Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Computer Science and Engineering

Master's Thesis

Simulation and optimization of laboratory processes

Bc. Dmitrij Sojma

Supervisor: doc. Ing. Přemysl Šůcha, Ph.D.

Study Programme: Open Informatics

Field of Study: Software Engineering

May 2022

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name:	**Sojma Dmitrij**	Personal ID number:	**474381**
Faculty / Institute:	**Faculty of Electrical Engineering**		
Department / Institute:	**Department of Computer Science**		
Study program:	**Open Informatics**		
Specialisation:	**Software Engineering**		

II. Master’s thesis details

Master’s thesis title in English:

Simulation and optimization of laboratory processes

Master’s thesis title in Czech:

Simulace a optimalizace procesu zpracování laboratorních vzorků

Guidelines:

A major trend among clinical laboratories is the automation of samples processing. However, existing scientific work and current practice show that the automation of these processes does not sufficiently use optimization methods that could increase their efficiency. The work aims to get acquainted with the testing process of laboratory samples on biochemical and immunochemical analyzers, create a simulation of this process, and design an appropriate optimization algorithm.
The diploma thesis has the following tasks:
1) research existing literature,
2) get acquainted with the DxA 5000 system and related biochemical and immunochemical analyzers,
3) design and implement a simulation based on the analysis of the systems mentioned above,
4) design and implement an optimization algorithm for the simulated process,
5) verify the simulation results against the system's real behavior and evaluate the impact of the optimization algorithm.

Bibliography / sources:

Laquanda T. Leaven (2015) Improving Hospital Laboratory Performance: Implications for Healthcare Managers, Hospital Topics, 93:2, 19-26.
Eline R. Tsai, Andrei N. Tintu, Derya Demirtas, Richard J. Boucherie, Robert de Jonge & Yolanda B. de Rijke (2019) A critical review of laboratory performance indicators, Critical Reviews in Clinical Laboratory Sciences, 56:7, 458-471.
Patrik Březina (2021) Data Analysis of Laboratory Workflow and Data-driven Laboratory Process Optimization, Diploma thesis, CTU.

Name and workplace of master’s thesis supervisor:

doc. Ing. Přemysl Šůcha, Ph.D.   Department of Control Engineering FEE

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **02.02.2022** Deadline for master's thesis submission: **20.05.2022**

Assignment valid until: **30.09.2023**

doc. Ing. Přemysl Šůcha, Ph.D. Supervisor’s signature	Head of department's signature	prof. Mgr. Petr Páta, Ph.D. Dean’s signature

III. Assignment receipt

The student acknowledges that the master’s thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master’s thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student’s signature

Declaration

Aknowledgements

I would like to express my sincere gratitude to doc. Ing. Přemysl Šůcha, Ph.D. and Ing. Antonín Novák for giving me the opportunity to work on this interesting topic and for providing guidance and a friendly environment that helped with the creation of this thesis.

Abstract

Automation of clinical laboratories enables laboratories to process a larger amount of samples and it also leads to requested test results becoming available faster than before. Modular laboratory automation systems make it possible to adapt the selection of components based on individual laboratory needs. This thesis presents a simulator that simplifies and speeds up the suitability analysis of technological laboratory designs that are based on modular laboratory automation systems DxA 5000 and DxA 5000 Fit. The second part of this thesis focuses on improving an already existing mathematical model that increases laboratory throughput by finding the optimal distribution of available tests among biochemical analyzers. The presented changes refine the model of analyzer DxC 700 AU.

Keywords: total laboratory automation, modular systems, discrete event simulation, throughput maximization, integer linear programming

Abstrakt

Automatizace provozu klinických laboratoří umožňuje zpracovávání vyššího množství vzorků ale také zkrácení doby do obdržení výsledků požadovaných testů. Modulární laboratorní automatizační systémy umožňují přizpůsobit výběr jednotlivých komponent podle individuálních potřeb laboratoře. Tato práce prezentuje simulátor, který usnadňuje a urychluje analýzu vhodnosti technologických návrhů laboratoří využívajících modulární automatizační systémy DxA 5000 a DxA 5000 Fit. Druhá část práce se zaměřuje na úpravy již existujícího matematického modelu, který umožňuje zvýšit průchodnost laboratorní sekce s biochemickými analyzátory nalezením optimalního rozložení prováděných testů. Prezentované změny zpřesňují model analyzátoru DxC 700 AU.

Klíčová slova: úplná laboratorní automatizace, modulární systémy, simulace diskrétních událostí, maximalizace průchodnosti, celočíselné lineární programování

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Being healthy is one of the greatest concerns for all of us. To cure our illnesses, physicians first need to diagnose them. Numerous medical conditions are diagnosed by some form of chemical tests. Many of these types of tests are performed from blood, urine, or other types of samples, which clinical laboratories then process to obtain the results of methods requested by the diagnosticians.

In times when the world is affected by a global pandemic, such as the ongoing case of COVID-19, the number of tests required can rapidly increase and put a lot of pressure on the healthcare system. The most common source of this pressure is the lack of trained personnel [1]. Therefore it is desirable to decrease the work performed by the medical staff as much as possible.

Laboratory automation can help with this problem in multiple ways. It has been shown that the use of auto-verification of results can increase productivity and relieve the laboratory personnel from the need to attend to most of the samples [2]. Many other benefits of laboratory automation have been presented in the literature, such as a significant decrease in turnaround times, decrease in financial cost per sample [3], reduction in errors, improved precision [4] and other benefits.

The construction costs of automated clinical laboratories are very high, and so are the upgrades or modifications to the ones that already exist. One of the ways to save some capital is to thoroughly explore and assess the choices of medical equipment to fit the needs based on the present and expected future demand. Modular systems enable the laboratory to choose the best-suited equipment, which can subsequently decrease the cost while keeping open doors for future extensions.

The problem with this approach is that the accurate prediction of whether or not a specific choice of laboratory equipment and configuration is sufficient while not being unnecessarily costlier than needed, based only on the demand and system parameters, is a very complex task. Therefore, it is necessary to find a way to support such claims by providing factual data.

This can be done by defining a set of metrics relevant to the problem and subsequently evaluating them by generating data with the use of virtual representations of the real-world systems, so-called ”digital twins” [5], digital modeling, and simulations. This kind of approach may be challenging to implement due to the complexity of such systems. However, the ability to use real-world data as input and the ability to collect and output a range of metrics, such as sample turnaround time, percentage of maximum load achieved, and causes of sample delay, corresponding to the concrete input data, are great benefits of this approach.

The main aim of this thesis is to gain a good understanding of a modular laboratory automation system DxA 5000 Fit [6] and its components and to use this knowledge to build a software solution that could be used to simulate different configurations of the DxA 5000 Fit system in order to find the best fit for small and mid-sized laboratories that wish to use this laboratory automation system. Chapter 3 describes the simulated laboratory automation system, and we present the implemented simulator in Chapter 4. An already existing mathematical model of a laboratory, which uses the DxA 5000 laboratory automation system, is revised, and changes improving the model's precision are presented in Chapter 6.

Chapter 2

Related work

2.1 Quality and performance indicators

One of the most important laboratory key performance indicators used in practice and literature is turnaround time (TAT). There exist a plethora of different TAT definitions; Breil et al. [7] fortunately performed an analysis of over 1000 articles leading to the identification of 162 different TAT definitions, from which 19 different TAT definitions were used in the laboratory domain. It is not easy to compare studies when most of them use different TAT definitions. Additionally, even when the same definition of TAT is used, the process of measuring it is different, i.e., measurements by solely using electronically available points in time versus manual records from shadowing the medical staff. For this reason, Breil et al. compiled a timeline for laboratory TAT, shown on Figure 2.1. According to them, the most common laboratory definition of TAT begins with the receipt of a specimen and ends when the results are available. These two milestones on the timeline presented in Figure 2.1 are called "receipt in lab" and "report available".

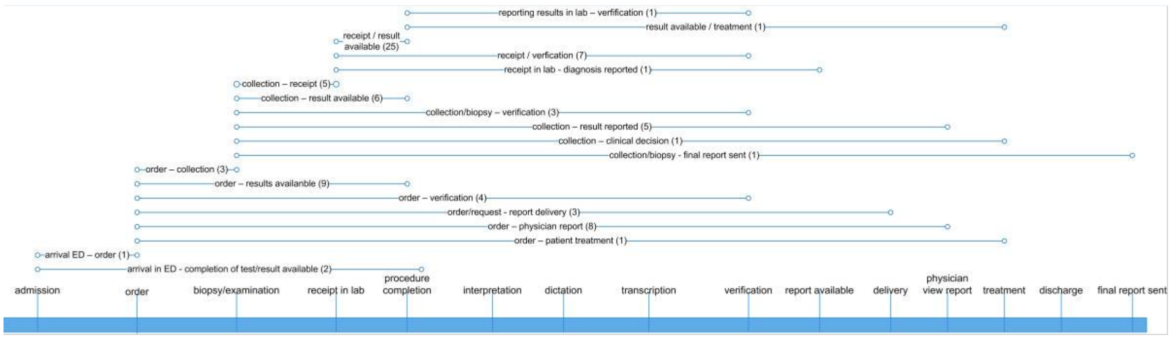

reporting results in lab – verification (1)
result available / treatment (1)
receipt / result available (25)
receipt / verfication (7)
receipt in lab - diagnosis reported (1)
collection – receipt (5)
collection – result available (6)
collection/biopsy – verification (3)
collection – result reported (5)
collection – clinical decision (1)
collection/biopsy - final report sent (1)
order – collection (3)
order – results availanble (9)
order – verification (4)
order/request - report delivery (3)
order – physician report (8)
order – patient treatment (1)
arrival ED – order (1)
arrival in ED - completion of test/result available (2)
admission
order
biopsy/examination
receipt in lab
procedure completion
interpretation
dictation
transcription
verification
report available
delivery
physician view report
treatment
discharge
final report sent

Figure 2.1: Turnaround time definitions in the laboratory domain [7]

Other quality indicators and their relations have been studied by Tsai et al. [8]. One other major indicator identified was cost, which includes numerous sources such as personnel cost, reagent cost, and maintenance cost. In the case of clinical laboratories working with low amounts of urgent specimens, a more meaningful indicator than TAT might be the efficient usage of laboratory resources. Resource usage, however, negatively correlates with TAT [8] and therefore, some laboratories may actually benefit from making changes that lead to a slight increase in TAT.

An evaluation process of laboratory changes based on measurements of quality indicators has been proposed by Miler et al. [9]. This process is shown on Figure 2.2 and has been used by the authors to evaluate the transition of hospital laboratory to total laboratory automation.

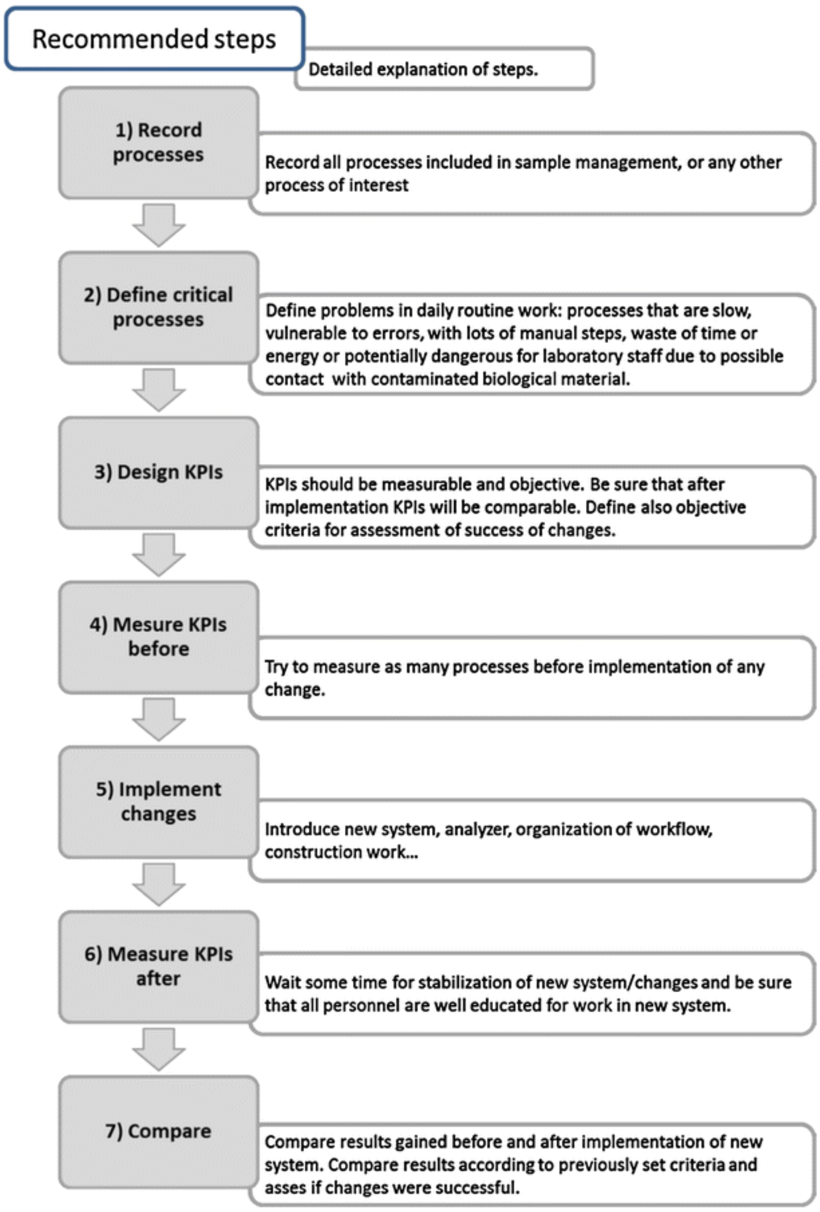
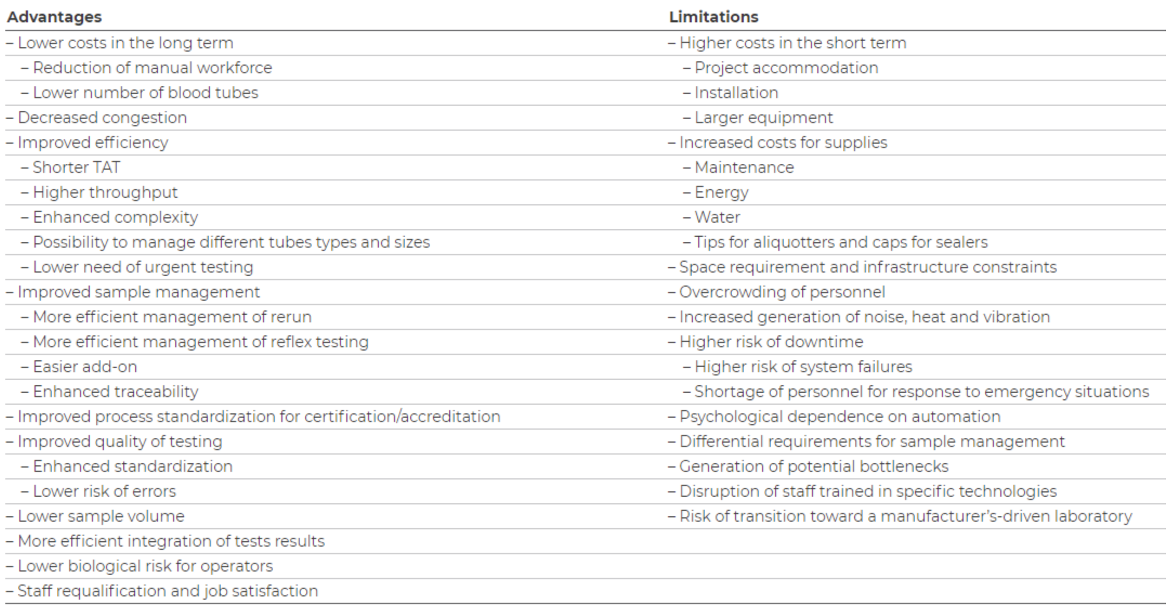

Recommended steps
Detailed explanation of steps.
1) Record processes
Record all processes included in sample management, or any other process of interest
2) Define critical processes
Define problems in daily routine work: processes that are slow, vulnerable to errors, with lots of manual steps, waste of time or energy or potentially dangerous for laboratory staff due to possible contact with contaminated biological material.
3) Design KPIs
KPIs should be measurable and objective. Be sure that after implementation KPIs will be comparable. Define also objective criteria for assessment of success of changes.
4) Mesure KPIs before
Try to measure as many processes before implementation of any change.
5) Implement changes
Introduce new system, analyzer, organization of workflow, construction work...
6) Measure KPIs after
Wait some time for stabilization of new system/changes and be sure that all personnel are well educated for work in new system.
7) Compare
Compare results gained before and after implementation of new system. Compare results according to previously set criteria and asses if changes were successful.

Figure 2.2: Flowchart of recommended steps for implementation of changes in laboratory with explained steps on the left side of the chart [9].

2.2 Total laboratory automation

Laboratory automation can be classified into multiple levels ranging from no automation at all to total laboratory automation (TLA), where most analyzers and many pre-analytical and post-analytical workstations are physically integrated as modular systems or connected by track systems and managed by software [10]. It has been shown that the implementation of TLA can yield many benefits such as a significant decrease in turnaround times, decrease in financial cost per sample [3], reduction in errors, improved precision [4], decrease in labor cost and improvements in TAT [11, 2]. In the case of intra-laboratory TAT, the results of Angeletti et al. [12] point to TLA having the most significant positive impact on the pre-analytical phase. The benefits of TLA can be substantial even when upgrading from a partially automated laboratory [13]. Lippi and Da Rin [10] compiled an overview of the pros and cons of total laboratory automation shown on Figure 2.3.



