lists all resources we included in our simulation dataset. We imported all the patients that made an appointment at one of these resources in the year 2007 into a new MySQL database. Attachment 7:Database diagram show the new MySQL database.

All appointments at the AMC hospital in the year 2007:

Total amount of appointments:	919,489
Total number of first consults:	93,171 (10%)

All Appointments from the selected resources in the year 2007:

Total amount of appointments:	95,448
Total number of first consults:	12,454 (13%)
Total number of patients:	47,443

4.1.1 Finding combinations

The AMC database misses crucial information about a patient's treatment process. Every appointment at the AMC is stored as a independent single appointment with no relation to other appointments that are within the same treatment process. Since our interest was to improve on combination appointments we need to obtain information about which appointments can be identified as combination appointments.

Therefore we analyzed the registration date of the appointments that were made by the same patient within a certain amount of time:

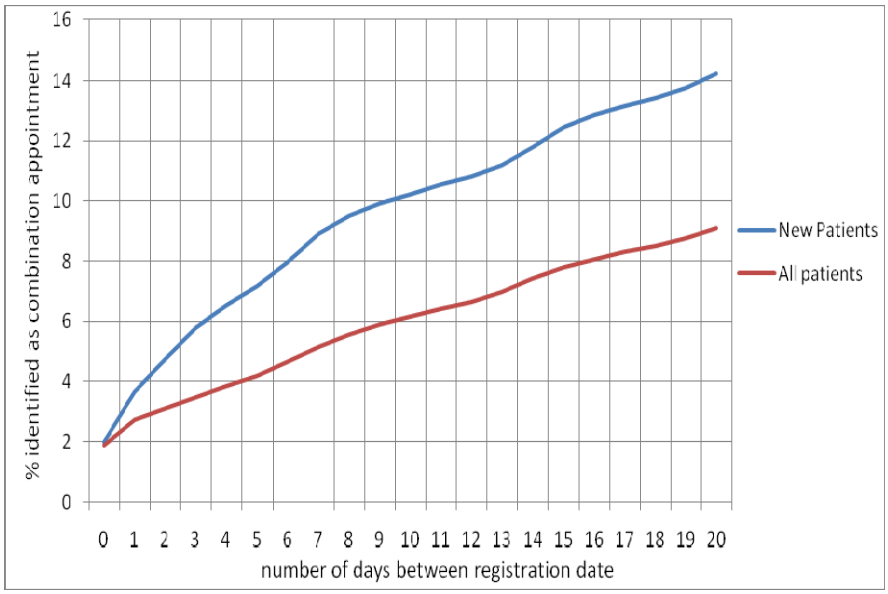


Figure 4.1-1 Identified combination appointments against number of days between registration

The diagram shows the number of identified combination appointments as a total and as part of first consults for new patients. A combination is identified when the appointments of the same patient are scheduled within a certain amount of days between the registrations of the appointment. We can see that new patients have a higher amount of combination appointments. The diagnosis of new patients mostly consists of several test and procedures, so this is an expected result. However we would have expected a higher amount of combination appointments and to see a more significant pattern within a certain amount of days.
The low number of identified combination appointments shows us that we don't have enough information available within the hospital to make correct assumptions about the number of combination appointments. These findings have been discussed with several employees at the hospital and they all state that the real number should be higher although they don't have any hard information to prove this.

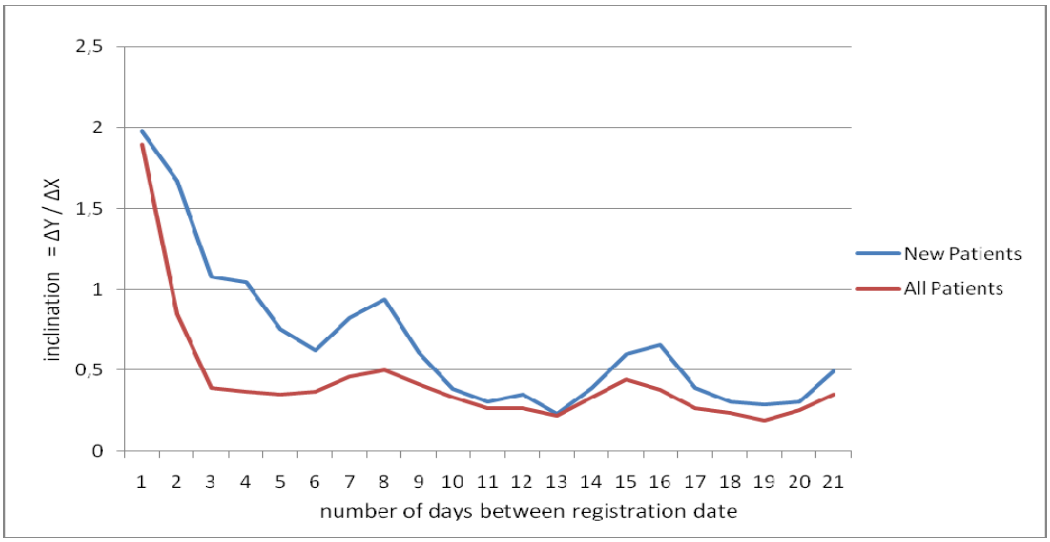


Figure 4.1-2 inclination of identified combination appointments

Figure 4.1-2 shows us the inclination (Δ % identified as combination appointment / Δ number of days between registration date) of *Figure 4.1-1***).** In this diagram we can see a pattern. The highest inclination for new patients is within three days, after that every week we can see a bigger inclination. As we will see in the next chapter this pattern is quite common in a hospital, since hospitals apply a 1 day, 1 week, 2 weeks and 3 weeks urgency classification.

For our simulation we have chosen all appointments which registration date is within three days to be a possible combination appointment, this is mainly based on the experience that forms don't take longer than three days to arrive at the requested department where the appointment is entered in the scheduling system.

4.1.2 Resource schedule

The simulation will also need to know the schedule of every single resource in order to schedule the appointments. The schedules of the resources are defined in X/Care which uses the oracle database. Our aim was to import these schedules from the oracle database. The complexity but also the different definition styles that departments applied to design their schedule made it impossible to automatically import the complete schedule of every single resource from the oracle database. X/Care uses several different techniques to define a schedule. They use layers over layers to define timeframes sometimes there are up to 4 layers. Then there are holidays which also use a different format.

Our approach is to use the appointment data to build a new schedule for every resource. We will use the first and last appointment as the start and end of the day for the scheduled resouce. This has the advantage that it also includes holidays and unexpected unavailability of the resource. We will not differentiate between different types of timeslots on the same resource we will only use a single timeframe a day which should give us more flexibility to coordinate appointments and better efficiency.

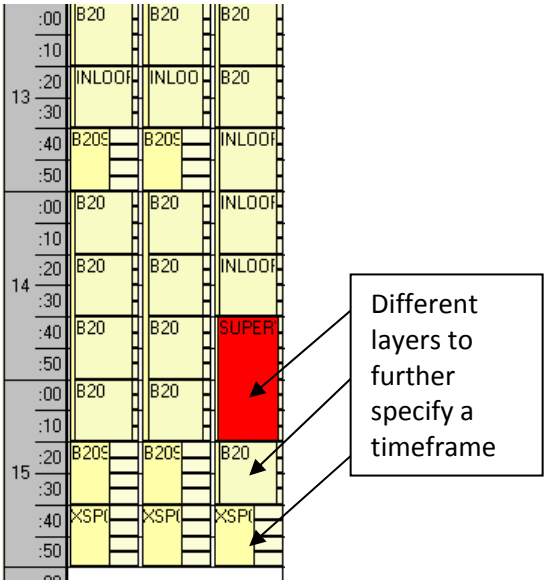


Figure 4.1-3 Layer over layer

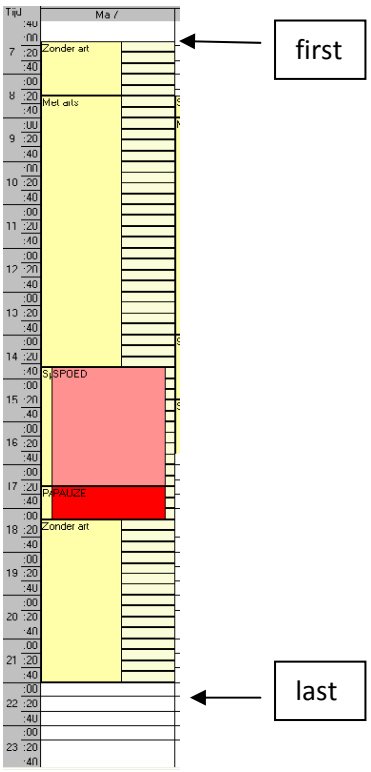


Figure 4.1-4 first and last appointment

Eliminating different types of timeslots is not a realistic representation of the current scheduling process however we think, better results might be achieved by applying this method. The MRI for example already applied this strategy to their scheduling process and based on interviews with schedulers at the MRI it has proven to perform better and give the schedulers more flexibility. It is not proven that this method will be benificial to all departments, it should be carefully invastigated whether this can be applied to all departments.

Most resources could see any kind of patient during the whole day the only reason different type of timeslots have been defined for different patient group is to reserve a certain amount of time for a certain patient group. However the current way of doing this(Figure 4.1-5) restricts the schedulers in their flexibility, a better approach could be to define a certain amount of time to certain patient groups but they can be planed any time of the day. The main advantage for our research is that it will give us a higher probability to coordinate appointments on the same day.

We have to note this approach doesn't include empty inbetween time such as lunch and breakes. Although not all resources have defined breaks we could manually define a certain amount of time a day that will be available to breakes, this way we should have a more realistic simulation. Onother practical issue that our simulation does not implement is the overbooking, we have seen a lot o overbooking at some departments. This maens more patients are seen than are normally possible based on the schedule. The lack of brakes and the overtime are issues that should be implemented in furture research for a more realistic setting.

4.2 Determining the urgency

The scheduling software X/care doesn't store any information about any requests forms. Since the AMC doesn't use digital request forms we have no information about the urgency of an appointment. The only information we have is the date the appointment was registered in the system and the date of the actual appointment, there is no other option to use this as the urgency although we know it is not the real urgency.

4.3 Simulation application

This simulation application has been programmed using the object oriented Java programming language. The main part of the source code can be seen under Attachment 10: Java source code. We have not included the whole source code because a huge part of the code is represents the graphical user interface which is not so interesting. Attachment 8: UML Class diagram also only includes the most relevant classes to our simulation.

Before the patient arrival process starts the simulation will create all the department and resource agents modeled as department class and resource class.

The simulation uses the local MySQL database with a simplified version of the hospital data. We will use a java thread to simulate the patient arrival. The patient arrival is queried in sequential order from our local MySQL database using the appointment registration date. The thread will create a patient object for each patient arrival. This patient object represents a patient agent. The patient agent will request timeslots at the departments, in case of a combination appointment, it will request timeslots from every department involved in the combination. The department agent will check whether shared resources apply to this request and will then request timeslots of its resources that qualify for the requested procedure. The resource agents will apply the scheduling method that can be defined on a simulation, department or resource level. If no specific scheduling method was defined it will use the global simulation scheduling setting. Global simulation settings can be defined on the start screen of the application and will apply to every resource and that don't have any parameters defined. In our simulation the scheduling method is a simple FCFS method that will assign the least cost to the earliest timeslots. The scheduling method object is programmed in such a way that it can easily exchanged by any other algorithm without further adjustments, the only condition a new scheduling method has to comply with is that is has to assign costs to its timeslots it returns to the patient agent.
Once the department agent receives the timeslots of its resource agents it will merge these timeslots into a single list. This list is provided to the patient agent who will sort this list by the cost of the timeslots in an ascending order. In case the patient agent has requested appointments at several department for a combination appointment the patient agent will try to find the best combination with the least total cost. Finally the patient agent will randomly select one of the cheapest timeslots or timeslot combinations out of a defined range. In our simulation we used a set of five cheapest appointments or combinations to randomly choose one, this parameter can be adjusted for every simulation.

The results of the simulations are stored in the local database and can be reviewed during and after the simulation. The application provides a double graph overview to compare certain characteristics of the resource.

