Delft University of Technology

Aircraft MRO, order fulfillment, multi- echelon, closed-loop supply chain, repairable components, availability contract, discrete time step simulation

Master thesis research project

Simulating order fulfillment for aircraft line replaceable unit strategy
A case study at KLM Component Services

Author:
H. J. M. Kalf (4261518)

Dr. J. Rezaei (chair)
Ir. M.B. Duinkerken (supervisor)
Dr. M.Y. Maknoon (supervisor)
P. van Voorbergen (supervisor, KLM)
E. Cornelisse (KLM)

February 24, 2021

Summary

Service levels are not reached

KLM Engineering and Maintenance Component services is a maintenance, repair, and overhaul organization part of Air France–KLM Group. Wherein Component services offers aircraft components on request to reduce capital employed at the the airline. When components are not on time, an aircraft could lose airworthiness and give the MRO a fine for not reaching the agreed service level.

The main problem is aircraft line replaceable unit (component type) maintenance, repair, and overhaul (organization) struggling to reach agreed service levels in availability contracts. An availability contract states that an airline pays per flight hours to exchange a failed component at any moment with a functioning component for 95% of the times within a predetermined period of days. Line replaceable units are repairable and are on an airport replaceable; the only exception is the main jet engine. Line replaceable units availability contracts are growing with 20% per year and take up 60% of the total capital in the total maintenance, repair, and overhaul.

Approach to solve the problem

This research provides new insights by performing a discrete time step simulation to increase the service level, considering strategies. The insights point out the aspects of why the service level is not reached. A discrete-time step simulation is applied because the environment changes over time and the repair TAT and removals are stochastic. The following steps are taken to find out *'what aspects make line replaceable unit maintenance, repair and overhaul not reach service levels in global commercial availability contracts?'*:

1. Search in literature for what is known about line replaceable units.
2. Describe order fulfillment at the KLM.
3. Build an order fulfillment model to evaluate the strategies from the literature.
4. Run the found strategies from literature with the created model.
5. Give an advice to the maintenance, repair, and overhaul from the results.

Result of the fixed TAT and increasing fleet size strategy

The two strategies from literature are calculating spares with a fixed repair TAT and increasing the fleet size. The system is a tree-like warehouse structure with three layers and will result in restocking requests from the in-between warehouses, and the end branch warehouses will determine if a demand request is a priority or no priority, both create feedback. Increasing the fleet size will reduce the service level if repair TAT is not met. Reaching the repair TAT and varying fleet size results in a 100% service level. The number of components sent from the in-between location decreases with increased fleet size and the fixed repair TAT results in a more extended shipment period. The long response period on no priority requests increases the number of priority requests in the system and reduces response period performance.

Solutions for KLM

KLM should reduce the fleet size to below 50 units and slightly reduce repair TAT or reach the fixed repair TAT. It is advised to revise the inventory control assumptions since this report points out improvements to increase validation. Future research in variability reduction and forecasting could improve service levels of certain types of requests. The model build model could be re-used for this purpose.

Contents

List of Figures

List of Tables

Nomenclature

List of abbreviations

ADI	Average Inter-Demand Interval
AFI	Air France Industries
AIP	Average In Process
AOG	Aircraft On Ground
BO	Back Order
CLSC	Closed-loop supply chain
CS	Component Services
CV^2	Coefficient of variation
EBO	Expected Back Order
E&M	Engineering and Maintenance
FE	Forward Exchange
IATA	International Air Transport Association
KLM	Royal Dutch Airlines
KPI	Key Performance Indicator
LC	Logistics center
LRU	Line Replacable Unit
MBK	Main Base Kit
MEL	Minimum Equipment list
MLC	Magazijn Logistiek Centrum
MRO	Maintenance and repair organization
MTBR	Mean Time Between Removals
MTBUR	Mean Time Between Unscheduled Repairs
MTBRUR	Mean Time Between Removal and Unscheduled Request
OEM	Original Equipment Manufacturer
ROCE	Return On Capital Employed
ROS	Return On Sales
RS	Internal repair shop or repair shop
SE	Serviceable component
SLA	Service Level Agreement
SL	Service Level
SR	Stock Replenishment
SRU	Shop Replacable Unit
TAT	Turn-Around-Time
US	Unserviceable component
VEN	External repair shop or vendor
WIP	Work in progress

1 Introduction

Report structure

This introduction chapter describes the position of KLM in the thesis, a summary of the system, the problem, the gap, the goal, and the research questions. Thereafter the structure from Figure 2 gives an overview of chapters, research questions, activities, and if it is theory, analysis or system. The second chapter searches in literature for what is known about line replaceable units and results in two strategies and in inventory control both influence the buffer theory from Hopp & Spearman (2011). The third chapter analyzes the order fulfillment and results in multi echelon structure and indicators. The fourth chapter uses the analysis from the chapter before and data to create a model and validate it. Thereafter is the results chapter discussing the scenario's and configurations to run based on the found strategies in literature. At last the reports recommends in the conclusion with how to improve the service level.

Line Replaceable Units at KLM

This research accredited by the TU Delft is conducted at KLM (de Koninklijke Luchtvaart Maatschappij) in the aircraft MRO (Maintenance and Repair Organization) sector. KLM E&M, CS (Engineering & Maintenance, Component Services) will supply the needed information to execute the case study. KLM is part of Air France KLM and can be divided into three main divisions, namely: passenger, cargo and E&M. These three main divisions are in the air transport or air transport support industry. Within the E&M division, the CS department controls all the logistics, warehousing and rotable component repairs to supply the client with serviceable components; supplying functional components is part of MRO business. The other two departments within E&M are for one Airframes which consists of aircraft modifications, hangar checks, and line maintenance. The last division is called Engine Services, where the main jet engine has repairs Hoed van den (2018)Haak (2019). Air France has its own MRO division called AFI (Air France Industries). To support the availability service, CS has more than half a billion euros of capital in rotable parts. With that, KLM is the world's biggest supplier in number and revenue for Boeing 787 components. Each year, over 240.000 rotable aircraft components are received from clients in the MLC (Main Logistics Center) at CS on Schiphol-Oost. From now on, the MLC will be called the depot.A serviceable, functional, or clean component is defined as a repaired unserviceable, or a dirty rotable component or part with an as good as new condition.

Summary of the system

Component services offer clients availability contracts for LRU's (Line Replaceable Unit) wherein an SLA (Service Level Agreement) is agreed. In aircraft MRO, an LRU is a rotable, defined as a repairable and track-able component. Additionally, a MRO will replace a LRU in a relatively short time at an airport. However the only component that is a rotable but can not be removed in a relatively short time is the aircraft's main jet engine. A jet engine is therefore not a LRU. The SL (Service Level) within the SLA is a threshold for which CS agrees to hand over a serviceable component to the shipping company or a predefined location within the agreed time window; CS commonly uses an SLA of 95%. KLM invoices a fixed price per component per flight hour or month per client aircraft availability contracts.

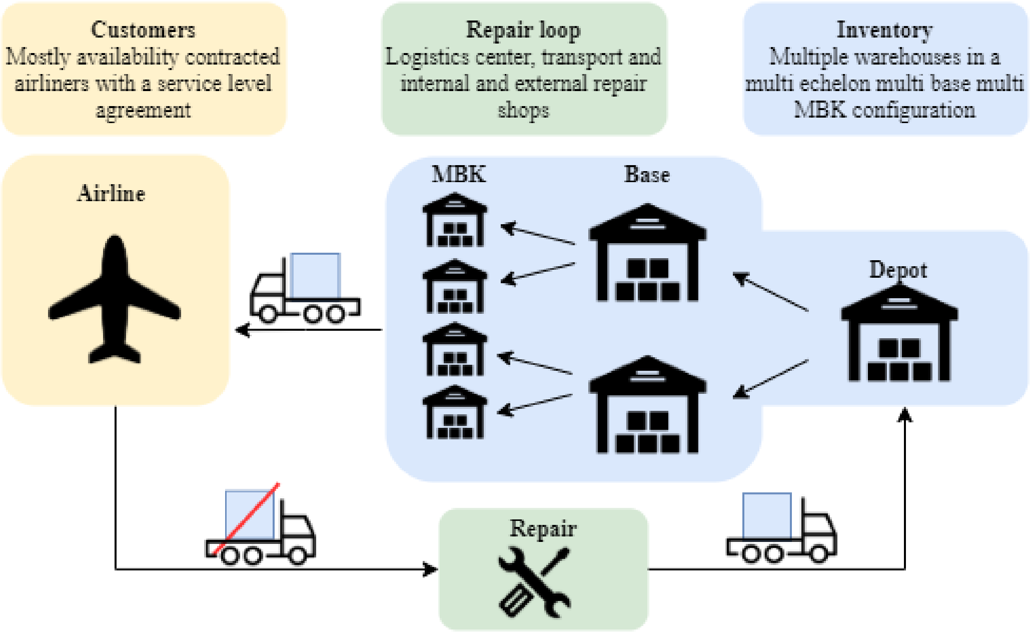


Figure 1: The closed-loop multi-echelon order fulfillment system. Wherein, the physical flow of components between client, inventory locations, repair, and adding components is visualized with an arrow and can not leave the system.

A repairable closed-loop multi-echelon (depot, remote base, and main base kit (MBK)) order fulfillment system enables LRU availability as a service. The property of closed-loop order fulfillment is stock items changing location and never leaving the system, named circulation stock. A MBK is a local stock at the airport leased by the contracted airliner. This is done in agreement with the MRO based on the MEL(Minimum Equipment List) of the aircraft manufacturer (in this thesis Boeing) and a protection level between 70-95%. The order fulfillment system could be divided in four functions on mezzo level as seen in Figure 1 and are: the repair loop(green), the inventory locations(blue), the customers(yellow) and components input/output(red). When a component failure is at the client(yellow), the client sends a request to the MRO starting two processes. One process arranges a serviceable component from one of the depots or remote base (forward-deployed inventory) inventory locations and the second process dispatches the unserviceable component to the repair loop. When no stock is available in one of the inventory locations, the request is delayed for a day. The dispatched unserviceable component is handled by the repair loop, which repairs the unserviceable component and restocks the stock level at the depot. After that, the depot will restock the remote base locations around the world to fulfill upcoming requests. The closed-loop will recirculate the components through the inventory and repair loop. When a client adds an aircraft to the contract, additional components are bought at the OEM(Original Equipment Manufacturer).

This type of request consists of forward exchange and stock replenishment/exchange. Forward exchange ships back the unserviceable component after the serviceable component has arrived. The stock replenishment request however directly ships back the unserviceable component while receiving the serviceable components later. The forward exchange request is used when there is no stock available in the MBK, shipping back the unserviceable component after removal. The forward exchange request could be misused because it has priority above a stock replenishment, by which the client would receive their components faster. If not fulfilled in time, a request results in a back-order to fulfill any day after. Essential components for protecting flight safety have technical limitations grouped in essentiality 1 and 2. Here essentiality 1 is a No-Go item and directly causes AOG

(Aircraft On Ground). Essentiality 2 is a Go-If item representing a specific time window by MEL-A, MEL-B, MEL-C, or MEL-D before the MRO replaces the components. If not, the aircraft will lose airworthiness. When an aircraft loses airworthiness, the client will have to commission a reserve aircraft or adjust scheduling resulting in huge cost and losing passenger's trust.

The context and practical relevance

Prior to this thesis and also under the supervision of P. van Voorbergen[1], a four month desk research by a four TU Delft students Chun et al. (2019) started a system dynamics model to give insights into the MRO processes and parameters. The assignment for this thesis was to finish the system dynamics model. However, Chun et al. (2019) pointed out that the interest is in the advantages of growth. Chun et al. (2019) recommends not to finish the system dynamics model but to apply the inventory control from Kilpi (2007) to reduce components, originally used in Sherbrooke (2004). Within KLM, Hofman (2017) applied the recommended basic version of inventory control optimization part of Sherbrooke (2004), it is available and the multi-echelon optimization is unknown how to use. Improving the inventory control model is declined. Back to system dynamics for a reflection on system dynamics, the causal loop diagrams, recommendations, and results from Tokgoz et al. (2017), Tracht et al. (2013), and Chun et al. (2019) did not catch the attention of KLM. Since growth is an important topic and pooling from Kilpi (2007) would reduce component with a bigger fleet size while a fixed TAT from Munsters (2019) and Driessen (2018) is resulting in a low service level. In combination with the recommendation from Sprong (2019) and Munsters (2019) to model all inventory locations it was decided to look in the consequences of the increasing fleet size and fixed TAT strategy on the order fulfillment performance.

Thus growth and inventory control is important within the LRU MRO, but why? Growth is relevant because next to doubling passengers in the upcoming ten years, according to IATA. The availability contract market is even growing harder Tracht et al. (2013), Tokgoz et al. (2017), and Palma-Mendoza (2014). Equivalent to this is the average growth of availability contracts within KLM CS with 19% in the past three years. When an availability contract is signed, the main activity undertaken by the MRO is buying spares.
Buying spares is relevant because about 40 % of airline cost drivers are in fuel, wages, and landing fees. A potential cost reduction within the cost drivers is the aircraft MRO. 10-15% of all flight cost originate from MRO costs Tokgoz et al. (2017) Wibowo et al. (2016). Holding cost of having capital in inventory components takes up around 20%. 70% - 80% of stock capital in inventory is tied up in repairable aircraft components Tracht et al. (2013). The financial aspects make spares asset management a top priority. KLM CS has around three-quarters of a billion in spares; improving efficiency or reducing the number of spares results in better performance for the client and a more profitable MRO.

Problem

The performance is below the promised service level agreement, but improving performance is complicated. The input parameter for spares calculation is a 99% service level. However, aircraft LRU MRO struggles to reach their service level agreements Tokgoz et al. (2017), Palma-Mendoza (2014). At KLM, the average service level for the Boeing 737 at KLM is 82%, the service level for the Boeing 787 is even lower Munsters (2019). The service level also accounts for components send from the remote base, which is 20% in practice instead of the intended 99%.
A cause for not reaching service levels is the fierce competition between MROs. To stay competitive, aircraft

---

[1] Manager business analysts at KLM Component Services

MRO calculates a new contract on an average repair cost and an initial investment in components based on theoretic(ideal) repair turnaround time values. Thus, calculating with fixed values(investing in more spares) will make the MRO lose competitiveness. The primary strategy is to reduce the repair turnaround time to the theoretic value with constant circulating components. Since the availability contracts are for 7-15 yearsMuhaxheri (2010), the resources are predetermined for the period.
The impact of a low service level or not reaching service levels is that it will suffer the MRO and the client airline. Delayed essential components for operations could result in the loss of airworthiness Tokgoz et al. (2017). Airworthiness loss will decline the brand experience. Moreover, the airline will have a massive cost to reschedule the flight and reimburse flight tickets. The MRO's concern is the contracted airline's penalty clause when the service level agreement fails. The use of a penalty clause is an exception. On most occasions, the client airline did not persist in their commitments as well. As a last resort, the client airline could decide to leave the MRO; this has happened in the past.

The problem is formulated as follows:

Problem: Agreed service levels are not reached in line replaceable unit global commercial availability contract aircraft maintenance and repair organizations.

Gap

Sprong (2019) and Munsters (2019) recommend in their research to model all the inventory locations in the order fulfillment system. The remote base as a continental stock location and commercial requests have not been described in published and KLM research.

Gap: The triple echelon order fulfillment structure is recommended and not described, analyzed or modeled.

Goal

This research aims by modeling to find relations between the configuration, the input, and the output. The input is based on the applied strategies from literature. The effect of MRO's applying strategies to be more competitive on the service level is unknown.
The configuration is the operational and physical structure of the system. Wherein the endogene steering abilities are the operational procedures and parameter; what airlines are contracted, how many components are bought, and how are those components handled. And the exogene influences are for example: the stochastic behaviour per component, external repair shop performance. Make the system a dynamic over time changing environment.
The output are service levels and performance indicators. The performance indicators need to be chosen in such a way that that the configurations feedback can be tested.
This research attempts to provide new insights by performing discrete time step simulation, considering applied strategies:

Goal: Provide new insights by performing discrete time step simulation, considering applied strategies.

Main and sub research questions

To solve the problem the following main question is formulated:

Main question: What aspects make line replaceable unit maintenance, repair and overhaul not reach service levels in global commercial availability contracts?

For answering the main question the suq-questions were formulated.

1. What is the state of art in LRU literature?
2. What is the current state of the order fulfillment system?
3. What is the discrete time-step simulation order fulfillment model?
4. What is the result of varying repair turn around times with fleet size?
5. What is recommended improve the service level?