Advantages	Limitations
– Lower costs in the long term	– Higher costs in the short term
– Reduction of manual workforce	– Project accommodation
– Lower number of blood tubes	– Installation
– Decreased congestion	– Larger equipment
– Improved efficiency	– Increased costs for supplies
– Shorter TAT	– Maintenance
– Higher throughput	– Energy
– Enhanced complexity	– Water
– Possibility to manage different tubes types and sizes	– Tips for aliquotters and caps for sealers
– Lower need of urgent testing	– Space requirement and infrastructure constraints
– Improved sample management	– Overcrowding of personnel
– More efficient management of rerun	– Increased generation of noise, heat and vibration
– More efficient management of reflex testing	– Higher risk of downtime
– Easier add-on	– Higher risk of system failures
– Enhanced traceability	– Shortage of personnel for response to emergency situations
– Improved process standardization for certification/accreditation	– Psychological dependence on automation
– Improved quality of testing	– Differential requirements for sample management
– Enhanced standardization	– Generation of potential bottlenecks
– Lower risk of errors	– Disruption of staff trained in specific technologies
– Lower sample volume	– Risk of transition toward a manufacturer's-driven laboratory
– More efficient integration of tests results	
– Lower biological risk for operators	
– Staff requalification and job satisfaction	

Figure 2.3: Potential advantages and limitations of TLA [10].

In hospital environments, statim (short TAT) tests are usually performed in dedicated laboratories; however, with the emergence of TLA systems, results for all commonly required laboratory tests can be reported in time consistent with statim needs [14]. Since TLA systems are expensive, hospitals may not afford to use TLA in all departments. Therefore a core laboratory using TLA may be built and shared by different departments, which is depicted in Figure 2.4.

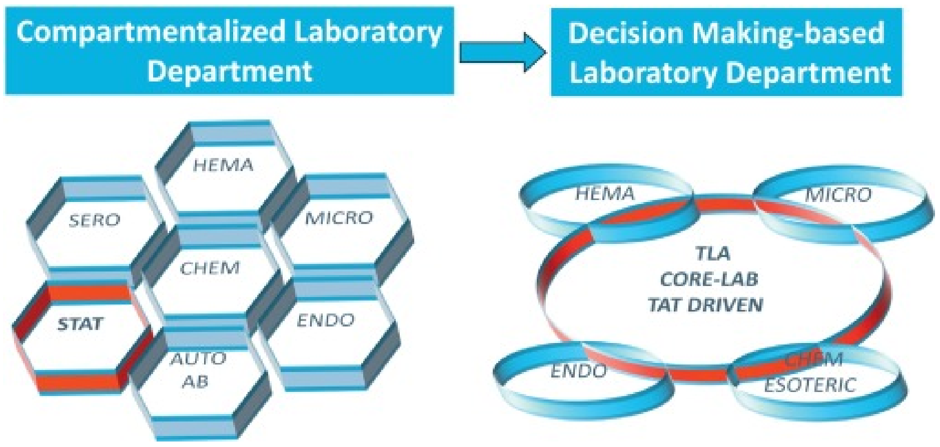

Compartmentalized Laboratory Department
Decision Making-based Laboratory Department
HEMA
SERO
MICRO
CHEM
STAT
ENDO
AUTO AB
HEMA
MICRO
TLA CORE-LAB TAT DRIVEN
ENDO
CHEM ESOTERIC

Figure 2.4: Moving from a compartmentalized laboratory department, in which stat testing is one of the independent laboratory sections, to a consolidated laboratory activity, where the core laboratory, using total laboratory automation (TLA), performs first-line tests and assures clinically suitable turnaround times (TAT) and satellite laboratories just execute specialized tests, may provide the occasion to create a decision making-based laboratory department [14].

According to [14], a well-designed TLA in a core laboratory allows eliminating the distinction between statim and ordinary orders by treating all of them as statim orders. The TLA should thus manage samples so that it is not forced to prioritize statim samples during load peaks. Sample arrivals to the laboratory should be optimized not to create large batches. The authors also stress the importance of identifying and limiting tests that should be included in the TLA. Infrequent tests, tests performed in very specific ways, and other tests needing careful result checks should be excluded from TLA or use specifically reserved TLA parts [14].

2.3 Laboratory simulation and modeling

The use of virtual models or simulations of clinical laboratories is almost non-existent in the publicly available literature. In 1994, Vogt [15] used discrete event simulation to model a hospital laboratory and was able to achieve results comparable to the real system. The simulation was then used to predict the amount of medical staff and analyzers needed for the laboratory to be able to process one order of magnitude increase in the number of samples.

More recently, Yang et al. [16] used a discrete event simulator to create a specific model for a laboratory with total automation. The model was then used to perform optimization of proposed rules and strategies to reduce sample TAT during periods of peak demand. A blood laboratory was also modeled in [17] to analyze bottlenecks and laboratory processes. The authors explored optimal allocation of medical staff and machines to reduce the throughput time of the system and increase the number of analyzed blood samples. As a result, the authors achieved significant improvements in desired system parameters.

Simulation of an arbitrary configuration of modular automated laboratory systems is not present in the literature. This makes it an attractive topic that could potentially bring new insights into the field.

2.4 DxA 5000 and DxA 5000 Fit laboratory automation systems

DxA 5000 [18] is a modular total laboratory automation system developed by Beckman Coulter, Inc. A smaller version of this system also exists and is called DxA 5000 Fit [6]. It is essential for us to understand how these systems and their parts operate. Fortunately, workflow analysis and optimization of a laboratory that uses DxA 5000 system have previously been done by Březina [19]. His work also contains a review on the use of optimization in the context of clinical laboratories.

Chapter 3

Laboratory automation system description

This chapter provides a description of a modular laboratory automation systems DxA 5000 [18] and DxA 5000 Fit [6]. Firstly, some commonly used terms are introduced in Section 3.1. The chapter then continues with the description of individual modules of the systems in Section 3.2, followed by the description of the sample processing workflow in Section 3.3.

3.1 Commonly used terms

Laboratories usually define two or three levels for TAT guarantees. Physicians appropriately mark one of these levels for each ordered test to indicate how urgent the results are for that particular test. Some results are acquired from chemical reactions with long incubation times, so it may not be possible to order these tests with the most strict TAT guarantee level. In the context of laboratory automation, each sample is assigned a priority based on the most strict TAT guaranty level associated with tests for that sample. Both the DxA 5000 and the DxA 5000 Fit laboratory automation systems distinguish between two levels of sample priority, **standard** priority and **high** priority. We will sometimes refer to samples with low priority as **routine** samples and to high priority samples as **statim** samples. Another commonly used priority level in a hospital environment is **vital**, which is also included in high priority samples.

Among the many tests that laboratories can perform, some are more important and more common. There are also certain pitfalls associated with the execution of some of the methods, and these pitfalls must be avoided to obtain accurate results.

An analytical technique for measuring ions in water-based solutions is **ISE** (Ion Selective Electrode) [20]. This technique is used by analyzers of the DxA laboratory automation systems to measure the concentration of three ions. The measured ions are Na^+, K^+ and Cl^-.

The quality of a blood sample can significantly impact the results of photometric tests; therefore, the blood quality needs to be measured [19]. The measurements are done by performing a set of methods called Lipemia, Icteric, and Hemolysis. It is common to refer to this set of methods as **LIH** methods.

Multi-step methods are tests that require measurements from multiple reactions to determine the final results. Therefore, these methods need multiple aliquots to be able to perform the different reactions. An example of such methods are methods BIL and BILK with their variants BIL blind and BILK blind; these methods determine the results by calculating the difference between two values measured where each value is gained from one version of the test [19].

Interfering methods are pairs of tests that may negatively affect their results if they are pipetted after each other. Mutual sensitivity of the used reagents causes this adverse effect, and a minimal number of washings of the pipetting needle is defined to minimize the probability of their occurrence [19].

All laboratory automation systems need to transport samples between multiple parts of the system to fulfill their function. One property of modules in modular TLA systems is **direct track sampling**. It is the ability to collect an aliquot from a sample that is placed on a conveyor, which transports the samples between the modules of the laboratory automation system without removing the sample from the conveyor and transporting it somewhere else first.

3.2 Description of modules

Both the DxA 5000 and the DxA 5000 Fit consist of multiple modules that perform pre-analytical and post-analytical processing of samples, and analyzers that perform the necessary reactions and analyses. A conveyor belt provides the connection between these different parts of the system.

Figure 3.1 depicts an example of the DxA 5000 system with some pre-analytical parts and processes colored in green, post-analytical parts colored in blue and analyzer connections in purple. Centrifugation (1), sample quality detection (2), tube decapping (3) and system entry point (4) are display as examples of pre-analytical parts and processes. Post-analytical examples of sample processing are sample storage (9), sample retrieval (10) and result auto-validation (13) [18].

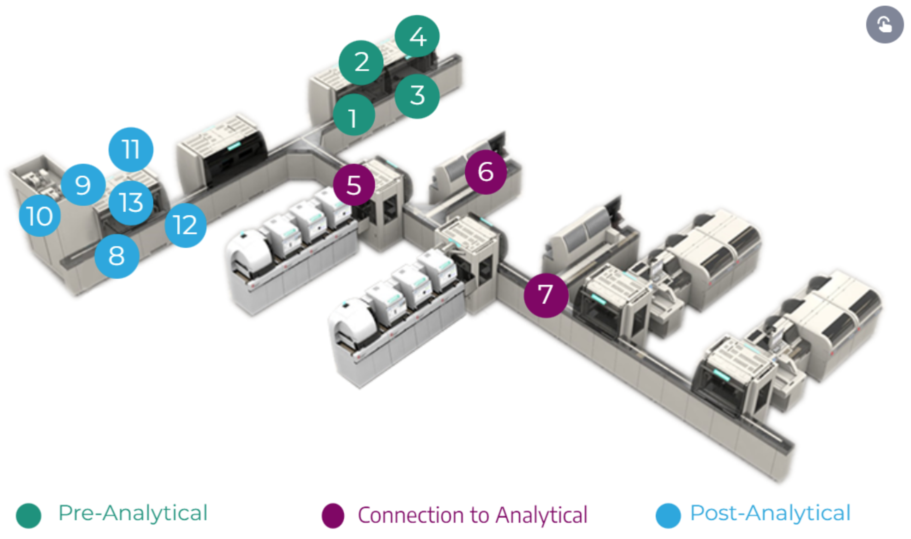


Figure 3.1: DxA 5000 Features [18].

3.2.1 Conveyor belt

A conveyor belt provides the transportation of samples between different modules of the laboratory automation system. There are two tracks present on the belt. The samples are usually transported via the inner track and only use the outer track to visit the connected analyzers or modules. The conveyor in the DxA 5000 system can additionally move a sample onto the track going in an opposite direction at specified places. Thus, the samples do not always need to travel around the whole length of the belt to reach their destinations.

3.2.2 Input-Output module

The input and output modules of the DxA 5000 system are combined into a single module in the DxA 5000 Fit system. The input module is where samples get inserted into the system, and the output module is used for their retrieval. The rest of this section describes the combined input-output module (IO module).

The IO module has two robots, one is used for removing the caps of samples on the conveyor belt, and the other robot handles the distribution of samples. New samples are introduced into the system by manual insertion into the input buffer of the IO module. The samples from the input buffer are then identified and moved into an internal buffer.

Samples from the internal buffer get handled in four different ways. Processed samples or samples that need to be processed outside the automation system are moved into the output buffer. Secondly, samples that do not need to be centrifuged are unloaded to the track. Thirdly, samples that must be centrifuged are moved into the input buffer of the centrifuge. Lastly, nothing is done with samples that are still waiting for some test results.

The distribution robot of the IO module can handle 375 samples per hour. The input module in the DxA 5000 system has two distribution robots instead of one, so it can manage two samples in parallel.

3.2.3 Centrifuge

The automation system may be equipped with a centrifuge. In the vast majority of cases, a centrifuge is present; however, its presence is not required. When the centrifuge module is present, it is located directly next to the input module or the combined IO module.

The input module prepares batches of samples for the centrifuge in specials buffers. There are four such buffers, with each holding up to 14 samples. In total, up to 56 samples can be centrifuged at once. The centrifuge spins for 4 minutes and then decelerates to a stop.

After the centrifugation finishes, the racks with spun samples are transferred from the centrifuge to a different part of the centrifugation module, where the samples are unloaded one by one onto the conveyor.

The system is designed such that the preparation of samples for centrifugation, the centrifugation of samples, and the sample unloading onto the conveyor happen simultaneously, each with different batches of samples.

The buffer exchange proceeds as follows. The racks with samples prepared for centrifugation are shifted from the input module into the centrifugation module. An empty rack is set aside from the racks in the unloading area to make space for a new rack. A rack with spun samples is moved from the centrifuge to the unloading area. A rack with prepared samples is then inserted into the centrifuge. Finally, an empty rack is moved from the unloading area among the racks with prepared samples. This cycle repeats additional three times with the remaining racks in order to move all four racks from each section to the next. The rack exchange process concludes with the input racks shifting back into the reach of the input module. The whole rack exchange process takes less than two minutes.

3.2.4 Rack Building Unit

A rack building unit (RBU) is used to connect analyzers which work with racks of samples instead of each sample individually. This module collects samples from the conveyor belt and inserts them into rack units. Priority samples are inserted into priority racks.

The racks are sent into the connected analyzer whenever a rack gets full or after a timeout is triggered. The timeout is configurable and can be set to a different value for racks of different priorities. The RBU's decision to send or not send a rack into the analyzer can also be influenced by the samples that are currently heading to it.

The RBU can also be configured to delay the process of unloading the samples back to the conveyor [19]. One of the reasons for doing so is the decreased time it takes the sample to repeat some tests on the connected analyzer. In those cases, the samples that need to redo some tests are moved to a new rack, usually a priority rack, and are sent for processing by the analyzer. The other samples from the original rack are then unloaded onto the conveyor [19].

The RBU can manipulate only one sample at a time. It takes only a few seconds to load or unload a sample. Each rack can hold up to 10 samples, and the RBU can hold up to 27 racks in its buffer.

3.2.5 Analyzer DxI 800

The DxI 800 is an immunoassay analyzer capable of direct track sampling. When a sample arrives in front of the analyzer, its label is scanned to identify it and determine how many aliquots the analyzer needs to create from the sample. The number of aliquots taken is calculated based on the total volume of the requested tests. One aliquot is pipetted if the total volume does not exceed 500 µl. If the total volume is greater than 500 µl, additional aliquots are pipetted, each covering at most another 500 µl of the total volume needed.

The analyzer can take an aliquot from a sample every 18 seconds. The aliquot pipetting itself is very fast and only takes up a small fraction of the time. The aliquot can be retrieved from the internal storage 9 seconds after the start of the pipetting cycle. Most of the time is then spent moving and washing the pipetting needle. The analyzer releases the sample after the required amount of aliquots gets pipetted. Since the aliquot pipetting itself is fast, the sample is not delayed much if only one aliquot is needed and the last pipetting of the previous sample started more than 18 seconds ago.

Four robotic arms are located inside the analyzer, and their job is to perform the requested tests. This is done by dividing the stored aliquots into reaction slots and by subsequently adding correct reagents to them [19]. Each arm can start the reaction for one method every 36 seconds. The four robotic arms are offset so that a reaction can be started every 9 seconds. However, only one arm may use an aliquot at any given time, so two arms cannot perform two different tests simultaneously for one sample if they come from the same aliquot. The necessary reagent for the given test may also be unavailable because another arm is using it; hence the execution of this test must also be postponed to a later time. Table 3.1 shows an artificial example of the inner pipetting mechanism.

Time [s]	Arm 1	Arm 2	Arm 3	Arm 4
0	M1	-	-	-
9	M1	M2	-	-
18	M1	M2	M3	-
27	M1	M2	M3	M4
36	M5	M2	M3	M4
45	M5	M6	M3	M4
54	M5	M6	-	M4
63	M5	M6	-	-
72	-	M6	-	-

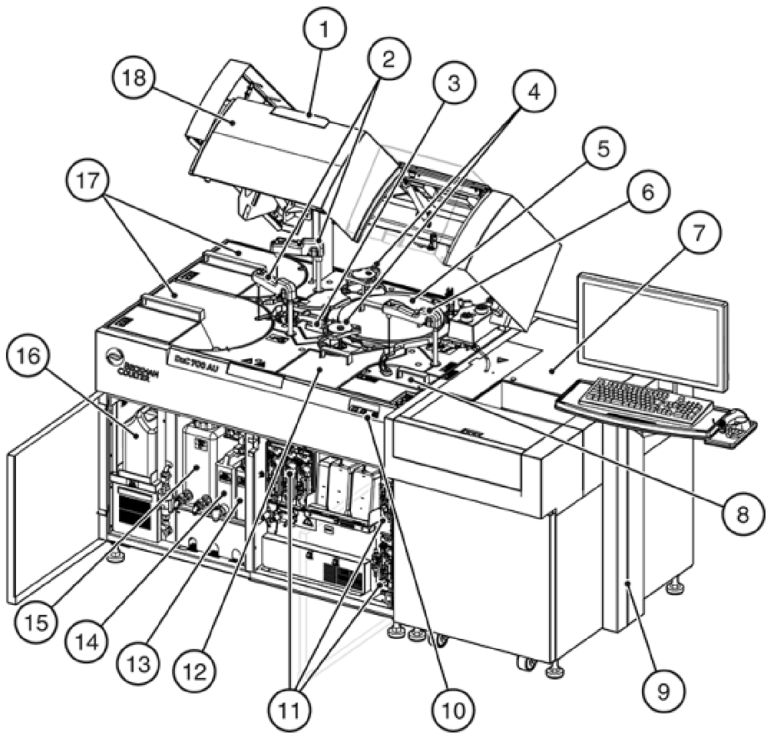
Table 3.1: An artificial example of the inner pipetting mechanism of the DxI 800 analyzer assuming the following aliquots were collected: A1={M1, M5}, A2={M2, M6}, A3={M3}, A4={M4}.

The analyzer can prioritize the execution of tests for statim samples over routine samples. The incubation time is different for each immunological method, and the results of performed tests are therefore reported separately.

A variant of this analyzer exists, and it is called DxI 600. The difference between them is that the performance of the DxI 800 is two times the performance of the DxI 600.

3.2.6 Analyzer DxC 700 AU

The DxC 700 AU is an analyzer used to perform biochemical tests, and is shown on Figure 3.2. The analyzer is capable of direct track sampling. The rack feeder module (7) is not present when direct track sampling is used and it is replaced by the track. A separate aliquot is pipetted (6) for each requested method of the sample. Multiple aliquots are collected to perform a single test if the method is a multi-step method. The length of the aliquot pipetting cycle is 4.5 seconds.