We have the following graphs available:

- Waiting time. Difference in days between the registration date and scheduled appointment date.
- Patients on time. Provides a graph which shows per urgency classification the on time %.
- Utilization. Depending on the available capacity of the resource and the scheduled timeslots the utilization per day is shown.
- Urgency. Number of urgent patients per urgency classification is shown.
- Arrival rate. Number of patients that arrive per day.
- Capacity. Total number of schedulable minutes is shown per day.

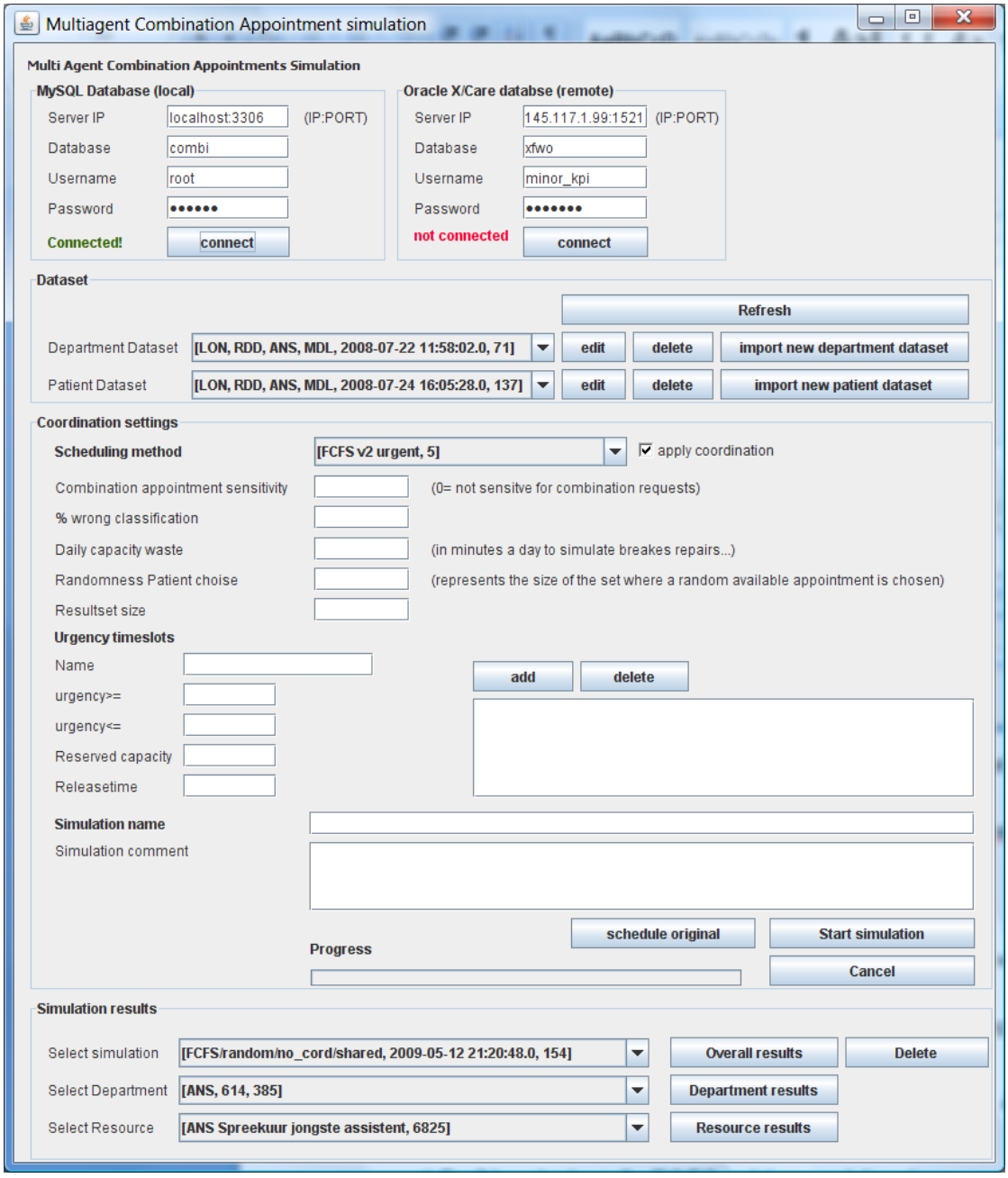


Figure 4.3-1 Main screen of the simulation application

Figure 4.2-1 shows the main application screen. At the top we have two database connection interfaces: local MySQL database connection interface and the remote Oracle database connection interface.

4.4 Simulation parameters

For our simulation we have defined several parameters we think should be included in the simulation because they are all parameters that in practice will have an impact on the overall performance of the scheduling method thus making the simulation more realistic. At every level (simulation, department and resource) parameters can be defined. If no parameters are defined for a resource it will use the department's parameters. If no department parameters are defined it will use the simulation parameters.

We have the following parameters:

- Combination sensitivity. The number represents the discount a patient agent gets if he request appointments that are part of a combination. A discount is represented as a reduction in cost it will assign to its available timeslots. This will increase the probability in finding a combination within the urgency timeframe of the patient. Not all resources will be sensitive to combination appointments.
- % wrong classification. This number represents the % of patients that are classified incorrect by the automatic classification module. We have no information about the possible value of this parameter, but it would represent the number of appointments that are not accepted by the medical doctors of the requested department. A possible value could be 10 %, then 10 out 100 would be classified wrong and possibly have to be rescheduled.
- Daily capacity waste. This parameter represents the amount of minutes a day that taken by to breaks, lunches and repairs. In case of overtime the parameter has a negative value. Some resources don't have any brakes and some resources have a lot of overtime, this parameter should be defined separate for every single resource.
- Randomness Patient choice. Determines the size of the top best possible timeslots where a random timeslot is chosen from. The best timeslots are defined as the cheapest timeslots return form the department agent. For our simulations we have set this value to five, so a random appointment out of five is chosen.

4.5 Simulations

4.5.1 FCFS scheduling method without coordination

The first simulation uses a standard First Come First Serve (FCFS) scheduling method. The patient will get one of the earliest available timeslots. The resource agent will assign a lower cost to earlier timeslots and a higher cost to later timeslots. The cost of a timeslot is calculated as the number of minutes between the date of patient arrival and the appointment date of the available timeslot. This will result that the patient agent will choose one of the earliest (cheapest) timeslots. Our FCFS method does not return a single appointment but a set of appointments that are as early as possible. If we would only return one single appointment this scheduling method would not give us enough opportunities to make a successful combination with other appointments. The FCFS scheduling method is not sensitive for combinations; it will

not make timeslots cheaper because the requesting patient agent is trying to coordinate several appointments.

Shared resources that were defined for this simulation that should be able to handle the same patients within their group:

Group1: (Anesthesiology)
Anesthesiology youngest assistant (ANS spreekuur jongste assistent)
Anesthesiology senior assistant (ANS spreekuur oudste assistent)
Anesthesiology chief of medical staff (ANS spreekuur stafarts)

Group2: (Radiology)
MRI scanner 1
MRI scanner 2
MRI scanner 3

Group3: (Radiology)
CT scanner 1
CT scanner 2

Group4: (Endoscopy)
GE-K132
GE-K213
GE-K219
GE-K120
GE-K127

The randomness Patient choice value for this simulation was set to 5. Meaning a random timeslot of the best 5 timeslots is chosen. This simulation doesn't use daily capacity waste parameter and wrong classification parameter.

For this simulation we have disabled the coordination mechanism of the patient agent, all appointments where scheduled as single appointments.

4.5.2 Simulation 2: FCFS scheduling method with coordination

The second simulation uses exact the same settings and parameters as the first simulation with the difference that the patient agent it will apply coordination. This simulation should show us if coordinating appointments has an impact on the efficiency of the resources. The coordination is done by combining all the retrieved timeslots of the department agents into a single list where the timeslots have a certain minimum and maximum timeframe between each other. This list is then again sorted by the on the cheapest total combination. Attachment Class: Patient shows the coordination method "find combinations()".

4.5.3 Simulation 3: Original scheduling method

The last simulation scheduled the appointments on the date they were originally scheduled. This way we can compare the first two simulations regarding to waiting, utilization and efficiency.

5 Simulation results

Every single simulation gave us information about waitingtime, amount of patients on time, utilization, urgency, arrivalrate and capacity of every resource. We will analyze two different resources that are of main interest to us, the MRI because of its capacity problems and the Anesthesiology because we replaced the fragmented schedule with a singel timeslot a day.

5.1 MRI 2 results

We will first analyze the waiting times of the different simulations. The waiting time is calculated as the difference in days between the the date of registration and the first appointment made. *Figure 5.1-1* shows us the result of the first simulation, we can see the a FCFS pattern, the waiting time is high at the beginning and then reaches 0 from day 75. The second simulation with coordination (*Figure 5.1-2*) shows nearly the same pattern, aparerently the extra coordination effort has no impact on the waiting compered to the first simulation. The last simulation represents the original appointment data, we see a completely different pattern. Our FCFS method didn't take into account the urgency of the patients and just tries to schedule as soon as possible. Especially for the MRI urgency is very important, therefore this scheduling mehtod will not be usable to the MRI in practice.

Figure 5.1-1 Waiting time MRI 2 FCFS without coordination with shared resources

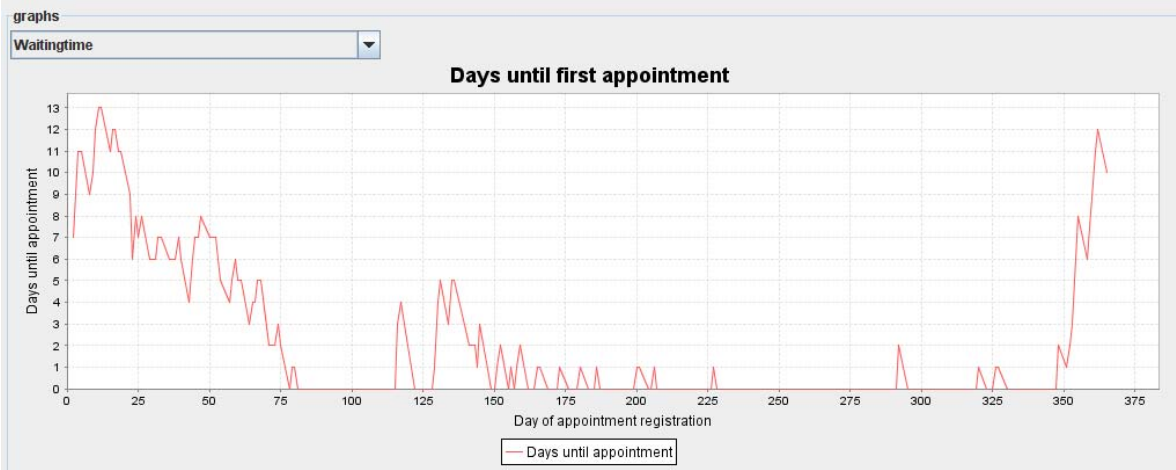


Figure 5.1-2 Waiting time MRI 2 with FCFS coordinated and shared resources

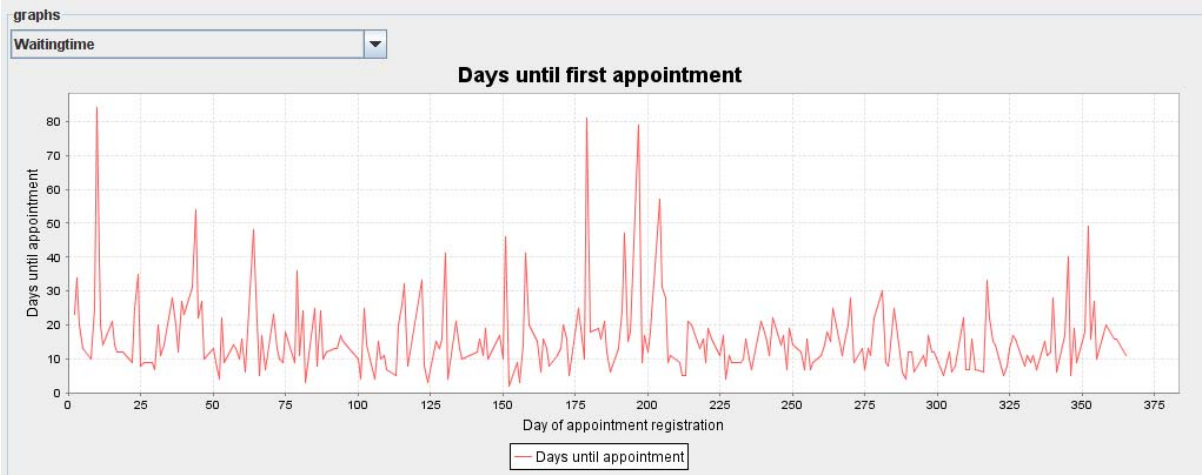


Figure 5.1-3 Waiting time MRI 2 with the original hospital data

To better understand the long waiting time at the beginning and end of the year we have to look at the capacity and patient arrival of the resource during that period. As we can see in *Figure 5.1-4* the capacity of the MRI 2 resource is low at the beginning and and of the year while the patient arrival is steady. It seems that during christmas and new year the MRI is closed or has limited opening hours during holidays. In Fact it takes the FCFS scheduling algoithm 75 days to recover from this period with limited availabilty. The capacity is shown in minutes available a day and is calculated by the time between the first and last apppointment that where scheduled in history records.