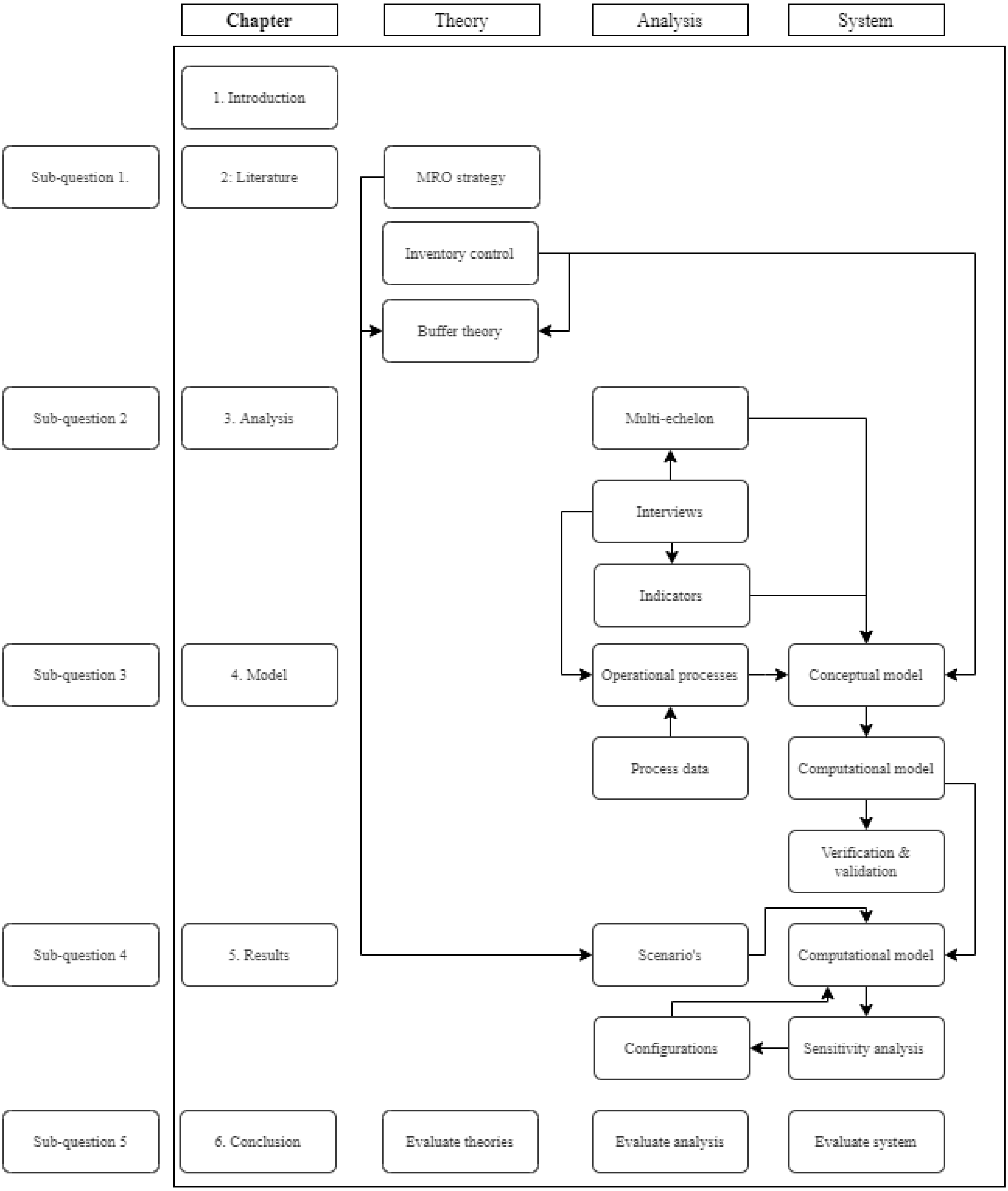


Figure 2: Report structure

2 Literature review on LRU's

This chapter answers the sub-question: "What is the state of the art in LRU literature?". And does it by reviewing LRU logistics literature within KLM and scientific literature. This chapter elaborates on: the structure of the system and the problem of not reaching service levels.

2.1 State of art in published literature

State of the art within general CLSC published literature

First the application of analytical models in general closed-loop supply chains are discussed in the literature reviews of Date et al. (2020), Kapoor & Ambekar (2015) and Govindan & Soleimani (2017). The following paragraph will discuss the research in the aircraft MRO with the assistance of Table 1.

Konyal gives an overview of models applied to closed-loop supply chains. wherein the most used models are mixed integer linear programming, mixed linear programming and different fuzzy logic models. Although most models are non-fuzzy methods. Besides that he describes that all the literature was focused on logistics, electric and electronic equipment and automotive sectors.

Kapoor gives an overview of the three main modeling methods used for inventory in multi-echelon closed-loop supply chains. The three methods are multi-echelon technique for recoverable item control (METRIC), queuing based models and level of repair analysis(LORA). He describes the developments and additions of the three modeling techniques.

Govindan describes only one CLSC application of a mathematical model and find it in Fahimnia et al. (2013). Fahimnia developed a mathematical planning model for green supply chain management and closed-loop supply chain in order to evaluate various scenarios for carbon prices. The decision making level is an integrated tactical-operational planning. They present a mixed integer-linear programming formulation of an actual case company in Australia.

The application of simulation in general supply chains is discussed in the literature review of Tako & Robinson (2012) and Govindan & Soleimani (2017). A gap mentioned before: there no solution in multi echelon CLSC from a periodic review replenishment and Kapoor & Ambekar (2015) suggests a discrete event simulation. If discrete event simulation will be applied the following gap is applicable although it need to researched if there is more recent literature on this topic "The lack of an analytical model with the periodic review inventory policy in the context of repairable items motivates us to look for other solution methodologies. Choice of the discrete – event – simulation due to its wider applicability and its flexibility in customizing the approach to a specific problem context without restrictive assumptions is promising one."

In the literature review of Tako & Robinson (2012), he describes the application of system dynamics and discrete event simulation on a wider supply chain perspective, wherein closed-loop supply chains are not named at all. For the tactical problem mentioned in this research he would recommend supply chain optimization, which is mainly concerned with the identification of optimal policies that optimize key performance indicators, such as profits, costs, product flows, etc. For supply chain optimization problems a discrete event simulation is applied in 88% of the articles in his review. Kapoor & Ambekar (2015)

State of the art within aircraft MRO published literature

General closed-loop supply chain literature does not describe the situation in aircraft MRO. Thus the following literature is specific on LRU's.Table 1 shows a list of published literature with LRU logistics as the topic.

Analytical models are the preferred choice, whereas simulations are on the low hand. Remarkable is the third column showing the third echelon not taken into account in any research. Even with the use of commercial availability contracts as in Kilpi & Vepsäläinen (2004) the echelon structure is the same as when not sharing resources. The third echelon location is between the depot and the MBK to supply local airlines on the other side of the world in a short period. Availability contracts are also possible in a two-echelon context when the contracted airline is nearby. An airline without shared resources and a double-echelon structure, this is when an airline has spares at one of it much visited destinations to reduce the risk of airworthiness loss.

A second strategy is found in Kilpi (2007). Kilpi proofs the component reduction advantage of using a pool of components with different fleet sizes in different strategy configurations without a service level reduction. MRO's try to benefit from this theory by enlarging their shared fleet sizes.

The sub-optimal performance of service level agreements confirmed in scientific literature by Wibowo et al. (2016), Tokgoz et al. (2017), and Tracht et al. (2013) .

Author	**Method**	**Triple-echelon**	**Description**
Ertogral & Öztürk (2019)	LP	No	Repair loop manpower and capacity planning
Visintin et al. (2012)	DTSS	No	Service delivery with product reorder point.
Tracht et al. (2013)	ANA	No	Double echelon repairable planning with service levels
Rezaei Somarin et al. (2017)	LP	No	Double-echelon heuristic stock allocation
Palma-Mendoza & Neailey (2015)	SD, DTSS	No	Business process redesign case study
Driessen (2018)	ANA	No	Effect of indenture level on availability in multiple branches
Kilpi & Vepsäläinen (2004)	ANA	No	Different pooling structures and its advantages
Chen et al. (2019)	ANA	No	Industry 4.0 leveraging on component types
Aisyati et al. (2013)	ANA	No	Continuous review inventory determination
Xie & Yao (2016)	ANA	No	Re-order quantity in limited warehouse space
Gross (1980)	ANA	No	Repair capacity as queuing
Sherbrooke (2004)	ANA	No	Repairable double-echelon optimization
Panteleev et al. (2014)	A	No	Service repair requests process

Table 1: A selection of aircraft LRU MRO models in published scientific literature. LP = linear programming, ANA = Analysis, DTSS = Discrete Time Step Simulation, A = Agent based, and SD = system dynamics.

The relevance of components

Both strategies mentioned in previous sections are related to minimizing the number of spares in stock. According to the aircraft MRO business logic map in Palma-Mendoza & Neailey (2015), cost and capital are used to enable optimal spare part availability, on the other hand revenue is generated with contract rates on the availability contracts. As mentioned in the Introduction, the capital employed in components accounts for up to 70-80% of the capital employed and is, with that, the most influential financial driver for component availability. Capital is invested in new components when a new client signs an availability contract. Within this 70-80% of capital in components the capital is divided between several less and more capital intense components.

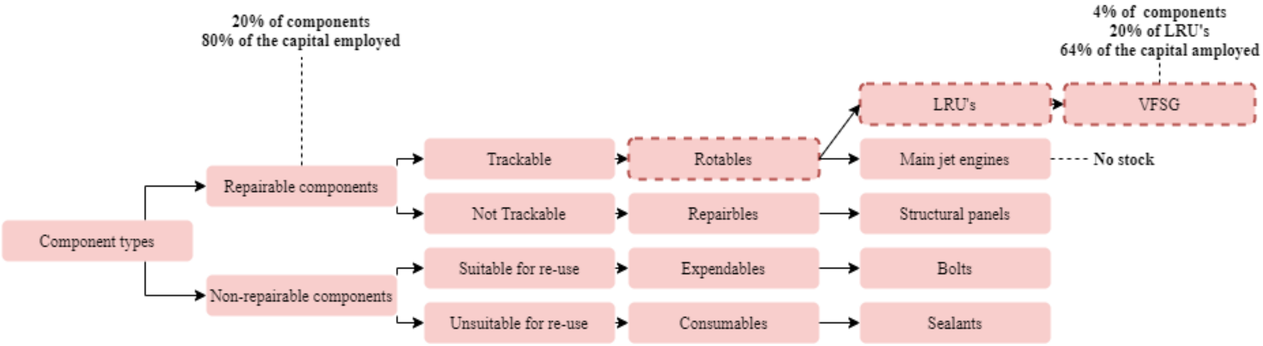


Figure 3: Specification of different component types and their characteristics, including an example Sprong (2019) and the double 20/80% pareto rule from

Figure 3 adjusted from Sprong (2019) divides all aircraft components in 4 groups: rotable, repairable, expendable and consumable items. Rotable components are classified as inventory of type A items and generally account for 20% of the inventory items and 70%-80% of the inventory value Xie & Yao (2016). Within the group of inventory group A again the top 20% account for 70-80% of the value. Within KLM CS, the top 14% of LRU's represent 80 % of the value from a total of 800 different rotable components families. There could be different versions of a component within a component code but all fit in the same type of aircraft. The Boeing 737 inventory has 1500 different LRU's with code names Hofman (2017).

2.2 State of the art within KLM literature

Appendix B.1 Table 29 gives an overview of all logistics related literature within KLM executed by TU Delft students between 2016-2020, Appendix B.1 Table 31 and 30 is an overview of the recommendations. Table 2 describes only the research within Component Services from Table 29. This section discusses the order fulfillment related to KLM literature, concluding with the problem, one of the strategies and the recommended goals.

There is a comprehensive application of operational excellence in the repair loop within KLM CS in the literature review period. Every report takes into consideration some form of the repair section in the closed-loop supply chain. Reoccurring in operational excellence research recommendations is to apply their approach to other industries but lacks research opportunities within the aircraft MRO. In contrast, the order fulfillment is except for Hofman (2017) only researched with a single inventory location, the depot. This research aims to give more insights into all the locations from depot to main base kit, the quality of order fulfillment, and priority requests. How and why is discussed in the proceedings of this chapter.

The recommendations from the summarized KLM research in Appendix B.1 Table 30 and 31 give the following two main recommendations. First both Sprong (2019) and Munsters (2019) apply a discrete time-step simulation. Munster applied it to efficient use of the loan desk, and Sprong used it to evaluate predictive maintenance's advantages. Sprong and Munster only have the depot as an inventory location and recommend taking all inventory locations into account. Both mention more accurate results to implement all locations around the world.
Sprong (2019) uses a single circulating component single echelon constant repair loop discrete time-step simulation. He validates the number of failures by the simulation, NFF's (Not Fault Found), and repair cost. While it is expected that the total repair cost aligns when the number of failures is correct, the repair cost is equal to the number of failures times the repair cost. To measure the business case cost, he uses a repair capacity for sending to much demand to external repairs and borrows when a request is not fulfilled within five days. Cited from Sprong:'The model only considers a single warehouse location while in reality there are more locations all around the world. This simplifies the research and eliminates the need for an algorithm that optimally distributes spare components in stock.'
Munsters (2019) uses a generating failure multi-component single echelon multiple process stochastic repair loop discrete time-step simulation. He uses the argument that the circulation stock level is inaccurate and calibrates the height of circulation stock to the service level in practice. To validate his simulation, colleagues knowing the system check the structure, and he validates the number of back-orders(borrows).
Hofman (2017) has the following recommendation: "The shipment of items between warehouses in the same echelon is called lateral transshipment in literature. Introducing the possibility of lateral transshipment in the model can increase the achieved service level with the same amount of stock." and "It might be interesting to analyze the effect of variation of the TAT on the required investment to achieve the optimization target. To analyze this effect, the TAT can be represented by a probability distribution." However, the report is missing validation. Hofman applied a multi-echelon VARI-METRIC optimization to commercial aviation availability contracts without analyzing how applicable it is. In the following section, the search for a validated commercial aviation circulation stock model is continued.

Munsters (2019) mentions the problem. In 2017, from a total of 10.223 component requests at the depot in Schiphol-East, 1.879 were not send in time as agreed in the SLA. This results in a service level average of 82% for Boeing 737 components. According to the supply chain specialists, the service levels of the 787 reach an average of approximately 70-80%[2].
Thereafter, Munsters mentions of a strategy in an interview with T. Knappers 'KLM tries to hold as little inventory as possible. The stock sizing calculations are based upon the TAT of a component. At this moment, the TAT in reality of spares is too long.' By reducing the the number of spares in circulation the investment is reduced to stay competitive with other MRO's.
The gap is best described from an overview of all KLM literature in Table 2. In comparison this research focuses on the not described MBK, different request types and the quality of order fulfillment.

---

[2] Supply chain specialists exists out of M. Konings, C. Cakiroglu and M. Zondag with 1-5 years experience and master degrees in airline operations or supply chain.

Author	**Model**	**Area's**	**Stock locations**	**priority requests**	**Stock management**	**Stochastic fail**	**OF quality**	**comp. spec. Shops**	**Work station**	**Workforce**	**Quality**	**MLC**	**TAT Repair loop**
KLM TU Delft thesis													
Hofman (2017)	IP	R,O	D,B		+								+
Sprong (2019)	DTSS	R,O	D			+							+
Munsters (2019)	DTSS	R,O	D			+						+	+
Vlamings (2020)	IP	R,O	D										+
Cornelisse (2018)	OE, DTSS	R, O	D			+						+	+
Papadopoulou (2015)	OE	R						+	+	+	+		
Sulyman (2020)	OE	R										+	+
Bosdijk (2019)	DTSS	R						+				+	+
Thijssens (2019)	ANA	-				+							
Hoogenboom (2019)	OE	R						+	+	+			
van Rijssel (2016)	OE, DTSS	R						+	+	+			
Spaan (2018)	MCA, LP	R							+	+	+		
My Goal													
Kalf, (2021)	DTSS	R,O	**D,B,M**	+	+	+	+						+

Table 2: Literature review of all TU Delft master thesis documents from 2016 to 2020 with an abstract including MRO or logistics. LP = Linear programming, DTSS = Discrete time step simulation, R = repair loop, D = Depot, B = Base, K = MBK, OF = order fulfillment

On component failures

Components in an aircraft have a stochastic failure distribution. The time between two connected failures is called MTBUR (Mean Time Between Unscheduled Removals). This is used as the main parameter for forecasting demand as described in the third column of Table 3. From the Table it can be concluded that there is no best practice. While the 1/removals is applied in practice because of its accuracy and simplicity. Besides that, the Table has a wide variety of failure distributions and methods comparable with published literature. For each purpose there is another best applied method. Noticeable, the Table only uses MTBUR, but for simulation a Mean Time Between Removal and Unscheduled Request (MTBRUR) could be more applicable. Because in MTBUR the system influence the time to assembly a component but within the MTBRUR not.

Method	Failure distribution	Measured	Used in
MLE censored	Poisson	MTBUR	Hofman (2017)
MLE censored	Exponential	MTBUR	Sprong (2019)
MLE uncensored	Binomial	MTBUR	Vlamings (2020)
Syntetos	Weibull, binomial	MTBUR	Munsters (2019)
1/ removals	Constant	MTBUR	Inventory model 787

Table 3: Different of methods of removal estimation and application. MLE = Maximum Likelihood Estimation.

Besides MTBUR based on the standard component, there are a few exceptions. Some components need to be replaced before a certain amount of flight hours. The time between two removals is called MTBR (Mean Time Between Removals). Requests for components with a constant time between removals are the hardest to fulfill because the inventory management calculation sheet calculates demand on MTBUR, not considering MTBR. Secondly, there are components called zero planners because there is no stock of them at KLM CS. Those components are rarely contracted and have a high MTBUR. To still fulfill the request, those components are always borrowed. Finally, components could also have different conditions, seals could dry out, and batteries lose capacity.

The relevance of fleet size increase

Kilpi & Vepsäläinen (2004) described the strategy of pooling between multiple airlines, this strategy is interesting because the MRO LRU availability contract market is growing. There are three main growing trends in the aircraft MRO market. First, the total aircraft industry market is growing. Secondly, there is a growth in availability contracts. At last, there is the recent Covid-19 crisis with its impact on the aircraft MRO market. The aviation industry's growth in billion passengers is plotted with three scenarios by IATA(International Air Transport Association) 2018. IATA is the world's biggest airline trade organization and announces a quadruple growth in the upcoming 20 years, doubling the demand in the upcoming 10 years. According to M. Koopmans [3], the growing markets are mainly Asia. The growth of availability contracts at KLM CS in the past two years is also seen in the growing number of contracts for the VSFG at KLM CS (the VFSG is a crucial component in the Boeing 787 aircraft and is contracted in almost every 787 availability contract within CS), as seen in Table 4.

Next to the growth of passengers, more and more airlines choose availability contracts. Kilpi & Vepsäläinen (2004) mentioned the trend of availability contracts to reduce cost by using pooling with an agreed service level. Since the fleet's size supported by the spare component inventory is the most important driver behind the

---

[3] Director of Component Services at KLM E&M CS.

Year	Aircraft	Growth
2018	154	N.A.
2019	188	+22%
2020	216	+15%

Table 4: The number of contracted VFSG aircraft according to the 787 inventory model at KLM CS

inventory cost, inventory pooling among several airlines is an intuitive way of exploiting the scale economies of availability services. This economies of scale benefit originates from the law of large numbers where variation reduces when the number of random events increases. This accounts for all the inventory locations from the depot: base up to the MBK. The reduction of inventory components effect from pooling is shown in Figure 4. The pooling gives MRO an opportunity to expand their client list with more aircraft. This situation results in shifting risks and uncertainties to the service provider from customers, Wibowo et al. (2016).
The future impact of the Covid-19 crisis on the aircraft MRO is unknown. In this thesis, the problem and the model evaluate the situation as before the out breach of Covid-19. It is still relevant because the market is expected to recover and thus will be growing relative from now. The model could even be used to simulate the effects of having a surplus in stock from the pre- Covid situation on the service level and cost.