1. Status light	10. Operation buttons
2. Reagent transfer components	11. Syringe components
3. Photometry component	12. ISE Module (option)
4. Mix bar component	13. Wash solution reservoir
5. Cuvette wheel component	14. Diluted wash solution tank
6. Sample transfer component	15. Deionized water tank
7. Rack feeder module	16. 5L wash solution tank
8. STAT table	17. Reagent refrigerator components
9. Monitor arm	18. Upper cover

Figure 3.2: DxC 700 AU Hardware Overview (Top and Front View) [21].

ISE methods are handled a little differently from all other methods on this analyzer. The same pipettor (6) is used to collect the ISE aliquot and all other aliquots; however, the ISE aliquot is handled differently inside the analyzer, because it is handled by the ISE module (12) instead of the photometry component (3). Only one aliquot is needed for the ISE methods, yet another ISE cannot be pipetted after 4.5 seconds but only after 18 seconds.

The requested methods of a sample are pipetted in a defined fixed order except for ISE methods which are pipetted as soon as possible. Interfering methods are handled by waiting for an appropriate number of pipetting cycles to pass to ensure sufficient washing of the pipetting needle. The fixed pipetting order mentioned earlier is created in order to minimize the waiting caused by needle washing.

The sample is released from the analyzer after the pipetting of all aliquots for the requested methods has finished. The analyzer reserves one pipetting cycle between two consecutive samples as shown in Table 3.2

Time [s]	Pipetted sample: method
0	Sample 1: A
4.5	Sample 1: B
9	-
13.5	Sample 2: A
18	Sample 2: C

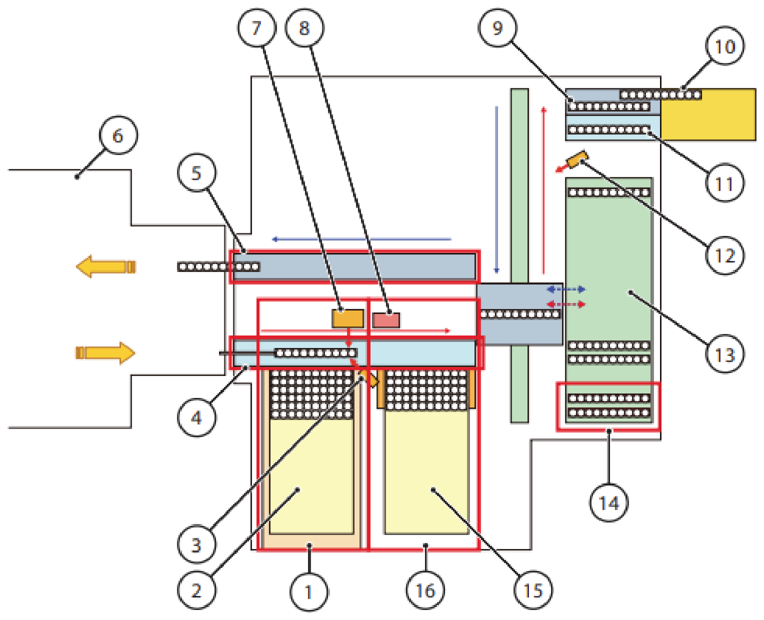
Table 3.2: DxC 700 AU pipetting example with skipped pipetting cycles (-).

The incubation time for biochemical tests is more or less the same for all of them. Therefore, the sample's test results of methods performed on this analyzer are reported collectively.

3.2.7 Analyzer series AU5800

The AU5800 is a series of biochemical analyzers that differ only in the number of photometric modules and the number of modules specialized for ISE methods. The analyzer can have anywhere from 1 to 4 photometric modules and from 0 to 2 ISE modules. The specific configuration can be derived from the name AU58XY, where X stands for the number of photometric modules and Y stands for the number of specialized ISE modules.

This analyzer is connected to the track via a rack building unit (RBU) described in Section 3.2.4 and a rack feeder unit [22] shown on Figure 3.3. The analyzer receives racks with samples from the RBU via the rack loader unit (4) and stores them inside a rack buffer unit (13). The buffer can hold up to 24 racks with up to 10 samples in each rack. When a rack is selected for processing, it is inserted into one of two processing lanes of the analyzer. The rack is inserted into a bypass lane (11) if the rack has high priority and into the primary sample transport lane (9) otherwise. The rack then visits the modules of the analyzer. The ISE modules are located before the photometric modules. When the rack reaches the end of the line it travels through the return lane (10) to the rack unloader unit (5) and to the RBU where the samples are unloaded back on the conveyor.



1. Rack set unit
2. Set tray
3. Rack ID barcode reader (LED BCR)
4. Rack loader unit from Beckman laboratory automation system
5. Rack un-loader unit to Beckman laboratory automation system
6. Beckman laboratory automation system
7. Sample ID barcode reader (Laser BCR)
8. Cup detector sensor
9. Primary sample transport lane
10. Return lane
11. Bypass lane
12. Collection rack ID barcode reader (LED BCR)
13. Rack buffer unit
14. Priority rack set unit
15. Collection tray
16. Rack collection unit

Figure 3.3: Rack Feeder Unit [22].

In the ISE module, one aliquot is pipetted from each sample with ISE requested from the rack. The pipetting order is the same as the order in the rack, and the length of this pipetting cycle is 12 seconds.

Each photometric module contains two reaction carousels that store sample aliquots. Chemical reagents are added to the aliquots to cause reactions that are used to determine the results of the requested tests. Each carousel operates with one robotic arm that pipettes the aliquots. The arm makes an aliquot every 3.6 seconds. The two carousels take turns, and so in ideal conditions, an aliquot is made every 1.8 seconds. The aliquots are pipetted only from one sample at a time, and the order of samples is given by their order in the rack. The rack can shift its position in time to allow the pipetting of an adjacent sample by the other carousel, i.e., an aliquot from the following sample can be taken 1.8 seconds after the aliquot from the previous sample.

One reason for skipping a pipetting cycle is the washing of pipetting needles caused by interfering methods. However, the methods are usually assigned to the carousels such that this happens as little as possible.

Another reason for skipping a pipetting cycle in the photometric module are odd methods [19] caused by the assignment of methods to carousels. Simply put, a pipetting cycle is skipped whenever a carousel does not have reagents for any of the not yet pipetted sample methods. Carousels contain reagents only for those methods that are assigned to them. This is done in order to save money. An example of skipped pipetting cycles due to the occurrence of odd methods is shown in Table 3.3, the missed cycles are denoted by "-".

Sample - methods	Time [s]	Inner carousel (A, B)	Outer carousel (C, D)
Sample 1 - A, C	0	A	
	1.8		C
Sample 2 - A, B	3.6	A	
	5.4		-
	7.2	B	
Sample 3 - D	9		D
Sample 4 - A	10.8	A	
	12.6		-

Table 3.3: An example of skipped pipetting cycles (-) caused by odd methods.

The rack feeder unit sends the racks from the buffer to the analyzer whenever a line has enough space for a new rack. The maximum number of racks in a module is 2 in a line. Samples from one are undergoing aliquot pipetting, and the other waits for the first to be done. The only exception is in front of the first module, where one extra rack can wait. The modules process racks in the bypass lane first.

The incubation time is the same as in the case of DxC 700 AU. Therefore, the sample's test results of methods performed on this analyzer are also reported collectively.

3.2.8 Other

Other modules and analyzers than the ones specified in Section 3.2 can be used with the DxA 5000 laboratory automation system. The system also supports analyzers for hematology [18], coagulation [19] and a selection of other third party instruments [18]. An example of this is an installation [19] of the DxA 5000 system that uses an immunological analyzer LiaisonXL. One commonly used module is a storage module that allows for easier management of processed samples.

3.3 Sample processing workflow

This section describes the decision processes of the laboratory automation system, which are tied to the sample processing workflow.

3.3.1 Sample routing

A sample's route is determined by its route plan. The route plan contains analyzers with assigned methods that the analyzer should perform for the sample. The sample visits the analyzers in the same order as specified in the route plan.

The route plan is created when the input module identifies the sample. The route plan creation process considers multiple factors when creating the route plan. A detailed description of this process is omitted to prevent the disclosure of trade secrets. In general, the process tries to prioritize analyzers according to user settings, balance analyzer loads, and minimize the number of analyzers visited for each sample, among other things.

3.3.2 Sample centrifugation

Samples, such as blood, that require the specimen to be separated into different layers with centrifugal force start by visiting the centrifuge. The system selects samples for centrifugation based on priority and expected load of the analyzers caused by the batch of selected samples.

When no samples are present inside the centrifuge and no other samples from the input can be included in the next centrifugation cycle, the system waits a certain amount of time for any newly inserted samples that could be included. This timeout duration depends on the highest sample priority from samples that are included in the next centrifugation cycle.

3.3.3 Repeated testing

Sometimes one or multiple tests must be repeated to ensure that the obtained results are correct. When this happens, the system waits for the sample to finish executing the current route plan, and then it creates a new route plan based on the tests that need to be repeated. The sample executes the new route plan and is usually treated as having high priority. The centrifuge is not revisited this time.

3.3.4 Custom rules

The users can implement custom system rules. In one installation of the DxA 5000 system, samples wait for biochemical results in an RBU of an AU5811 analyzer in order to decrease the time it takes to perform repeated tests [19]. The user can also influence the route plan creation process by specifying rules for assigning specific methods between analyzers, which is demonstrated in the same installation as mentioned above.

3.3.5 Laboratory configurations

The relative placement of modules to each other in a laboratory is highly dependent on multiple factors. Available space, test prioritization, general laboratory workflow, and many other aspects are considered. Analyzers from the same group, such as biochemistry or hematology, are generally placed close together. An example of laboratory configuration with the DxA 5000 Fit system is an L-shaped conveyor with the IO module at one end, one or two DxI analyzers in the middle, and two DxC 700 AU analyzers at the other end. However, another laboratory might prefer the IO module to be in the middle with immunology on one side and biochemistry on the other side of the IO module.

Chapter 4

Simulator design

This chapter addresses the design of the application that simulates the DxA 5000 Fit laboratory automation system. First, the general application requirements are presented in Section 4.1. The architectural design is explained in Section 4.2, followed by description of inputs and outputs in sections 4.3 and 4.4. Finally, some information about the implementation and used technologies is provided in Sections 4.5, 4.6 and 4.7. Additionally, images depicting the user interface and graphical outputs are attached in the appendix.

4.1 Requirements

4.1.1 Functional requirements

1. Supports simulation of the DxA 5000 Fit laboratory automation system.
2. Supports the following laboratory modules:
    (a) Input-Output
    (b) Centrifuge
    (c) DxI 800
    (d) DxC 700 AU
    (e) AU5811
3. User can create custom laboratory configurations.
4. User can use laboratory data as input for the simulation.
5. User can view comprehensive graphical analysis of the results.
6. User can download simulation results for further analysis.

4.1.2 Non-functional requirements

1. The application should run locally without accessing the internet.
2. The application should be compatible with Windows 10 operating systems.

3. The application should be easy to use.
4. Simulations should be fast.
5. Simulations should faithfully imitate the behavior of real-world laboratories.
6. The application should be extensible for future additions of modules.

4.2 Architecture

The application is quite simple from an architectural point of view. Users interact with the application via their preferred browser. A simple local HTTP server serves content to the user based on rules defined for specified local URL routes. When appropriate, the server prompts the operating system to spawn a new process that performs a discrete event simulation. The input data for the simulation process and its output are stored as files in the application's directories on the file system. Figure 4.1 depicts the simulator's simple architecture.

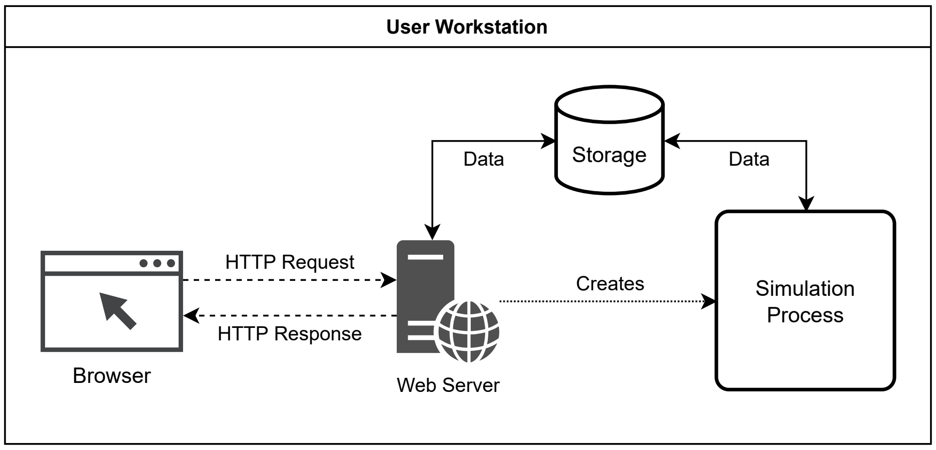


Figure 4.1: General architecture of the simulator.

There are many benefits to the selected architectural design. First of all, the architecture naturally promotes separation of concerns, specifically the user interface, data validation, and simulation. Additionally, the simplicity allows for easy adaptations. For example, it would be easy to have a company's employees use the simulator remotely by hosting the simulator on the company's server. An inclusion or addition of authorization and a transition to database storage should not be problematic. Similarly, the simulation processes could be queued and easily modified to be executed by a group of workers.

4.3 Input data

The simulator requires two files with input data to perform a simulation. One file contains information about samples and requested tests, and the other describes various parameters of the simulated laboratory system. Section 4.3.1 describes the sample inputs and Section 4.3.2 describes the file with laboratory parameters.

4.3.1 LIS simulation data

The simulator needs information about samples and tests to simulate. This is provided in a CSV (comma-separated values) file. Laboratories use laboratory information systems (LIS) to gather and manage information about requested and performed sample tests. This data can be used as an input to the simulator. So, the input CSV file uses one of the LIS data formats. The fields are listed below in order of their occurrence in the CSV file.

1. sample identifier
2. date of receipt
3. time of receipt
4. method
5. stat
6. material
7. execution date
8. execution time
9. date of collection
10. time of collection

A semicolon is used as a separator for the fields. All dates are in the format "YYYY-MM-DD" and time uses the format "HH:MM". The simulator uses the date and time of receipt as the time when the sample is available for processing. Samples that contain multiple different times of receipt are ignored and listed to the user in a simulation's report. The execution time is used to graph information about the real performance of a laboratory. The collection time is unused.

The method field specifies a method that needs to be performed for the sample. The stat field describes the sample's priority. Samples with standard priority are represented by the value "0" and samples with high priority by "1". The material field is used to decide whether or not a sample needs to be centrifuged. Only serum and plasma samples are centrifuged.

4.3.2 System description file

A JSON [1] file is used as another input to the simulator to describe various parameters of the simulated system. The file is validated against a JSON schema [2]. However, some things cannot be validated by the schema alone, so extra validation is implemented and performed by the HTTP server. The file contains three main groups of parameters called Methods, Modules, and Configuration.

[1]https://www.json.org/json-en.html
[2]https://json-schema.org/

The Methods group contains parameters for every method that can be simulated. All four parameters in Table 4.1 must be specified for each method under its identifier used as a key.

Parameter	Unit/Type
duration	seconds
panel	string
steps	aliquots
volume	µl

Table 4.1: Method parameters.

The duration parameter describes the method's incubation time. The panel parameter defines a group of methods to which the method belongs. The steps parameter describes the number of aliquots needed to perform the method. This is used by the DxC 700 AU and the AU5811 analyzers. Finally, the volume parameter defines the method's aliquot volume pipetted by the DxI 800 analyzer.

The Module group contains parameters for simulation models of the DxA 5000's modules. Parameters of the Input-Output module and the Centrifuge are summarized in Table 4.2 and Table 4.3. Analyzer parameters are listed in Table 4.4 (DxI 800), Table 4.5 (DxC 700 AU) and Table 4.6 (AU5811). The parameters of a rack building unit are listed in Table 4.7.

Parameter	Unit
sample manipulation time	seconds

Table 4.2: Input-Output module's simulation parameters.

Parameter	Unit
capacity	samples
sample manipulation time	seconds
spinning duration	seconds
rack exchange duration	seconds
timeout	
- priority	seconds
- routine	seconds

Table 4.3: Centrifuge's simulation parameters.

Parameter	Unit
aliquot pipetting duration	seconds
method pipetting duration	seconds
max. aliquot volume	µl
max. theoretical hour capacity	tests

Table 4.4: DXI 800's simulation parameters.

Parameter	Unit
photometry pipetting duration	seconds
ISE pipetting duration	seconds
max. theoretical hour capacity	tests

Table 4.5: DxC 700 AU's simulation parameters.

Parameter	Unit/Type
photometry pipetting duration	seconds
ISE pipetting duration	seconds
max. ISE queue	racks
max. photometry queue	racks
max. theoretical hour capacity	tests
RBU	RBU parameters

Table 4.6: AU5811's simulation parameters.

Parameter	Unit
sample manipulation time	seconds
rack capacity	samples
transport time	seconds
timeout	
- priority	seconds
- routine	seconds

Table 4.7: RBU's simulation parameters.

The last group of parameters describes the laboratory configuration and some simulation settings. The parameters are listed in Table 4.8 with details for some map values in Table 4.9.

Parameter	Unit/Type
with centrifuge	boolean
method fail percentages	map (method panel $\rightarrow$ number from range [0, 100])
LIH policy	one of: never / always / <specific method panel>
ISE methods	list of methods
modules	module map
method assignment	assignment map

Table 4.8: DxA configuration parameters.

As stated in Section 3.3.3, it is sometimes necessary to repeat some tests. A percentage of methods that will need to be repeated can be defined separately for each group of methods. LIH methods test blood quality and can either not be performed at all, be always performed, or be performed only for samples with tests from a selected group. A list of methods from the input file described in Section 4.3.1 that together form ISE methods must be specified.