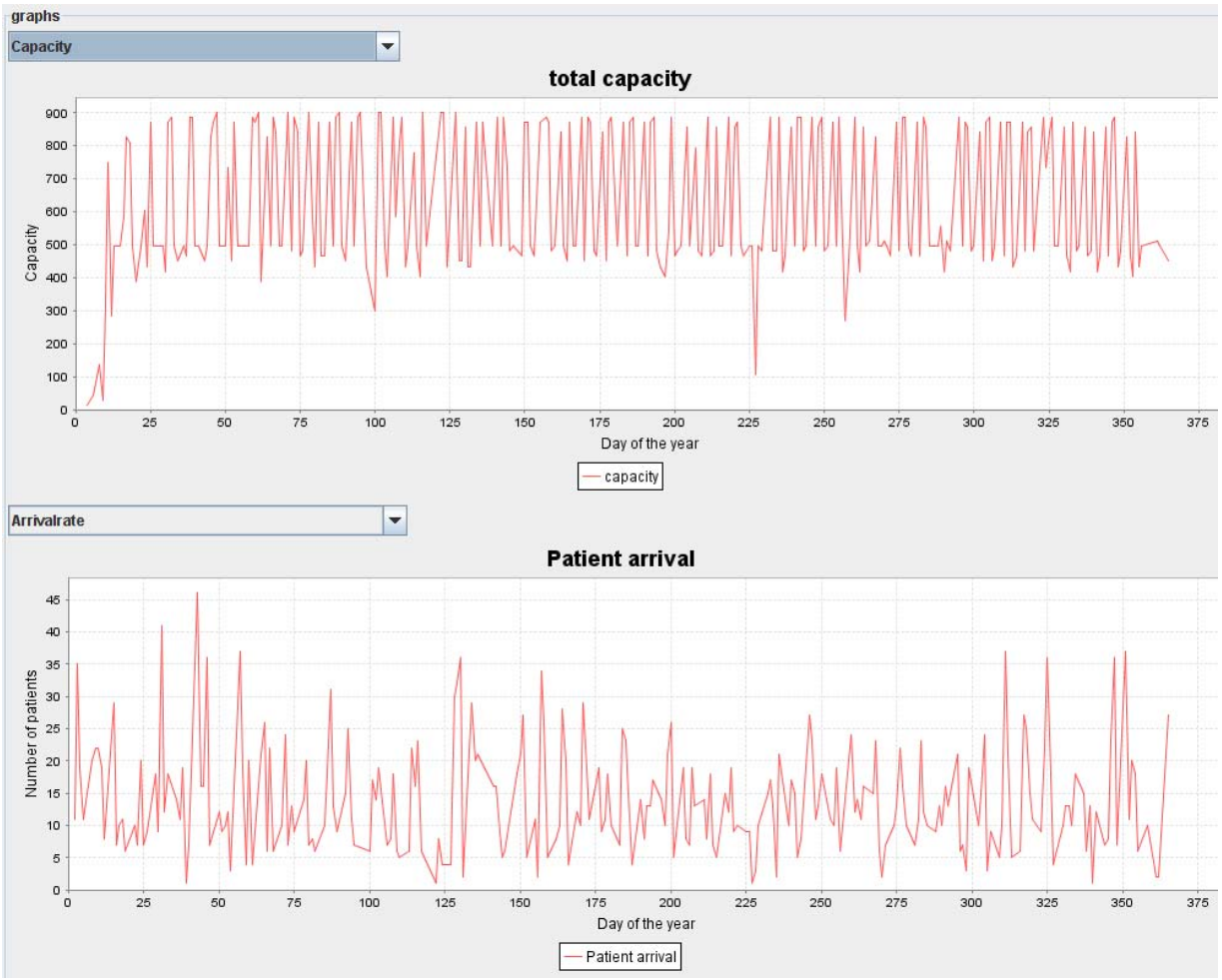


Figure 5.1-4 Resource capacity and Patient arrival of MRI 2

These results show us that if we would have had adequate capacity during holiday seasons we could eliminate waiting times and even implement a FCFS scheduling method at the MRI serving every patient on time. Not having enough capacity at certain holiday seasons is quite common in hospitals and will properly remain a problem.

The utilization graph (Figure 5.1-5) of the first and second simulation with resource MRI 2 shows us that until day 75 we have a nice utilization of nearly100%.

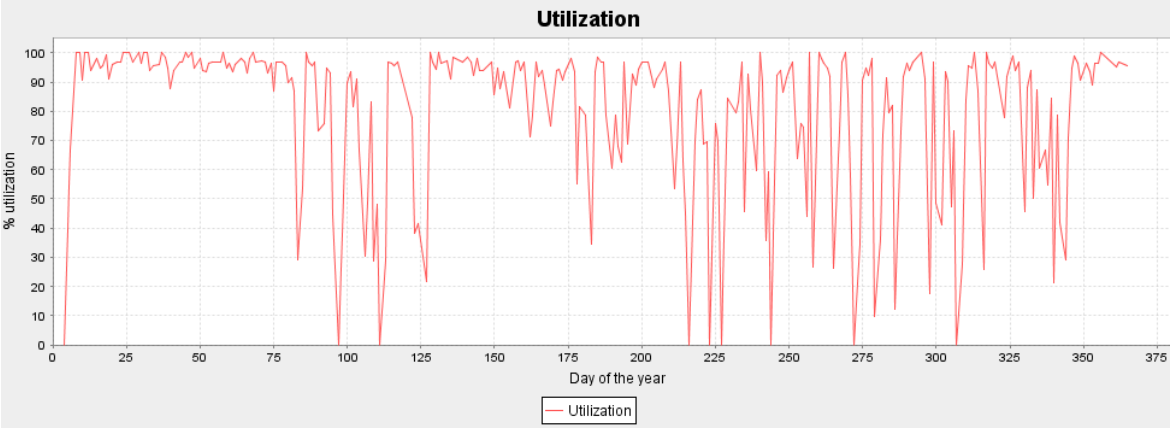


Figure 5.1-5 MRI 2 utilization FCFS method coordinated appointments

The simulation application also provides us with information about the estimated urgency classification per resource calculated from the difference in days between date of registration and date of original appointment. For MRI 2 we can (*Figure 5.1-6*) recognize a certain pattern, every seven days we have pike in the graph. This is the result of the classification system used by hospitals who will schedule a patient with urgency >3 weeks exactly in three weeks for example.

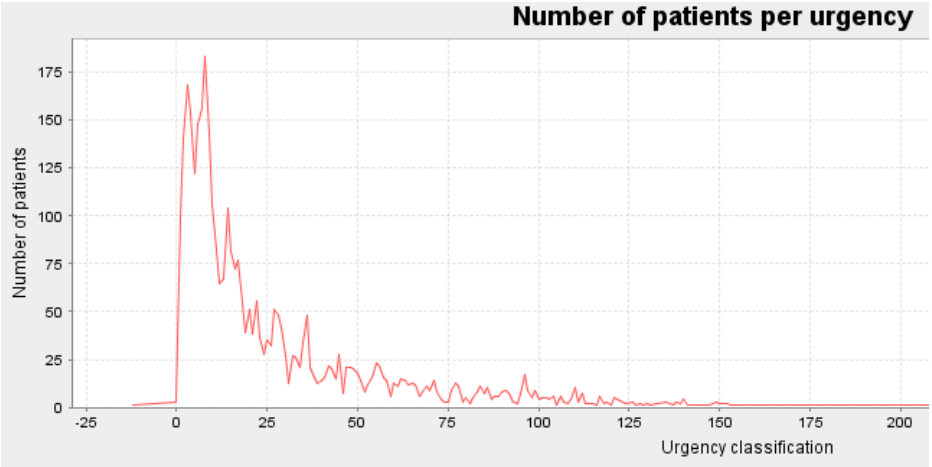


Figure 5.1-6 MRI 2 urgency classification

5.2 Anesthesiology results

For the anesthesiology we had some very interesting results. With both simulations FCFS and FCFS with coordination we could completely eliminate waiting time from day one. During the whole year we could serve every patient one time with FCFS meaning immediately. The first reason for this is the single time slot a day without any urgency classification. It prevents unused timeslots, increasing the efficiency. Second we load balanced all Anesthesiology request over the three available resources that are all capable of performing the same requests. This leveled out any fluctuations in resource patient arrival and the available resource capacity.

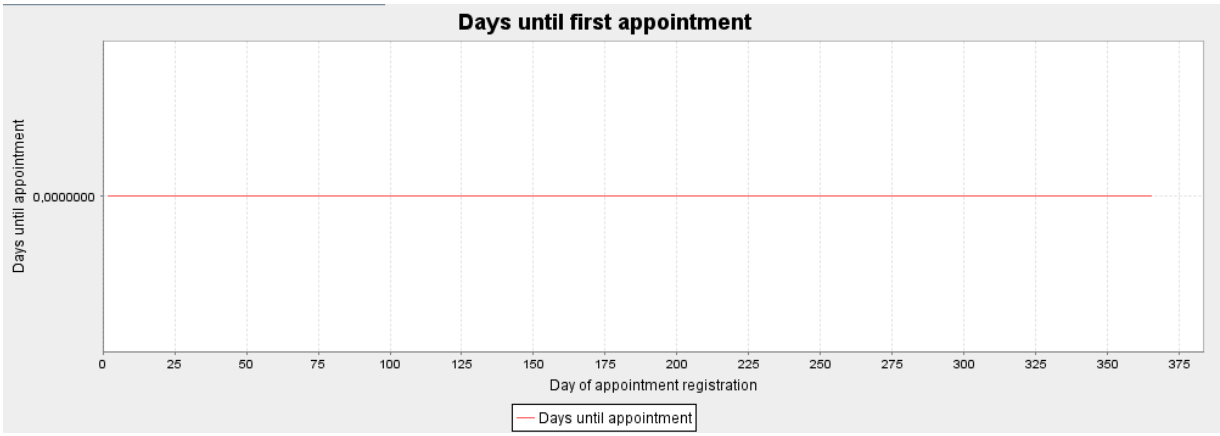


Figure 5.2-1 Waiting time at the Anesthesiology. With FCFS method using shared resources and a single timeslot per day. (junior assistant, senior assistant and chief of medical staff all have the same results in our simulation)

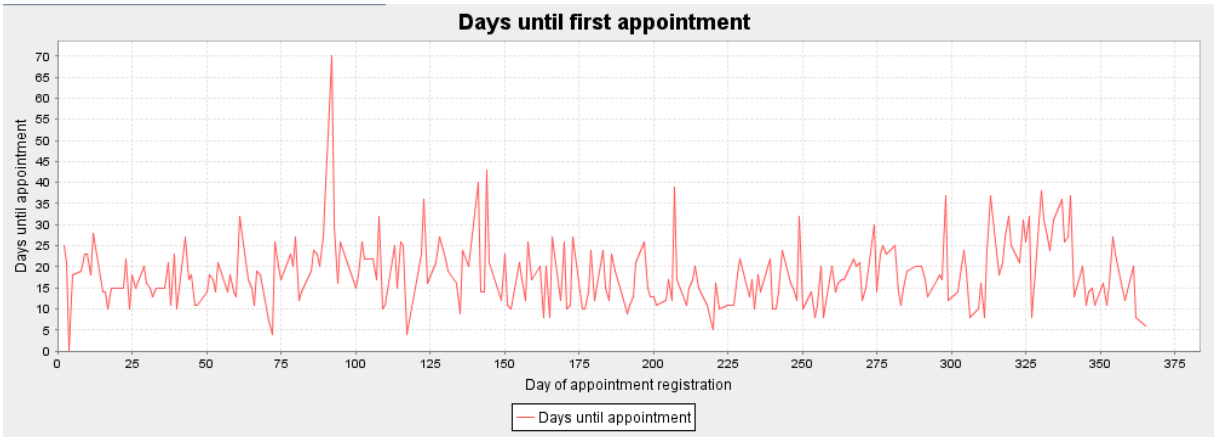


Figure 5.2-2 Waiting time at the Anesthesiology. Original hospital data during the year 2007.