Desk research into inventory control

The strategy of not reaching TAT and pooling are both based on reducing spares to stay competitive. Since spares are only bought when an availability contract is added this section will explain the inventory control. The buying of components within availability contracts is called initial provisioning. The inventory control are mathematical formulas to determine how many spares need to be added for an additional client. The investment in spares and the cost of repairs determine the value of a contract.
The inventory control model calculates the number of components in the remote base and depot locations around the world. For MBK locations a protection level is agreed with the client based on the same theory. The theory from Sherbrooke (2004) is applied by Hofman (2017) on the Boeing 737 comparable with the Boeing 787 but without optimization. The Boeing 787 inventory control within KLM is found in this section and used in this thesis. The choice for Boeing 787 inventory control is because it is no optimization thus a small set of components is not influenced by the parameter of all components.
Based on the number of removal in the past two years and the increase of aircraft in the upcoming year a forecast is done in the number of removals for upcoming year. This value is the 12 month removals. Equation 1 uses the number of 12 month removals times the TAT to calculate the average time spares not available per year. Divided by the number of days in a year it will result in the average number of components in process per day.
AIP(Average In Process) **With**:

$$AIP_{depot} = \frac{12\, month\, removals * TAT}{365} \tag{1}$$

The same is done but then for the number of aircraft connected to a remote base location to determine the demand at a remote base.
With:

$$AIP_{remotebase} = \frac{Fleetunits\, remotebase * shipping\, time\, remotebase * 12\, month\, removals}{Fleetunits\, total * 365} \tag{2}$$

Now the demand at the depot and each remote base location is known the variability is added. To cope with variability the cumulative inverse standard deviation is used. Equation 3 determines the maximum number of requests on a day covering 99% of the days times the number of days the component in process.

$$OLV = Normal_{inverse}(probability\ =\ SL, Mean\ =\ AIP, standard\,deviation\ =\ \sqrt{AIP}) \tag{3}$$

For the components in the depot the result is rounded up as in equation 4.

$$Depot\,stock\,threshold\,level = max\,(RoundUP(OLV, 0), 0) \tag{4}$$

Whereas for a remote base it depends on the type of components. See in equation 5, 6, and 7 the follow up for different essentially and MEL categories. For essentially 1:

$$essentiallity = 1 \wedge \frac{Fleetunits\,remotebase * 12\,month\,removals}{Fleetunits\,total} > 2 \tag{5}$$

For essentially 2 and MEL is A or B:

$$essentiallity = 2 \wedge \frac{Fleetunits\,remotebase * 12\,month\,removals}{Fleetunits\,total} > 2 \wedge MEL = A \vee B \tag{6}$$

For essentially 2 and MEL is C or D:

$$essentiallity = 2 \wedge \frac{Fleetunits\,remotebase * 12\,month\,removals}{Fleetunits\,total} > 5 \wedge MEL = C \vee D \tag{7}$$

If equation 5, 6 or 7 is true **then** apply equation 8.

$$Remotebase\,stock\,threshold\,level = max\,(RoundUp(OLV, 0), 0) \tag{8}$$

Else apply equation 9

$$Remotebase\,stock\,threshold\,level = max\,(Round(OLV, 0), 0) \tag{9}$$

It could be directly noted that less important components with essentiality 2 C,D or 3 are almost not kept in stock at the remote base locations.

Besides that the working of pooling could be made understandable. Pooling is explained with Figure 4 in which the filled dotted blocks represent the buffer to cope with variability. A seen in the Figure for 300 fleet units this amount is relatively less than at 100. 2.3 spares per 100 at a fleet size of 300 compared to 4 spared per 100 at a fleet size of 100. This is the result of more less variability thus there has to be less spares to buffer high variability in demand. Less buffer also has the disadvantage of making the system more vulnerable for other variability not caused by contracted demand for example variability from repair, shipping, or loan desk.

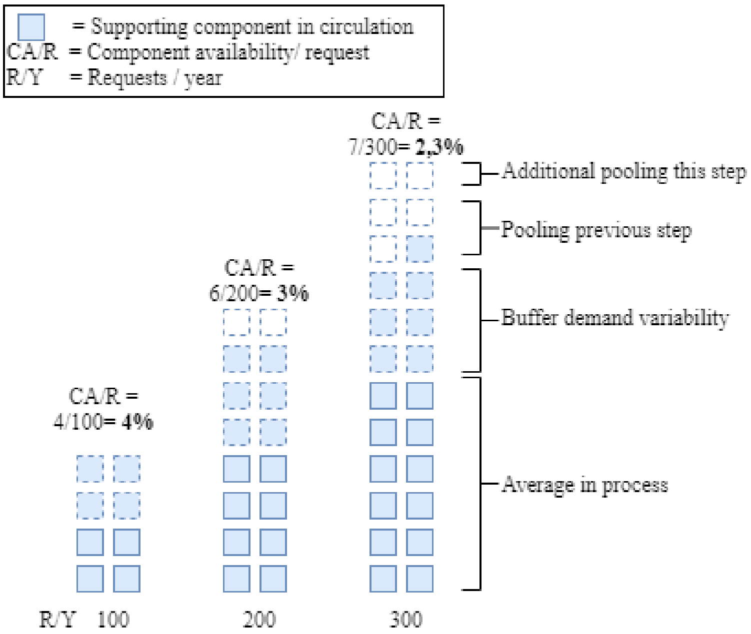


Figure 4: The pooling effect explained with a made up example. The relative amount of spares needed for an additional set of fleet units will reduce.

2.3 Manufacturing buffer theory

The previous section ends with the conclusion that pooling will result in less safety stock and before was mentioned that the TAT is not reached both impacting the buffer of the closed-loop system. The relation between financial and physical aspects is based on the theory of Hopp & Spearman (2011). There are three ways to synchronize demand and supply. High variability in demand and supply is solved with the buffering law. The buffering law: Systems with high variability must be buffered by some combination of inventory, capacity, and or time. An illustration of the buffering law is shown in Figure 5. According to Cornelisse (2018), the interpretation of the buffering law in aircraft MRO is as follows: if you cannot pay to reduce variability, you will pay in terms of high work in progress (WIP), underutilized capacity, or reduced customer service. Following from the buffering law comes the variability law. Variability Law: Increasing variability always degrades the performance of a production system. Higher demand variability requires more safety stock for the same level of customer service, and higher TAT variability requires longer lead-time quotes to attain the same level of on-time delivery time.

The application in aircraft MRO practice of paying for more WIP is in Figure 5. The strategies itself influence each other and the system. Enlarging the fleet size will result in lower demand variability Sherbrooke (2004) and not reaching the TAT is the cause of limited and slow resources. The resulting WIP is better explained as follows:

- Reaction time for a request will take longer due to low availability of spares. A reaction time longer than a day will result in the loss of service level Sherbrooke (2004).
- Shipment time will be longer due to sending spares from a further away location.
- More borrowing because the spares are not available within serveral daysHofman (2017), Sprong (2019), and Munsters (2019)

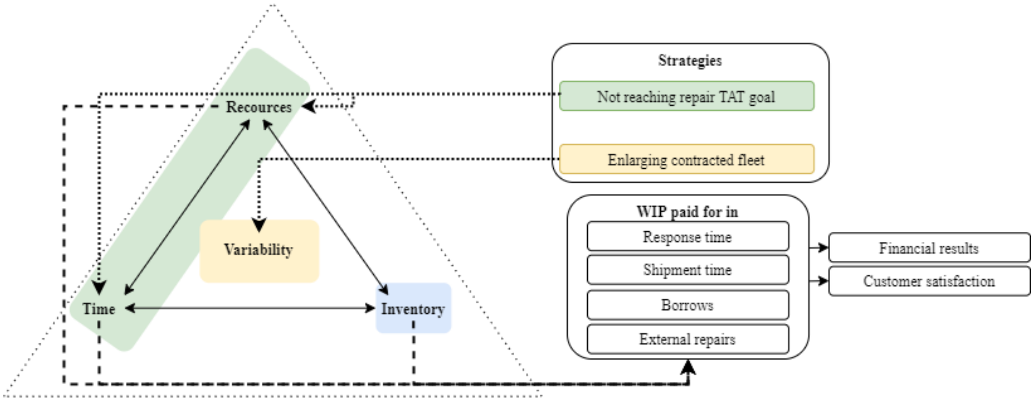


Figure 5: The influence of the two strategies on the order fulfillment WIP according to the theory to synchronize demand and supply by Hopp & Spearman (2011)

- Less loaned spared due to lower availability Munsters (2019).
- External repairs when a the repair capacity is reached Sprong (2019).

2.4 Conclusion

From this chapter can be concluded that not reaching service levels is a problem according to KLM literature Munsters (2019) and according to scientific literature Chen et al. (2019), Tokgoz et al. (2017), and Tracht et al. (2013). Not reaching the service levels is done deliberately to stay competitive. By not reaching the repair turn around time Munsters (2019) or the other way around by not buying enough components. A second strategy applied by MRO's to stay competitive is enlarging the contracted fleet size Kilpi et al. (2009). The addition to scientific literature is recommended by Sprong (2019) and Munsters (2019) by giving insights in all inventory locations within the order fulfillment system. To the best of the authors knowledge the description of a triple echelon structure with corresponding operational procedures is not available in scientific literature.

3 The current state

This chapter answers the sub-question: '*What is the current state of the order fulfillment system?*'. The structure and specification of the order fulfillment is analyzed with the use of interviews and literature. First, the inventory locations are described because it is recommended by Sprong (2019) and Munsters (2019). Thereafter, it is the system boundary that scopes the thesis project. In that section, a description of the sub-systems within the system boundary is given. At last the indicators for the system are described.

Inventory locations

A multi-echelon inventory structure is formed like a tree structure where each location can have one predecessor and multiple successors. The top-level will always contain one single location named the depot. In a closed-loop system, the physical objects (components) do not leave the system. Here those are repairable components that can be recycled multiple times, after which it will restock the depot until scrapped as described by Hofman (2017). The order fulfillment is the process of receiving, processing, and delivering a request to the client.

Figure 6 shows the flow of components between a removal and the assembly of a component. M. Bosch added the remote base, and the MBK at KLM to the swimlane from Cornelisse (2018), Sulyman (2020), and Sprong (2019). The depot, remote base, and MBK all have a different function within the order fulfillment. The depot is restocked from the repair loop with serviceable components and restocked from the lease company if there is no stock in the client's transport route. In reality, a component from the lease company will be sent directly to the MBK. After receiving components, the depot distributes the components to the base locations or fulfills requests. The base locations only serve as a forward-deployed local stock location so that the client is served with a shorter lead time. The service level is successful when the component is handed over to the shipping company within the agreed upon time frame even if it is sent from the most remote location. This base location is restocked from the depot when its stock levels are below the threshold, and the depot has above minimum threshold stock levels. The MBK is a small inventory at the client's airport location based on a protection level agreed upon between the client and KLM. The client manages the MBK stock, although the stock is KLM's property where KLM leases it from another company.

A client could make several different types of requests when a component fails to serve himself with a serviceable component. The agreed handling days of a component when a stock replenishment request comes in is summed in Section 3.1. Below are the different types of requests from the client airline.

- **Forward exchange**
  A component failed and there is no stock in the MBK. Thus first, the serviceable component is sent to the airlines and after removal, the unserviceable component is sent back. Forward Exchange requests have to be fulfilled according to contract, mostly within two days. In practice, there is a negotiation with the client, and the regulation limitations are used.
- **Stock replenishment/ Exchange**
  When a component fails unexpectedly and the airline has stock in the MBK, it sends a stock replenishment request. The airline already sent back the unserviceable to the repair loop and replaced it for the one in the MBK stock before placing the request. The request still has to be fulfilled within about 2 days but there are no technical problems ahead causing the loss of airworthiness. An exchange request is the same

but then requesting for another part serial number.

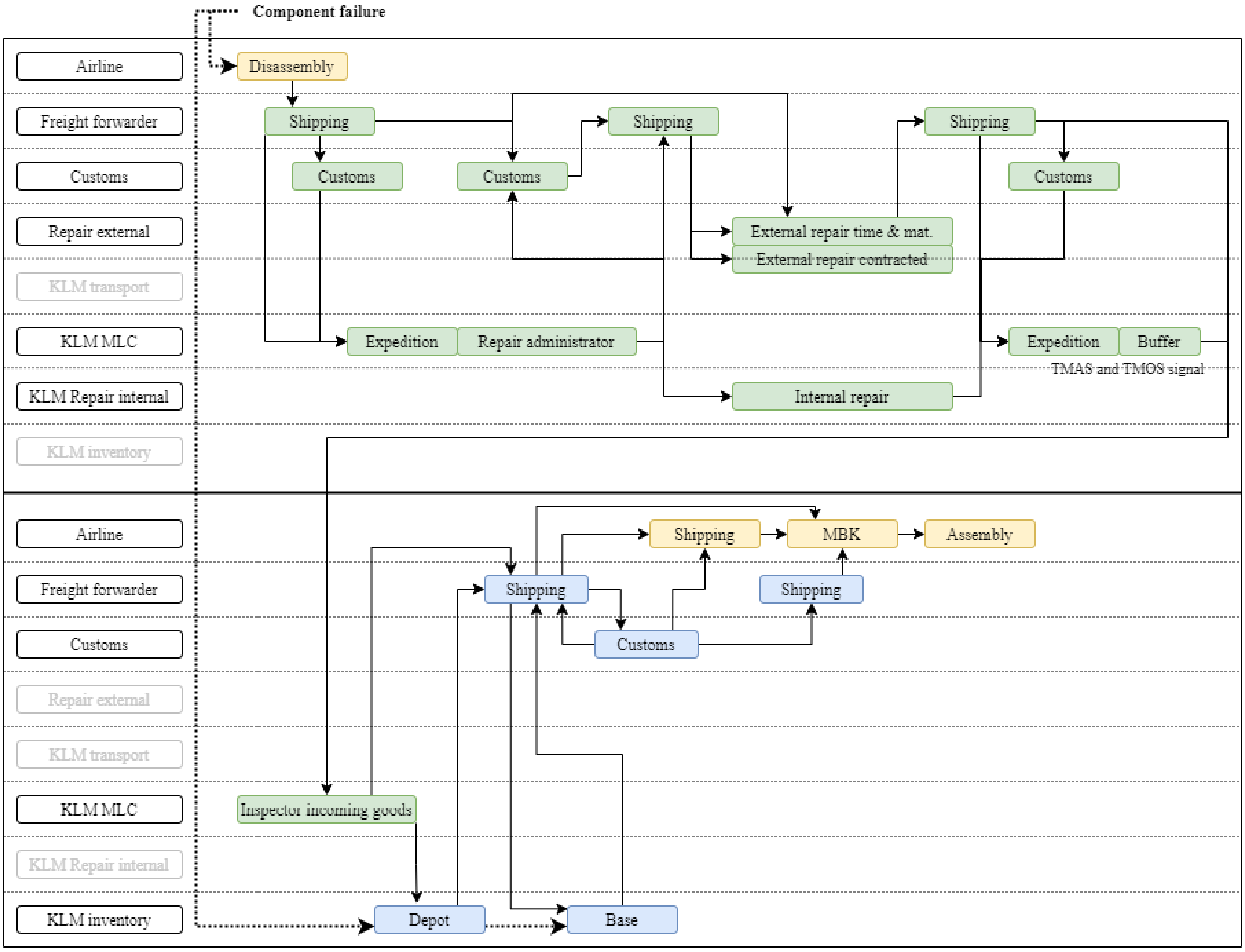


Figure 6: The flow of a component from removal to assembly animated from a combination of sources. the repair processes from: Cornelisse (2018), Sulyman (2020) and Sprong (2019). The remote base and MBK is added from an interview by M.Bosch

3.1 System boundaries

The previous section describe the inventory locations structure to research, this section scopes the research and describes the different aspects of the order fulfillment. The following bullet points give an overview of the scope.

- There are no market forces on pricing. This means that all external market contract prices, component prices, and leasing prices have a constant, not influenced rate from outside the system and are always available.
- The system should recognize when an order is fulfilled but has no insight if a faster-handled request will lead to more airworthiness, which results in more flight hours.
- The system is handling LRU components for aircraft MRO.

The commercial aircraft MRO supply chain is structured in four main departments depicted in Figure 6. These components are: the repair loop(green Section 3.1), the inventory locations(blue more on this in Section 3), and the clients(yellow more on this in Section 3.1). Figure 6 is an overview of the flow for a single components. The business process for a request with only a depot is best described in Palma-Mendoza & Neailey (2015), but here a different configuration is taken. In a simplified matter, the business process shows that when parts fail at the client (yellow on the left), two processes are started. One process arranges a replacement part from one of the blue inventory locations. The second process dispatches the unserviceable component to the repair loop (green).