Additionally, analyzers and modules used by the simulated laboratory must be defined. Each position on a conveyor is mapped to a module. The position is expressed as a non-negative integer, and the module is defined by parameters listed in Table 4.9.

An IO module must always be present. One module must also always be connected to the conveyor's base position, i.e., the position at index 0. The ”travel time” parameter describes how long it takes a sample to travel from the base position to the connected module. However, the parameter corresponds to the time it takes a sample to travel around the whole conveyor when the module's position is the base position.

Parameter	Unit/Type
id	unique string
type	one of: IO / DXI 800 / DXC 700 AU / AU5811
priority	integer
travel time	seconds

Table 4.9: Module map's value parameters.

Lastly, each analyzer has a pool of tests it can perform. A list of available methods is mapped to the unique identifier of each connected analyzer. A list of available methods is assigned to each carousel for the AU5811 analyzers.

4.4 Output data

The simulation process generates two data outputs. Each data entry in the first output file consists of the following information. The second pipetting time is relevant only for methods that were measured repeatedly. Otherwise, it is equal to the first pipetting time.

- sample identifier
- method identifier
- timestamp of the time at which the method's results were available
- identifier of the analyzer that performed the method
- timestamp of the first time the sample's method was pipetted
- timestamp of the second time the sample's method was pipetted

This data output is used to generate graphical outputs provided by the simulator. The graphs are interactive, and the user can download each graph as an image or export the underlying data with a single click. The simulator provides the following graphs.

- Analyzer utilization over time
- Breakdown of methods executed on the analyzers
- Comparison of analyzer workload over time
- Histogram of sample TAT differentiated by priority

- Distribution of sample TAT differentiated by priority
- Sample TAT based on sample arrival time
- Pre-analytical time and method execution time based on sample arrival time
- Histogram of sample arrival time by priority
- Histogram of sample arrival time by methods
- Histogram of sample arrival time by methods separately for each sample priority
- Histogram of sample arrival time by sample type
- 2D visualization of 5-dimensional data about sample methods
  - Method type
  - Analyzer that performed the method
  - Sample priority
  - Sample TAT
  - Type of methods requested by the sample
- Analysis of the real laboratory data
  - Histogram of sample TAT differentiated by priority
  - Distribution of sample TAT differentiated by priority
  - Sample TAT based on sample arrival time
  - Method execution time based on sample arrival time

Examples of all graphs are available in the appendix. In addition to the graphs, the simulator provides information about the simulated configuration, a list of ignored methods and samples, and the total number of simulated methods and samples.

The second output file provides the simulation's event log. Each data entry consists of a timestamp, sample's identifier, event type, and additional information relevant to the reported event. The different event types mimic events reported by laboratory automation systems and are listed in Table 4.10.

Event type	Additional information
CENTRIFUGATION START	-
METHOD PIPETTING	method id, analyzer id and carousel
NEW RESULT	method id, analyzer id and result validity
NEW ROUTE PLAN	route plan
NEW SAMPLE	sample priority
RACK TRANSPORT	analyzer id and transport direction
SAMPLE ARRIVAL	module id
SAMPLE PROCESSED	-
SAMPLE RELEASE	module id
SAMPLE SCHEDULED FOR RERUN	-

Table 4.10: List of logged events with additional information about details provided.

4.5 Simplification of the simulated systems

The simulation models the systems primarily based on the description provided in Chapter 3. However, some processes were simplified with the permission of the users.

4.5.1 IO module

The limited rack capacity of buffers in the IO module is not modeled. Additionally, the sample manipulation is performed in a round-robin manner. One sample is moved in each round unless there are no tubes available in which case the round is skipped without any time loss.

4.5.2 DxI 800

Reagent availability during the inner method pipetting performed by the DxI 800 analyzer is not modeled. Instead, the internal pipetting process is simplified. Each method's inner pipetting time is computed based on the number of methods performed from the same aliquot. Assuming that k methods are pipetted from the same aliquot, and d equals the inner method pipetting cycle's duration, then one method would be pipetted every d seconds. By averaging over all methods from the aliquot, we get that the additional delay caused by inner pipetting is equal to $\frac{d(k+1)}{2}$ for each method from the aliquot.

4.5.3 DxC 700 AU and AU5811

Cycles reserved for needle washing between the pipetting of interfering methods are not considered. This simplification is possible since the methods are pipetted in order such that the additional washing is rarely needed.

4.5.4 Repeated tests

As shown in Table 4.8, one input parameter is the percentage of repeated tests for each group of methods. To achieve consistent simulation results, the methods that will be repeated are preliminarily decided in a deterministic manner. Assuming p percent of methods from group g should be repeated, we can compute the number n of consecutive methods that are not repeated according to (4.1).

$$n = \frac{100 - p}{p} \tag{4.1}$$

After n sample methods from group g, the following sample method from group g is repeated. This continues with another n methods that are not repeated and then one repeated method again.

4.6 Technological stack

The simulator is implemented in Python and is compatible with versions 3.9 and higher. SimPy [3], Plotly [4] and Flask [5] are the main libraries used to implement the simulator. Flask provides a built-in HTTP server and allows for clear and easy web interface implementation. Plotly provides a high-quality interactive graph functionality, and SimPy is a library for creating discrete event simulations. Additionally, PyInstaller [6] was used to bundle the application into a single package for distribution.

4.6.1 SimPy workflow

The SimPy library provides utilities to create and execute discrete event simulations. In essence, any process can be described as a series of events with some time passing between them. The library allows the developer to register events and processes that can then be executed in the correct order. The processes are implemented by defining python generators that generate related events. The simulated system's parts are essentially implemented by describing various processes that occur in the simulated system with additional encompassing logic. The simulation then runs until all samples are processed.

4.6.2 Simulation speed

The number of generated events heavily influences the execution speed of simulated processes. Fewer events result in faster simulations. The number of events scales with the number of input samples and methods; however, it also depends on the implementation of models for the DxA5000 system and its components. Multiple possible implementations of various processes were considered in order to decrease the number of events generated by the models. An example of such optimized implementation is the pipetting on analyzers AU5811. A naive implementation would generate an event for each started pipetting cycle; however, the analyzers are never used 100% of the time. Since the AU5811 starts a new pipetting cycle every 1.8 seconds, this implementation would generate a lot of events that would be ignored. The clever implementation generates pipetting events only when there are samples to pipette aliquots from.

4.7 Object models

The simulated system was primarily implemented using an object-oriented programming approach. This section provides a brief summary of the main classes used by the simulator.

4.7.1 Data classes

- LoggedEvent - holds information about a logged simulation event
- Method - holds parameters of a method as per Table 4.1

[3]https://simpy.readthedocs.io/en/latest/
[4]https://plotly.com/python/
[5]https://flask.palletsprojects.com/en/2.1.x/
[6]https://pyinstaller.org/en/stable/

- Sample - holds information about a sample similar to those listed in Section 4.3.1
- RoutePlan - sample's route plan describing the order of analyzers and methods performed on them
- ModuleInfo - holds information about a module connected to the conveyor

4.7.2 Enums

- EventType - types of logged events
- SpecialMethods - specially treated methods such as LIH and ISE
- SamplePriority - differentiated sample priorities
- MethodResultValidity - validity of performed tests
- CentrifugeState - states of the centrifuge module

4.7.3 System models

- ExtendedSimpyEnvironment - extends SimPy's simulation environment with access to simulation logger and functionality providing mapping of the simulation time to corresponding date and time and vice versa
- SimulatedMethodResult - holds information about a performed method with additional functionality necessary for the simulation
- SimulatedSample - encapsulation of Sample that provides the functionality necessary for the simulation
- SampleRack - models racks for samples
- SampleRackFactory - provides new racks with consistent parameters
- RackBuildingUnit - models rack building unit for analyzers without direct track sampling
- Module - abstract class defining the common interface of system modules
- DTSAnalyzer - abstract class defining the common interface of analyzers with direct track sampling
- RBUAnalyzer - abstract class defining the common interface of analyzers without direct track sampling
- Conveyor - connect system modules and moves samples between them
- Centrifuge - models the centrifuge functionality
- IO - models the IO module functionality
- DXI800 - models the functionality of DxI 800

- DXC700AU - models the functionality of DxC 700 AU
- AU5811 - models the functionality of AU5811
- DxA5000Fit - abstracts the simulation process
- PlannedAnalyzerLoadsTracker - tracks analyzer loads based on route plans
- MonitoredQueue - encapsulation of a queue [7] from standard library that allows the tracking of operations performed on the queue with SimPy events
- LIHPolicy - policy deciding planning of LIH methods
- MethodFailurePolicy - decides repeated method testing
- RoutePlanCreationPolicy - creates sample route plans
- MaxAnalyzerLoadPolicy - allows to make analyzer load balancing decisions

[7] collections.deque

Chapter 5

Simulator testing

This chapter presents testing methodology used during the development process of the simulator in Section 5.1. Sections 5.2 and 5.3 inform about simulation speed benchmarks and user acceptance testing.

5.1 Development tests

The simulator development relied on a great understanding of the simulated systems. For this reason, the majority of time was spent on the creation of a detailed specification. At first, a specification written in natural language was created, and then it was reviewed by the users to ensure its correctness. The reviewed specification was then formalized according to Table 5.1 to function as a base for the system's design.

Unique identifier	Name of the requirement
Purpose	Description of the purpose behind this requirement
Action	Description of actions taken to generate the output from the input
Input	List of inputs
Output	List of outputs

Table 5.1: Structure of requirements in formalized system specification.

Static testing was also performed on the source code during the development process. Static type checking played a significant role since the simulator was developed in Python, a dynamically typed language. In addition to type checking, Pylance [1] was used for other source code diagnostics including analysis for potential errors.

The development process consisted partially of test-driven development. In other cases, the code was first written and then tested. An unresolved problem with the use of the SimPy library halted the implementation of automated tests for units described by process generators used for the discrete event simulation. So, the corresponding unit and integration tests were instead performed manually.

Component and system correctness was also verified with end-to-end testing. Artificial simulator inputs were created for each testing scenario, and the simulation log was compared

[1]https://marketplace.visualstudio.com/items?itemName=ms-python.vscode-pylance

to the expected behavior. The testing scenarios included simple simulation speed tests. More precise performance testing is described in Section 5.2.

The testing process concluded with user acceptance tests that are described in Section 5.3. Figure 5.1 shows the development and testing process.

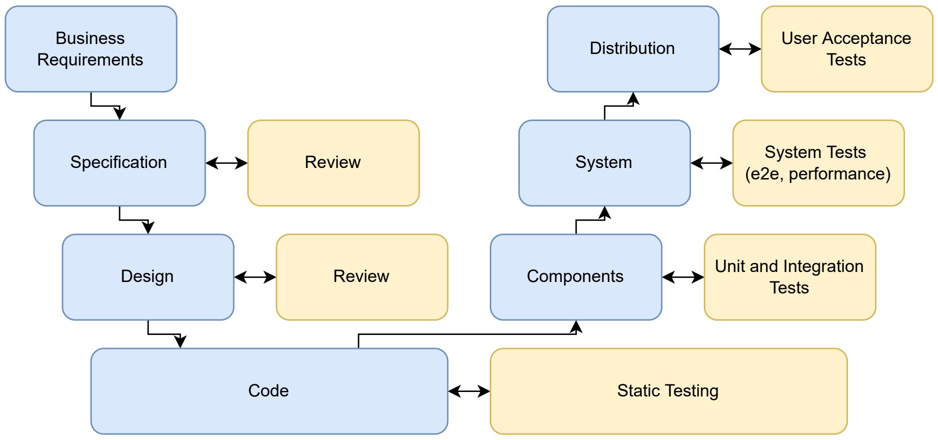


Figure 5.1: Development and testing process.

5.2 Simulation speed benchmarks

Section 4.1.2 lists non-functional requirements imposed on the simulator. One of them is short simulation time. In the worst case, simulations should not take more than a few minutes. This section describes how the system's performance was measured and presents the results.

5.2.1 Methodology

Multiple testing scenarios were prepared. The testing scenarios differed in either the number of simulated samples and requested methods or the laboratory system's configuration. The execution time of the simulation process was measured using Python's performance counter clock [2]. The execution time for each scenario was measured five times. Mean values including 99% confidence intervals were computed from the measurements according to the standard error margin formula

$$e = 2.576\frac{\sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(x_i - \bar{x}\right)^2}}{\sqrt{N}}. \tag{5.1}$$

Parameters of the machine used for performance testing are listed below.

- Intel Core i5-8250U
- Microsoft Windows 10 Home (x64) Build 19043.1645
- 8GB 1200MHz CL10 Dual-Channel RAM
- 256 GB SSD

[2]https://docs.python.org/3/library/time.html#time.perf_counter

5.2.2 Results

Tables 5.2, 5.3, 5.4 and 5.5 describe measured configurations and their performances. The first three columns describe the measured configuration, the corresponding number of simulated samples, and the number of simulated methods. The fourth column contains the mean execution time in seconds including the range of the 99% confidence interval. At last, the overloaded parameter indicates configurations in which the number of requested methods exceeded the maximal analyzer load of the selected configuration for a prolonged amount of time.

Configuration	Number of Samples	Number of Methods	CPU Time [s] (99% confidence)	Overloaded
1x DxI 800	334	563	1.22 ± 0.09	No
1x DxI 800	668	1126	1.61 ± 0.08	No
1x DxI 800	1336	2252	2.31 ± 0.19	Yes
2x DxI 800	334	563	1.44 ± 0.07	No
2x DxI 800	668	1126	1.56 ± 0.09	No
2x DxI 800	1336	2252	2.38 ± 0.28	No
2x DxI 800	2004	3378	3.06 ± 0.17	Yes

Table 5.2: Performance of configurations with analyzers DxI 800.

Configuration	Number of Samples	Number of Methods	CPU Time [s] (99% confidence)	Overloaded
1x DxC 700 AU	1126	8876	3.77 ± 0.11	Yes
2x DxC 700 AU	1126	8876	3.46 ± 0.06	No
2x DxC 700 AU	2252	17752	6.88 ± 0.16	Yes

Table 5.3: Performance of configurations with analyzers DxC 700 AU.

Configuration	Number of Samples	Number of Methods	CPU Time [s] (99% confidence)	Overloaded
1x AU5811	1126	8876	3.99 ± 0.06	No
1x AU5811	2252	17752	7.06 ± 0.21	Yes
2x AU5811	1126	8876	4.10 ± 0.07	No
2x AU5811	2252	17752	7.36 ± 0.19	No
2x AU5811	4504	35504	15.24 ± 0.39	Yes

Table 5.4: Performance of configurations with analyzers AU5811.

Configuration	Number of Samples	Number of Methods	CPU Time [s] (99% confidence)	Overloaded
2x DxI 800 + 2x DxC 700 AU	1232	9439	3.56 ± 0.10	No
2x DxI 800 + 2x DxC 700 AU	2464	18878	7.14 ± 0.20	Only DxC analyzers

Table 5.5: Performance of configurations with mixed analyzers.

The results indicate that the execution time of configurations with biochemical analyzers scales linearly with the number of requested methods. Figure 5.2 compares the execution times together with corresponding linear regressions. In conclusion, the simulator satisfies the imposed speed requirements.

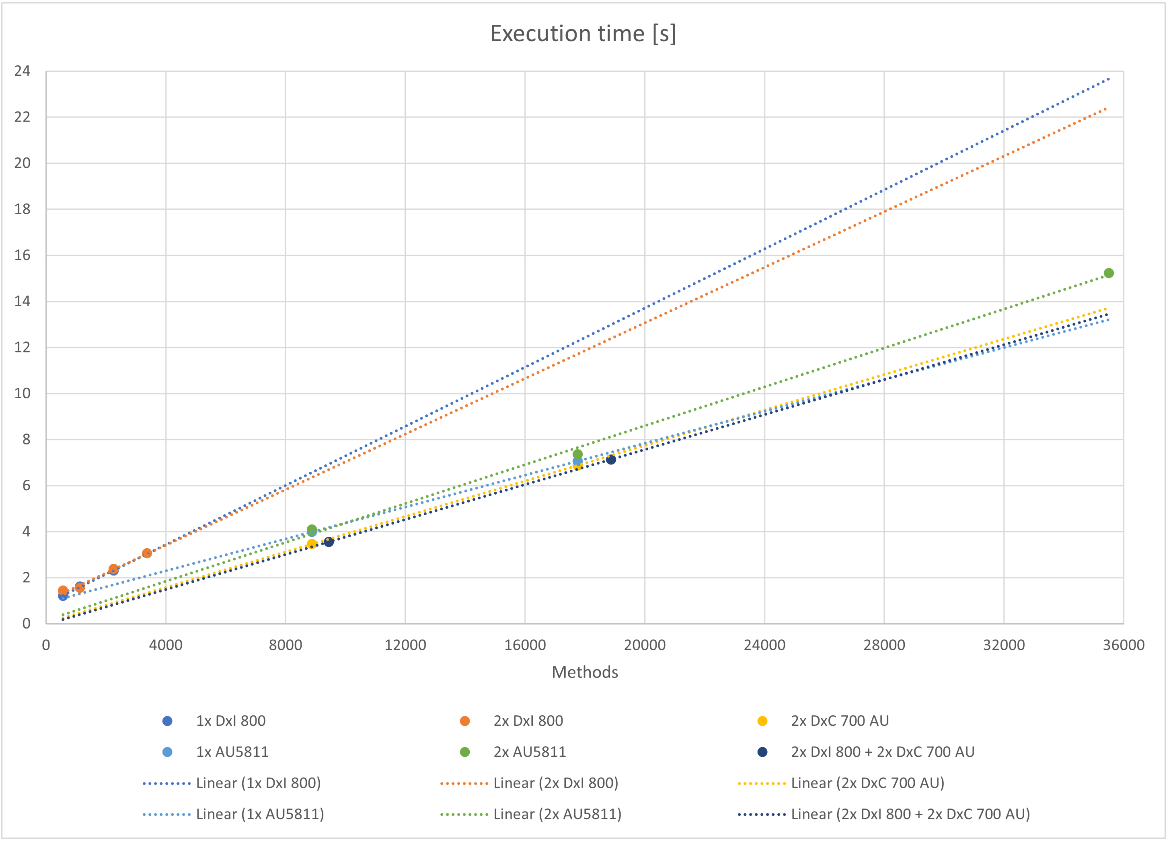


Figure 5.2: Comparison of execution times for different configurations.