5.3 Coordination results

In total we had identified 1437 combinations of appointments, which is only about 3% of the total appointments made during the year 2007 of the selected resources in our study.

The second simulation FCFS with coordination managed to successfully coordinate 1436 of 1437 combinations (99,9%) on the same day. If we compare this with the original hospital data which was only able to schedule 322 combinations (22,4 %) on the same day. These results can be explained by the coordination friendly environment we created. Following changes in the scheduling procedure had influence on the well performing coordination environment:

- Shared resources
- Single timeslot a day
- Better resource utilization and less patient waiting time
- No patient classification nor urgency applied, this is a crucial environmental parameter which makes our simulation application a toy application.

5.4 Single time slot per day efficiency

One of the concerns we had when implementing a single timeslot a day was that it could create empty spaces between appointments which would cause an inefficient resource usage. In our simulations we implemented FCFS with a random factor of the top 5 best timeslots. This random factor has discussed in the previous section as a simulation parameter. We expected this to result in some kind of inefficiency since the appointments are not always scheduled one after another.

We implemented the scheduling algorithm (Attachment 10.7 FCFS2 method) in such a way that no empty spaces were created. The reason for this is rather simple but very effective. The scheduling algorithm queried the total time available and then divided the duration of timeframe by the duration of the requested appointment, which returns the number of appointments that are schedulable. In *Figure 5.1-1* we can see that with the MRI 2 for example it results in very efficient schedule, however since all timeslots are a multitude of 15 minutes it is not that surprising. In case timeslots are not a multitude of each other one could apply a algorithm that will always ensure that no smaller timeslots are created than is necessary for the shortest procedure for that resource.

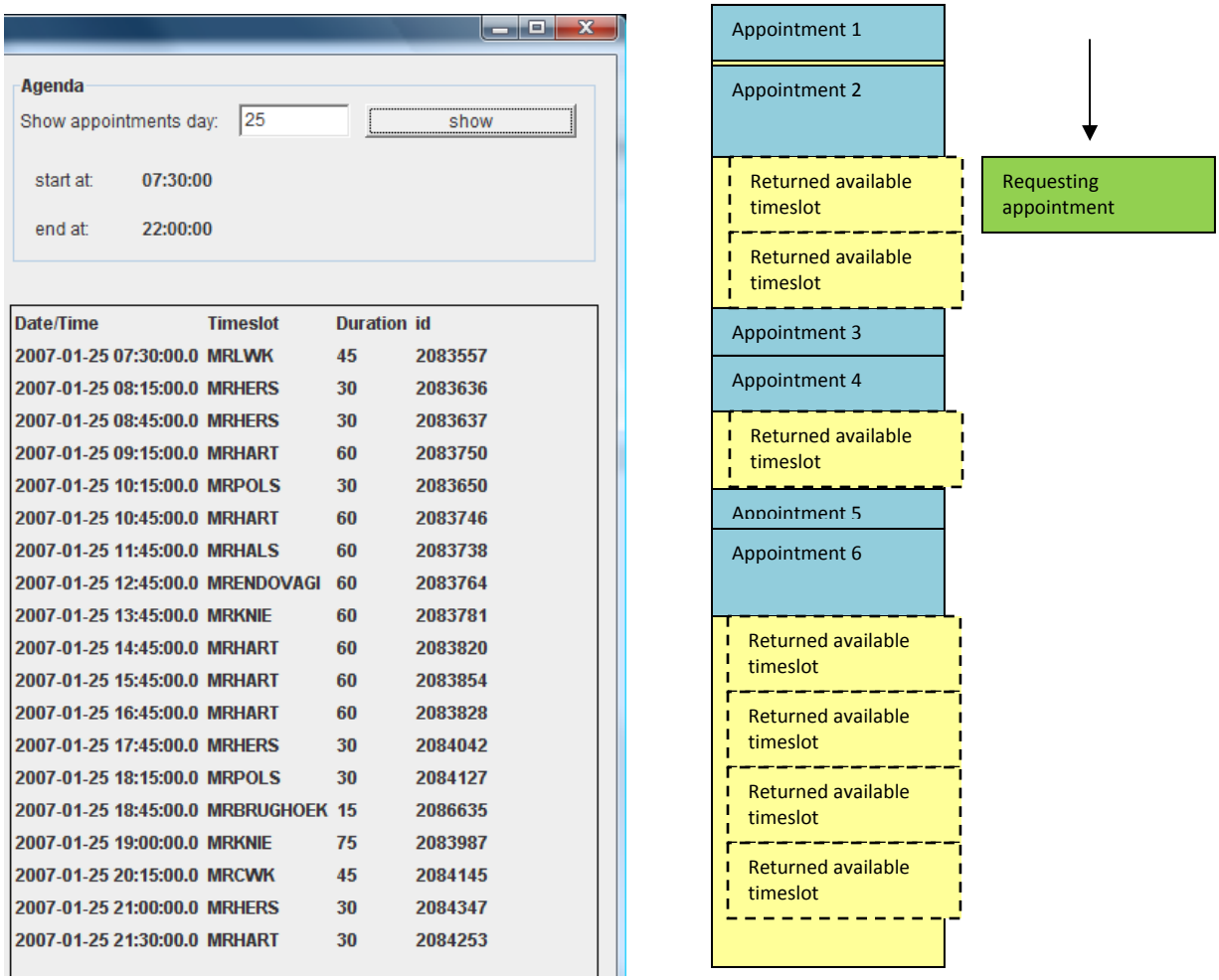


Figure 5.4-1 Schedule of MRI 2 on day 25 even with random timeslots no empty spaces

6 Conclusion and recommendations

The primary aim of our study was to research if coordinated scheduling would be possible without a decrease in the efficiency of the hospital resources. We analyzed the current hospital scheduling process and addressed several issues that could be implemented to further improve their efficiency and at the same time would enable us to coordinate appointments. As a result our implemented simulation application shows us some interesting results.

We only identified about 3% appointments as a possible combination. This is a rather low percentage we expect that the real number would be substantially higher. Because the hospital doesn't store any information about the treatment plan of their patients we are not able to retrieve the real number of possible combination appointments.

Our simulation uses a single thread that sequentially creates patients agents. A more realistic simulation would be to create a new java thread for every single created patient agent so they can perform their request in parallel. Patients which appointments have been registered on the same day should all be created in parallel to simulate any possible problems that may occur. One of the issues that would be addressed this way would be the locking of timeslots as we have been discussing in the previous chapter. We did not implement this because it is a far more complex and demanding simulation which would take more time. For further research we would recommend to implement a system with multiple threads to analyze its behavior.

We would like to recommend the following issues to reconcider when implementing a new scheduling software at the AMc hospital:

1. Introduce digital request forms. It would provide valuable information about patients urgency and whether or not a patient is served whithn its urgency classification. Further more it would be the first step to automatic classification. Digital request forms could provide information about the outpatient doctors capabilities classifying patients not of their own specialism.

2. Reducing the number of different types of timeslots of a single resource. It will provide better scheduling flexibility and better coordination posibilities.

3. Combine resources that are capable performing the same procedures to reduce unequal distribution between patient arrival and resouce capacity. It will increase better coordination possibilities, better efficiency and shorter waitingtimes.

4. A Multi agent system is better suited for hospital enviroment that the currently used scheduling software. Implementing a multi agent system would bring several opportunities:
    a. Real time information, monitoring and alerting of resource efficiency and waiting times.
    b. Beter patient service by letting the patient change appointments or define preferences using a webinterface that instructs the patient agent.

c. Would enable automatic classification. Agents can autonomiously decide whether or not certain doctors a capable of classifying patient whithout the interaction of a medical doctor from the requested department.

6.1 Possible simulations for further research

With the simulation application it is possible to implement different scheduling methods with different parameters for every single resource. Depending on the patient arrival pattern and their urgency a resource should be adjusted to optimally serve its patients within their urgency classification. For this kind of simulation we would need information about the real urgency of patient which is not available at the current time.

Different scheduling methods would all attach a cost value to their available timeslots. It would be interesting to so see how the resource agents and department agents would perform with different scheduling methods within the same environment.

Adding an automatic classification value would be the next step in a more realistic setting for the simulations. Currently there is no information available about the degree in which doctors are able to perform classifications that are not within their own specialism. However we could still perform simulations with a random value, for example 20% is classified wrong, to see what the impact would be.

6.2 Research question

"Is it possible to implement a scheduling system at the AMC hospital that will enable us to automatically coordinate different appointments of a patient on the same day without any impact on the departments' performance?"

Our results is a first step in showing that it is possible to use a multi agent system for coordinating appointments within a hospital and it performs very well. But since we haven't included patient urgency in our simulation application yet we strongly recommend further research should implement a scheduling algorithm that takes into account the urgency of the patient. Once the urgency has been implemented into the scheduling method we can evaluate how many patients are served within their urgency timeframe.
But even if this scheduling method is implemented, the hospital currently has no digital information available on the real urgency of their patients. The urgency we extracted from the hospital was calculated as the difference in time between appointment registration and the actual appointment, this is also not the real urgency.
For a realistic simulation application the hospital must provide information about the real urgency.

7 References

[1] I.B. Vermeulen, S.M. Bohte, D.j.A Somefun, and J.A. La Poutré , Multi-agent Pareto appointment exchanging in hospital patient scheduling**,** *Service Oriented Computing and Applications(2007)1:185--196,* 2007.

[2] I.B. Vermeulen, S.M. Bohnte, S.G. Elkhuizen, J.S.Lameris, P.J.M. Bakker, and J.A. La Poutré, Adaptive Optimization of Hospital Resource Calendars**,** *Proceedings of the 11th conference on Artificial Intelligence in Medicine:305-315,* 2007.

[3] Decker K, Li J, Coordinating mutually exclusive resources using GPGP, *Auton Agent Multi Agent Syst 3(2):133-157,* 2000.

[3] Anke K. Hutzschenreuter et al, Agent-based patient admission scheduling in hospitals, *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems: industrial track:45-52*, 2008.

[4] Vili Podgorelec and Peter Kokol, Towards More Optimal Medical Diagnosing with Evolutionary Algorithms, *Journal of Medical systems 25(3)*:195-219, 2001.

[5] Pieter Buzing et al., Coordinating self-interested planning agents, *Auton Agent Multi-Agent Sys (2006)12:199-218,* 2006.

[6] Jan M.h. Vissers, Patient flow-based allocation of inpatient resources: a case study, *European Journal of Operational Research 105(1998):356-370*, 1998.

[7] Neal J, Moreno A, Agent-based applications in health care, *Nealon J, Moreno A(eds) applications of software agent technology in the health care domain. Birkhueser*, pp 3-18, 2003.

[8] M.Wooldridge and N.r. Jennings. Intelligent agents: Theory and practice. *The Knowledge Engineering Review, 10(2)*:115-152, 1995.