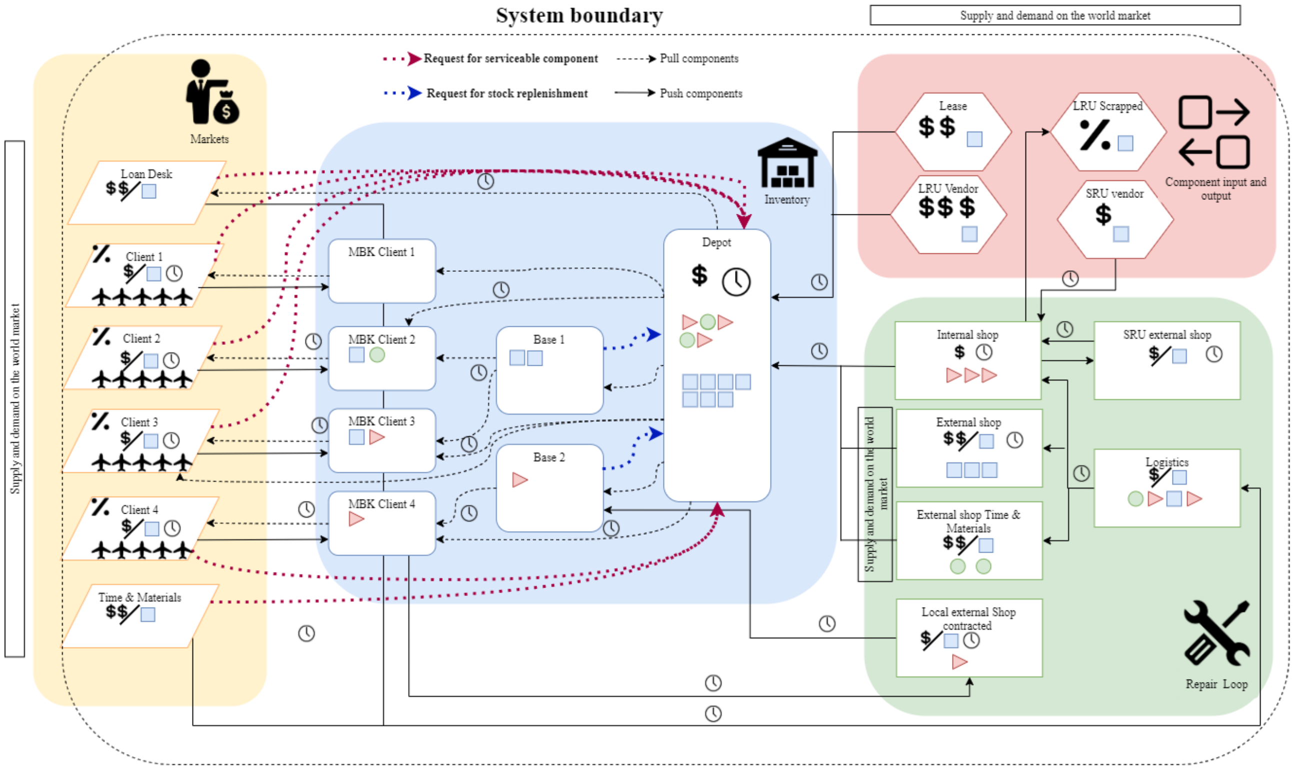


Figure 7: Overview of component flows from location perspective

Essential components

When a component fails, the urgency to replace it depends on the type of component. The urgency is obviously different for a start motor that will cause AOG than for one coffee maker when there are seven other coffee makers on board an aircraft. That is why each component has a priority to which the MRO should comply if the component is requested for exchange handling rather than the standard stock replenishment. The following list of components fast handling. The different request types are discussed in Section 3. The times named here are the technical limitations on rules and regulations.

- **Essentiality 1**, also known as No-Go items. If these components fail, the aircraft will lose airworthiness. This could thus resolve in an AOG. KLM generally promises a 95% service level and hands the component over to the shipping company within 3 hours.
- **Essentiality 2**, also known as Go-If items. If these components fail, several restrictions will cause the aircraft to lose airworthiness in a short while. This could thus resolve in an AOG.
  - **MEL-A**, KLM generally promises a 95% service level and hands the component over to the shipping company within 24 hours in case of AOG then within 3 hours.

  - **MEL-B**, KLM generally promises a 95% service level and hands the component over to the shipping company within 48 hours in case of AOG then within 3 hours.
  - **MEL-C**, KLM generally promises a 95% service level and hands the component over to the shipping company within 72 hours.
  - **MEL-D**, KLM generally promises a 95% service level and hands the component over to the shipping company within 240 hours.
- **Essentiality 3**, also known as Go items because the aircraft will not lose airworthiness when this component fails. KLM generally promises a 95% service level and hands the component over to the shipping company within 240 hours. This is also named a stock replenishment.

Clients & contracts

The client pays in the form of revenue to the MRO for a delivered service. All contract types are described in Wibowo et al. (2016). Excluded because they are not offered at KLM are the service offer and the dry-lease agreement. The delivered service at KLM exists out of one of the following contracts. The three types of contracts are described in order of the productization from low to more. Productization involves taking a skill or service that has been used internally and developing it into a standard, fully-tested, packaged, and marketed product Wibowo & Tjahjono (2017).

- **Time & materials contracts** are similar to an auto garage where you bring your car and pick it up when it is repaired. The airline communicates the work scope, and the MRO arranges manpower and capacity to deliver pure MRO to the customer. For example, in this situation, the airline has to wait until their unserviceable component is repaired to receive it serviceable back.
- **Component lend out contracts** are available via the loan desk. A component is lent out for a short while to a client for extra revenue. This is done when stock levels are high and spare repair capacity is high Munsters (2019).
- **Availability contracts** give a total component solution. This solution combines service and products to fulfill the total customer's requirements by adding a tangible product to the service. "Under this type of contract, customers buy a predetermined level of service availability, instead of paying the maintenance and spare parts costs directly to the service provider. In other words, the contractual arrangement of the availability contracts is centered on buying the performance outcome rather than the spare parts availability or repair activities. In fact, this puts the responsibility onto the service provider to fulfill the demand at an agreed service level "(Mirzahosseinian and Piplani 2011). In the case of the service provider failing to do so, it is penalized by a predetermined penalty cost" [Aghil Rezaei Somarin 2017]. The availability contracts are agreed upon for 10-15 years Wibowo & Tjahjono (2017).

Availability contracts are the main business of CS. They differ between contracted airliner. The contracts variate on the following conditions with an influence on the system buffer:

- **Location** where the MBK of the client is stationed. An exception to this is Polish Airlines with an MBK at two locations where they often fly between.
- **Supported components** by the airline, where the quantity is determined per supported component. It is optional to change the priority of a component to a higher level. Airlines at the Shanghai base have an AOG order fulfillment time of 30 minutes.
- **Flight hours** impact the number of removals and could be constant or variable.

- **MBK stock levels** are based on the protection level, the capital involved, and the importance of the component.
- **Service levels** can differ per airline or even per component priority. For example, Virgin Atlantic negotiated a 98% service level.
- **Incoterms** defining the responsibilities and locations of component hand over are in the transport agreement. Currently, CS is trying to remove all ex-works from the incoterms in the contracts.
- **Percentage of forward exchange** requests may vary between 10-15 % per contract. With fewer aircraft and lower MKB stock levels, these values are on the high side of the variation.
- **False returns** may not exceed a certain threshold per year. A false return occurs when, after removal, a functional component is returned. This happens when the airline does faulty diagnostics of what has failed.

Repair loop

The repair loop's main function is to repair unserviceable components dispatched from the aircraft into a serviceable component to restock the depot as seen in Figure 6. The dispatched component is shipped from the customer to MNL on Schiphol, where the component is shipped to internal repair shops or external repair shops. A component is repaired internally or externally based on the type of component. When the internal shop's maximum capacity is reached, the component is sent to an external shop. This rarely happens.

There is a set of differences within the repair loop, different per component and type of failure as could be seen in Figure 6

- **Component repair shop**, 60% of specific components is repaired in internal shops at Schiphol-Oost Driessen (2018). For external shops, also known as shop vendors, there are two options. External shops could have a contract with KLM with agreements about price, quality, and repair time. Other external shops are repairing without a contract. This is called time and materials in Figure 6 and could take even longer because there are no time and price agreements.
- **Location** and incoterms determine the logistics route. When contracted at Schiphol, the component does not have to be shipped or agreed to send to the country's customs.
- **Direct to the repair shop**, for some components, it evident to the client what the repair is about. Then it is also a possibility to directly send the component from the airline to the repair shop, eliminating expedition and repair administration.
- **Repair shop capacity** of the repair shop is dependent on manpower and resources. Resources exist out of consumables and SRU's. Not only the components in the aircraft are modular and rotable, but even the sub-components are. This is called an indenture level. Keeping more indenture level components on the stock could be a cheap way to reach higher service levels.

3.2 Indicators

The service level and response period mentioned in section 4.1 are the main indicators of performance. The response time existing out of the reaction time plus the shipping time. The reaction time is the period which it takes to handle over a request to a shipping company. For visualization in a single graph it is a required that all indicators are normalized with the same units. The service level with as unit % has a maximum and minimum and is fit for the job. To give insights in the response period performance the percentage of components that have to be send from a remote base will cause a shorter shipping time. Secondly the percentage of forward exchange request will give insight in the response time of stock replenishment request taking longer to restock. The indicators can be found in Table 5.

Indicators	Unit
Service level request	[%]
Forward exchange requests	[%]
Shipped from base requests	[%]
Spares per contracted component	[%]

Table 5: The two indicators for more client value with their measured units. Due to additions made in a later the stadium of the research the two indicators have different units.

After finishing the model and results in a conversation with M. Koopmans, he mentioned the importance of a high priority request sent from the nearest location. Because if a priority request is sent from the nearest location without reaching the service level, it could be faster at the clients' location, which means more value. Thus the total reaction time plus transshipment time are equally important as reaching the service level for high priority requests.

3.3 Conclusion

This chapter concludes that there is a multi-echelon inventory structure existing of a depot, remote base, and MBK. The MBK determines the type of request either being a forward exchange or a stock replenishment request dependent on the MBK stock level. Wherein, the stock replenishment takes less component availability than the forward exchange request. This research is scoped to the availability contracts with any external pricing forces. To evaluate the system the four main indicators are service level, percentage forward exchange requests, shipped from base percentage, and spares per contracted component.

4 Order fulfillment modeling

The qualification, verification, and validation processes from Abrahamson (1979) are applied to develop the order fulfillment model. First the model objectives are described. Then side by side sub-systems are analyzed and modeled, the sub-systems are separated per swimlane row, answering the sub questions: 'What is the discrete time-step simulation order fulfillment model?' . The swimlane and assumptions base their processes and parameters on interviews and data analysis. After creating the concept computational model it is verified. The report will do two validations per section up to seven validations, using the indicators as a guideline.

4.1 Model objectives

The order fulfillment model objective is to give insights into stochasticity, feedback, and operational procedures between MRO strategies and the service level. To reach the objectives, Figure 8 depicts the model's black box with a structure, input, output, and performance. The black box structure is based on the physical structure described in the upcoming sections' operational processes. For setting up operational processes and indicators, the use of interviews and data analysis was used. The number of fleet units input determines the circulation stock in the simulation according to inventory control. Other in going parameters, the model's input exists out of the number of fleet units and the repair turnaround stochastic used in the next chapter as the experimental plan. The performance is based on Sherbrooke (2004) while the output is an executed number of years with a fleet size over that period.
To evaluate the two strategies alternatives in their potential a model is needed to mimic the real world. All order fulfillment research mention the stochastic failure behavior of aircraft components. Combined with changing availability over time makes it a dynamic over time changing environment. Both arguments have made the choice for a discrete-time step model.

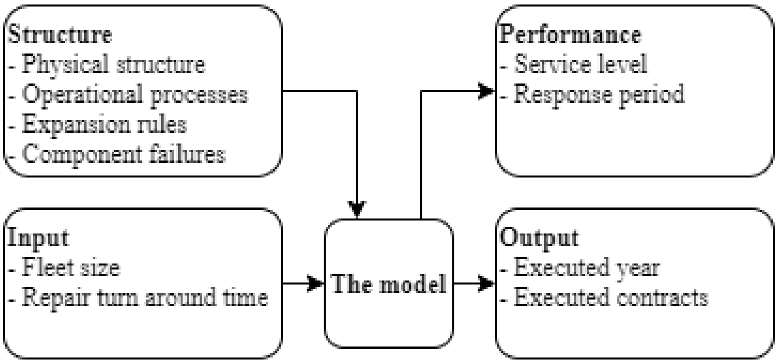


Figure 8: The simulation as a system mapped according to Delft systems approach

4.2 Conceptual model and assumptions

The following sections will separately describe the swimlane blocks from Figure 9. This Figure shows that a failing component is noticed and filed by the pilot. The filed request is handled by airline logistics and determined whether it is a forward exchange request or a stock replenishment request. If it is a forward exchange request, a spare undergoes the following processes in sequence: handle, assemble, and repair. If the request is a stock replenishment, the repair and handling start at the same time. While the restocking of the remote base only occurs when a component is sent from a remote base location. From the swimlane could be concluded that a single request will activate processes for a single component.

The first assumption is about the total process in relation to finance:

1. **Only the capital in components directly impacts financial results,**
   The data shows for external repair, borrow, and fine clause only outlier data. The fine clause is only used once in the past year according to Voorbergen, but none of the service levels is reached. External repairs in the data are not found with the exemption of local repairs. For borrowing the data in Appendix C Table 33 shows the validation of a borrow assumption but the validation data shows barely borrows. The validated borrow assumption was that if a component takes more than five days to handle, it will be borrowed. The transport cost and repair cost are linear with the number of added contracts and therefore not interesting. Thus only the holding cost of capital employed in components is a financial result of the system.

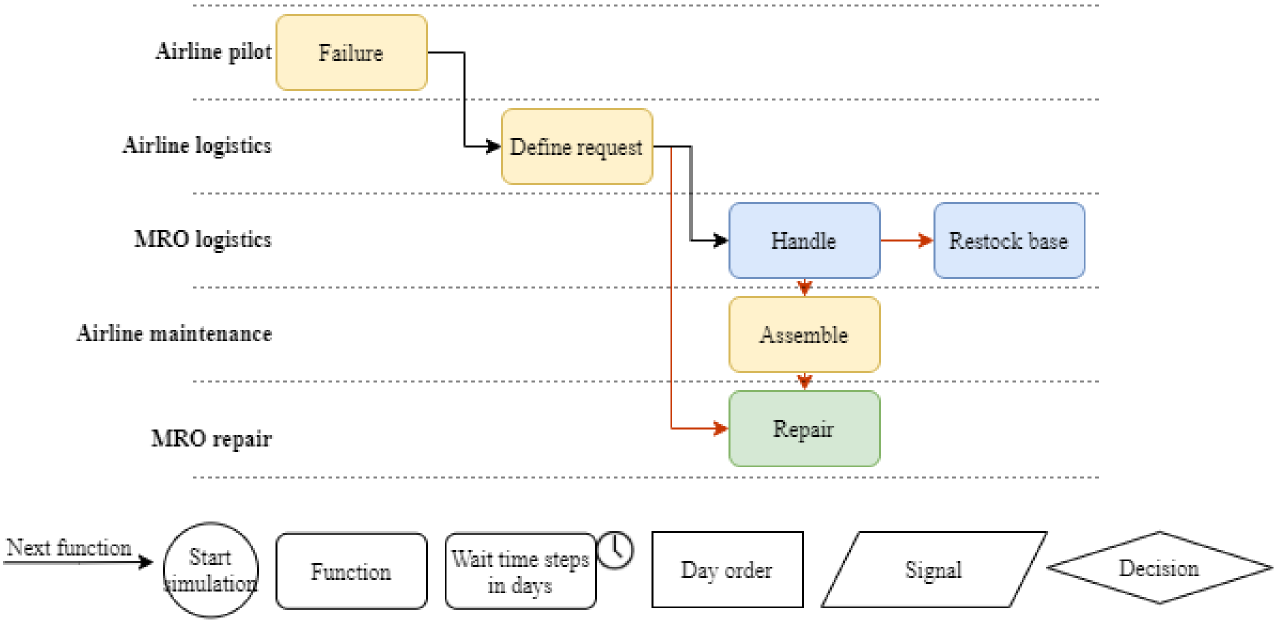


Figure 9: The MRO client request lane section from the total swimlane using as input a FE or SR handling signal. Locating stock and shipping it to the client. The process may activate a remote base restocking or assembly signal

Conceptual model and assumptions airline pilot lane

This sub-process visualized in Figure 10 is a combination of both the starting procedure and the pilot. When a year starts all pilots fly in an aircraft with components. When a component has failed the pilot gets a signal from the aircraft that a function is not working. In most cases flights can continue since components are redundant but a failed component need to be replaced within a certain period, more about the replacement periods in Section 3.1. The failed component is replaced and the aircraft with pilot can return to their duty. Due to incomplete and missing it is chosen to make failures independent of repair, same as in Sprong (2019), Hofman (2017) and Munsters (2019).

2. **Airlines differentiate on flight hours, location, contacted components, and number of aircraft,**
   with that all availability contracts have the same contracted service level, response time and incoterms. The MTBR is constant per airline. The MTBR differentiates between airlines because it is dependent on flight hours, flight cycles, and maintenance procedures. Unfortunately, there is not enough data to support airline dependent significant MTBR.
3. **Removals are cumulative exponential distributed based on MTBUR, calculated by KLM and constant throughout the year.**
   The MTBUR calculated by KLM in the Boeing 787 inventory model is used in a cumulative exponential distribution to generate the time up to the next failure from the time step of the past failure. There is no seasonality according to the number of failures per day in 2016, 2017, 2018 and 2019 analyzed bySulyman (2020) and Munsters (2019).
4. **A total of ten components are selected, of which two components of each MEL and two from each essentiality. These components must have the highest capital and have at least 20 connected data rows in the last six months.**
   First, a list of the top 100 components with the most capital employed in total stock is made (new price times number of spares). They are sorted ascending on high capital employed. A corresponding data row exists of a component part and a serial number matched between SAP and Aero exchange from the last six months of requests. Essentiality 3 does not have any component that fulfills the requirements, which is not a problem because those components can not result in a loss of airworthiness and is with that the least important. The list of selected components with specification is shown in Table 32
5. **The simulation is started for every component and airline times aircraft and QPA with time steps of one day.**
   Operations are active every day of the year and have time steps of one day. This assumption simplifies the service level agreement from fulfilling requests within hours or minutes.

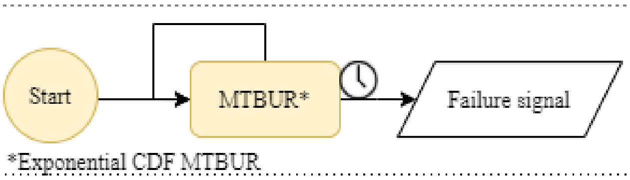


Figure 10: The MRO client request lane section from the total swimlane using as input a FE or SR handling signal. Locating stock and shipping it to the client. The process may activate a remote base restocking or assembly signal

Conceptual model and assumptions airline logistics lane

Figure 11 depicts the airline logistics process with the most important function to determine the type of request. From the interviews used for operational processes, the supply chain specialists underlined the importance of the different request types for higher customer satisfaction in order of priority:*AOG-request, TBR-request, forward exchange request, stock replenishment request, and remote base restocking request.* However,the AOG-request is complex to model due to its multiple location failure probabilities, and the TBR-request represents only a minor group of components; that is why those request types are not taken into the simulation.