5.3 User acceptance tests

A piece of software needs to meet the predefined set of requirements and get accepted by the end-users to be considered a finished product. The simulator was thoroughly tested by multiple experts in the domain of laboratory design and analysis. Simulations of several laboratories were performed, and the simulation results were compared to the real operational data analysis results. The simulator shows errors within the range expected by the users, so it meets the precision requirements and is well-suited for laboratory analysis. As a result, the simulator will significantly help with future laboratory designs since it will be used to verify the suitability of technological proposals.

Chapter 6

Laboratory optimization problem

This chapter presents an improved version of a mathematical model balancing loads of biochemical analyzers in the DxA 5000 laboratory automation system. A detailed description of the optimization problem and the original model is presented in Březina's master's thesis [19]. A brief summary of the optimization problem and the original model is given in Section 6.1. The improved model and the reasoning behind the introduced changes are presented in Section 6.2. Březina's mathematical notation is adopted for easy comparison of the models.

6.1 Problem and Model Description

The section briefly introduces the laboratory, and the corresponding load balancing problem addressed by Březina in his thesis [19]. The optimization problem is introduced in Section 6.1.1. Section 6.1.2 introduces and briefly describes Březina's mathematical model that is revised and improved in Section 6.2.

6.1.1 Problem statement

Laboratories may suffer from imbalanced distribution of work among the available analyzers. Karel Gevančiak [23] was the first to observe an imbalanced workload distribution in a hospital laboratory on system Cobas 8000. The imbalance is caused by a suboptimal distribution of chemical reagents among the available reaction carousels given the ordinary spectrum of samples and requested tests for the laboratory. The test results are generated based on different reactions each requiring a specific reagent. This means that different tests are assigned to different reaction carousels based on the reagents available for the given carousel. The imbalance could be avoided by distributing all reagents to all carousels. However, this solution greatly increases material costs for the laboratory since it is common to assign almost every method to only one of the reaction carousels.

The imbalance causes high utilization of some carousels, which increases waiting times in front of the affected analyzers. This results in the overall increase in sample TAT. Therefore, the goal is to achieve uniform carousel utilization levels. The maximal carousel workload decreases when the workload is distributed more uniformly among the carousels, so the problem can be formulated as minimizing the maximal workload among the carousels. This is expressed in Equation (6.1) where C is a set of reaction carousels, c is a carousel from the set C, and l_c is the workload of carousel c. We assume a scenario with three biochemical carousels, one carousel in the DxC 700 AU analyzer and two in the AU5811 analyzer.

$$\min \max_{c \in C} l_c \tag{6.1}$$

A carousel workload depends on sample methods executed by the given carousel. The spectrum of samples and requested tests varies and cannot be controlled. However, a carousel's workload can be influenced by changing the set of available tests which are associated with the given carousel. So, the idea is to find an optimal assignment of methods to carousels that minimizes the maximal workload among the carousels.

6.1.2 ILP Representation

This section starts with Březina's ILP (integer linear programming) model and lists all used sets, parameters, decision variables, and their respective meanings. A brief explanation of the mathematical representation follows. The model is taken from Section 4.2 of Březina's master's thesis [19].

The model uses parameters based on laboratory operational data, i.e., a log with information about requested and performed tests across multiple days, so its complexity is dependent on the data size.

$$\begin{aligned}
\min \sum_{D} \max \Bigg\{ & pt^1 \left(\sum_{J} p_j s_{d,j} x_j^{1,1} + \sum_{I_d} h_{d,i}^{1,1} \right) + pt^1 \sum_{I_d} \max\{0, n_{d,i}^{1,2} - n_{d,i}^{1,1}\}, \\
& pt^1 \left(\sum_{J} p_j s_{d,j} x_j^{1,2} + \sum_{I_d} h_{d,i}^{1,2} \right) + pt^1 \sum_{I_d} \max\{0, n_{d,i}^{1,1} - n_{d,i}^{1,2}\}, \\
& pt^2 \left(\sum_{J} p_j s_{d,j} x_j^{2,1} + \sum_{I_d} (1 - h_{d,i}) + ise_d \right) \Bigg\}
\end{aligned}$$

$$\sum_{A}\sum_{B^a} x_j^{a,b} = 1 \qquad \forall j \in J \tag{6.2}$$

$$x_j^{a,b} + x_{j'}^{a,b} \leq 1 \qquad \forall \left(j, j'\right) \in F, \forall a \in A, \forall b \in B^a \tag{6.3}$$

$$x_{BIL}^{1,1} = 0; x_{BILK}^{1,1} = 0 \tag{6.4}$$

$$h_{d,i}^{1,1} + h_{d,i}^{1,2} = h_{d,i} \qquad \forall d \in D, \forall i \in I_d \tag{6.5}$$

$$\frac{|J_i| - \sum_{J_i} x_j^{2,1}}{|J_i|} \geq h_{d,i} \geq 1 - \sum_{J_i} x_j^{2,1} \qquad \forall d \in D, \forall i \in I_d \tag{6.6}$$

$$n_{d,i}^{1,2} = \sum_{J_i} p_j x_j^{1,2} + h_{d,i}^{1,2} \qquad \forall d \in D, \forall i \in I_d \tag{6.7}$$

$$n_{d,i}^{1,1} = \sum_{J_i} p_j x_j^{1,1} + h_{d,i}^{1,1} \qquad \forall d \in D, \forall i \in I_d \tag{6.8}$$

$$t_{d,i} \geq \sum_{B^a} x_j^{a,b} + \sum_{B^{a'}} x_{j'}^{a',b} - 1 \qquad \forall d \in D, \forall i \in I_d, \; \forall j, j' \in J_i : j \neq j', \forall a, a' \in A : a < a' \tag{6.9}$$

$$\alpha \geq \frac{1}{|D|} \sum_{I_d} \left(1 - st_{d,i}\right) t_{d,i} \tag{6.10}$$

Sets:

- D - The set of all considered days.
- I - The set of all samples. I_d denotes a set of samples processed in day d.
- J - The set of all biochemical methods except LIH and ISE. J_i represents regular methods requested by sample i.
- A - The set of biochemical analyzers (AU and DXC).
- B^a - The set of carousels of analyzer a.
- F - The set of all pairs of interfering methods.

Parameters:

- pt^a - The pipetting operation length of analyzer a.
- p_j - The number of aliquot collections needed to process method j.
- $s_{d,j}$ - The total quantity of samples requesting method j in day d.
- ise_d - Estimation of the ISE methods performed on the DXC analyzer.
- $st_{d,i}$ - Binary parameter equal to one if sample i requests long immunological method. If so, the transport of such a sample is not considered.
- α - Upper bound on the average daily amount of occurred transports.

Decision variables:

- $x_j^{a,b}$ - A binary variable equal to one if method j is assigned to a carousel b of analyzer a. Note that AU has two carousels, whereas DXC has only one.
- $h_{d,i}$ - A binary variable determining whether the LIH methods of sample i from day d will be performed on the AU analyzer or not. The variable is equal to one if LIH are performed by the AU and zero otherwise.
- $h_{d,i}^{a,b}$ - If AU performs the LIH methods, it is necessary to decide on which carousel. This binary variable is equal to 1 if LIH of sample i are performed by carousel b of the AU analyzer.
- $n_{d,i}^{a,b}$ - The number of aliquots collected by the carousel b of analyzer a in order to process sample i. This variable determines the number of odd methods.
- $t_{d,i}$ - A binary variable. Equal to one, if sample i from day d needs to visit both biochemical analyzers.

The model's criterion function is a version of the criterion expressed in Equation (6.1) which minimizes the maximal workload among the biochemical carousels. It is essentially formed from the workload of the three available carousels. The workload of the carousels can be expressed as time spent by active pipetting at and some idle time it. Equation (6.11) expresses the workload of AU5811's inner carousel, and Equation (6.12) expresses the workload of AU5811's outer carousel. DxC 700 AU's workload is expressed in Equation (6.13).

$$w_d^{1,1} = at_d^{1,1} + it_d^{1,1} \tag{6.11}$$
$$w_d^{1,2} = at_d^{1,2} + it_d^{1,2} \tag{6.12}$$
$$w_d^{2,1} = at_d^{2,1} + it_d^{2,1} \tag{6.13}$$

The active pipetting time for the AU5811's inner and outer carousels is shown in Equations (6.14) and (6.15). The second sum expresses pipetting of LIH methods, and the first sum includes the pipetting of all other methods. ISE methods are not included in the active time, because they are performed separately on the AU5811.

$$at_d^{1,1} = pt^1 \left(\sum_J p_j s_{d,j} x_j^{1,1} + \sum_{I_d} h_{d,i}^{1,1} \right) \tag{6.14}$$

$$at_d^{1,2} = pt^1 \left(\sum_J p_j s_{d,j} x_j^{1,2} + \sum_{I_d} h_{d,i}^{1,2} \right) \tag{6.15}$$

The problem of odd methods, which is introduced in Section 3.2.7, causes the AU5811 to sometimes skip pipetting cycles. The idle time of AU5811's carousels is made of these skipped pipetting cycles and is expressed in Equations (6.16) and (6.17).

$$it_d^{1,1} = pt^1 \sum_{I_d} \max\{0, n_{d,i}^{1,2} - n_{d,i}^{1,1}\} \tag{6.16}$$

$$it_d^{1,2} = pt^1 \sum_{I_d} \max\{0, n_{d,i}^{1,1} - n_{d,i}^{1,2}\} \tag{6.17}$$

There is no idle time for the workload of the DxC 700 AU's carousel; however, the active time is made of three parts. The first two parts correspond to the same methods as in the case of AU5811, but an additional term caused by the pipetting of ISE methods is present. The DxC's active time is expressed in Equation (6.18) and idle time in Equation (6.19).

$$at_d^{2,1} = pt^2 \left(\sum_J p_j s_{d,j} x_j^{2,1} + \sum_{I_d} (1 - h_{d,i}) + ise_d \right) \tag{6.18}$$

$$id_d^{2,1} = 0 \tag{6.19}$$

Constraint (6.2) denotes that every method must be assigned to exactly one carousel. Note that this does not include ISE and LIH methods, which are handled separately because both are assigned to all biochemical analyzers by default. Constraint (6.3) assures that no pair of interfering methods, which are explained in Section 3.1, is assigned to the same carousel to prevent chemical interference. A special constraint is applied to two methods. The two methods are BIL and BILK, which the AU5811's inner carousel cannot perform, and so lead to constraints (6.4).

Constraint pairs (6.6) decide whether sample's LIH methods are performed by the AU5811 or by the DxC 700 AU analyzer. The rule enforced by constraints (6.6) is that if a sample visits the DxC 700 AU analyzer, then the LIH methods are performed there. Otherwise, the LIH methods are performed by the AU5811 analyzer. However, when the AU5811 performs the LIH methods, it is necessary to decide which of the two AU5811's carousels will perform them. Constraint (6.5) ensures that when AU5811 performs sample's LIH methods, they are performed by one and only one of its two carousels.

Constraints (6.7) and (6.8) calculate the number of aliquots collected by the AU5811's carousels. The last two constraints limit the number of samples that visit both analyzers. A high amount of samples visiting both analyzers may increase their turn-around times. Equation (6.9) determines whether or not a particular sample visits both analyzers, and Equation (6.10) limits the daily amount. The extra time caused by visiting both analyzers is not meaningful when samples request immunoassays with long incubation times, so these samples are excluded from the daily limit.

6.2 ILP Model Revision

This section describes and explains two changes to Březina's model. Both changes relate to the function of the DxC 700 AU analyzer and are described in sections 6.2.1 and 6.2.2. The new mathematical representation is presented in Section 6.2.3.

6.2.1 ISE on DxC 700 AU

One shortcoming of Březina's model is the handling of ISE methods by the DxC 700 AU analyzer. His model does not compute the number of ISE methods performed on the DxC analyzer; instead, the amount is approximated by a constant. However, it is possible to determine which of the two biochemical analyzers performs the ISE methods for each sample. This can then be used to accurately calculate the number of ISE methods performed on the DxC analyzer.

Since the AU5811 analyzer contains a separate carousel reserved only for ISE methods, ISE is performed primarily by the AU5811. This means that whenever the AU5811 executes at least one method of the sample, the ISE is also performed there. However, the DxC 700 AU performs the sample's ISE methods when no biochemical method of the sample is performed by the AU5811. The rule is used in the daily operation of the modeled laboratory.

Let's define a new binary decision variable $m_{d,i}$ which determines whether or not the ISE methods of sample i from day d are performed by the DxC 700 AU. The variable is equal to one if the ISE methods of the given sample are performed by the DxC 700 AU and zero otherwise. This means that the sample's ISE methods are performed by the AU5811 when the variable is equal to zero.

We can now formulate the rule with the two following inequalities.

$$\frac{|J_i| - \sum_{J_i}\left(x_j^{1,1} + x_j^{1,2}\right)}{|J_i|} \geq m_{d,i} \qquad \forall d \in D, \forall i \in I_d \tag{6.20}$$

$$1 - \sum_{J_i}\left(x_j^{1,1} + x_j^{1,2}\right) \leq m_{d,i} \qquad \forall d \in D, \forall i \in I_d \tag{6.21}$$

When any method j of sample i is assigned to either of the AU5811's two carousels the sum $\sum_{J_i}\left(x_j^{1,1} + x_j^{1,2}\right)$ in Equation (6.20) becomes a positive non-zero number. This causes the nominator of the fraction to be smaller than the denominator, and so the binary decision variable $m_{d,i}$ is forced to be zero, i.e., perform ISE on AU5811. The fraction cannot become negative since the sum is at most equal to $|J_i|$, which happens when all sample methods are assigned to the AU5811. The decision variable's value is not constrained when no sample methods are assigned to the AU5811. The Equation (6.21) is active when no sample methods are assigned to the AU5811 analyzer. This forces the variable to be equal to one, so the ISE methods get performed by the DxC 700 AU.

Now we need to adjust the criterion function to use this newly defined variable to calculate the number of ISE methods executed by the DxC 700 AU. First, we need to realize that ISE methods are not performed for every sample, so we define a new binary parameter $q_{d,i}$. This parameter equals one when sample i from day d requests ISE methods and zero otherwise. Now the product $q_{d,i}m_{d,i}$ is equal to one whenever sample i requests ISE methods and the DxC 700 AU analyzer performs them. The product is equal to zero in all other cases. Finally, we sum the product over all samples in the given day as shown in Equation (6.22).

$$\sum_{I_d} q_{d,i}m_{d,i} \tag{6.22}$$

The DxC 700 AU's active pipetting time can now be calculated more precisely by replacing the ise_d constant in Equation (6.18) by the contents of Equation (6.22). The update version of DxC's active pipetting time is shown in Equation (6.23).

$$at_d^{2,1} = pt^2\left(\sum_J p_j s_{d,j} x_j^{2,1} + \sum_{I_d}(1 - h_{d,i}) + \sum_{I_d} q_{d,i}m_{d,i}\right) \tag{6.23}$$

6.2.2 Skipped pipetting cycles on DxC 700 AU

The DxC 700 AU analyzer in Březina's model does not suffer from any irremovable idle time as shown in Equation (6.19). However, this is not accurate. As stated in Section 3.2.6, one pipetting cycle is skipped between two consecutive samples. Since these skipped pipetting cycles cannot be prevented they should be reflected in the idle time of the DxC 700 AU analyzer.

It is important to realize that the number of skipped pipetting cycles actually depends on the workload of the DxC 700 AU. No pipetting cycles are skipped when there are no two samples visiting the analyzer one just after the other. However, when the analyzer is utilized 100 % of the time only, the cycles are skipped on every sample exchange, so the total amount of pipetting cycles skipped would be equal to one less than the number of samples that visit the DxC 700 AU analyzer. In reality, the majority of samples comes in batches due to the centrifugation process. If we knew the number of sample batches, we could compute the number of skipped pipetting cycles by Equation (6.24).

$$\#\ \textit{of skipped DxC cycles} = \#\ \textit{of samples visiting DxC} - \#\ \textit{of batches} \tag{6.24}$$

Equation (6.24) can work for both edge cases discussed above since a different number of batches can characterize both cases. One batch per sample characterizes the case of sparse utilization of the DxC 700 AU analyzer, whereas one batch in total characterizes the 100 % utilization case. However, samples from multiple large consecutive batches may merge and essentially form a single batch. In that case, the actual number of skipped pipetting cycles would be greater than the value computed by Equation (6.24). On the other hand, some batches may not even contain samples that visit the DxC analyzer, so the computed value might be skewed in the other direction when compared to the actual amount.

To compute the number of skipped pipetting cycles in Equation (6.24) we also need to know the number of samples that visit the DxC 700 AU analyzer. We would usually do this by introducing a new binary variable for each sample that decides whether or not the sample visits the DxC 700 AU analyzer. Then we would create constraints that set the variable's value to one if at least one sample method is assigned to the DxC analyzer. However, we used a different approach.