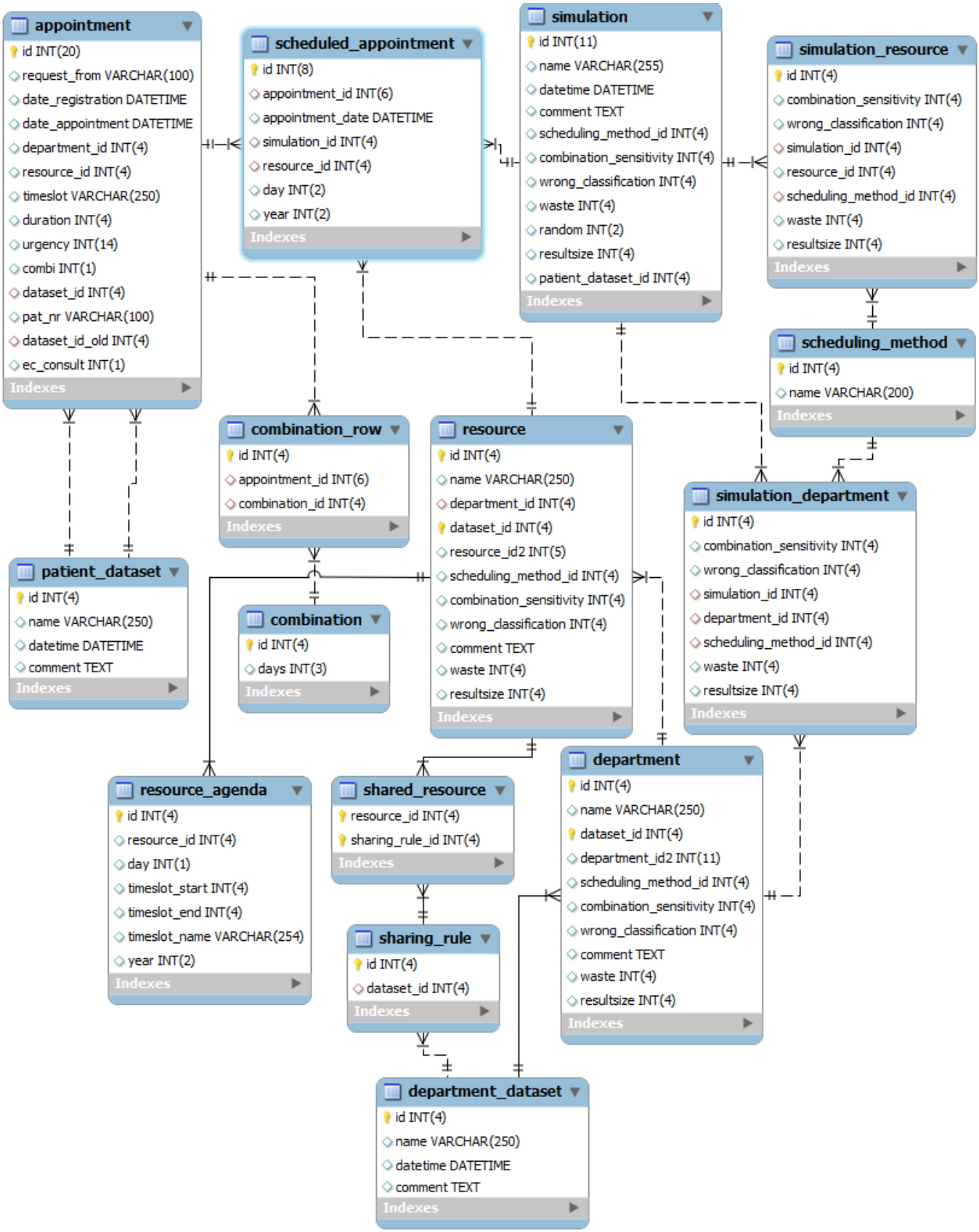
[9] Mathijs de Weerdt, Adriaan ter Mors, Cees Witteveen, *Multi-agent Planning: An introduction to planning and coordination*, 2005.

[10] Poisson process, http://en.wikipedia.org/wiki/Poisson_process

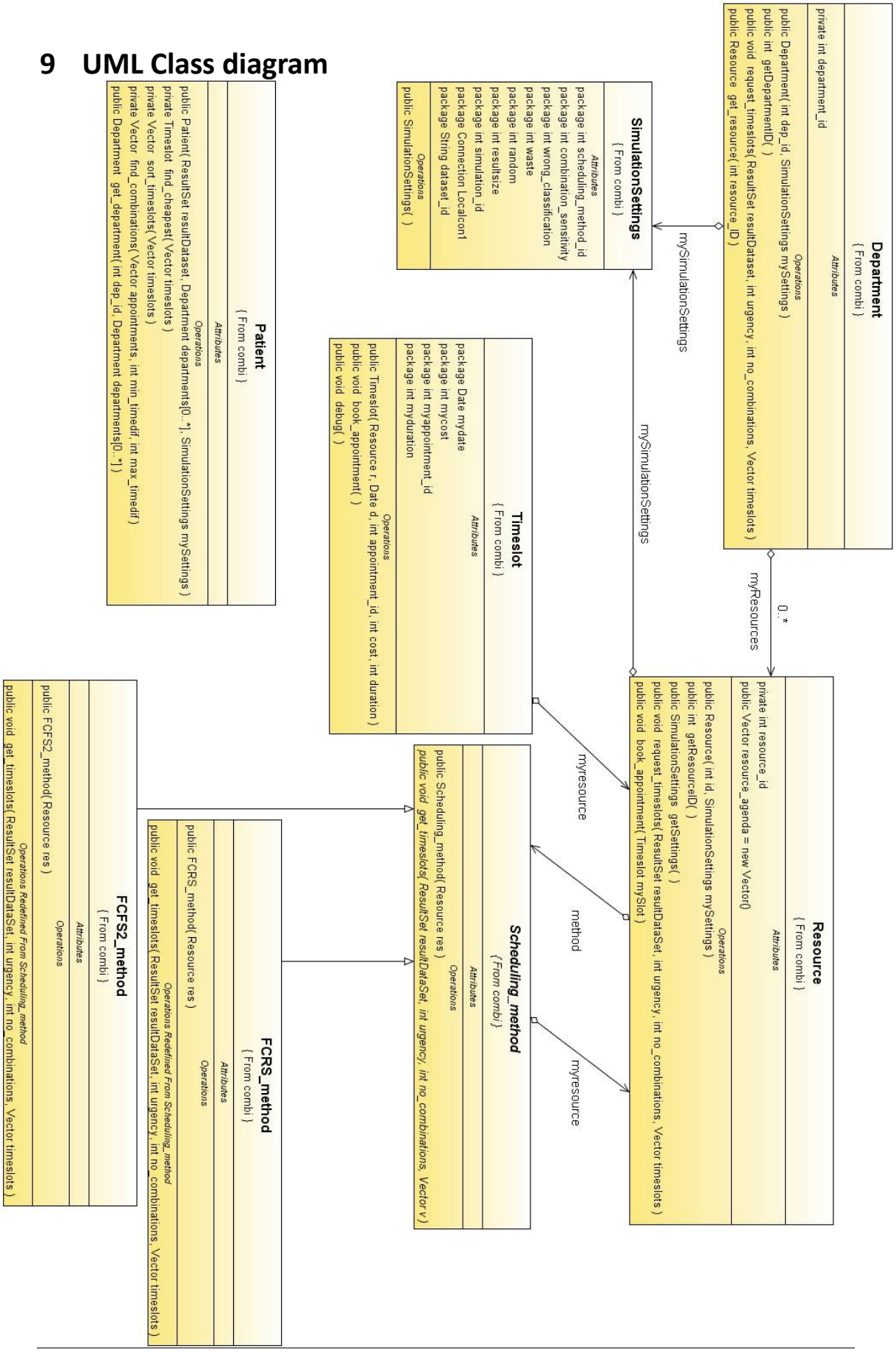
[11] N.M. van Dijk, "To pool or not to pool? "The benefits of combining queuing and simulation"," wsc, vol. 2, pp.1469-1472, 2002 Winter Simulation Conference (WSC'02) - Volume 2, 2002

List of figures

8 Database diagram



9 UML Class diagram



10 Resources used with simulation

Anesthesiologie

147 ANS Spreekuur jongste assistent
148 ANS Spreekuur oudste assistent
149 ANS Spreekuur stafarts

Radiologie

150 conventioneel km 1
151 conventioneel km 2
152 conventioneel km 3
153 conventioneel km 4
154 ct scan 1
155 ct scan 2o
156 echo mammo kamer 3
157 echografie kamer 1
158 echografie kamer 2
159 echografie kamer 4
160 echografie kamer 5
161 echografie kamer 6
162 echografie kamer 7
163 GE kamer 1
164 GE kamer 2
165 interventie kamer 1
166 interventie kamer 2
167 mammografiekamer
168 mri scan 1 - F0 418
169 mri scan 2 - C1 418
170 mri scan 3- C1 418
171 mri scan 4 - B1 418

Endoscopie

172 GE-K120
173 GE-K124
174 GE-K127
175 GE-K132
176 GE-K213
177 GE-K218
178 GE-K219

Longfucntie

179 Masterlab, 416: spiro + dlco + box + monddrukken
180 MS Body, 138: spiro + box (kind)
181 MS Body, 417: spiro + box
182 MS Capno, 422: spiro + capnografie
183 MS Diffusion, 417: spiro + dlco + monddrukken
184 MS FRC, 140: spiro + dlco + helium (kind)

185 MS Oxycon Pro, 424: ergometrie
186 MS PFT, 416: spiro + dlco + helium
187 MS PFT, 417: spiro + dlco + helium
188 MS Pneumo, 138: spiro + histamine + io (kind)
189 MS Pneumo, 416: spiro + histamine
190 MS Pneumo, 417: spiro + histamine
191 Remstar Autopap 1
192 Remstar Autopap 2
193 Remstar Autopap 3
194 Remstar Autopap 4
195 Remstar Autopap 5