Thus the pilots send a failure signal to the airline logistics. Airline logistics sends out a forward exchange (FE) signal when the MBK stock level is below threshold. If not then it will replace the component in the aircraft with a spare from the MBK and sent the failed component to repair with a repair signal. After a few days the client will request for a serviceable spare to restock the MBK by sending a stock replenishment (SR) request. For the airline logistics there are the following two assumptions:

6. **Airliners will place a FE request when MBK stock level is below threshold based mentioned by the supply chain specialists.**

7. **Every day, first the depot is restocked from the repair loop, secondly FE requests can take place, thirdly SR requests are prioritized, and at last, the remote base is restocked according to M.Palm** [4]

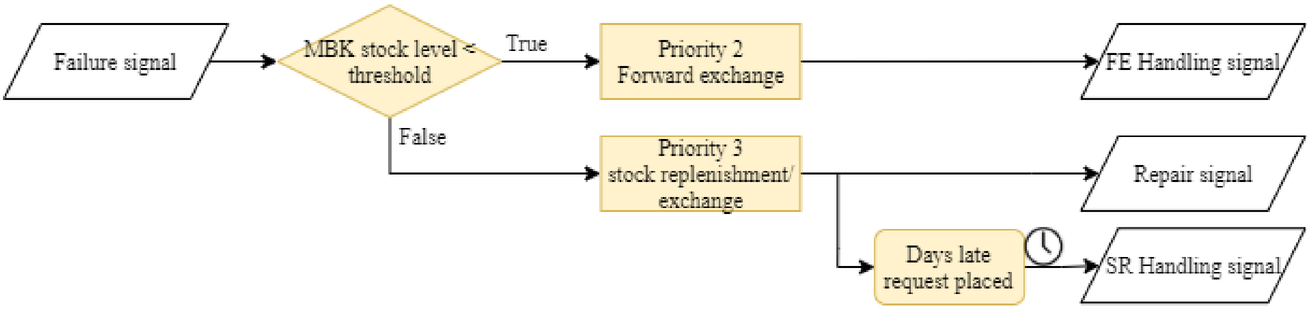


Figure 11: The MRO client request lane section from the total swimlane using as input a FE or SR handling signal. Locating stock and shipping it to the client. The process may activate a remote base restocking or assembly signal

Conceptual model and assumptions MRO repair lane

The repair sub-process in Figure 12 can be activated by two processes. When a failure occurs and there is stock in the MBK (as mentioned in the previous section) or after a component is assembled as a follow up of a forward exchange request. The unserviceable component is in the MBK and send by the airline logistics for repair. The repair system concerts the unserviceable component into a serviceable component and restocks the the depot at the first moment in the day. For the repair loop the following assumption is made by simplifying the complex set of processes in Figure 6.

8. **The aggregate repair loop exists of waiting days with a normal distribution.**
   The repair capacity is adjusted for demand; thus, the repair loop's duration and deviation impacts order fulfillment. Due to time constraints, a standard deviation was made fit to the repair data. The repair loop data is given per component and is defined as the time from airline removal to restocking the depot. No other distributions were evaluated. This data is extracted from the connected data rows from the last six months(August 2020), mentioned in the previous assumption.

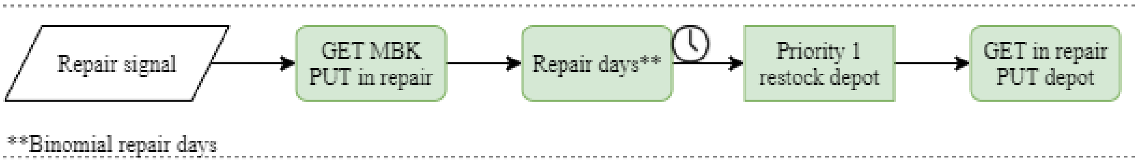


Figure 12: The MRO client request lane section from the total swimlane using as input a FE or SR handling signal. Locating stock and shipping it to the client. The process may activate a remote base restocking or assembly signal

Conceptual model and assumptions MRO airline logistics lane

Figure 13 shows the sub process of handling a request at the MRO logistics department. The main function is to determine the location where the spare is send from to the client MBK. First is checked if a connected remote base has stock, if so then the component is sent from the remote base location and the restocking of remote base process is activated. If not then is checked of there is stock in the depot, if so then the component is send from the depot, if not then MRO logistics waits for the next day restarting the handling process. If the to handle request is of the type forward exchange then after arriving at the MBK the assembly process is activated. To modeling this process the following four assumptions have been made:

9. **A request is handled the first day from the remote base or from the depot and at last at the day after according M.Palm**

10. **A request is handled successfully when a component is sent from remote base or depot on the first day.**
    Palm mentioned that most of the service level requests are reached if shipped within 2-3 days, but there are many exceptions, for example, 45 minutes, 30 minutes, 24 hours, or delivered at a checkpoint within a certain amount of days. The request data show that the median response time average for all components is 0 days. For simplification, the service level is only reached if send on the first day.
11. **Initial stock quantity for the remote base, depot, and MBK are calculated by the 787 inventory model for the remote base and depot, and by a protection level for the MBK.**
    According to the supply chain specialists, an MBK is filled with the same calculation for calculating the remote base location but then adjusted for MBK values with a service level of 60%. The remote base and depot initial values are calculated by the same method as the Boeing 787 inventory model. The equations of this are in Section 2.2.
12. **Shipping days from depot to remote base, depot to MBK or remote base to MBK are constant and based on Hofman (2017) for to the remote base. From remote base or depot to the MBK is based on request data 2017-2018.**
    From the 2017 and 2018 request data received from S. Zeedijk all requests are grouped by airline and pivoted by the shipped from location, resulting in the average travel times between depot and remote base to an airline per airline. The transport data is filtered to eliminate all travel times above ten days due to many outliers in data. Outlier data make the travel times up to 5 times higher. Those outliers could consist of travel times between several months and a year in time.

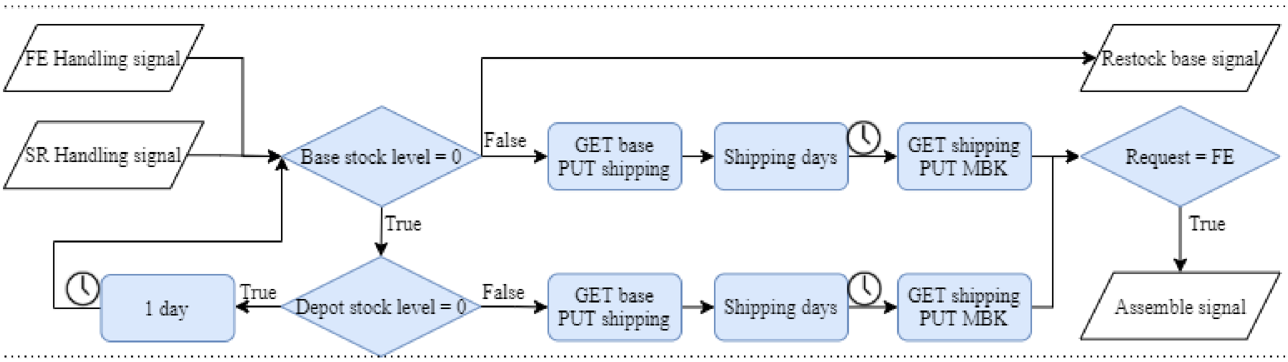


Figure 13: The MRO client request lane section from the total swimlane using as input a FE or SR handling signal. Locating stock and shipping it to the client. The process may activate a remote base restocking or assembly signal

Conceptual model and assumptions airline assembly lane

Figure 14 shows the spare arrival process. After arrival assembly is planned in the airline executes assembly. During assembly the unserviceable component is removed where the next step is to send it to repair. For this process there is only one assumption determining the assembly and waiting period.

13. **When a forward exchange request arrives it takes several days to plan and execute assembly.** The 6 month AeroExchange data proves component specific days to assembly a serviceable spare. This is the time between arrival of the shipping company and the removal date of the ariline.

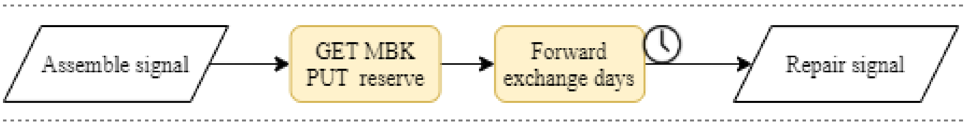


Figure 14: The MRO client request lane section from the total swimlane using as input a FE or SR handling signal. Locating stock and shipping it to the client. The process may activate a remote base restocking or assembly signal

Conceptual model and assumptions MRO restocking logistics lane

The last sub-process to desribe has the function of restocking the remote base and is seen in Figure 15. The remote base restock process is activated after the handling sub-process has sent a component from the remote base location. The last assumption of this chapter:

14. **The remote base is restocked at the end of every day when the depot stock is above 1 and takes place for the duration of a repair loop after a request is handled from the remote base according to G. la Fontaine** [5]

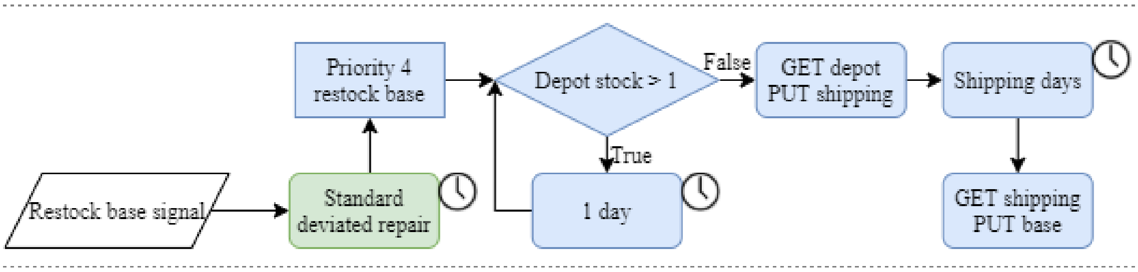


Figure 15: The MRO client request lane section from the total swimlane using as input a FE or SR handling signal. Locating stock and shipping it to the client. The process may activate a remote base restocking or assembly signal

Case study specific details

From the start of this chapter up to this section the information from interviews and data is used from KLM CS. The case study specific details are high. The model is applicable for MRO's contracting LRU availability in a global setting. This application is recognized by an extra continental warehouse which is not described in literature but present in other MRO's, according to the website of Lufthansa Technic, AAR, and Turkish Technic.
The commonalities with other MRO details are: first as described in literature the use of service levelsSherbrooke (2004), pooling strategy Kilpi (2007), and components with different essentiality ans stochastic propertiesXie & Yao (2016). Since other MRO's also use the continental warehouse and availability contracts the following can be concluded. The client airline will have different types of requests based on priority and the MBK stock level. Next to the airline request the remote base request for servicable components to restock the inventory.
Specific to KLM is the handling or operations of requests thus in what order and on what conditions to handle a request. Secondly the inventory control determines the number of spares in circulation and the remote base stock level. At last all used parameters are based on the Boeing 787 performance and KLM data.

4.3 Model verification, validation, and implementation

Verification

The warm-up period for the simulation is set at 1 year for simplification. This is executed by eliminating the first year results from all components 30 year run time. From the results, it can be seen that the number of removals in the first year is within the 95% interval and it takes a maximum of 6 months to reach the average in the repair shop. With that, the warm-up period is slightly longer than the 3 months warm-up period in Sprong (2019). Logical, because Sprong only used the VFSG with a relative short MTBUR.
For verification, the bullet points below were executed:

- **Input checks**, all input values are checked if logical by checking the type and if it is between certain values.
- **Balance checks**, for example, the starting stock levels for the depot and the remote base should be equal to the inventory control outcome, the total stock level should be equal every year and stock levels cannot be negative. Each function is programmed to create an error at the moment that the output value is incorrect.
- **Run time checking**, the three request types were followed for three total loops.

- **Result running checks**, the results from the different configurations show expected behavior.

Model validation

To validate the model seven parameters are chosen mainly on the available data from KLM CS. In upcoming sections the validation parameters are discussed. As a starting point the number of removals is validated, with that the number of starting events are correct. For checking the correct feedback of stock replenishment requests the percentage of stock replenishment request from total is validated. To check the feedback from the remote base location the percentage of component send from the remote base relative to the connected demand is checked. With the number of from remote base the shipping time is known, but for the response period also the reaction need to be validated. That is why the reaction time of a stock replenishment and a forward exchange request is validated. At last the service level is validated as the most prominent indicator and the MBK stock levels due to hard to extract data about the stock levels.

During the validation many inaccuracies occurred. Getting the validation correct is considered the most difficult activity in this research. The inaccuracies are described in each validation step.

Validation of the removals and percentage of stock replenishment requests

Table 6 shows that all except for one of the failure counts in the past 12 months (June) fit within the 95% interval of the result from the simulation. A forecast for 2020 is used as validation. The forecast uses the number of contracted aircraft in 2020 and divides it with the sum of the contracted aircraft in 2018 and 2019 divided by two. Before reaching such an accurate confidence interval the number of removals were extremely low due to the long KLM MTBUR. But when making use of the number of removals per it came closes. Since the service level is from the past 12 month with current components a forecast for 2020 is taken. For the VFSG, using the removal count results in an MTBUR with a difference of 0.5% compared to Sprong (2019) using a maximum likelihood analysis with censored data.

Two of the stock replenishment percentages are within the model's 95% interval. Actually most of them are within a 15% range of the actual % of stock replenishment request. Before the % stock replenishment requests was extremely off because a protection level of 90% which was used in agreement with the supply chain specialists. Because it was so far off the MBK stock levels were checked with a far lower ratio of stock replenishment requests, this is done in section 4.3.

Component code	**95% interval removals**	**Forecasted removals 2020**	**95% interval SR**	**Actual SR**
870056	75-82	78	61%-70%	67%
870087	77-84	85	44%-48%	40%
870180	105-114	108	39%-42%	29%
870261	44-50	44	17%-19%	56%
871010	233-245	239	39%-42%	55%
871754	64-70	67	16%-18%	55%
871915	76-82	78	8%-10%	25%
872318	528-544	539	51%-52%	52%
872517	224-236	225	29%-31%	50%
888003	169-180	176	21%-22%	40%

Table 6: Confidence T test with a sample of 29 years, the service level per component and %stock replenishment requests components per component. The green cells are values matching with the confidence interval.

Validation of the service level and shipped from location

Table 7 shows that none except for one of the service levels fit within the 95% confidence interval. Most of the values are less than 10% off from the model interval. A service level is reached by the simplified rule that every component send after the first day is a not reached service level request. The fourth column in Table 7 shows whether the component is sent within the first day from the past 6 months. This results in higher and lower service levels from which it can be concluded that service level fluctuate during the year in contrary with the small confidence interval from the model. KLM having a service level of 95% is not incorporated in the validation data because it is administered in another system. Adding the KLM service level would raise the average service of the actual data.

Four of the shipped from remote base percentages fit within the 95% confidence interval. Stock is sent from the depot to the remote base when there is more than 1 stock unit available in the depot at the end of the day. When the shipped from remote base is 100% then all request arriving at the base are shipped from the same base. The 20% is the same for each component because the average from all 787 components is taken from 2017-2018 request data. Since most spares are not stocked at a remote base this actual average parameter may be discussed as to low, from which the assumption might be incorrect.

component	**First day interval service level**	**Actual service level last 12 month**	**First day SL data last 6 month**	**Interval % shipped from base**	**Actual % total shipped from base**
870056	61%-74%	71%	48%	14%-21%	20%
870087	83%-92%	73%	86%	27%-34%	20%
870180	81%-92%	86%	65%	20%-28%	20%
870261	100%-100%	88%	75%	37%-45%	20%
871010	70%-80%	56%	63%	15%-20%	20%
871754	90%-97%	82%	67%	31%-37%	20%
871915	12%-20%	37%	25%	-1%-3%	20%
872318	100%-100%	79%	72%	18%-19%	20%
872517	94%-97%	66%	77%	22%-25%	20%
888003	71%-84%	82%	52%	0%-23%	20%

Table 7: Confidence T test with a sample of 29 years with service level per year average per component and %stock replenishment requests components per component. The green cells are values matching with the confidence interval.

Validation forward exchange and stock replenishment reaction time

Table 8 shows both the response time for a forward exchange and a stock replenishment. The model is in excess of buffer and is capable of shorter response times. In the data, outliers with long response time dominate the average, while for almost all requests, the median is 0 days. The actual response time originates from the average time per component between receiving a request and handing it over to the shipping company. The data is from request data in the past 6 months of August 2020.

Two notes have to made about the reaction periods. First was assumed that components were send deliberately on the last day of reaching the service level. This was based first on the technical limiting period (3 hours - 10 days) which are longer than the agreed periods for components (30 minutes - 3 days). It resulted in to high service levels and the response time data proofed a response time with a median of 0 days. In the first instance this also supported the choice for components with different essentiality, because the reaction times would have been different. The second note is that only two actual borrows have been taken place by all the components together in the past year (Table 33), which is low. In combination with a few very long actual response times which should have been borrowed according to the assumption of Munsters (2019) and Sprong (2019), to borrow a component when the response time is longer than 5 days. Extra interviews with the supply chain specialists turned out that components are only borrowed in extreme situations for example when a certain repair shop is not returning serviceable components due to transport problems or supply problems.

Component code	**Reaction time SR / component**	**Actual reaction time SR**	**Reaction time FE / component**	**Actual reaction time FE**
870056	1.7-3.2	6.8	1.6-3.3	4.8
870087	0.3-0.9	0.7	0.2-0.9	3.2
870180	0.3-1.3	2.3	0.2-0.7	0.5
870261	0	3.6	0	20.2
871010	0.8-1.5	3.4	0.5-1.2	1
871754	0-0.4	3.2	0-0.3	0
871915	17.6-30.3	24.2	12.9-23.3	2.6
872318	0-0	1.3	0-0	2.6
872517	0.1-0.2	3.5	0-0.1	2.2
888003	0.7-1.8	4.2	0.5-1.5	0.6

Table 8: Confidence T test with a sample of 29 years with % stock replenishment requests per year per component and response time components per year per component. The green cells are values matching with the confidence interval.

Validation of the MBK stock levels

Since the % of stock replenishment requests was off due to a received protection level of 90%, the actual MBK stock levels were requested. In Table 9, the sum is taken of the 2020 MBK stock values per component received from a supply chain specialist. The numbers 870087, 870261, and 888003 all have a high priority thus are kept for safety in high number in the MBK relative to other components code with less priority. To reduce time spend on implementing the correct values from the .txt file with MBK stock levels, the protection level is calibrated in steps of 5% to match as good as possible actual stock. In reality airlines differ between each other in protection level since each airline has a different stragey on components.

Component code	Model MBK total stock level	Actual total MBK stock	Stock split over # airliners
870056	10	4	2
870087	14	7	5
870180	14	21	8
870261	14	6	5
871010	15	9	7
871754	13	6	6
871915	14	3	3
872318	22	3	2
872517	14	10	7
888003	14	15	9

Table 9: The total stock level generated by the MBK stock assumption and the actual MBK stock level and in how many locations the stock is placed.