When analyzing the original model, we observed that the variables corresponding to LIH methods could be used to calculate the number of samples visiting the DxC 700 AU. Section 6.1.2 states that the variable $h_{d,i}$ is equal to one if and only if the AU5811 performs LIH methods of sample i. The rule for LIH methods which is enforced by constraints (6.6) states that the DxC 700 AU analyzer always performs the LIH methods if the sample visits the analyzer. The AU5811 performs the LIH methods only when the sample does not visit the DxC analyzer. Additionally, LIH methods are performed for every sample [19]. This means that the variable $h_{d,i}$ is equal to zero if and only if sample i visits the DxC 700 AU analyzer. So we can use this finding with Equation (6.25) to calculate the number of samples visiting the DxC analyzer.

$$\#\ \textit{of samples visiting DxC} = \sum_{I_d} (1 - h_{d,i}) \tag{6.25}$$

The DxC 700 AU's idle time can now be calculated as the product of the number of skipped pipetting cycles and the duration of one pipetting cycle. A new constant v_d is introduced. This parameter represents the number of sample batches for a given day d. Its value can be calculated from the laboratory data. Equation (6.26) shows the newly derived idle time of the DxC 700 AU.

$$id_d^{2,1} = pt^2 \left(\sum_{I_d} (1 - h_{d,i}) - v_d \right) \tag{6.26}$$

6.2.3 Revised ILP Representation

By combining the results from Section 6.2.1 and Section 6.2.2 we get a more precise expression of the DxC 700 AU's workload. Note that a majority part of the idle time from Equation (6.26) can be combined with the workload caused by the pipetting of LIH methods. The resulting workload of the DxC 700 AU analyzer is shown in Equation (6.27).

$$w_d^{1,2} = at_d^{2,1} + it_d^{2,1} = pt^2 \left(\sum_J p_j s_{d,j} x_j^{2,1} + 2 \sum_{I_d} (1 - h_{d,i}) - v_d + \sum_{I_d} q_{d,i} m_{d,i} \right) \tag{6.27}$$

The changes are reflected in the model's criterion function and in the addition of constraints (6.33). The revised ILP model is shown below.

$$\begin{aligned}
\min \sum_D \max \Bigg\{ & pt^1 \left(\sum_J p_j s_{d,j} x_j^{1,1} + \sum_{I_d} h_{d,i}^{1,1} \right) + pt^1 \sum_{I_d} \max\{0, n_{d,i}^{1,2} - n_{d,i}^{1,1}\}, \\
& pt^1 \left(\sum_J p_j s_{d,j} x_j^{1,2} + \sum_{I_d} h_{d,i}^{1,2} \right) + pt^1 \sum_{I_d} \max\{0, n_{d,i}^{1,1} - n_{d,i}^{1,2}\}, \\
& pt^2 \left(\sum_J p_j s_{d,j} x_j^{2,1} + 2 \sum_{I_d} (1 - h_{d,i}) - v_d + \sum_{I_d} q_{d,i} m_{d,i} \right) \Bigg\}
\end{aligned}$$

$$\sum_{A}\sum_{B^a} x_j^{a,b} = 1 \qquad \forall j \in J \tag{6.28}$$

$$x_j^{a,b} + x_{j'}^{a,b} \leq 1 \qquad \forall \left(j, j'\right) \in F, \forall a \in A, \forall b \in B^a \tag{6.29}$$

$$x_{BIL}^{1,1} = 0; x_{BILK}^{1,1} = 0 \tag{6.30}$$

$$h_{d,i}^{1,1} + h_{d,i}^{1,2} = h_{d,i} \qquad \forall d \in D, \forall i \in I_d \tag{6.31}$$

$$\frac{|J_i| - \sum_{J_i} x_j^{2,1}}{|J_i|} \geq h_{d,i} \geq 1 - \sum_{J_i} x_j^{2,1} \qquad \forall d \in D, \forall i \in I_d \tag{6.32}$$

$$\frac{|J_i| - \sum_{J_i} \left(x_j^{1,1} + x_j^{1,2}\right)}{|J_i|} \geq m_{d,i} \geq 1 - \sum_{J_i} \left(x_j^{1,1} + x_j^{1,2}\right) \qquad \forall d \in D, \forall i \in I_d \tag{6.33}$$

$$n_{d,i}^{1,2} = \sum_{J_i} p_j x_j^{1,2} + h_{d,i}^{1,2} \qquad \forall d \in D, \forall i \in I_d \tag{6.34}$$

$$n_{d,i}^{1,1} = \sum_{J_i} p_j x_j^{1,1} + h_{d,i}^{1,1} \qquad \forall d \in D, \forall i \in I_d \tag{6.35}$$

$$t_{d,i} \geq \sum_{B^a} x_j^{a,b} + \sum_{B^{a'}} x_{j'}^{a',b} - 1 \qquad \forall d \in D, \forall i \in I_d, \quad \forall j, j' \in J_i : j \neq j', \forall a, a' \in A : a < a' \tag{6.36}$$

$$\alpha \geq \frac{1}{|D|} \sum_{I_d} \left(1 - st_{d,i}\right) t_{d,i} \tag{6.37}$$

Sets:

- D - The set of all considered days.
- I - The set of all samples. I_d denotes a set of samples processed in day d.
- J - The set of all biochemical methods except LIH and ISE. J_i represents regular methods requested by sample i.
- A - The set of biochemical analyzers (AU and DXC).
- B^a - The set of carousels of analyzer a.
- F - The set of all pairs of interfering methods.

Parameters:

- pt^a - The pipetting operation length of analyzer a.
- p_j - The number of aliquot collections needed to process method j.
- $s_{d,j}$ - The total quantity of samples requesting method j in day d.

- $st_{d,i}$ - Binary parameter equal to one if sample i requests long immunological method. If so, the transport of such a sample is not considered.
- α - Upper bound on the average daily amount of occurred transports.
- $q_{d,i}$ - Binary parameter equal to one if sample i from day d requests ISE methods.
- v_d - The number of different sample batches processed by the DxC 700 AU analyzer in day d.

Decision variables:

- $x_j^{a,b}$ - A binary variable equal to one if method j is assigned to a carousel b of analyzer a. Note that AU has two carousels, whereas DXC has only one.
- $m_{d,i}$ - A binary variable determining whether the ISE methods of sample i from day d will be performed on the DXC analyzer or not. The variable is equal to one if ISE for this sample are performed by the DXC and zero otherwise.
- $h_{d,i}$ - A binary variable determining whether the LIH methods of sample i from day d will be performed on the AU analyzer or not. The variable is equal to one if LIH are performed by the AU and zero otherwise. Note when LIH methods are performed for every sample and variable $h_{d,i}$ equals zero that also means that the sample needs to visit the DXC analyzer.
- $h_{d,i}^{a,b}$ - If AU performs the LIH methods, it is necessary to decide on which carousel. This binary variable is equal to 1 if LIH of sample i are performed by carousel b of the AU analyzer.
- $n_{d,i}^{a,b}$ - The number of aliquots collected by the carousel b of analyzer a in order to process sample i. This variable determines the number of odd methods.
- $t_{d,i}$ - A binary variable. Equal to one, if sample i from day d needs to visit both biochemical analyzers.

Chapter 7

Optimization results

This section presents experimental results of laboratory throughput optimization according to the revised model described in Section 6.2.3. The experimental setup is explained in Section 7.1 with the results presented in Section 7.2.

7.1 Experimental setup

The same experiments that were performed by Březina [19] to evaluate his model were repeated with the new model to allow a direct comparison between them. The same sets of laboratory operational data were used. The optimization set consists of 5 days of laboratory operational data and the data from all other days forms the evaluation set. Two experiments were performed to find the optimal method assignments to carousels. The experiments differ in the use of the side criterion that limits the average number of sample transports between the two modeled biochemical analyzers. The found method assignments are compared with assignments found by the same experiments with the original model. The results show the differences in the number of sample transports, occurrences of odd methods, biochemistry throughput, and analyzer utilization. The analyzer utilization is computed from the analyzer's active time, which was introduced in Section 6.1.2. Note that this is different from the total analyzer workload which also includes any analyzer idle time.

7.2 Experimental results

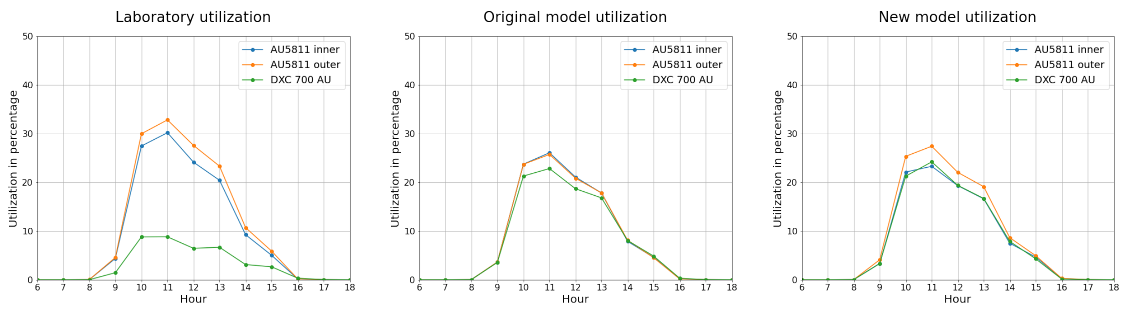
The first experiment does not introduce any limit on the daily average number of samples transported, so the optimization is solely focused on minimizing the maximal carousel workload. Whereas the second experiment limits the number of daily sample transports to match the real laboratory. The matching method assignment was found with the limiting parameter set to $\alpha = 35$. The summarized results are presented in Table 7.1 followed by an analysis of the first experiment in Section 7.2.1 and the second experiment in Section 7.2.2.

Model	Average Throughput [Samples per Hour]	Odd Method Average [Occurrences per Day]	Average Transports [Samples per Day]
Laboratory	1919.24	425.86	63.05
Original	2113.09	197.10	174.29
New	2144.81	357.38	163.33
Original (limited transports)	2018.87	285.05	63.95
New (limited transports)	1997.20	281.90	62.05

Table 7.1: ILP result overview of performed experiments.

7.2.1 Experiment without transport limits

Graphs in Figure 7.1 depict average carousel utilization for both analyzers. Each point represents an average for a one-hour-long interval starting at that hour. Figures 7.1b and 7.1c show that the distribution of carousel active time is balanced in both cases contrary to the actual method assignment used in the laboratory as shown in Figure 7.1a. As a result, the biochemical throughput is increased by 11.7% for the method assignment found with the new model in comparison to the actual assignment used in the laboratory. The original model's methods assignment leads to a smaller throughput increase of 10.1%.

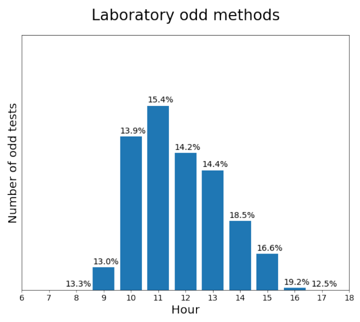


(a) Laboratory (b) Original model (c) New model

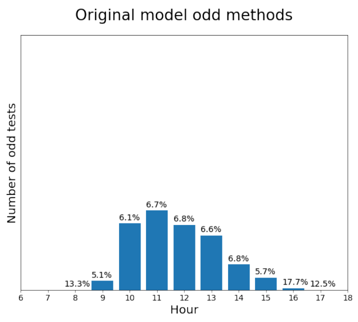
Figure 7.1: Comparison of carousel utilization without transport limits

Note that the values presented for laboratory throughput in Table 7.1 were calculated according to the new model's criterion function. When the original model's criterion is used for the evaluation, the reported throughput increase of the old model is over 30%. However it does not account for any idle time on the DxC 700 AU and does not compute but only approximates the active time caused by ISE methods. Based on the method assignment found by the new model, the laboratory would still benefit from using the new method assignment. Although the actual impact may be smaller than previously expected.

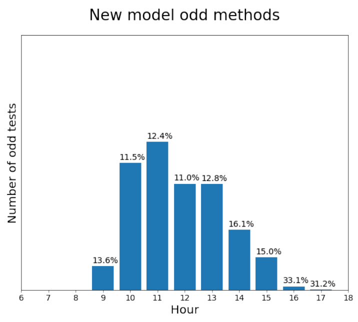
The method assignment found by the new model reduces the average number of odd method occurrences by 16.1% compared to the laboratory. However, the reduction is much higher for the original model's method assignment. Figure Figure 7.2 depicts the average number of odd method occurrences during the day, where the percentage above the bars describes the ratio between the number of odd pipetting events and the total number of requested tests during the corresponding hour.



(a) Laboratory



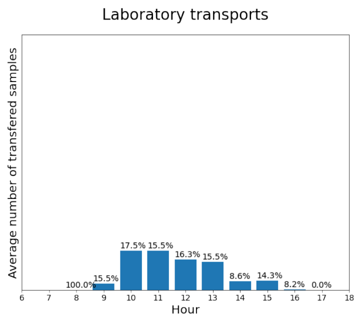
(b) Original model



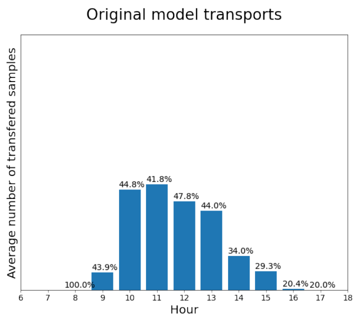
(c) New model

Figure 7.2: Comparison of average odd method occurrences without transport limits

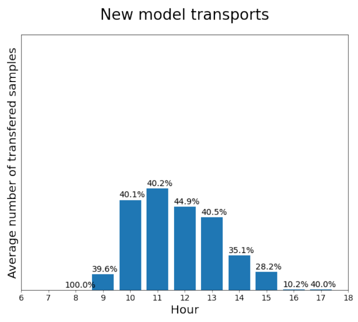
Odd method occurrences may increase TAT for some samples. The same is true for sample transport between analyzers. An average of 163 samples per day or around 40% of samples are transported and visit both analyzers when the method assignment found by the new model is used. This is a significant increase compared to the real laboratory, where only 63 samples per day are transported on average. The higher number of transports is expected since the carousel workload cannot be effectively balanced without sample transports. Nevertheless, the total increase in sample transfers is smaller than that caused by the assignment found with the original model. Figure 7.3 compares the hourly average number of transports.



(a) Laboratory



(b) Original model



(c) New model

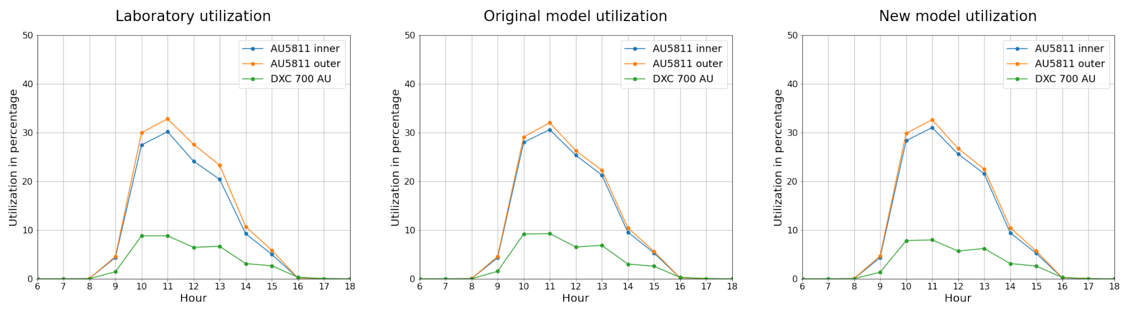
Figure 7.3: Comparison of average daily sample transports without transport limits

7.2.2 Experiment with transport limits

We have learned in Section 7.2.1 that the method assignment obtained from the new model causes a significant increase in the number of transported samples. We would like to know if it is also possible to increase laboratory throughput while keeping the number of transports the same. Thankfully a parameter introduced in the original model can be used to limit the sample transfers. We ran the optimization multiple times with different values of the parameter until we found a method assignment that caused the same number of sample transports as the assignment used by the laboratory. The corresponding assignment was found by setting the parameter equal to $\alpha = 35$.

Based on the analyzer utilization shown in Figure 7.4, we can say that the method assignment used in the real laboratory leads to similar active utilization of the carousels when compared to the assignment found by the new model given the constraints on the average number of transported samples.

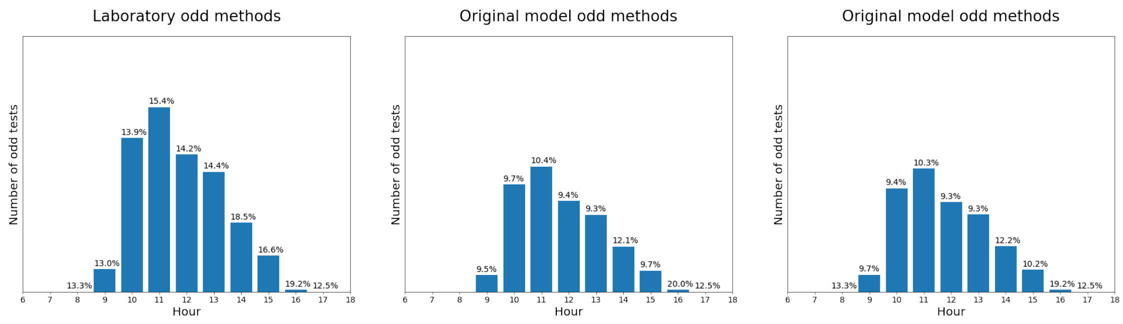
7.2.3 Sample transport limit



(a) Laboratory (b) Original model (c) New model

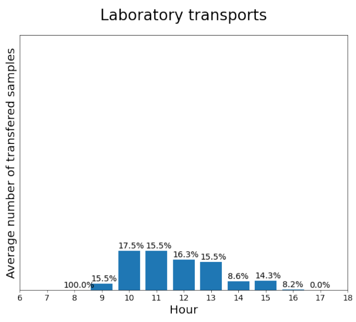
Figure 7.4: Comparison of carousel utilization with limited transports

The throughput difference between the method assignment found by the original and the assignment found by the new model is minor. However, both solutions decrease the average number of odd method occurrences by about 33%, resulting in increased throughput in comparison to the assignment used by the laboratory. Figure 7.6 clearly shows that both models work with almost the same average number of daily transports as the laboratory.

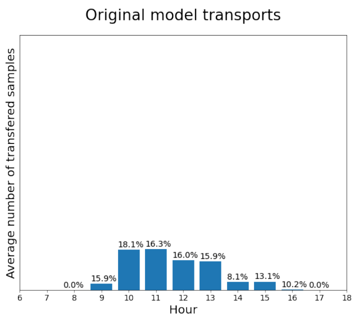


(a) Laboratory (b) Original model (c) New model

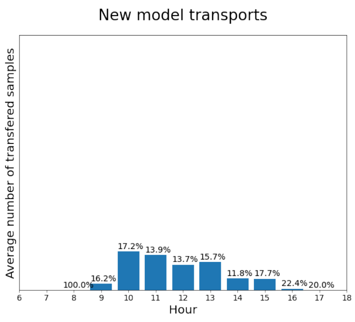
Figure 7.5: Comparison of average odd method occurrences with limited transports



(a) Laboratory



(b) Original model



(c) New model

Figure 7.6: Comparison of average daily sample transports with transport limits

Chapter 8

Conclusion

This thesis mainly focused on gaining an insight into the automation of clinical laboratories. The principles of clinical testing, modular laboratory automation systems, and laboratory processes were introduced. Consequently, the function of DxA 5000 and DxA 5000 Fit laboratory automation systems were described.