11 Java Source Code

11.1 Class: Patient

```
package combi;

import java.sql.*;
import java.lang.*;
import java.util.Random;
import java.util.Vector;
/**
 *
 * @author Robin
 */
public class Patient {

  public Patient(ResultSet resultDataset,Department[] departments,SimulationSettings mySettings) {

  int combination_id;
  int no_combinations=0;
  String patient_id;
  Vector appointments = new Vector();


      try {


         combination_id = resultDataset.getInt("combination_id");
         patient_id = resultDataset.getString("pat_nr");
         int urgency=resultDataset.getInt("urgency");

         //get max urgency and no_combinations by looking at the next record in the result set
         do{
            no_combinations++;
            if(resultDataset.getInt("urgency")>urgency)
               urgency=resultDataset.getInt("urgency");

            resultDataset.next();
         }while ((resultDataset.getInt("combination_id")!=0)&&(combination_id ==
resultDataset.getInt("combination_id"))&&(patient_id.equals(resultDataset.getString("pat_nr"))));



         // go back number of combination records in the resultset
         for(int l=0;l<=no_combinations;l++)
            resultDataset.previous();



        // urgency=0;


         //loop through combinations and request appointments at departments
         for(int l=0;l<no_combinations;l++){
            resultDataset.next();

             //request appointments from department and add to list of appointments
            Vector timeslots = new Vector();

get_department(resultDataset.getInt("department_id"),departments).request_timeslots(resultDataset,urgency,no_combinations,ti
meslots);
```

```
            appointments.add(timeslots);
            System.out.println("Request appointment at department: "+resultDataset.getInt("department_id"));

        }



        // if two or more appointments then coordinate these appointments and make a new list
        if(no_combinations>=2){
            System.out.println("coordinate");

            //find combinations returning a list with total cost of combination and the timeslots of a combination ordery by cost in a
ascending order
            Vector combination_list=find_combinations(appointments,30,120);

            //if no timeslots can be matched book them seperetaly
            if(combination_list.size()==0){
                for(int i=0;i<no_combinations;i++){

                    //get the timeslots and sort them by costs
                    Vector timeslots=sort_timeslots((Vector)appointments.get(i));


                    //select a random timeslot
                    int randomIndex;

                    Random generator = new Random();
                    int seed;

                    //if less then random resultset size then adjust the random to the size of the results
                    if(timeslots.size()<mySettings.random){
                        seed=timeslots.size();
                    }
                    else
                        seed=mySettings.random;

                    randomIndex = generator.nextInt(seed);
                    //System.out.println("randomindex:"+randomIndex);

                    //book randomly picked timeslot
                    Timeslot mySlot = (Timeslot)timeslots.get(randomIndex);

                    //Timeslot mySlot = find_cheapest(timeslots);


                    //Timeslot mySlot = (Timeslot)timeslots.get(0);
                    mySlot.book_appointment();


                }
            }
            else{

            /*
            //find the cheapest combination
            int cheapest=0;
            for(int i=0;i<=combination_list.size();i++){
                Vector combination=(Vector)combination_list.get(i);

                if((Integer)combination.get(0)<cheapest){
                    cheapest=i;
                }

            }
```

```
   */

      Random generator = new Random();
      int seed;

      //if less then random resultset size then adjust the random to the size of the results
      if(combination_list.size()<mySettings.random){
         seed=combination_list.size()-1;
      }
      else
        seed=mySettings.random;

      int randomIndex = generator.nextInt(seed);
      System.out.println("randomindex:"+randomIndex);

      // get combination record
      Vector combination =(Vector)combination_list.get(randomIndex); //get combination record
      Vector timeslots=(Vector)combination.get(1); //get timeslots of the combinations

      //book the combinations
      for(int i=0;i<timeslots.size();i++){
         Timeslot mySlot=(Timeslot)timeslots.get(i);
         mySlot.book_appointment();
         mySlot.debug();  //debug information

      }

    }

}
else{
  //randomly pick an appointmet
   // System.out.println("choose appointment");

   //get the timeslots and sort them by costs
   Vector timeslots=sort_timeslots((Vector)appointments.get(0));


   //select a random timeslot
   int randomIndex;

   Random generator = new Random();
   int seed;

   //if less then random resultset size then adjust the random to the size of the results
   if(timeslots.size()<mySettings.random){
      seed=timeslots.size();
   }
   else
      seed=mySettings.random;

   randomIndex = generator.nextInt(seed);
   System.out.println("randomindex:"+randomIndex);

   //book randomly picked timeslot
   Timeslot mySlot = (Timeslot)timeslots.get(randomIndex);

   //Timeslot mySlot = find_cheapest(timeslots);


   //Timeslot mySlot = (Timeslot)timeslots.get(0);
   mySlot.book_appointment();


}
```

```
    }
    catch(Exception e) {
            //textoutput.append("\nException: " + e.getMessage());
            System.err.println("Exception: " + e.getMessage());
    }



}


private Timeslot find_cheapest(Vector timeslots)
{
   Timeslot t1;
   t1=(Timeslot)timeslots.get(0);
   int cost=t1.mycost;
   int k=0;

   for(int j=0;j<timeslots.size();j++){
    t1 =(Timeslot)timeslots.get(j);
    if(cost>t1.mycost){
       cost=t1.mycost;
       k=j;
    }
   }

 return (Timeslot)timeslots.get(k);
}

private Vector sort_timeslots(Vector timeslots){

   //---- Insert sort algorithm---------- sort by cost ascending
   for(int i=1;i<=timeslots.size()-1;i++){

    Timeslot temp=(Timeslot)timeslots.get(i);

    int j=i-1;
    while((j>=0)&&((((Timeslot)timeslots.get(j)).mycost)>temp.mycost)){

       timeslots.set(j+1,timeslots.get(j));
       j--;
    }
    timeslots.set(j+1,temp);
   }


   return timeslots;
}

private Vector find_combinations(Vector appointments,int min_timedif,int max_timedif){


        // new vector containing the mathching timeslots
        Vector combination_list = new Vector();

        //get the timeslots of first retrieved set
        Vector timeslots=(Vector)appointments.get(0);

        // find other timeslots in sets received form other departments
        Timeslot t1,t2;
```

```
Vector timeslots2,combination,combination_record;
int cost;

//main loop first set
for(int j=0;j<timeslots.size();j++){


  t1 =(Timeslot)timeslots.get(j);

  //vector to hold list of timeslots
  combination = new Vector();

  //vector to hold total cost and the list of timeslots that can be combined
  combination_record = new Vector(2);

  // initialse cost of the total combination
  cost=0;

  //loop within other sets
  for(int i=1;i<appointments.size();i++){
    timeslots2=(Vector)appointments.get(i);

    //loop within selected other set
    for(int k=0;k<timeslots2.size();k++){

      //get timeslot to compare
      t2=(Timeslot)timeslots2.get(k);


      //calculate time difference in ms, then calculate to minutes
      int time = Math.abs(((int)(t1.mydate.getTime()-t2.mydate.getTime())/60000));


      //substract the duration of the appointment
      if(t1.mydate.getTime()>t2.mydate.getTime())
        time-=t2.myduration;
      else
        time-=t1.myduration;

      //if within timeframe break loop and add to list---- presuming first found is cheapest when timeslots are ordered
      if(time>=min_timedif&&time<=max_timedif){
        cost+=t2.mycost;
        combination.add(t2);
        break;
      }
    }
  }

  //if the size of combination matches the no of combination than add to the list
  if(combination.size()==(appointments.size()-1)){

    //add the first record
    combination.add(t1);

    //add the cost of the first record
    cost+=t1.mycost;

    //add total cost
    combination_record.add(cost);
    //add the timeslots
    combination_record.add(combination);
    //add combination record containg list of timeslots that can be combined and their total cost
    combination_list.add(combination_record);
  }
}
```

```
    //sorting the combination list by cost in ascending order   ---- Insert sort algorithm ------
    for(int i=1;i<=combination_list.size()-1;i++){

      Vector temp=(Vector)combination_list.get(i);

      int j=i-1;
      while((j>=0)&&(((Integer)((Vector)combination_list.get(j)).get(0))>((Integer)temp.get(0))) ){

         combination_list.set(j+1,combination_list.get(j));
         j--;
      }
      combination_list.set(j+1,temp);
    }



    return combination_list;
  }


  public Department get_department(int dep_id,Department[] departments){

    //find corresponding department in array
   for(int j=0;j<departments.length;j++){
      //find department object
      if(dep_id==departments[j].getDepartmentID())
         return departments[j];
   }

   return null;
  }



}
```

11.2 Class: Department

```
package combi;

import java.sql.*;
import java.util.Vector;

/**
 *
 * @author Robin
 */
public class Department {
private Resource[] myResources;
private int department_id;
SimulationSettings mySimulationSettings;

   public Department(int dep_id,SimulationSettings mySettings){
      //init all resources
      department_id=dep_id;

      mySimulationSettings = mySettings;

      try {

         // Create a statement to send SQL
         Statement db_statement1 = mySimulationSettings.Localcon1.createStatement();
         Statement db_statement2 = mySimulationSettings.Localcon1.createStatement();
         String query1;

         //SELECT resources
         query1="select * from resource WHERE department_id="+ department_id+" AND dataset_id="+
mySimulationSettings.dataset_id;
         System.out.println("MySQL query: " + query1);
         ResultSet resultDataset = db_statement1.executeQuery(query1);
         if(resultDataset==null)
            System.out.println("MySQL null: " + query1);

         //create new resources array with the size of the fetched result
         resultDataset.last();
         myResources = new Resource[resultDataset.getRow()];
         resultDataset.beforeFirst();

         //loop through resource results
         while(resultDataset.next()){

            //----------------------------INSERT NEW RESOURCE SIMULATION RECORD----------------------------------
            query1="INSERT INTO
simulation_resource(combination_sensitivity,wrong_classification,simulation_id,resource_id,scheduling_method_id,waste,resultsize
)
VALUES("+resultDataset.getInt("combination_sensitivity")+","+resultDataset.getInt("wrong_classification")+","+mySimulationSetting
s.simulation_id+","+resultDataset.getInt("id")+","+resultDataset.getInt("scheduling_method_id")+","+resultDataset.getInt("waste")+
","+resultDataset.getInt("resultsize")+")";
            int resultDataset2 = db_statement2.executeUpdate(query1);
            if(resultDataset2==0)
              System.out.println("\nMySQL null: " + query1);


           //create new settings
           SimulationSettings mySimulationSettings_res = new SimulationSettings();



           mySimulationSettings_res.random=mySimulationSettings.random;
           mySimulationSettings_res.Localcon1=mySimulationSettings.Localcon1;
           mySimulationSettings_res.simulation_id=mySimulationSettings.simulation_id;
```

```
            if(resultDataset.getInt("scheduling_method_id")==0){
              mySimulationSettings_res.combination_sensitivity=mySimulationSettings.combination_sensitivity;
              mySimulationSettings_res.resultsize=mySimulationSettings.resultsize;
              mySimulationSettings_res.scheduling_method_id=mySimulationSettings.scheduling_method_id;
              mySimulationSettings_res.wrong_classification=mySimulationSettings.wrong_classification;
              mySimulationSettings_res.waste=mySimulationSettings.waste;
            }
            else  //else get settings
            {
               mySimulationSettings_res.combination_sensitivity=resultDataset.getInt("combination_sensitivity");
               mySimulationSettings_res.resultsize=resultDataset.getInt("resultsize");
               mySimulationSettings_res.scheduling_method_id=resultDataset.getInt("scheduling_method_id");
               mySimulationSettings_res.wrong_classification=resultDataset.getInt("wrong_classification");
               mySimulationSettings_res.waste=resultDataset.getInt("waste");
            }

       //add resource to array, second parameter specifies scheduling algorithm 1=FCFS,2=FCFS_DAY_URGENT
       myResources[resultDataset.getRow()-1] = new Resource(resultDataset.getInt("resource_id2"), mySimulationSettings_res);

     }


  } catch(Exception e) {
          System.err.println("Exception: " + e.getMessage());
  }


}

public int getDepartmentID(){
  return department_id;
}

public void request_timeslots(ResultSet resultDataset,int urgency,int no_combinations,Vector timeslots){


 try{
     Statement db_statement1 = mySimulationSettings.Localcon1.createStatement();
     String query1;

     //----------------------------SELECT SHARING RESROUCES UNION WITH ITSELF IN CASE NULL RESULTS ----------------------

     query1="(SELECT r.resource_id2 FROM shared_resource sr1" +
           " inner join resource r on r.id=sr1.resource_id" +
           " " +
           " WHERE sr1.sharing_rule_id IN " +
           " " +
           " (SELECT sr2.sharing_rule_id FROM shared_resource sr2" +
           " inner join sharing_rule srule on srule.id=sr2.sharing_rule_id AND srule.dataset_id="+ mySimulationSettings.dataset_id+
           " inner join resource r2 on r2.id=sr2.resource_id" +
           " WHERE r2.resource_id2='"+resultDataset.getInt("resource_id")+"') " +
           " " +
           " AND r.resource_id2!='"+resultDataset.getInt("resource_id")+"')" +
           " " +
           " UNION" +
           " " +
           "(SELECT '"+resultDataset.getInt("resource_id")+"')";

     ResultSet resultDataset2 = db_statement1.executeQuery(query1);
     query1=null;
     if(resultDataset==null)
       System.out.println("MySQL null: " + query1);

     while(resultDataset2.next()){
       //----------------------------------------REQUEST TIMESLOTS---------------------------------------
```

```
            get_resource(resultDataset2.getInt("resource_id2")).request_timeslots(resultDataset,urgency,no_combinations,timeslots);
          }


      }
      catch(Exception e) {
              System.err.println("Exception: " + e.getMessage());
      }

    }


    public Resource get_resource(int resource_ID){

      //find corresponding resource in array
      for(int j=0;j<myResources.length;j++){

              //find department object
              if(resource_ID==myResources[j].getResourceID())
                 return myResources[j];
      }

      return null ;
    }

}
```