Model implementation

This subsection discusses the model implementation. After building a model there is a need to describe how to use it. The first paragraph describes the used hardware and software. Then there is the format for the input data and the interface with the user. At last a description of what the result data format is.
The simulation is run on an Elitebook 8560W with a 2.5 GHz Intel Core i5-2520M processor. The script is coded in Python version 3.7.6. In Python the Simpy environment is used for modeling the discrete time-step simulation. The pandas tool is used for uploading and downloading data from excel. For operational procedures ia made use of itertools, Random, numpy, math, and scipy.stats. All within combined in the Jupyter interface.

For setting up the model a folder need to be created including the simulation script from appendix E and the excel data tabs according to Table 10, Table 11, Table 12, Table 13, and Table 14. In the script itself is a more elaborate description of what the input values must be specifically. The three separate scripts in the appendix need to be positioned in a single .py file in the order of the appendix. The Script in the appendix has five functions:

1. Row 1-134 in data_import.py uploads data from the excel into Python matrices
2. Row 1-387 in processes.py codes the operational processes and prints per year the results in a matrix.
3. Row 1-251 in initiate_classes scripts the different scenario's and configurations.
4. Row 251-689 in initiate_classes initiates the locations, components and airline classes.
5. Row 689-793 in initiate_classes starts the events for each component and at the end of the script prints the results in an excel sheet

For the input data the excel format is used with four tabs. The first sheet must be named 'airliners_50'. Each row is a single airline.

A	B	C	D	E	F	G
Number	Contracted	Flight hours day	AMS	RMIA	RKL	SHA

Table 10: Columns in import data excel tab ’airliners_50’ part 1

H	I	J	K	L	M	N	O	P	Q
870056	870087	870180	870261	871010	871754	871915	872318	872517	888003

Table 11: Columns in import data excel tab ’airliners_50’ part 2, if components are changed also change the the headers with the according KLM family code.

The second excel tab must be named:’components_50’ and must have ten components in ten rows. The code name must correspond with the the code name used in the other tabs.

A	B	C	D	E	F	G	H	I	J	K
Name	Code name	Price	QPA	MTBR [hours]	Repair cost	ess	mel	sl	tat	12 month removals

Table 12: Columns in import data excel tab ’components_50’ part 1

M	N	O	P	Q	R	S	T	U	V
avg repair time	std.dev repair time	% forward exchange	waiting time FE	real stock diff	Real MTBR	EFF 2018	EFF 2019	EFF 2020	EFF KUL

Table 13: Columns in import data excel tab ’components_50’ part 2.

The third excel tab must be named:’locations’. The used name in ’Base’ must correspond with the base name used in the other tabs. The tab could have multiple rows but at least 1.

The fourth excel tab must be named:’parameters’ which in model contains a lot of data which is not used.

A	B	C
Base number	Base	Transport time depot to base

Table 14: Columns in import data excel tab ’locations’ existing of all the base locations.

Place in this tab on the first row column A ’total simulation time’ and in column N ’Random seed’.

When the model is set up and able to run it is possible to adjust the configurations. This is done by changing ’true’ into ’false’ or the other way around after ’run’ in the last coded row of the following experiments.

- Row 14-26 is the validation scenario
- Row 29-46 is the scenario with distributed actual repair time
- Row 49-67 is the scenario with a fixed repair time as real repair time
- Row 73-107 is a configuration testing the MTBUR from KLM

- Row 110-140 is configuration with three different protection levels of 30%, 60%, and 90%
- Row 146-170 is configuration with all aircraft contracted at the base, at the depot, or as in the current situation
- Row 182-197 is a configuration in which each component has [-2, -1, 0, 1, 2] additional spares per component code
- Row 199-223 is configuration adjusting the square root of the inventory control
- Row 225-264 is a configuration in which the inventory control service level is reduced to 70% and the MBK protection levels are equal.
- Row 267-291 is a scenario with a deterministic actual repair TAT

The next step is to run the model which is done by using run button in the Jupyter interface within Python. The clock time time is extremely dependent on the type of scenario and configuration. A single scenario with all fleet sizes takes approximately 60-90 minutes, but the exact time is not measured in the script. Running the total experimental plan in this report would take around 4-8 hours.

After running the data is printed in an excel sheet. Each row in the excel sheet represents the summary of the results from of one year for a particular component and experimental plan. An additional experiment could result in 5000 rows. To convert this to the graphs and confidence intervals this report uses google sheets. The google sheets are to messy to share or explain.

4.4 Conclusion

To conclude Chapter 4 the sub-research questions 'What is the current state of the order fulfillment system?' and 'What is the model structure?' have to be answered. In contrary with what is discussed in literature the order fulfillment system does not borrow when request reaction is to long and does not sent to external repair when repair capacity is limited. Secondly, instead of sending all the demand from a single location demand is send from multiple locations to the MBK.

The model structure can be best described as: the demand originates from contracted components based on the component's last failure. The physical structure is a triple echelon with a depot as global warehouse, a remote base as continental warehouse and an MBK as local (airport) warehouse (MBK). The local warehouse protection level determines the demand class. The demand class of a request, if a request is send from a remote base location, and the stochastic TAT determine how long a component is not available for a single loop. Influencing the service level and the response time.

A summary of actual performance and model performance see Chapter 6 Table 27.

5 Results of fixed TAT and fleet size increase

In this chapter, the experimental plan's sensitivity analysis will evaluate the indicators. Thereby it will answer the sub-question: 'What is the result of varying repair turn around times with fleet size?'. This chapter combines the validated simulation from Chapter 4 with the experiments from Chapter 2.

First, this chapter has an experimental plan divided into scenarios and configurations. Three scenarios and two configurations make up the main experimental plan. The scenarios have a constant fixed repair turnaround time, stochastic practical repair turnaround time, or a constant practical repair turnaround time. The two configurations are added to the constant fixed repair turnaround time scenarios and change the inventory model's service level and protection level. All the mentioned main scenarios and configurations have in common that they are divided into increasing fleet size scenarios. The first experiment also describes the number of supporting components in the closed-loop per contracted component equivalent to all the scenarios, thus not the configurations.

5.1 Experimental plan

Equal to the validation, each experiment has ten replications with a run time of ten years. In contrast with the validation, the simulation does not compensate experiment stock levels with the actual stock levels, which means that the inventory calculated stock level is the experiment's stock level. Secondly, the contracts are not normalized to 50 aircraft. The experiment difference between the validation depicted with a 0 in Table 15 and the experiment is evident. Secondly, the evaluation of configurations is on indicators from section 3.2 instead of validation parameters.

Experimental plan input scenarios

The simulation experimental plans input scenarios are based on the literature analysis in Chapter 2. All the mentioned main scenarios in this section and the next are varied over [50,100,150,200,250,300,350,400] aircraft. Starting at 50 aircraft to represent less contracted components. Up to 400 because KLM currently has around 200 contracts, and ITAA forecasts doubling demand in 10 years. Variable contracts are made possible by normalizing the contracts to 50 aircraft.

The rows in Table 15 mention the scenarios in the experimental plan. As discussed in Section 5.1 the 0 experiment represents comparison with the validation. Scenario 1 and 2b only differentiate fixed turnaround time in scenario one and practical turnaround time in 2b. Next, 2a is different from 2b because it uses a stochastic repair time instead of a constant. Table 16 in the following section describes the configurations chosen from the evaluated scenarios.

#	Experiment	Stock	Normalized	Stochastic TAT	Repair TAT	SL	MBK
0.	Validation	Actual	Not	Standard dev.	Actual	99%	[20%-80%]
1.	Fixed TAT	Calculated	[50-400]	Constant	Fixed	99%	[20%-80%]
2.	Actual TAT	Calculated	[50-400]	Standard dev.	Actual	99%	[20%-80%]
2.a	Actual constant TAT	Calculated	[50-400]	Constant	Actual	99%	[20%-80%]

Table 15: The partial experimental plan consisting of the validation and scenarios in the rows and the changed parameters in the columns

Experimental plan input configurations

The results of scenario 1 in Table 15 give a reason for an extra set of configurations. The service level's performance reaches 100% in the simulation of scenario one, from which it is hard to make conclusions. Thus as seen in Table 16 configuration 1a is present with a calculated 70% inventory to give more insights. Configuration 1a resulted in lower service levels for only a few components. To search for the cause of deterioration, configurations with varying QPA, removals, MTBUR, stock calculations, and forward exchange waiting days did not result in a clear relation. Making all the protection level equal to the average of 40% for each component gave a clear relation resulting in configuration 1b. Equal to the scenarios the configurations are varied over [50,100,150,200,250,300,350,400] aircraft.

#	Experiment	Stock	Normalized	Stochastic TAT	Repair TAT	SL	MBK
1.a	Theo. TAT, SL	Calculated	[50-400]	Constant	Fixed	70%	[20%-80%]
1.b	Theo. TAT, MBK, SL	Calculated	[50-400]	Constant	Fized	70%	40%

Table 16: The partial experimental plan consisting of the configurations in the rows and the changed parameters in the columns

5.2 Results from experimental plan

Experimental plan scenario 1. fixed repair turnaround time

This section first describes the capital employed in the scenarios named. After describing the capital employed in components, the results of scenario one are described. Figure 16 depicts the number of stock components as a percentage of the total components contracted. The rule of large numbers will result in more likely constant demand. Fewer extremes in demand make it possible to have relatively fewer stock components. For more elaborate explanation on pooling see Chapter 2.

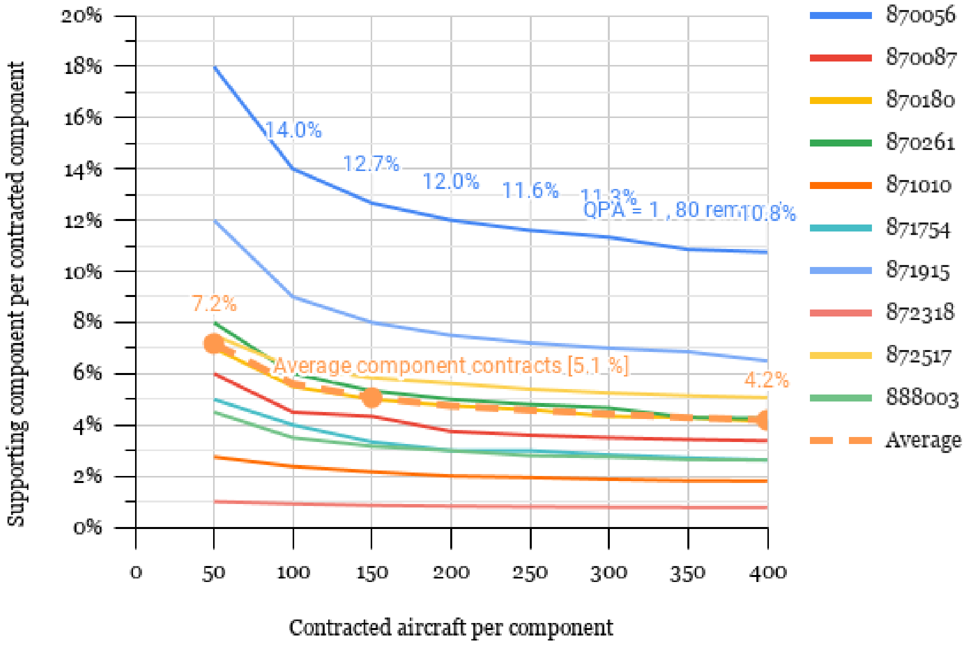


Figure 16: Number of spares in stock per contracted component overview of all aircraft main and sub scenario's including the validation configuration

The relative advantage of having fewer components in stock between 150 and 400 aircraft for the chosen ten components is 0.9%, from Figure 16. From which 150 aircraft is not the same but comparable with the number of aircraft in reality. Since the chosen ten components all have many removals per year, this is not comparable to the components' complete set. Even component 870056 (QPA = 1 and removals per year = 80) with the blue line is a conservative estimate regarding the components' complete set. 870056 describes an advantage of 0.7% between 150 and 200 aircraft. The improvement for less contracted components is greater due to fewer failures and has less value for the MRO. Table 17 depicts the capital employed and average MBK stock.

Configuration	**Capital in depot and remote base**	**MBK stock average[#]**
50	11,067,612 $	5.8
100	17,125,558 $	7.8
150	22,558,290 $	10.2
200	27,591,171 $	10.8
250	32,911,879 $	11.8
300	37,509,962 $	13.7
350	42,143,884 $	15.3
400	47,172,925 $	16

Table 17: Capital employed overview of all aircraft main and sub scenario's including the validation configuration

From here the evaluation per of the scenario simulation results start. Compared to the validation, there is no stock level compensation with reality, normalized contracts, and the simulation TAT is equal to the fixed TAT. This scenario gives insights into inventory deficiency and the goals' feasibility in an ideal situation with a met TAT.

Figure 17 shows the maximum and constant 100% service level for all the fleet size variation scenarios(this also accounts for all components), concluding that the current inventory management is buying enough components to reach it predetermined inventory threshold of 99% service level (which is higher than the 95% in the SLA). Secondly, results show that the inventory management as a minimal service level threshold is compliant with the simulation. However, unclear is the actual deficiency caused by a maximum simulation service level of 100%, the reason for an additional configuration with an inventory management service level of 70% instead of 99%. The lowered service level will give room to show the deficiency.

The inventory model calculates a service level of 99% for remote base location requests. On the contrary, the Figure shows a descending send from 63% to around 41%. It proves the difference between the inventory model and the simulation. The inventory model requests a component after handling from the remote base at the depot. The simulation repairs the same handled component before returning it to the remote base. Components being longer unavailable is not accounted for in the inventory model. Thus, the overall service level is reached, but the remote base's goal is unclear if meant to be 99%. A different experiment is recommended with a similar ship to remote base location policies in the inventory model and the simulation. This extra experiment is not executed in this thesis.

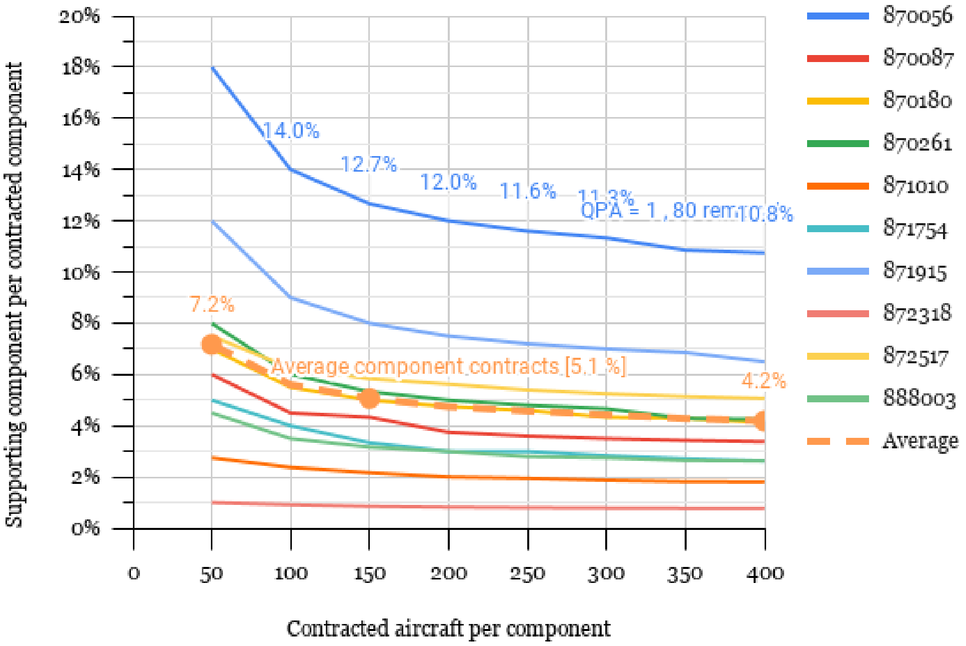


Figure 17: Components averaged performance of the order fulfillment in configuration 1.

Finally the last not discussed indicator, the response time of both types of requests and all fleet size variation scenarios is in Figure 18. From left to right are the number of aircraft in rising order; per aircraft scenario,

	Service level	Forward exchange %	Base shipped %
50	99.9%-99.6%	48.9% -47.3%	58.2%-57.1%
100	99.9%-99.7%	44.7%-44.2%	44.8%-43.8%
150	99.8%-99.6%	42.4%-41.6%	37.4%-36.6%
200	99.8%-99.6%	39.7%-39.1%	36.3%-35.6%
250	99.9%-99.8%	38.7%-38.2%	32.7%-32.1%
300	99.9%-99.7%	37.4%-36.8%	32%-31.4%
350	99.9%-99.6%	35.9%-35.5%	32.1%-31.6%
400	99.8%-99.7%	35.5%-35.1%	31.1%-30.7%

Table 18: Experimental plan scenario 1. fixed repair turnaround time with 95% confidence intervals of the service level, percentage forward exchange requests, and percentage shipped from remote base requests.

describing both request types' response time.

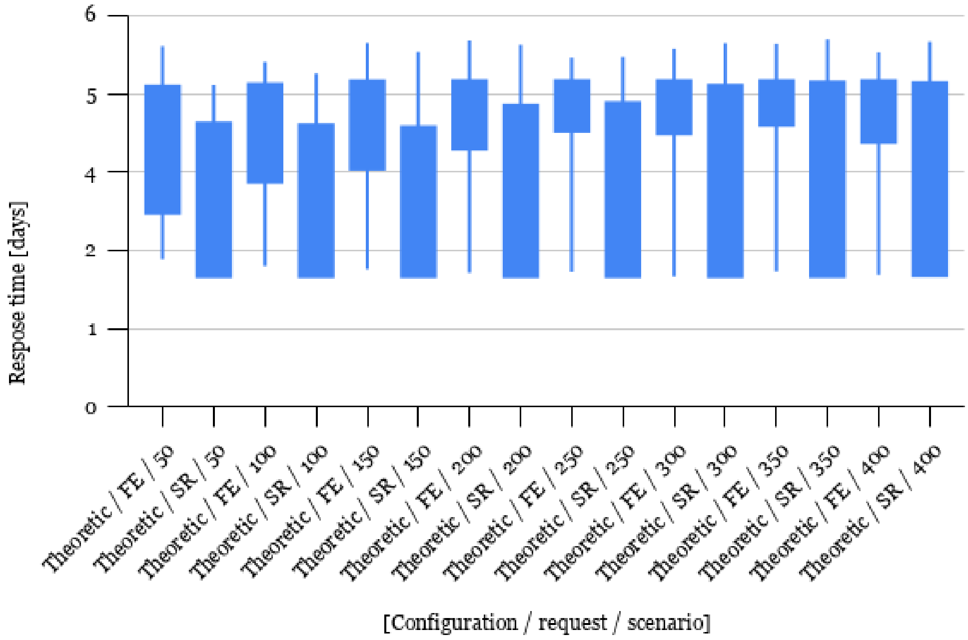


Figure 18: Experimental plan scenario 1. fixed repair turnaround time with minimum, second quartile, third quartile, and maximum response times on contracted aircraft, and type of requests.