The field displayed an interest in software solutions as tools for simulation and analysis of laboratory designs and processes. A simulator design was proposed to create a piece of software for simulation and analysis of the DxA 5000 Fit laboratory automation system.

The designed simulator has been implemented and thoroughly tested. It enables fast and accurate simulations of custom configurations of the laboratory automation system. The simulator shows errors within the range expected by the users. In conclusion, the simulator is and will be used to help with laboratory designs by verifying the suitability of proposed technological solutions. The simulator should make this process faster, cheaper, and easier for the users.

This thesis also visited the domain of laboratory process optimization. An existing integer linear programming model that is used to optimize the throughput of biochemical analyzers was introduced. Two changes were proposed to more accurately model the workload of a biochemical analyzer DxC 700 AU.

The proposed changes were implemented, and the revised model was compared with the original. The results confirm that the model can be used to find method assignment that significantly improves the laboratory throughput compared to the method assignment proposed by experienced technicians. However, the increase in the laboratory throughput should be smaller than previously expected. In conclusion, laboratory designers will always benefit from the optimization since the number of odd method occurrences cannot be optimized manually.

Bibliography

[1] V. Matheeussen, V. M. Corman, O. Donoso Mantke, E. McCulloch, C. Lammens, H. Goossens, D. Niemeyer, P. S. Wallace, P. Klapper, H. G. Niesters, C. Drosten, M. Ieven, on behalf of the RECOVER project, and collaborating networks, "International external quality assessment for sars-cov-2 molecular detection and survey on clinical laboratory preparedness during the covid-19 pandemic, april/may 2020," *Eurosurveillance*, vol. 25, no. 27, 2020. [Online]. Available: https://www.eurosurveillance.org/content/10.2807/1560-7917.ES.2020.25.27.2001223

[2] M. Krasowski, J. Kulhavy, C. Morris, D. Nelson, S. Teul, D. Voss, D. Aman, T. Bebber, J. Blau, I. Clark, C. Crone, S. Davis, D. Drees, and J. Fahnle, "Autoverification in a core clinical chemistry laboratory at an academic medical center," *Journal of Pathology Informatics*, vol. 5, no. 1, p. 13, 2014. [Online]. Available: https://doi.org/10.4103/2153-3539.129450

[3] K. Culbreath, H. Piwonka, J. Korver, M. Noorbakhsh, and E. McElvania, "Benefits derived from full laboratory automation in microbiology: a tale of four laboratories," *Journal of Clinical Microbiology*, vol. 59, no. 3, pp. e01 969–20, 2021. [Online]. Available: https://journals.asm.org/doi/abs/10.1128/JCM.01969-20

[4] J. R. Genzen, C.-A. D. Burnham, R. A. Felder, C. D. Hawker, G. Lippi, and O. M. Peck Palmer, "Challenges and Opportunities in Implementing Total Laboratory Automation," *Clinical Chemistry*, vol. 64, no. 2, pp. 259–264, 02 2018. [Online]. Available: https://doi.org/10.1373/clinchem.2017.274068

[5] E. VanDerHorn and S. Mahadevan, "Digital twin: Generalization, characterization and implementation," *Decision Support Systems*, vol. 145, p. 113524, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167923621000348

[6] "Workflow automation system dxa 5000 fit," Beckman Coulter. [Online]. Available: https://www.beckmancoulter.com/products/automation/workflow-automation-system-dxa-5000-fit

[7] B. Breil, F. Fritz, V. Thiemann, and M. Dugas, "Mapping turnaround times (TAT) to a generic timeline: A systematic review of TAT definitions in clinical domains," *BMC Medical Informatics and Decision Making*, vol. 11, no. 1, May 2011. [Online]. Available: https://doi.org/10.1186/1472-6947-11-34

[8] E. R. Tsai, A. N. Tintu, D. Demirtas, R. J. Boucherie, R. de Jonge, and Y. B. de Rijke, "A critical review of laboratory performance indicators," *Critical Reviews*

in Clinical Laboratory Sciences, vol. 56, no. 7, pp. 458–471, Aug. 2019. [Online]. Available: https://doi.org/10.1080/10408363.2019.1641789

[9] M. Miler, N. N. Gabaj, L. Dukic, and A.-M. Simundic, "Key performance indicators to measure improvement after implementation of total laboratory automation abbott accelerator a3600," *Journal of Medical Systems*, vol. 42, no. 2, Dec. 2017. [Online]. Available: https://doi.org/10.1007/s10916-017-0878-1

[10] G. Lippi and G. D. Rin, "Advantages and limitations of total laboratory automation: a personal overview," *Clinical Chemistry and Laboratory Medicine (CCLM)*, vol. 57, no. 6, pp. 802–811, 2019. [Online]. Available: https://doi.org/10.1515/cclm-2018-1323

[11] R. S. Seaberg, R. O. Stallone, and B. E. Statland, "The Role of Total Laboratory Automation in a Consolidated Laboratory Network," *Clinical Chemistry*, vol. 46, no. 5, pp. 751–756, 05 2000. [Online]. Available: https://doi.org/10.1093/clinchem/46.5.751

[12] S. Angeletti, M. D. Cesaris, J. G. Hart, M. Urbano, M. A. Vitali, F. Fragliasso, and G. Dicuonzo, "Laboratory automation and intra-laboratory turnaround time: Experience at the university hospital campus bio-medico of rome," *Journal of Laboratory Automation*, vol. 20, no. 6, pp. 652–658, 2015, pMID: 25609253. [Online]. Available: https://doi.org/10.1177/2211068214566458

[13] H.-Y. E. Yu, H. Lanzoni, T. Steffen, W. Derr, K. Cannon, J. Contreras, and J. E. Olson, "Improving Laboratory Processes with Total Laboratory Automation," *Laboratory Medicine*, vol. 50, no. 1, pp. 96–102, 06 2018. [Online]. Available: https://doi.org/10.1093/labmed/lmy031

[14] A. Dolci, D. Giavarina, S. Pasqualetti, D. Szőke, and M. Panteghini, "Total laboratory automation: Do stat tests still matter?" *Clinical Biochemistry*, vol. 50, no. 10, pp. 605–611, 2017, improving laboratory Performance Through Quality Indicators. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0009912016307378

[15] W. Vogt, S. Braun, F. Hanssmann, F. Liebl, G. Berchtold, H. Blaschke, M. Eckert, G. Hoffmann, and S. Klose, "Realistic modeling of clinical laboratory operation by computer simulation," *Clinical chemistry*, vol. 40, pp. 922–8, 07 1994.

[16] T. Yang, T.-K. Wang, V. C. Li, and C.-L. Su, "The optimization of total laboratory automation by simulation of a pull-strategy," *Journal of Medical Systems*, vol. 39, no. 1, Dec. 2014. [Online]. Available: https://doi.org/10.1007/s10916-014-0162-6

[17] D. Kadı, Y. Kuvvetli, and S. Çolak, "Performance analysis of a university hospital blood laboratory via discrete event simulation," *SIMULATION*, vol. 92, no. 5, pp. 473–484, 2016. [Online]. Available: https://doi.org/10.1177/0037549716643167

[18] "Workflow automation system dxa 5000," Beckman Coulter. [Online]. Available: https://www.beckmancoulter.com/en/products/automation/dxa-5000-lab-automation-system

[19] P. Březina, "Data analysis of laboratory workflow and data-driven laboratory process optimization," Master's thesis, Czech Technical University in Prague, Czech Republic, 2021.

[20] P. M. V. Raja and A. R. Barron, “Ion Selective Electrode Analysis,” 21 Mar. 2021, [Online; accessed 2022-04-18].

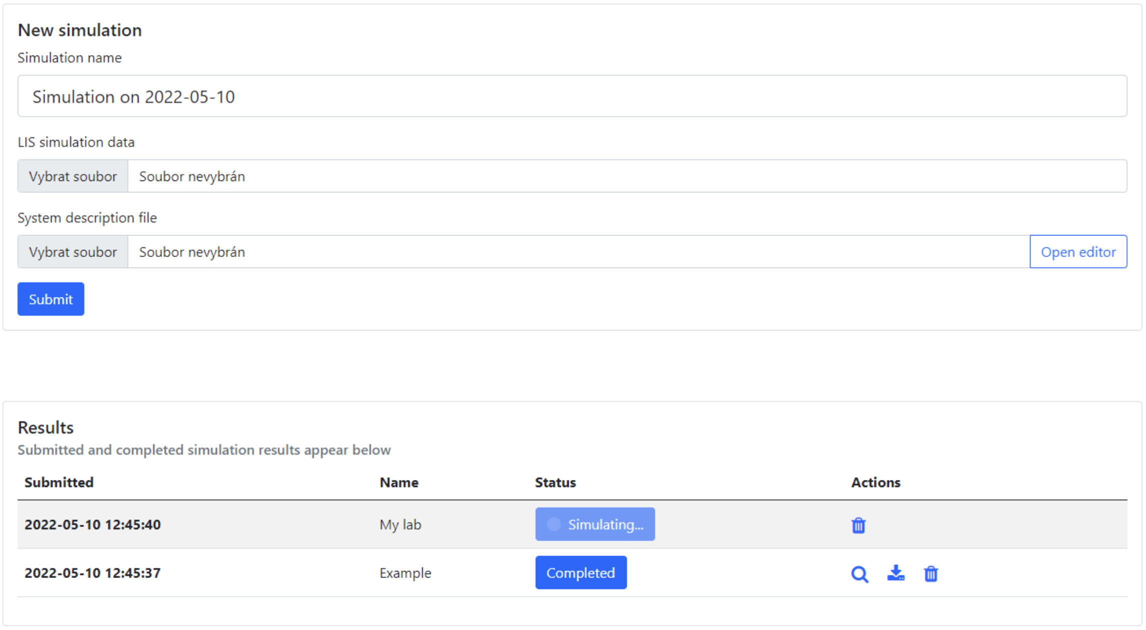
[21] “DxC 700 AU IFU Manual,” Beckman Coulter, 16 Apr. 2021. [Online]. Available: https://www.beckmancoulter.com/download/file/wsr-231081/B71495AF?type=pdf

[22] “AU5800 Laboratory Automation Connecting Kit,” Beckman Coulter, 22 Apr. 2013. [Online]. Available: https://www.beckmancoulter.com/download/file/wsr-112805/B08501AB?type=pdf

[23] K. Gevančiak, “Optimization of Samples Processing in Medical Laboratories: Problem Analysis Based on Data,” Master’s thesis, Czech Technical University in Prague, Czech Republic, 2019.

User interface examples

The attached figures depict user interface of the simulator and examples of graphical simulation results for an artificial system configuration.



New simulation

Simulation name

Simulation on 2022-05-10

LIS simulation data

Vybrat soubor | Soubor nevybrán

System description file

Vybrat soubor | Soubor nevybrán | Open editor

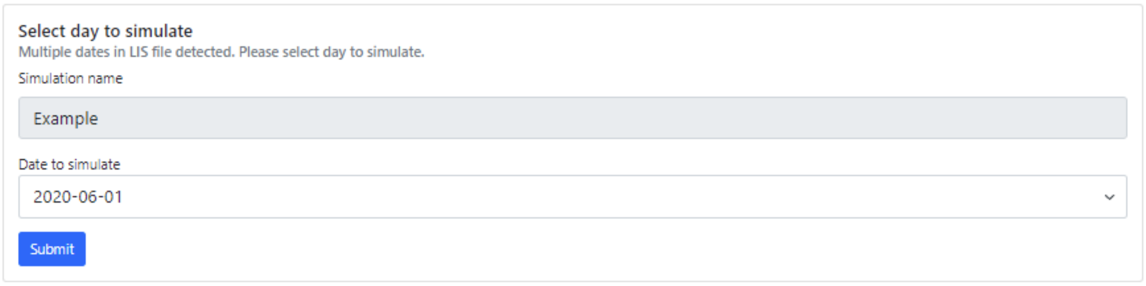
Submit

Results

Submitted and completed simulation results appear below

Submitted	Name	Status	Actions
2022-05-10 12:45:40	My lab	Simulating...	
2022-05-10 12:45:37	Example	Completed	

Figure 1: Main - new simulations, status of past simulations



Select day to simulate

Multiple dates in LIS file detected. Please select day to simulate.

Simulation name

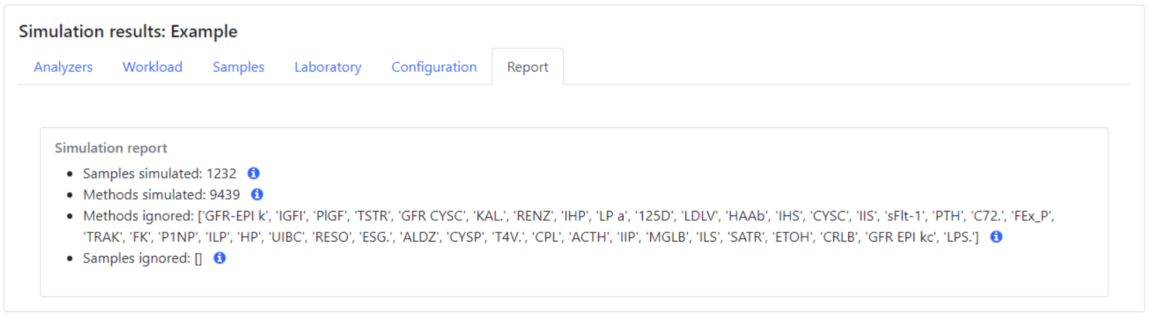
Example

Date to simulate

2020-06-01

Submit

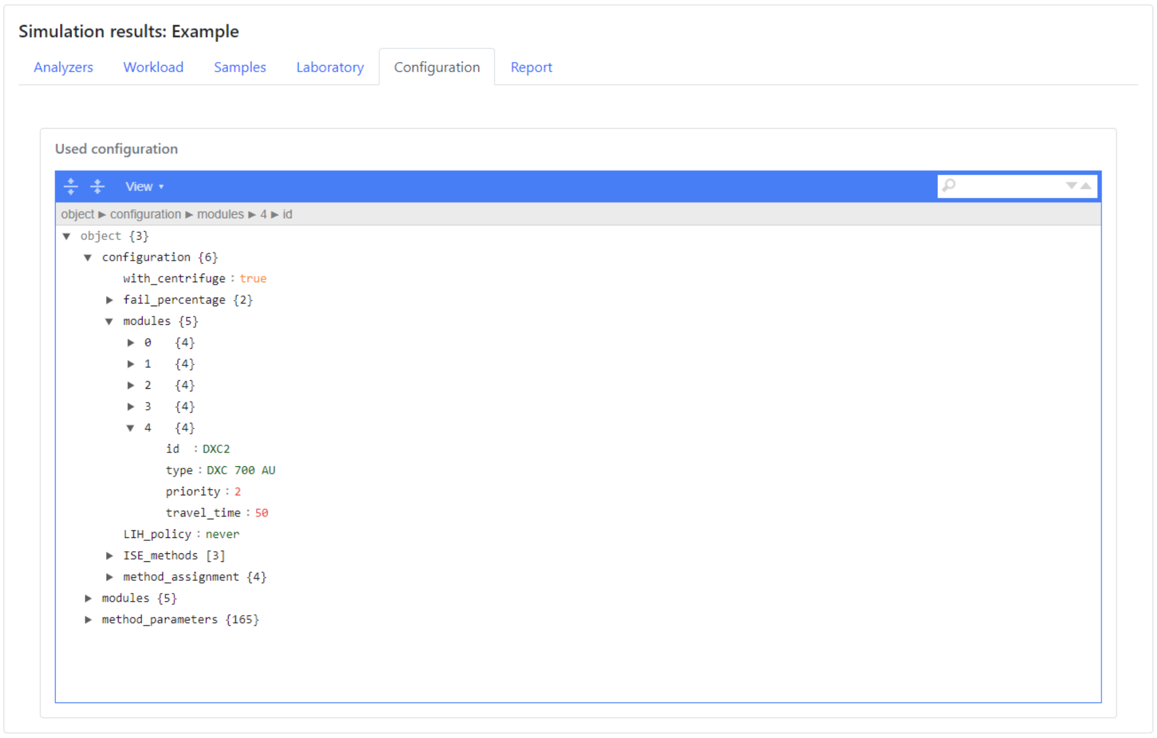
Figure 2: New simulation - day selection



Simulation report

- Samples simulated: 1232
- Methods simulated: 9439
- Methods ignored: ['GFR-EPI k', 'IGFI', 'PlGF', 'TSTR', 'GFR CYSC', 'KAL.', 'RENZ', 'IHP', 'LP a', '125D', 'LDLV', 'HAAb', 'IHS', 'CYSC', 'IIS', 'sFlt-1', 'PTH', 'C72.', 'FEx_P', 'TRAK', 'FK', 'P1NP', 'ILP', 'HP', 'UIBC', 'RESO', 'ESG.', 'ALDZ', 'CYSP', 'T4V.', 'CPL', 'ACTH', 'IIP', 'MGLB', 'ILS', 'SATR', 'ETOH', 'CRLB', 'GFR EPI kc', 'LPS.']
- Samples ignored: []

Figure 3: Report - simulation report


Simulation results: Example
Analyzers
Workload
Samples
Laboratory
Configuration
Report
View
object ▶ configuration ▶ modules ▶ 4 ▶ id

Used configuration

```
▼ object {3}
  ▼ configuration {6}
      with_centrifuge : true
    ▶ fail_percentage {2}
    ▼ modules {5}
      ▶ 0   {4}
      ▶ 1   {4}
      ▶ 2   {4}
      ▶ 3   {4}
      ▼ 4   {4}
          id   : DXC2
          type : DXC 700 AU
          priority : 2
          travel_time : 50
      LIH_policy : never
    ▶ ISE_methods [3]
    ▶ method_assignment {4}
  ▶ modules {5}
  ▶ method_parameters {165}
```