11.3 Class: Resource

```
package combi;

import java.sql.*;
import java.text.SimpleDateFormat;
import java.util.Vector;

/**
 *
 * @author Robin
 */
public class Resource {

private int resource_id;
private Scheduling_method method;
private SimulationSettings mySimulationSettings;
public Vector resource_agenda=new Vector();
//private Date first;

public Resource(int id,SimulationSettings mySettings){


   resource_id=id;
   mySimulationSettings=mySettings;
  // first=null;

   //initialize scheduling method for resource
   switch(mySimulationSettings.scheduling_method_id){
      case 1:  method = new FCFS_method(this);
      break;
      case 3:  method = new FCFS_DAY_URGENT(this);
      break;
      case 2:  method = new FCRS_method(this);
      break;
      case 4:  method = new FCFS2_method(this);
      break;
      case 5:  method = new FCFS2_urgent_method(this);
      break;
   }

   // init agenda
   try {

      Statement db_statement1 =  mySimulationSettings.Localcon1.createStatement();

      String query1="select `year`,`day`,timeslot_start,timeslot_end FROM resource_agenda WHERE  resource_id="+resource_id+"
ORDER by `year`,`day`";
      ResultSet resultDataset2 = db_statement1.executeQuery(query1);

      if(resultDataset2==null)
         System.out.println("MySQL null: " + query1);

      boolean result=resultDataset2.next();

      for(int i=0;i<=365;i++){
         int times[]= new int[2];
         times[0]=0;

        if(!result)
         times[0]=0;
        else if(resultDataset2.getInt("day")==i){
            times[0]=resultDataset2.getInt("timeslot_start");
            times[1]=resultDataset2.getInt("timeslot_end");
            result=resultDataset2.next();
```

```
        }
        resource_agenda.add(i,times);

      }


   } catch(Exception e) {
         System.err.println("Exception: " + e.getMessage());
   }
}

public int getResourceID(){
  return resource_id;
}

public SimulationSettings getSettings(){
  return mySimulationSettings;
}



public void request_timeslots(ResultSet resultDataSet,int urgency,int no_combinations,Vector timeslots){
   //Vector v = new Vector();

   method.get_timeslots(resultDataSet,urgency,no_combinations,timeslots);


}

public void book_appointment(Timeslot mySlot){
    try{
       //insert new dataset
       Statement db_statement1 = mySimulationSettings.Localcon1.createStatement();

       //format the java date to a mysql datestring
       SimpleDateFormat SQL_DATE_FORMAT =
       new SimpleDateFormat("yyyy-MM-dd HH:mm:00");
       String sqlDate = SQL_DATE_FORMAT.format(mySlot.mydate);

       String query1="INSERT INTO scheduled_appointment(appointment_id,resource_id,appointment_date,simulation_id)
VALUES("+mySlot.myappointment_id+","+mySlot.myresource.getResourceID()+",'"+sqlDate+"',"+mySimulationSettings.simulation_i
d+")";
       int resultAppointment = db_statement1.executeUpdate(query1);
       query1=null;
       if(resultAppointment==0)
          System.out.println("\nMySQL null: " + query1);
       //else
         System.out.println("\nTimeslot booked: "+mySlot.myappointment_id+" rid: "+mySlot.myresource.getResourceID()+" dt:
"+sqlDate);

       }catch(Exception e) {
          System.err.println("Exception: " + e.getMessage());
       }


}



}
```

11.4 Class: Timeslot

```
package combi;

import java.util.Date;
import java.util.Calendar;
import java.text.SimpleDateFormat;

/**
 *
 * @author Robin
 */
public class Timeslot {
   Resource myresource;
   Date mydate;
   int mycost, myappointment_id,myduration;

   public Timeslot(Resource r,Date d,int appointment_id,int cost,int duration){
      myresource=r;
      mydate=d;
      mycost=cost;
      myduration=duration;
      myappointment_id=appointment_id;
   }

   public void book_appointment(){
      myresource.book_appointment(this);
   }

   public void debug(){
      System.out.println("book appointment_id:"+myappointment_id);
  }

}
```

11.5 Class: Scheduling method

```
package combi;

import java.sql.*;
import java.util.Vector;

/**
 *
 * @author Robin
 */
public abstract class Scheduling_method {

Resource myresource;

   public Scheduling_method(Resource res){
   myresource=res;
   }

public abstract void get_timeslots(ResultSet resultDataSet,int urgency,int no_combinations, Vector v);


}
```

11.6 Class: SimulationSettings

```
package combi;

import java.sql.Connection;

/**
 *
 * @author Robin
 */
public class SimulationSettings {
   int scheduling_method_id;
   int combination_sensitivity;
   int wrong_classification;
   int waste;
   int random;
   int resultsize;
   int simulation_id;
   Connection Localcon1;
   String dataset_id;

   public SimulationSettings(){

   }
}
```

11.7 Class: FCFS2_method

```
package combi;

import java.sql.*;
import java.util.Calendar;
import java.util.GregorianCalendar;
import java.util.Vector;

/**
 * Will only return 1 timeslot, the first available.
 *
 * @author Robin
 */
public class FCFS2_method extends Scheduling_method {


public FCFS2_method(Resource res){
  super(res);
}

public void get_timeslots(ResultSet resultDataSet,int urgency,int no_combinations, Vector timeslots){

   try {
      int combi_extension=0;

      //if part of combination appointment extend the timeframe with 15 days
      if(no_combinations>1)
         combi_extension=30;

      Statement db_statement1 =  myresource.getSettings().Localcon1.createStatement();

      //query all appointments within timeframe
      String query1="SELECT YEAR(appointment_date) as `year`,DAYOFYEAR(appointment_date) as
`day`,time_to_sec(sa.appointment_date)/60 as timeslot_start,(time_to_sec(adddate(sa.appointment_date,INTERVAL a.duration
MINUTE))/60) as timeslot_end FROM scheduled_appointment sa"
```

```
            +" left join appointment a on a.id=sa.appointment_id"
            +" WHERE  sa.appointment_date>=DATE('"+resultDataSet.getString("date_registration")+"') AND
sa.appointment_date<=adddate(date('"+resultDataSet.getString("date_registration")+"'), INTERVAL
"+(myresource.getSettings().resultsize+combi_extension)+" DAY) AND sa.resource_id="+myresource.getResourceID()+" AND
sa.simulation_id="+myresource.getSettings().simulation_id+" ORDER BY `year`,`day`,time_to_sec(sa.appointment_date)/60 ASC";

    ResultSet resultDataset2 = db_statement1.executeQuery(query1);
    boolean result=resultDataset2.next();

    query1="select DAYOFYEAR('"+resultDataSet.getDate("date_registration")+"') as
dag,YEAR('"+resultDataSet.getDate("date_registration")+"') as jaar";
    Statement db_statement2 =  myresource.getSettings().Localcon1.createStatement();
    ResultSet resultDataset3 = db_statement2.executeQuery(query1);

    resultDataset3.next();
    int day=resultDataset3.getInt("dag");
    int year=resultDataset3.getInt("jaar");



      int timeslot_end=0,timeslot_start=0,start_day=0,end_day=0,timeframe=0,day2=0,k=0,endloop=0;
      boolean found=false,setsize=false;

      endloop=day+myresource.getSettings().resultsize+combi_extension-1;

      for(int i=day;i<=endloop;i++){

        //if days exceed 365 then go to next year
        if(i>365){
          k=i-365;
          year=resultDataset3.getInt("jaar")+1; //next year
        }
        else
          k=i;

        timeslot_start=((int[])myresource.resource_agenda.get(k))[0];

        if(timeslot_start!=0){ //check if that day appointments can be scheduled at all

          if((!result)||(k!=resultDataset2.getInt("day"))){  //if not a single appointment in resultset or not for that day the whole
day is available
            timeframe=((int[])myresource.resource_agenda.get(k))[1]-timeslot_start;



            Calendar calendar2 = new GregorianCalendar();

            //calculate number of possible timeslot iwthin timeframe
            int no_timeslots=timeframe/resultDataSet.getInt("duration");
            //System.out.println("duration: " + resultDataSet.getInt("duration"));

            //loop through the number of calculated timeslots
            for(int j=0;j<no_timeslots;j++){

              //calculate the starttime in minutes
              int start_time=timeslot_start+j*resultDataSet.getInt("duration");

              //set calendar to calculted timeslot
              calendar2.set(year,0,k,0,start_time);

              //create the timeslot, with calculated cost
              //COST is represented as the difference in minutes from the current day, the earlier the cheaper
              Timeslot mySlot = new
Timeslot(myresource,calendar2.getTime(),resultDataSet.getInt("appointment_id"),((i*1440)+start_time)-
(day*1440),resultDataSet.getInt("duration"));
```

```
                //add the timeslot to the vector
                timeslots.add(mySlot);

                /*
                if(!result)
                   System.out.println("EMPTY timeslot added "+myresource.getResourceID()+", start: "+year+"-"+k+":"+start_time);
                else
                   System.out.println("NO RECORD THIS DAY timeslot added "+myresource.getResourceID()+", start: "+year+"-
"+k+":"+start_time);
                */
              }
            }
            else{
                int l=resultDataset2.getInt("day");

                while(k==l){ //get only records from the current day

                  //first appointment of the day
                  if(day2!=resultDataset2.getInt("day")){
                    day2=resultDataset2.getInt("day");
                    timeslot_end=timeslot_start;
                    //found=true;
                  }

                  if(resultDataset2.getInt("timeslot_start")-timeslot_end>=resultDataSet.getInt("duration")){
                     timeslot_start=timeslot_end;
                     timeframe=resultDataset2.getInt("timeslot_start")-timeslot_end;
                     found=true;
                  }


                  if(found){
                     Calendar calendar2 = new GregorianCalendar();

                     //calculate number of possible timeslot iwthin timeframe
                     int no_timeslots=timeframe/resultDataSet.getInt("duration");
                     //System.out.println("duration: " + resultDataSet.getInt("duration"));

                     //loop through the number of calculated timeslots
                     for(int j=0;j<no_timeslots;j++){

                        //calculate the starttime in minutes
                        int start_time=timeslot_start+j*resultDataSet.getInt("duration");

                        //set calendar to calculted timeslot
                        calendar2.set(year,0,k,0,start_time);

                        //create the timeslot, with calculated cost
                        //COST is represented as the difference in minutes from the current day, the earlier the cheaper
                        Timeslot mySlot = new
Timeslot(myresource,calendar2.getTime(),resultDataSet.getInt("appointment_id"),((i*1440)+start_time)-
(day*1440),resultDataSet.getInt("duration"));

                        //add the timeslot to the vector
                        timeslots.add(mySlot);

                       //System.out.println("NORM timeslot added "+myresource.getResourceID()+", start: "+year+"-
"+k+":"+start_time);

                     }
                  }

                  //get next record
                  found=false;
                  timeslot_end=resultDataset2.getInt("timeslot_end");
                  if(result=resultDataset2.next())
```

```
                    l=resultDataset2.getInt("day");
                  else
                    l=0;
                }

                //from the last record till the end of the day
                if(((int[])myresource.resource_agenda.get(k))[1]-timeslot_end>=resultDataSet.getInt("duration")){
                    timeslot_start=timeslot_end;
                    timeframe=((int[])myresource.resource_agenda.get(k))[1]-timeslot_end;

                    Calendar calendar2 = new GregorianCalendar();

                  //calculate number of possible timeslot iwthin timeframe
                  int no_timeslots=timeframe/resultDataSet.getInt("duration");
                  //System.out.println("duration: " + resultDataSet.getInt("duration"));

                  //loop through the number of calculated timeslots
                  for(int j=0;j<no_timeslots;j++){

                    //calculate the starttime in minutes
                    int start_time=timeslot_start+j*resultDataSet.getInt("duration");

                    //set calendar to calculted timeslot
                    calendar2.set(year,0,k,0,start_time);

                    //create the timeslot, with calculated cost
                    //COST is represented as the difference in minutes from the current day, the earlier the cheaper
                    Timeslot mySlot = new
Timeslot(myresource,calendar2.getTime(),resultDataSet.getInt("appointment_id"),((i*1440)+start_time)-
(day*1440),resultDataSet.getInt("duration"));

                    //add the timeslot to the vector
                    timeslots.add(mySlot);

                   //System.out.println("END timeslot added "+myresource.getResourceID()+", start: "+year+"-"+k+":"+start_time);

                  }


                }


            }



          }

          //adjust the timeframe starting from the first available timeslot
          if((!setsize)&&(timeslots.size()!=0)){
             setsize=true;
             endloop=k+myresource.getSettings().resultsize+combi_extension-1;
          }
          else if(!setsize)
            endloop++;


        }

    } catch(Exception e) {
            System.err.println("Exception: " + e.getMessage());
    }

}

}
```