In all the aircraft scenarios from Figure 25, 99.9% of the reaction times is one day. Thus the results in the Figure are nearly entirely influenced by the shipping time. There are two observations, the average rise and higher certainty of forward exchange requests. The second observation is the rising third quartile and lower certainty of the stock replenishment requests.

The reason for longer forward exchange response times originates from the MBK stock assumption, indirectly determining the number of forwarding exchange requests. On average, airlines connected to a remote base have one and a third more aircraft in their contract. With more contracted aircraft, the MBK stock assumption will stock an MK with more spares than an MBK at depot connected airlines. Resulting in 78% of all forward exchange request first approaching a depot and 22% a remote base.

Experimental plan configuration 1a. fixed repair turnaround time with 70% inventory

For an increased insight into the inventor model deficiency, the scenario in Section 5.2 advises an additional configuration with 70% inventory management service level. Figure 19 shows a simulation service level descend of 90% down to 85% and an average overestimated service level of 17%.

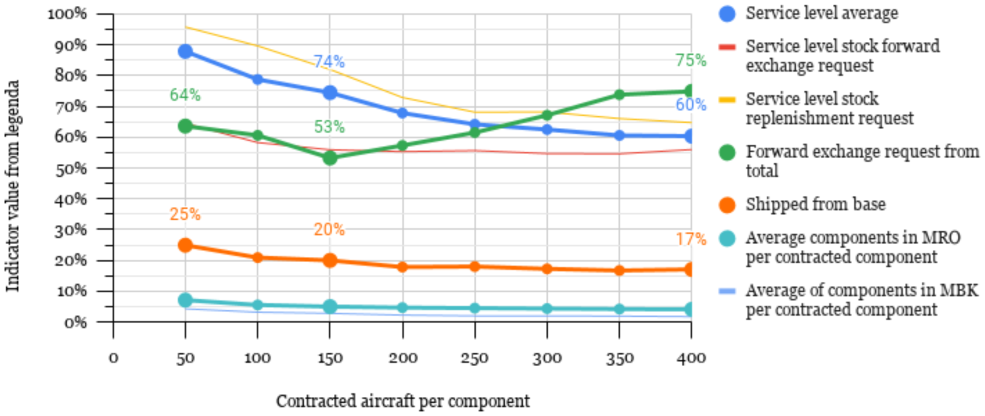


Figure 19: Components averaged performance of the order fulfillment in configuration 1.a

Service level	Forward exchange %	Base shipped %
90.6%-88.7%	32.2% -30.8%	59.2%-58.2%
89.5%-87.9%	25.1%-24.7%	46.5%-45.6%
86%-84.3%	21.3%-20.5%	40.6%-39.8%
87.2%-86%	21.3%-20.7%	39.5%-38.6%
88.9%-87.5%	21%-20.5%	36.9%-35.9%
87.8%-86.1%	20.1%-19.6%	37%-36.1%
86.4%-84.8%	19.3%-18.7%	38.3%-37.5%
85.8%-84.4%	18.7%-18.4%	37.8%-37.1%

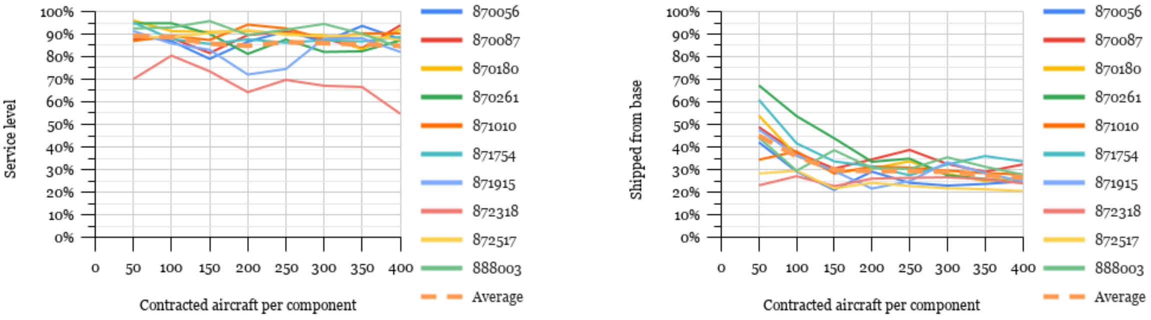
Table 19: Experimental plan scenario 1.a fixed repair turnaround time and 70% inventory service level depicting 95% confidence intervals of the service level, percentage forward exchange requests, and percentage shipped from remote base requests.

In the fixed turnaround scenario, the simulation overestimates the inventory model by 17%. In contrast, the inventory model does not account for the extra days needed for a forward exchange request. Thus sending more spares from the depot instead of the remote base will result in more availability to enable a 17% service level improvement and counteract the longer forward exchange requests.

Besides a higher service level, the graph shows an unexpected low forward exchange service level. The forward exchange service level is lower than the average service level. As already mentioned in the previous section:

78% of all forward exchange requests first approached a depot and 22% a remote base. A request approaching a remote base will have a second chance of succeeding at the depot with a higher service level outcome. A request approaching a depot will only have a single change of succeeding and have a shipment time of 5 days. Those requests will result in a longer response time (more shipment days) and a lower service level for a forward exchange request.

The shipped from remote base parameter depicts a steep descent from 25% to 19% between 50 and 150 aircraft. Up to 400 aircraft, it will stay relatively constant. The negative decreasing slope is a result of less availability in components caused by increased fleet size. Figure 20 describes the service level's unsolved behaviors and the shipped from remote base percentage.



(a) Service level for different request types with averages from 20

(b) MBK stock levels at different contracted aircraft levels with averages from 21

Figure 20: Experimental plan scenario 1.a fixed repair turnaround time and 70% inventory service level depicting service level and shipped from remote base per component as a function of the number of contracted aircraft.

The shipped from a remote base performance per component is in Figure 20b. The components with a high number of removal per year show directly from the beginning a constant shipped from the remote base performance. In contrast, the components with a low number of removals start with a high shipped from remote base performance and later converge to the average service level dotted line. At last, the cause of a descending service level.

The descending line is an average of all the service levels found per chosen component in Figure 20a. Within this Figure, components 872318 and 871915 both depict irregular or descending behavior. The found cause for this is in the next section's configuration, making the MBK stock levels equal.

Experimental plan configuration 1b. fixed repair turnaround time with 70% inventory and equal protection levels

In Section 5.2 the descend of a simulation service levels of 5% when adding aircraft is noted and linked to components 872318 and 871915. Tests with different configurations based on varying the: QPA, forward exchange waiting days, removals, MTBUR, and stock calculation scaling ways did not give any relation. Except for changing the protection level of the MBK. The choice for testing configurations is on outstanding component parameters.

The MBK protection level is a unique percentage per component. In the simulation, this percentage varies between 20-80%; it dependents on each component. Chosen for all components in this configuration is a service

	870056	870087	870180	870261	871010	871754	871915	872318	872517	888003
50	87.2%-93%	86.3%-91.5%	94.3%-97.5%	93.5%-97.2%	86.1%-90.5%	94.3%-98.1%	86.1%-92.7%	66.3%-74.6%	85.7%-91.6%	91.3%-95%
100	82.7%-88.8%	86.2%-91.1%	88.8%-94.3%	92.7%-97.1%	89%-93.3%	84.1%-90.6%	85.7%-92.4%	74.9%-80.8%	87.7%-92.9%	88.7%-92.4%
150	76.5%-84.6%	83.1%-87.6%	86.5%-91.6%	87.7%-93.4%	86.8%-91.9%	78.4%-85.3%	79.9%-87.1%	64.1%-70.2%	84.9%-89.8%	93.9%-96.5%
200	84.8%-89.3%	85.2%-90.5%	88.4%-92.5%	81.4%-88.3%	90.6%-94.2%	85.8%-90%	71.4%-77.8%	63%-67.5%	90.8%-94.5%	89%-92.9%
250	80.3%-86.4%	91.2%-94.6%	91.8%-95.1%	89.4%-94%	90.3%-94.3%	83.5%-89%	73.3%-78.9%	68.1%-72.8%	89.4%-94.1%	89.8%-93.1%
300	86.7%-92.7%	84.7%-89.8%	89.3%-93.6%	82%-88.7%	89.5%-93%	83.9%-89.3%	86.1%-91.6%	64.7%-69.2%	86.2%-91.5%	93.8%-96.5%
350	87.4%-92.4%	82.3%-87.7%	86.8%-91.6%	79%-85.8%	86.9%-91.3%	87%-90.7%	81.9%-87.9%	63.8%-67.7%	87.3%-91.4%	89.1%-92.7%
400	88.9%-93.4%	90.8%-94.5%	87.3%-91.5%	83.3%-88.2%	87%-90.8%	87.2%-91.6%	78%-83.3%	53.1%-56.6%	84.4%-88.9%	85.2%-89%

Table 20: Experimental plan scenario 1.a fixed repair turnaround time and 70% inventory service level depicting the service level 95% confidence interval of the service level per component belonging to Figure 20a.

	870056	870087	870180	870261	871010	871754	871915	872318	872517	888003
50	27.6%-32.4%	32.3%-36.8%	35.8%-40.3%	43.7%-49.9%	22.8%-25.4%	40.5%-45.8%	31.4%-38.3%	14.5%-18%	18.8%-20.8%	29.3%-32.7%
100	19.3%-21.4%	24.8%-27.7%	24.1%-26.4%	35.2%-40.2%	25.5%-27.5%	27.5%-31.5%	23.6%-28.1%	17.7%-19.9%	19.6%-21.5%	19.7%-21.4%
150	13.8%-15.9%	20%-22.3%	19.2%-21%	29%-32.6%	18.9%-20.4%	22%-25.2%	19.4%-22.3%	15%-16.5%	14.5%-15.6%	26%-27.5%
200	19.4%-21.1%	23%-25.4%	20.5%-21.9%	21.9%-25.9%	21.2%-22.4%	20.8%-23%	14%-16.2%	17.7%-18.2%	16.3%-17.1%	20.7%-22%
250	16%-17.7%	26%-27.7%	22.8%-24%	23.2%-25.5%	20.9%-21.8%	18.4%-20.1%	16.3%-18.5%	17.9%-18.5%	15.4%-16%	20.5%-21.5%
300	15.4%-16.5%	21.6%-23.6%	19%-20.2%	18%-20.8%	20.1%-20.8%	21.4%-23.6%	21.9%-24.5%	18.1%-18.6%	14.7%-15.3%	24%-25.2%
350	15.9%-16.9%	19.3%-21.2%	16.7%-17.9%	17.1%-19.1%	19.3%-20%	24.1%-26.1%	18.9%-20.9%	17.5%-18%	14.5%-15%	21.2%-22.1%
400	16.7%-17.5%	21.8%-23%	17.7%-18.7%	16%-17.5%	19%-19.5%	22.6%-24.1%	16.3%-18%	16.4%-16.7%	14%-14.4%	18.9%-19.5%

Table 21: Experimental plan scenario 1.a fixed repair turnaround time and 70% inventory service level depicting the service level 95% confidence interval of the send from location per component belonging to Figure 20b.

level of 40% equal to the average protection level.

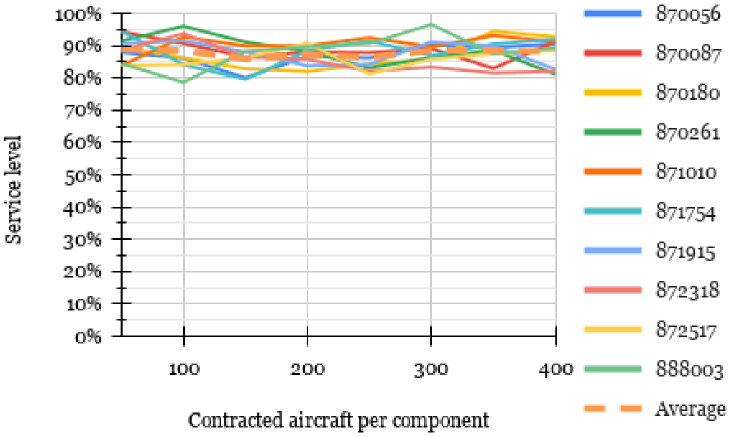


Figure 21: Service level average per component for configuration 1.b

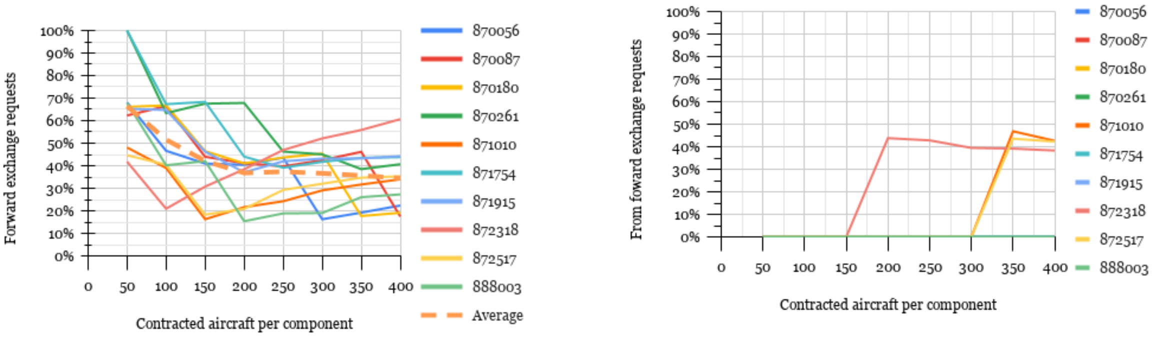
	870056	870087	870180	870261	871010	871754	871915	872318	872517	888003
50	86.2%-92.8%	89.2%-94.9%	91.1%-96%	90.3%-96.1%	82.9%-89%	91.3%-97.5%	90.8%-94.6%	86.3%-91.8%	83.9%-90.8%	83.7%-89.3%
100	81%-87%	88.9%-93.4%	81.2%-87.7%	93.3%-97.3%	91.1%-94.7%	84%-89.7%	88.9%-93.5%	92.3%-95.1%	82.4%-88.3%	79.5%-85.5%
150	78.4%-85.5%	81.8%-87.7%	82.6%-88.8%	86.8%-92.8%	85.9%-90.4%	78.9%-85.6%	87.1%-92.2%	83.3%-88%	84.5%-89.4%	87.8%-91.8%
200	83.9%-88.8%	86%-90.8%	84.6%-89.8%	85.2%-90.4%	89.4%-92.9%	83.6%-89.4%	78.8%-85.1%	83.6%-88.2%	90%-93.9%	88.8%-92.6%
250	82.5%-88.4%	90.3%-93.5%	83.1%-88.6%	86.5%-92%	88.2%-92%	85.8%-90.7%	81.8%-87.6%	83.1%-87.4%	86.7%-91.6%	91%-93.8%
300	88.4%-92%	88.1%-92.2%	82.1%-87.3%	83.1%-89.7%	89%-92.2%	85.1%-89.9%	90.2%-94.7%	80.9%-85.6%	85.9%-90.8%	94.1%-96.8%
350	88.2%-92.7%	82.5%-88%	87.9%-92.2%	80.9%-87.7%	90%-93.5%	88.7%-93.1%	84.9%-90.3%	79.5%-84.1%	86.1%-91%	87.8%-92.1%
400	88.8%-93.9%	90.3%-94.4%	90%-94.1%	79.1%-87%	87.3%-91.2%	87.3%-91.4%	84.3%-88.9%	74.5%-79%	86.5%-91%	86.7%-91.2%

Table 22: Experimental plan scenario 1.b fixed repair turnaround time and 70% inventory service level and equal protection levels depicting the 95% confidence interval of the service level per component belonging to Figure 20a

.

With the MK protection level set to 40%, Figure 21 shows a rising service level from 86% to 90%. It creates for the specific two components with a protection level before lower than 40% more forward exchange request and the other way around for components with a protection level higher than 40%. There is the same total amount of forward exchange requests, but the ratio of requests within each component is balanced. With a better balance, the MBK stock will work as an order fulfillment buffer.

The low performing components in Figure 21 do not show erratic behavior on the right side. To find the cause of the erratic less line, get back to the forward exchange assumption.



(a) MBK stock levels at different contracted aircraft levels

(b) Requests originated from a second components in MBK stock.

Figure 22: Experiment with 70% service level calculation no repair deviation perfect repair, fleet size variation and same MBK protection levels

Each request generated with an MBK level between zero and threshold is a forward exchange request. It will only apply to MBK stock levels higher than one. Since Figure 22a does not show an erratic line on the right side for the previous weak performing components, those could have an MBK value higher than one. Figure 22b shows the number of forward exchange requests originated from an MBK level between zero and threshold. If taken into account, this could result in even a more significant improvement in the service level. For the other configurations, this effect only occurs at 250 aircraft and over.

Experimental plan scenario 2 practical stochastic repair turnaround

Compared to the validation, there is no stock level compensation with reality, and there are normalized contracts. However, compared to the fixed TAT, the simulation TAT is the same as the practical repair period and deviation. This scenario gives insights into the impact of the current repair loop situation.
Figure 23 shows the same metrics as in Figure 17 but now with the actual TAT fleet size variation configuration. When adding more aircraft, the Figure depicts a negative declining service level. As mentioned before, this is due to the reduction of spares availability. The difference with the fixed TAT configuration is the longer and stochastic TAT. To find out which of both additions have the most impact. The next section's configuration has a constant actual TAT to show the difference. From here onwards, it describes the current configuration.

The overall performance Figure shows forward exchange and stock replenishment service levels nearing the average. The average service level is descending and nearing an equilibrium with more aircraft. The shipped from the remote base is nearing 17%, approximately equal to the ideal repair duration results.
The rising forward exchange request is a cause for the decreasing service level. The amount of a forward exchange request is 12% higher at 150 aircraft than the ideal repair duration configuration. Figure 24a depicts the service level per component.