Figure 4: Configuration - configuration of the simulated system

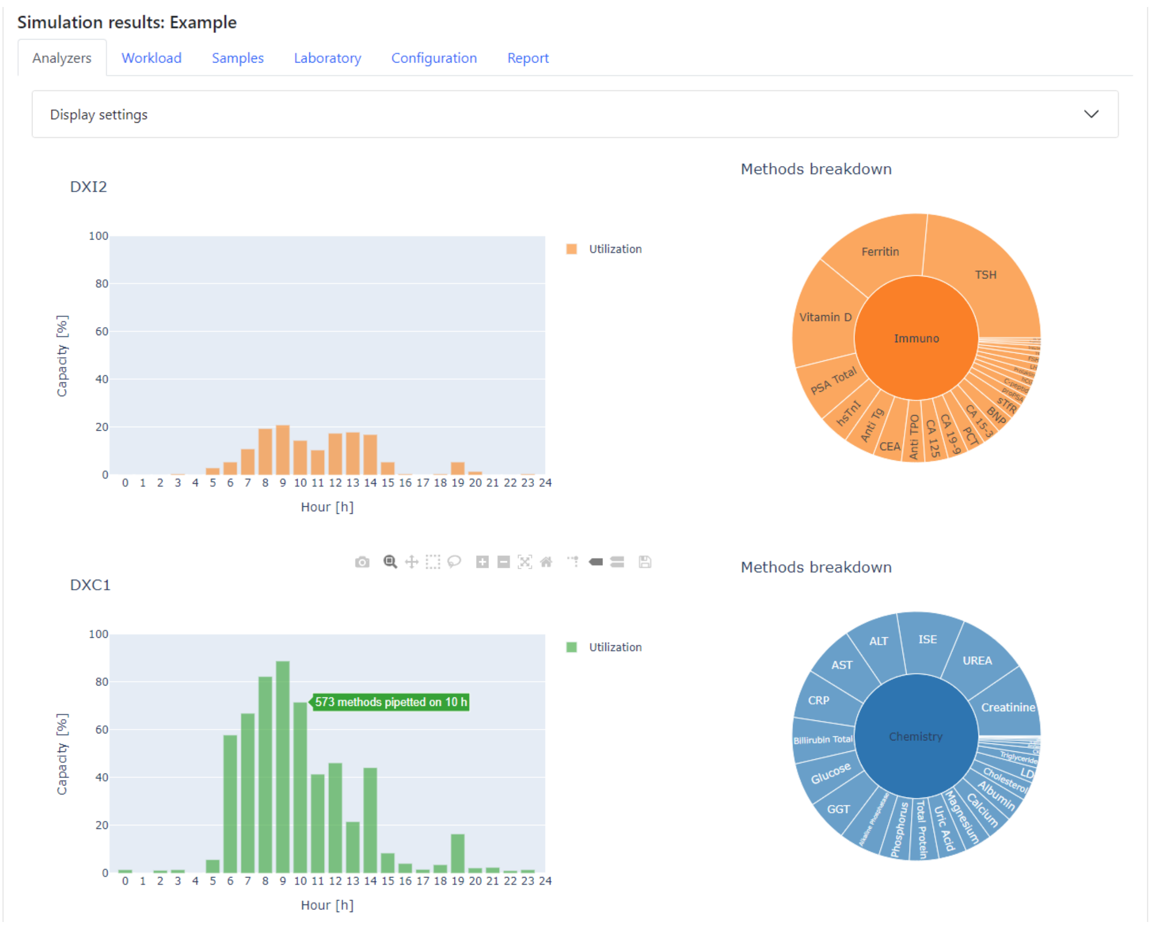


Figure 5: Analyzers - analyzer workload and method breakdown

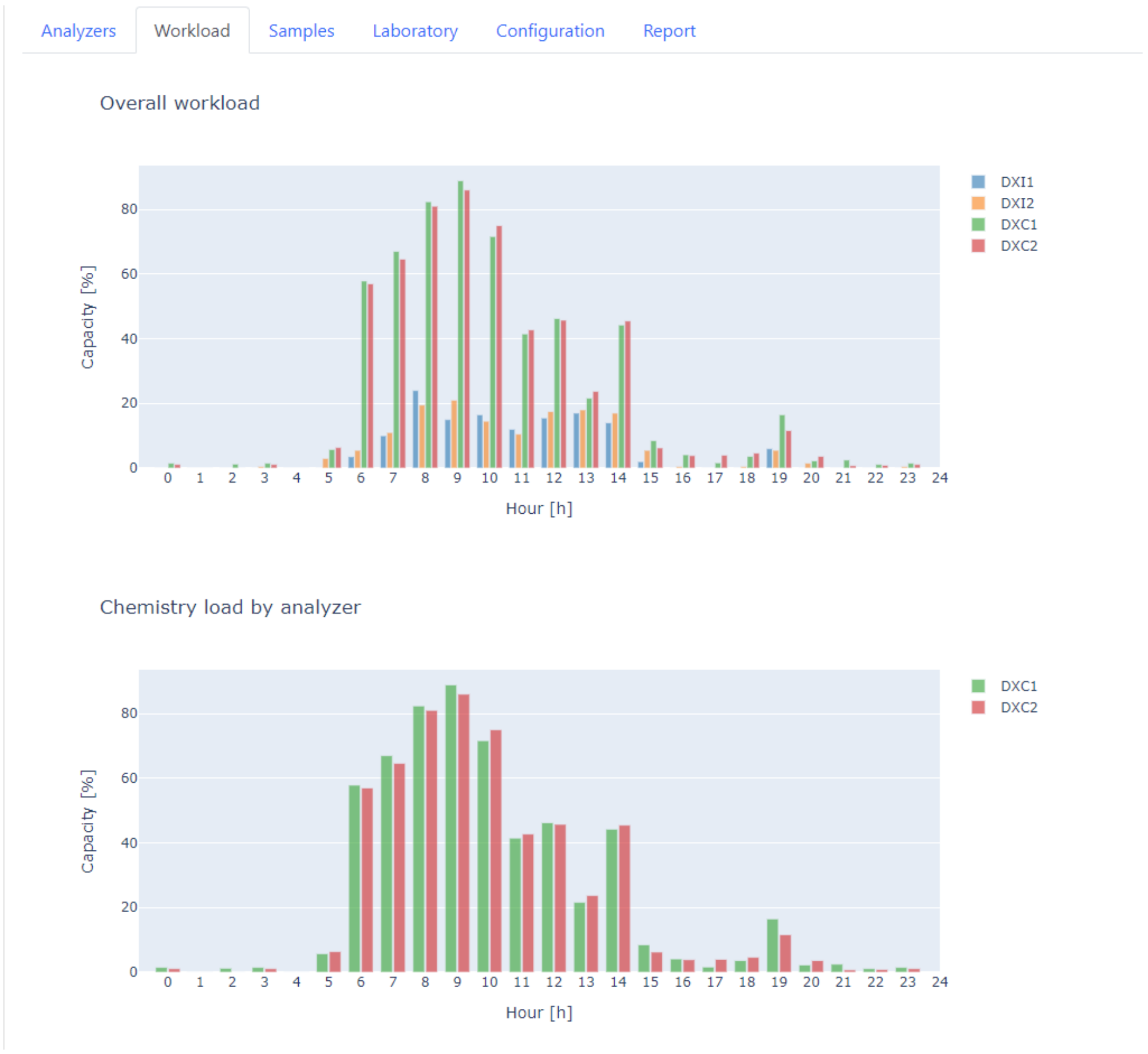


Figure 6: Workload - analyzer workload comparison

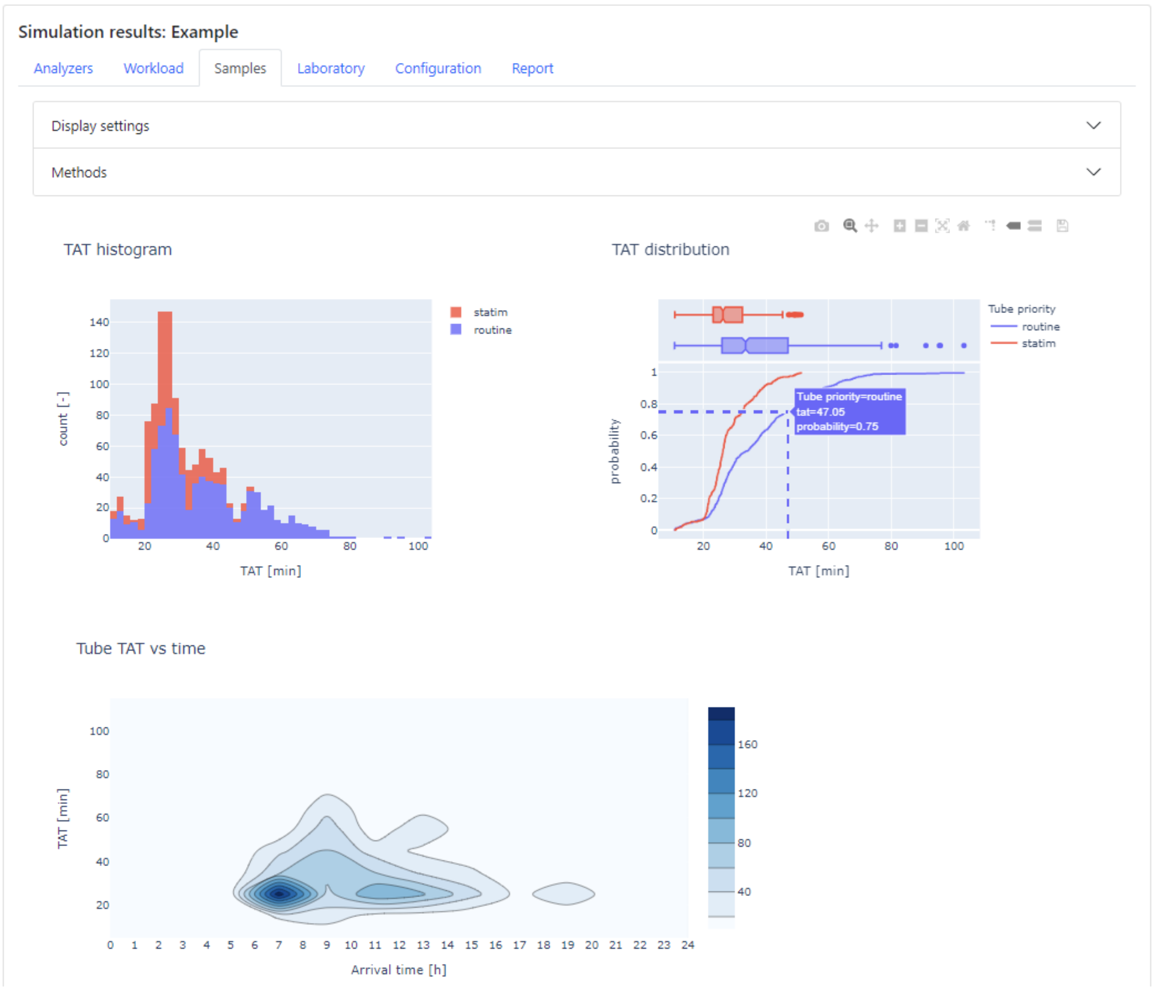


Figure 7: Samples - sample TAT histogram and distribution graph, sample TAT vs. arrival time

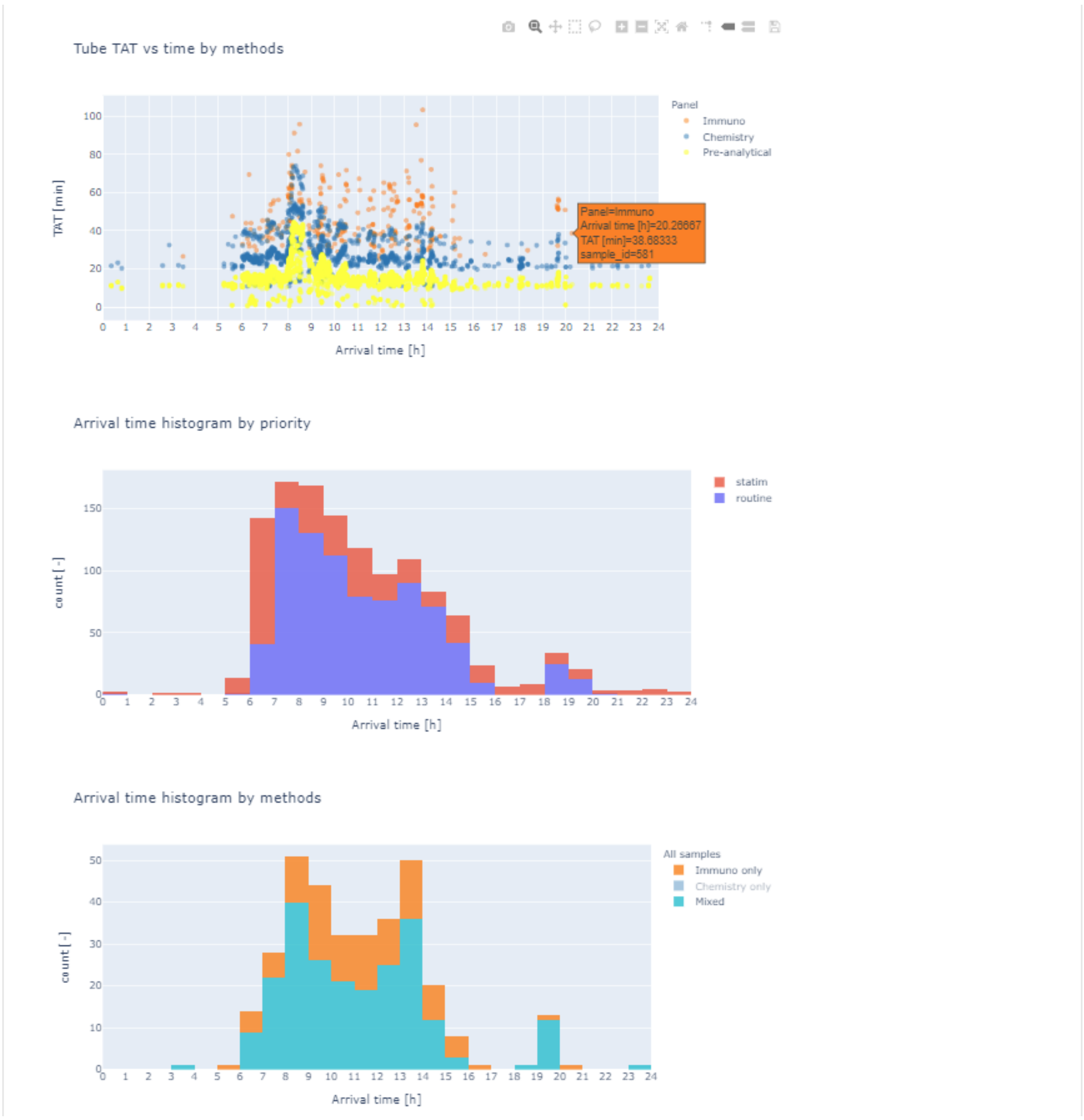


Figure 8: Samples - pre-analytical and method result time, arrival time histograms by priority and method types

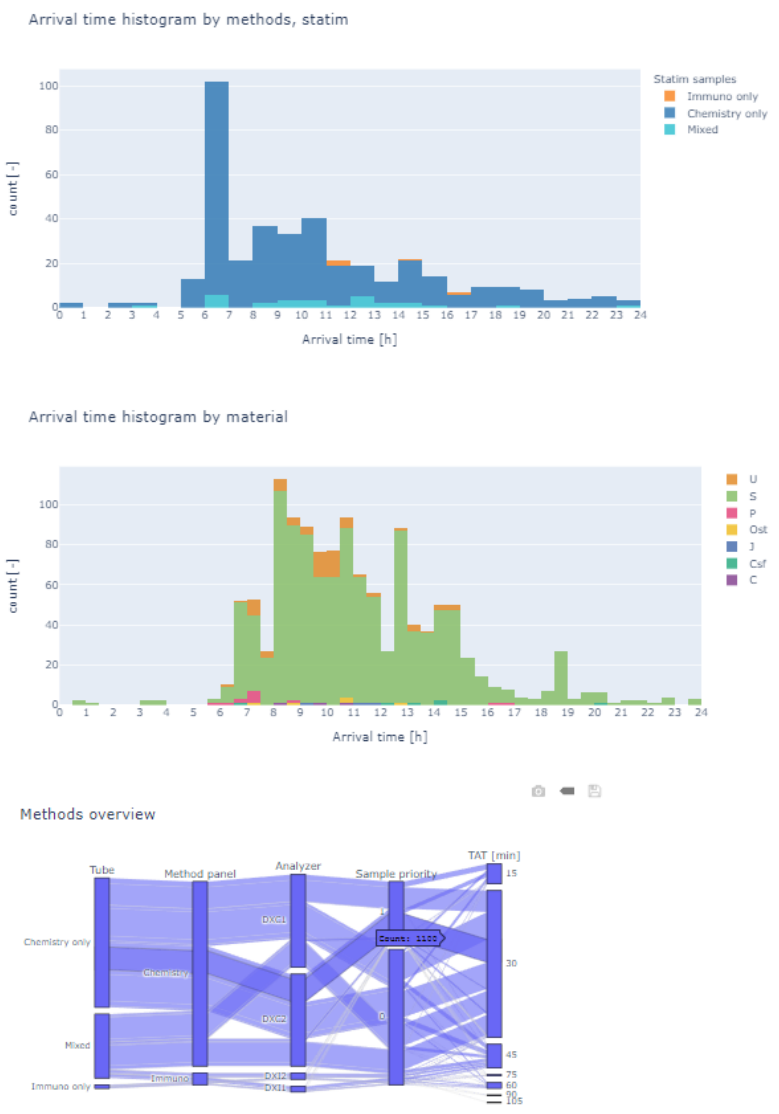


Figure 9: Samples - arrival time histograms by method type + priority and sample type, 2D visualisation of 5D data