11.8 Class: FormMain

```
package combi;

import java.awt.Color;
import java.awt.Font;
import java.sql.*;
import java.util.Vector;
import javax.swing.JComboBox;

/**
 *
 * @author  Robin
 */
public class FormMain extends javax.swing.JFrame {

private Connection con1 = null;
private Connection con2 = null;
private boolean simulation=false;
private boolean first_start=true;
private Department[] myDepartments;
private SimulationSettings mySimulationSettings = new SimulationSettings();
private int simulation_id;

   /** Creates new form FormMain */
   public FormMain() {
     initComponents();
     /*
     String query1="SELECT MAX(id) as NewID FROM dataset";
     ResultSet ResultDatasetID = db_statement1.executeQuery(query1);
     ResultDatasetID.next();
     Integer DatasetID = ResultDatasetID.getInt("NewID");
     System.err.println("\nMySQL DatasetID: " + DatasetID);
      */
   }

private void fillcombo(String table,JComboBox temp){

   //Fill Patient dataset ComboBox
   temp.removeAllItems();
   try {

      // Create a statement to send SQL
      Statement db_statement1 = con1.createStatement();
      String query1;

      //insert new dataset
      query1="select * from "+table+" ORDER BY id DESC";
      System.out.println("MySQL query: " + query1);
      ResultSet resultDataset = db_statement1.executeQuery(query1);
      if(resultDataset==null)
         System.out.println("MySQL null: " + query1);
      else
         System.out.println("Done MySQL query: " + query1);



      System.out.println("done query");
      // While more rows exist, print them
      while(resultDataset.next()){
         System.out.println("Dataset: " + resultDataset.getString("name"));
         Vector v = new Vector();
```

```
            v.add(resultDataset.getString("name"));
            if(!table.equals("scheduling_method"))
               v.add(resultDataset.getString("datetime"));
            v.add(resultDataset.getString("id"));
            temp.addItem(v);
         }

      } catch(Exception e) {
               System.err.println("Exception: " + e.getMessage());
      }
   }


private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
   FormPatientDatasetEdit EditPatientForm = new FormPatientDatasetEdit();
  try{
      Vector item = (Vector) jComboBox_patient.getSelectedItem();
      System.out.println("selected dataset:"+ item.get(0));
      EditPatientForm.setConnection(con1,con2,""+item.get(2));
      EditPatientForm.setVisible(true);
   }
   catch(Exception e){
      System.out.println("No dataset selected!");
   }

}

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
   FormDepartmentEditDataset EditDepartmentForm = new FormDepartmentEditDataset();
   try{
      Vector item = (Vector) jComboBox_department.getSelectedItem();
      System.out.println("selected dataset:"+ item.get(0));
      EditDepartmentForm.setConnection(con1,con2,""+item.get(2));
      EditDepartmentForm.setVisible(true);
   }
   catch(Exception e){
      System.out.println("No dataset selected!");
   }
}

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

   FormDepartmentDataset NewDepartmentForm = new FormDepartmentDataset();
   NewDepartmentForm.setConnection(con1, con2);
   NewDepartmentForm.setVisible(true);
}

private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
   try{
      Vector item = (Vector) jComboBox_patient.getSelectedItem();
      System.out.println("selected dataset:"+ item.get(0));

      try {

         // Create a statement to send SQL
         Statement db_statement1 = con1.createStatement();
         String query1;

         //delete selected patient dataset
         query1="delete from patient_dataset WHERE id="+ item.get(2);
         System.out.println("MySQL query: " + query1);
         Integer resultDataset = db_statement1.executeUpdate(query1);
```

```
        if(resultDataset==0)
          System.out.println("MySQL null: " + query1);
        else
          System.out.println("Done MySQL query: " + query1);

        System.out.println("done query");

        //delete appointments
        query1="delete from appointment WHERE dataset_id="+ item.get(2);
        System.out.println("MySQL query: " + query1);
        resultDataset = db_statement1.executeUpdate(query1);
        if(resultDataset==0)
          System.out.println("MySQL null: " + query1);
        else
          System.out.println("Done MySQL query: " + query1);

        System.out.println("done query");


      } catch(Exception e) {
            System.err.println("Exception: " + e.getMessage());
      }

      jComboBox_patient.removeAllItems();
      fillcombo("patient_dataset",jComboBox_patient);

    }
    catch(Exception e){
      System.out.println("No dataset selected!");
    }
}

private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
    try{
      Vector item = (Vector) jComboBox_department.getSelectedItem();
      System.out.println("selected dataset:"+ item.get(0));

      try {

        // Create a statement to send SQL
        Statement db_statement1 = con1.createStatement();
        String query1;

        //delete selected dataset
        query1="delete from department_dataset WHERE id="+ item.get(2);
        System.out.println("MySQL query: " + query1);
        Integer resultDataset = db_statement1.executeUpdate(query1);
        if(resultDataset==0)
          System.out.println("MySQL null: " + query1);
        else
          System.out.println("Done MySQL query: " + query1);

        System.out.println("done query");

        //delete department
        query1="delete from department WHERE dataset_id="+ item.get(2);
        System.out.println("MySQL query: " + query1);
        resultDataset = db_statement1.executeUpdate(query1);
        if(resultDataset==0)
          System.out.println("MySQL null: " + query1);
        else
          System.out.println("Done MySQL query: " + query1);

        System.out.println("done query");
```

```
          //delete resources
          query1="delete from resource WHERE dataset_id="+ item.get(2);
          System.out.println("MySQL query: " + query1);
          resultDataset = db_statement1.executeUpdate(query1);
          if(resultDataset==0)
             System.out.println("MySQL null: " + query1);
          else
             System.out.println("Done MySQL query: " + query1);

          System.out.println("done query");


       } catch(Exception e) {
              System.err.println("Exception: " + e.getMessage());
       }

       jComboBox_department.removeAllItems();
       fillcombo("department_dataset",jComboBox_department);

    }
    catch(Exception e){
       System.out.println("No dataset selected!");
    }
}

private void jButton11ActionPerformed(java.awt.event.ActionEvent evt) {


try {

    //MySQL database
    Class.forName("com.mysql.jdbc.Driver").newInstance( );
    con1 = DriverManager.getConnection("jdbc:mysql://" + jTextField_con1_ip.getText() + "/" +
jTextField_con1_db.getText(),jTextField_con1_user.getText(),jTextField_con1_pass.getText());


    if(!con1.isClosed()){
    System.out.println("\nSuccessfully connected to " +
    "MySQL server using TCP/IP...");

    con1text_connected.setForeground(new Color(51,102,0));
    con1text_connected.setText("Connected!");
    }

} catch(Exception e) {
      System.err.println("Exception: " + e.getMessage());
}

//fille the dataset comboboxes
fillcombo("department_dataset",jComboBox_department);
fillcombo("patient_dataset",jComboBox_patient);
fillcombo("simulation",jComboBox_TotalResults);
fillcombo("scheduling_method",jComboBox_method);
//fill_sim_department();

    // TODO add your handling code here:
}

private void jButton12ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:

    try {

    //MySQL database
    Class.forName("oracle.jdbc.OracleDriver").newInstance( );
```

```
   con2 = DriverManager.getConnection("jdbc:oracle:thin:@" + jTextField_con2_ip.getText() + ":" +
jTextField_con2_db.getText(),jTextField_con2_user.getText(),jTextField_con2_pass.getText());

   if(!con2.isClosed()){
   System.out.println("\nSuccessfully connected to " +
   "MySQL server using TCP/IP...");

   con2text_connected.setForeground(new Color(51,102,0));
   con2text_connected.setText("Connected!");
   }

} catch(Exception e) {
     System.err.println("Exception: " + e.getMessage());
}
}

private void jButton13ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
jComboBox_department.removeAllItems();
jComboBox_patient.removeAllItems();
fillcombo("department_dataset",jComboBox_department);
fillcombo("patient_dataset",jComboBox_patient);
}


private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:

//initialise departments for selected department dataset

   if(first_start){

      Vector department_dataset = (Vector) jComboBox_department.getSelectedItem();
      Vector coordination_method = (Vector)  jComboBox_method.getSelectedItem();
      Vector patient_dataset = (Vector) jComboBox_patient.getSelectedItem();
      int coordination_method_id=Integer.parseInt((String)coordination_method.get(1));

      try {
               Statement db_statement1 = con1.createStatement();
               Statement db_statement2 = con1.createStatement();

                //----------------------------INSERT new SIMULATION records----------------------------------

               String query1="INSERT INTO
simulation(name,comment,datetime,scheduling_method_id,combination_sensitivity,wrong_classification,waste,random,resultsize,p
atient_dataset_id)
VALUES('"+jText_simulation_name.getText()+"','"+jTextarea_simulation_comment.getText()+"',NOW(),"+coordination_method_id+"
,'"+jTextField_sens.getText()+"','"+jTextField_wrong.getText()+"','"+jTextField_waste.getText()+"','"+jTextField_rand.getText()+"','"+j
TextField_size.getText()+"','"+(String)patient_dataset.get(2)+"')";
               int resultDataset2 = db_statement1.executeUpdate(query1);
               if(resultDataset2==0)
                 System.out.println("\nMySQL null: " + query1);

               //get the simulation_id
               ResultSet resultDataset = db_statement1.getGeneratedKeys();
               resultDataset.next();
               simulation_id=resultDataset.getInt(1);
               mySimulationSettings.simulation_id=resultDataset.getInt(1);
               mySimulationSettings.combination_sensitivity=Integer.parseInt(jTextField_sens.getText());
               mySimulationSettings.resultsize=Integer.parseInt(jTextField_size.getText());
               mySimulationSettings.scheduling_method_id=coordination_method_id;
               mySimulationSettings.wrong_classification=Integer.parseInt(jTextField_wrong.getText());
               mySimulationSettings.waste=Integer.parseInt(jTextField_waste.getText());
               mySimulationSettings.random=Integer.parseInt(jTextField_rand.getText());
               mySimulationSettings.Localcon1=con1;
```

```
            //------------------------- CREATE DEPARTMENTS-----------------------------------------------------
            query1="SELECT * FROM department WHERE dataset_id="+(String)department_dataset.get(2);
            resultDataset = db_statement1.executeQuery(query1);

            if(resultDataset==null)
               System.out.println("MySQL null: " + query1);
            else
               System.out.println("Done MySQL query: " + query1);

            //create new department array with the size of the fetched result
            resultDataset.last();
            myDepartments = new Department[resultDataset.getRow()];
            resultDataset.beforeFirst();

            //loop through the department results
            while(resultDataset.next()){

                //----------------------------INSERT new DEPARTMENT SIMULATION record----------------------------------

                query1="INSERT INTO
simulation_department(combination_sensitivity,wrong_classification,simulation_id,department_id,scheduling_method_id,waste,re
sultsize)
VALUES("+resultDataset.getInt("combination_sensitivity")+","+resultDataset.getInt("wrong_classification")+","+simulation_id+","+re
sultDataset.getInt("id")+","+resultDataset.getInt("scheduling_method_id")+","+resultDataset.getInt("waste")+","+resultDataset.getI
nt("resultsize")+")";
                resultDataset2 = db_statement2.executeUpdate(query1);
                if(resultDataset2==0)
                  System.out.println("\nMySQL null: " + query1);
                else
                  System.out.println("\nDepartment created:");

                //----------------------------CREATE new Department object-----------------------------

                SimulationSettings mySimulationSettings_dep = new SimulationSettings();

                //if scheduling method is not defined of department use general simulation settings
                mySimulationSettings_dep.random=mySimulationSettings.random;
                mySimulationSettings_dep.Localcon1=mySimulationSettings.Localcon1;
                mySimulationSettings_dep.simulation_id=mySimulationSettings.simulation_id;
                mySimulationSettings_dep.dataset_id=(String)department_dataset.get(2);


                if(resultDataset.getInt("scheduling_method_id")==0){
                   mySimulationSettings_dep.combination_sensitivity=mySimulationSettings.combination_sensitivity;
                   mySimulationSettings_dep.resultsize=mySimulationSettings.resultsize;
                   mySimulationSettings_dep.scheduling_method_id=mySimulationSettings.scheduling_method_id;
                   mySimulationSettings_dep.wrong_classification=mySimulationSettings.wrong_classification;
                   mySimulationSettings_dep.waste=mySimulationSettings.waste;
                }
                else  //else get settings
                {
                   mySimulationSettings_dep.combination_sensitivity=resultDataset.getInt("combination_sensitivity");
                   mySimulationSettings_dep.resultsize=resultDataset.getInt("resultsize");
                   mySimulationSettings_dep.scheduling_method_id=resultDataset.getInt("scheduling_method_id");
                   mySimulationSettings_dep.wrong_classification=resultDataset.getInt("wrong_classification");
                   mySimulationSettings_dep.waste=resultDataset.getInt("waste");
                }

                 myDepartments[resultDataset.getRow()-1]= new
Department(resultDataset.getInt("department_id2"),mySimulationSettings_dep);
```