	870056	870087	870180	870261	871010	871754	871915	872318	872517	888003
50	62.7%-68.1%	61.4%-67.8%	61.1%-67.1%	100%	42.8%-47.1%	100%	61.5%-66.6%	40.9%-43.1%	43.5%-46.8%	64.4%-68.2%
100	45.3%-48.4%	65.1%-68.4%	65.3%-68.4%	62.2%-67.9%	38.2%-40.7%	64.7%-68.2%	64.8%-68.7%	20.8%-22.6%	39.5%-42.8%	38.3%-40.6%
150	38.3%-41.8%	44.6%-48.1%	44.4%-47.3%	62.7%-67.9%	15.9%-17.6%	65.5%-69.1%	42.6%-46.3%	30.8%-33.6%	16.5%-18.6%	40.2%-42.1%
200	39.3%-41.4%	37.6%-40.3%	39%-41.6%	63.9%-67.7%	20.4%-22.1%	43%-46.5%	38%-41.5%	37.4%-39.8%	20.1%-21.8%	14.9%-16.7%
250	42%-44.7%	38.5%-41.1%	41.4%-44%	42.9%-47%	23.9%-25.7%	36.5%-39.1%	37.9%-40.8%	43.1%-45.9%	25.3%-27.3%	17.8%-19.3%
300	15.5%-17%	40.4%-42.4%	43.2%-45.7%	43.8%-47.2%	27.7%-29.5%	39.2%-41.7%	39.2%-41.7%	49.3%-53.5%	29.3%-31.8%	20%-21.4%
350	18.2%-19.9%	42.4%-45.1%	16.9%-18.7%	36.3%-39.6%	30.1%-32%	39.4%-42.4%	40.7%-43.1%	54.7%-60.4%	32.2%-35.5%	24.3%-26.2%
400	19.8%-22%	16.1%-17.7%	17.7%-19.9%	38.1%-41.7%	34.1%-36.1%	41.4%-44%	42.9%-44.7%	64.4%-71.9%	35.4%-37.7%	26.9%-28.8%

Table 23: Experimental plan scenario 1.b fixed repair turnaround time and 70% inventory service level and equal protection levels depicting the 95% confidence interval of the percentage of forward exchange requests per component belonging to Figure 22a

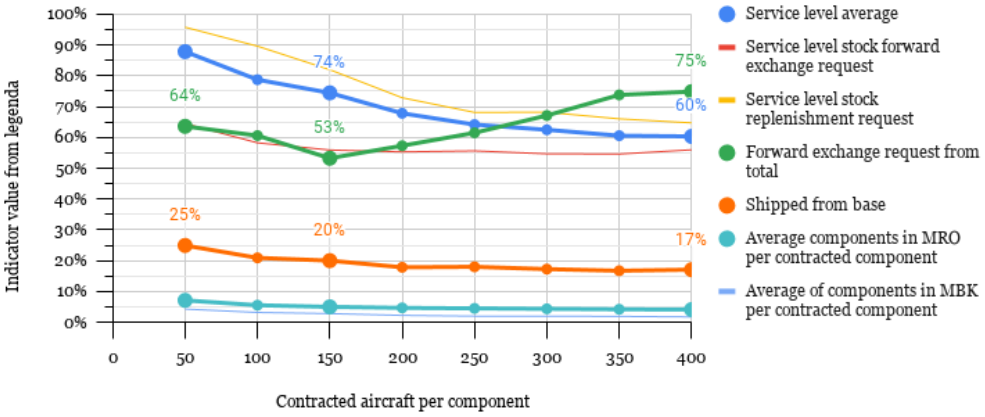
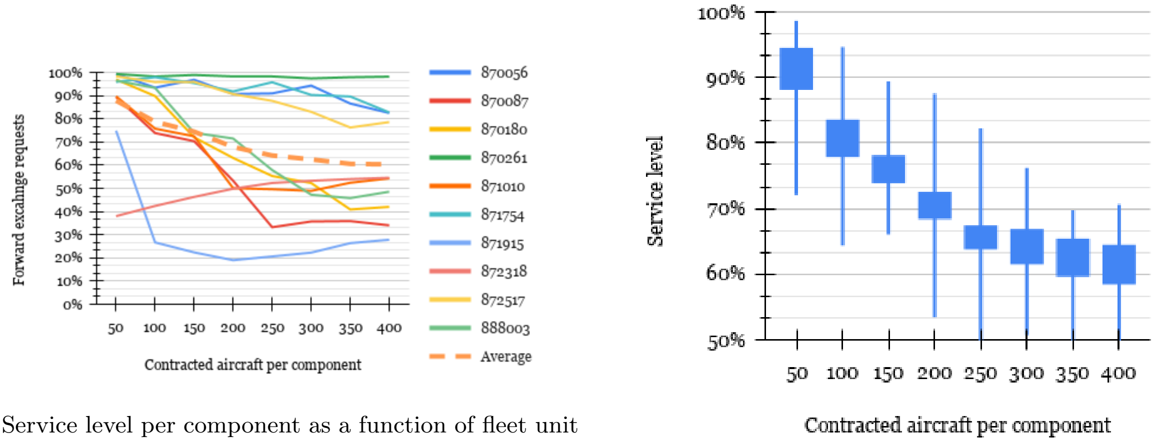


Figure 23: Components averaged performance of the order fulfillment in configuration 2.

	Service level	Forward exchange %	Base shipped %
50	90%-88.5%	90%-88.5%	90%-88.5%
100	85.6%-83.5%	85.6%-83.5%	85.6%-83.5%
150	81.3%-79.1%	81.3%-79.1%	81.3%-79.1%
200	74.4%-72.1%	74.4%-72.1%	74.4%-72.1%
250	70.3%-67.7%	70.3%-67.7%	70.3%-67.7%
300	67.8%-66.1%	67.8%-66.1%	67.8%-66.1%
350	65.3%-63.4%	65.3%-63.4%	65.3%-63.4%
400	64.9%-63.1%	64.9%-63.1%	64.9%-63.1%

Table 24: Experimental plan scenario 2 practical stochastic repair turnaround depicting the 95% confidence interval of the indicators in the top column



(a) Service level per component as a function of fleet unit size

(b) Average service level minimum, second quartile, third quartile, and maximum in different fleet unit sizes

Figure 24: Experimental plan scenario 2 practical stochastic repair turnaround depicting the service level average per component and total minimum, second quartile, third quartile, and maximum

	870056	870087	870180	870261	871010	871754	871915	872318	872517	888003
50	98.8%-99.9%	86.2%-92.6%	95.1%-98.8%	98.7%-100%	87%-92.5%	92.9%-98.9%	67.4%-82.4%	33.7%-42.3%	97.3%-99.3%	94.5%-98.4%
100	90.8%-96.1%	69.4%-78.3%	86.5%-93.1%	97.1%-99.4%	71.1%-80.4%	96.4%-99.5%	19.7%-33.7%	41.5%-43.4%	94.2%-97.5%	91.2%-95.4%
150	95.1%-98.6%	65.6%-75.1%	65.9%-78.1%	98.2%-99.6%	68.1%-76.8%	93.7%-97.1%	19%-25.8%	45.6%-47.1%	94.2%-97.7%	69.7%-78.2%
200	87.7%-93.7%	46.9%-59.7%	56.5%-69.7%	97.3%-99.3%	46.1%-54%	89.3%-94.4%	17.8%-20.2%	49.3%-50.4%	88.1%-93.2%	65.5%-77.5%
250	88.4%-93.5%	30.1%-36.4%	49.3%-61.3%	97.1%-99.5%	46.7%-52.6%	93.9%-97.6%	19.8%-21.5%	51.8%-52.9%	85%-90.5%	53.5%-62.3%
300	92.4%-96.2%	32.2%-39.2%	47.3%-57.3%	96%-98.8%	47.6%-50.3%	87.2%-93.4%	21.5%-23.1%	52.9%-53.7%	79.5%-86.5%	43.7%-50.9%
350	83.3%-89.9%	34.2%-37.5%	37.2%-44.6%	96.7%-99.2%	51.8%-53%	87.1%-92.1%	25.6%-27.2%	53.7%-54.3%	72.1%-80.4%	43.3%-48.4%
400	77.9%-87.1%	32.2%-36%	38%-46%	97.2%-99.2%	53.6%-55%	78.7%-86.8%	26.9%-28.8%	54.2%-55%	75.6%-81.6%	46.1%-51.1%

Table 25: Repair with stochastic values 95% confidence intervals Experimental plan scenario 2 practical stochastic repair turnaround depicting the 95% confidence interval of the service level per component

Figure 24a shows a descending service level average and a per component service level with different characteristics. Take the green straight line of 870261, which keeps a constant service level. Alternatively, 872318, 871915, and 870087 all started with a steep decline and end with slight growth. The remaining component codes all have a descending service level. The lines' erratic behaviors are related to additional MBK stock for each aircraft scenario, lowering the forward exchange requests.

The component with the highest amount of a forward exchange request does have the lowest service level. For example, 872318, 871915, and 870180 all have 100% forward exchange requests. The high amount of a forward exchange request is caused by the order fulfillment deterioration due to the longer TAT. The deterioration will cause longer reaction times for stock replenishment and result in more forward exchange requests, further deteriorating the system.

Take a look at Figure 24b depicting the stochastic behavior of the average service level. It first shows a more narrow interval of the second and third quartile from left to right, while the interval is getting wider at the far right. For the minimum and maximum values, it is the other way round. It is unknown why the service level diverges at high fleet unit sizes.

Figure 26 depicts the response time per request per aircraft scenario. Wherein the response time of a forward exchange request is relatively stable and slightly getting more accurate. On the other hand, the stock replenishment requests are rising. The graph only shows an average, and it is interesting to analyze on a component level.

Thus Figure 25 shows the response time per request type, per component, and aircraft scenario. The strange jump in stock replenishment response time at 150 aircraft from Figure 26 has an origin at two components: 871919 and 872318. Those two only produce forward exchange requests. On the components level, each request is more accurate when adding aircraft. Components identified with a long stock replenishment response time

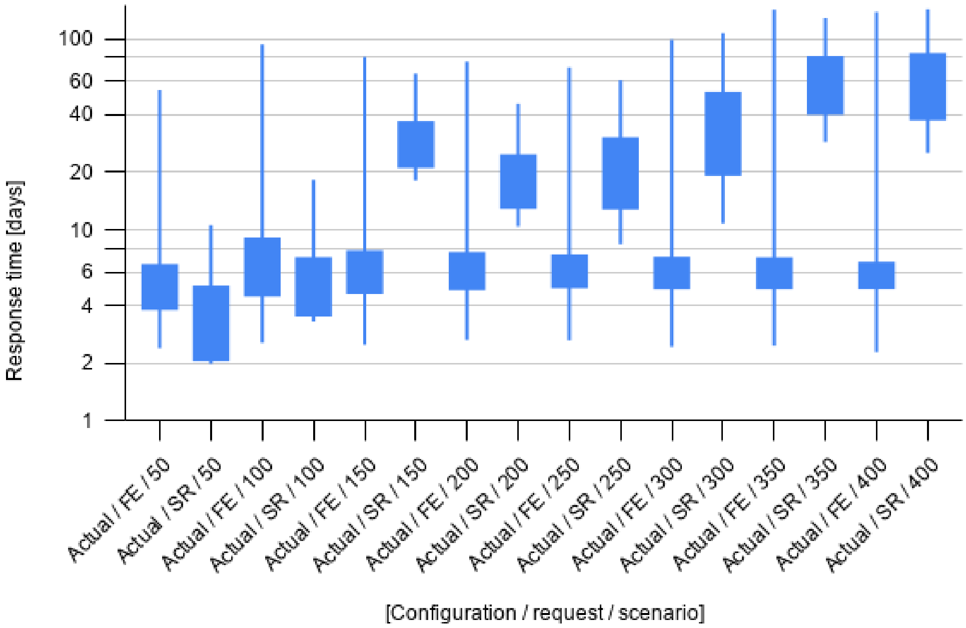


Figure 25: Minimum, second quartile, third quartile, and maximum for a request based on all components' averages in a configuration.

also have long TAT combined with a high amount of components contracted.

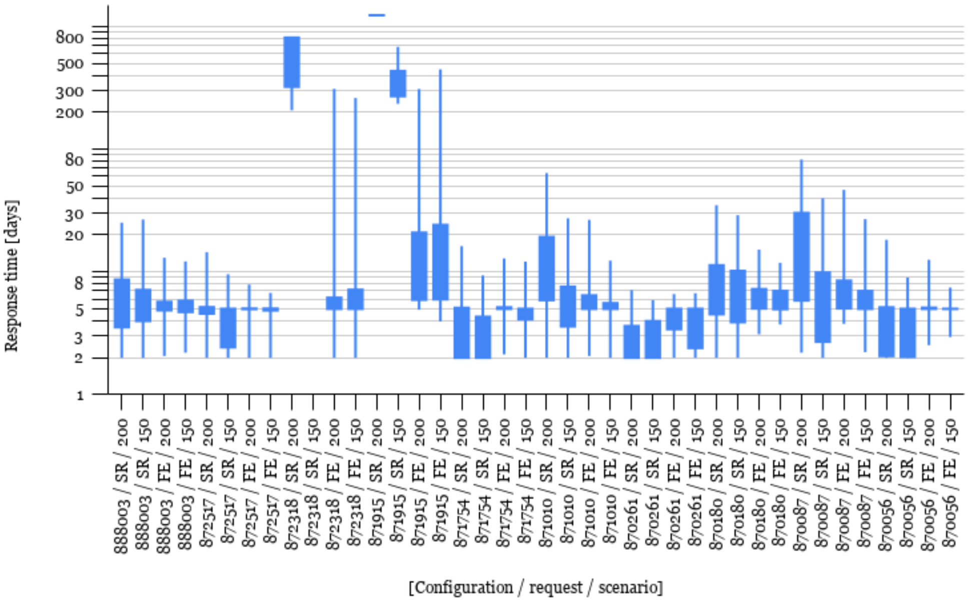


Figure 26: Experimental plan scenario 2 practical stochastic repair turnaround depicting the minimum, second quartile, third quartile and maximum of the response time fleet unit 150-200 and both request types per component

Experimental plan scenario 2.a practical constant repair turnaround

The actual and fixed configurations differentiate on a longer and stochastic TAT. Adding an actual TAT configuration with constant duration will point out the influence of stochasticity. Figure 27 shows a low impact of stochasticity on the actual TAT performance. The service level difference at 150 aircraft is an average improvement of 3%.

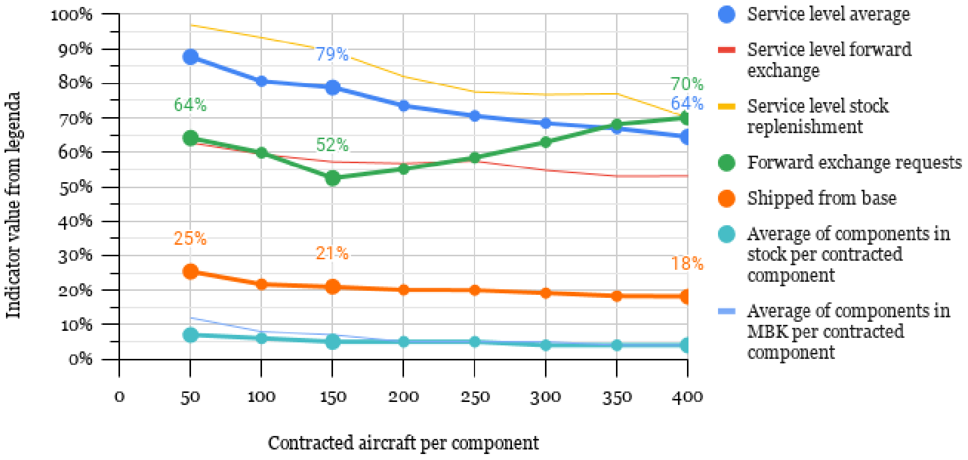


Figure 27: Components averaged performance of the order fulfillment in configuration 2.a

	Service level	Forward exchange %	Base shipped %
50	90.8%-89.2%	36.5% -34.8%	58.5%-57.3%
100	88.2%-86.8%	30.6%-29.9%	49.3%-48%
150	86.5%-84.7%	28.3%-27.3%	41.8%-40.7%
200	80.7%-78.5%	24.9%-24%	43.2%-41.9%
250	77.1%-75.3%	23.6%-22.9%	44.8%-43.3%
300	74.4%-72.8%	22%-21.3%	47.8%-46.1%
350	72.3%-70.8%	20.7%-20.2%	52.7%-50.9%
400	69.6%-67.6%	20.6%-20%	53.5%-51.7%

Table 26: Experimental plan scenario 2.a practical constant repair turnaround depicting the indicators per component for Figure 27

5.3 Conclusion

To answer 'What is the result of varying repair turn around times with fleet size?'. The scenarios originate from the strategy not to reach TAT and increase fleet size. Processing the result discovered a side effect of the protection level assumption; more forward exchange requests are requested at the depot than at a remote base. Since a remote base has a second chance on succeeding at the depot results in higher service levels for stock replenishment requests.

First what if the TAT is reached and constant. This remote base scenario has a 100% service levels and in the higher fleet size regions a 40% forward exchange equal to the average protection level of 40%. The low shipped from based is due to the operational procedure of allocated spares for a location instead of ordering.
A constant fixed TAT with 70% inventory control proofs the importance of MBK spares on component level. Secondly, the send from remote base ratio is around 20% lower in the full range of fleet units, whereas a higher number for forward exchange requests raises after 150 fleet units.
The impact of extending fleet units with current repair period is a worse performance in the higher fleet size regions. The send from remote base location has the same but the percentage of forward exchanges rises when adding more fleet units. Making the repair time constant only slightly improves the service level with 3%
For an overview of the results see Chapter 6 Table 28.

6 Conclusion

6.1 Conclusion

To answer the main question stated in the introduction as: '**What aspects make line replaceable unit maintenance, repair and overhaul not reach service levels in global commercial availability contracts?**', the sub-questions are answered with the following sub-conclusions:

What is state of the art in LRU literature?
Not reaching the service level is a general LRU MRO problem. Literature within KLM is focused on the repair loop instead of on order fulfillment. MRO's apply two strategies: increasing fleet size Kilpi et al. (2009) and secondly a fixed TAT from Munsters (2019) and Driessen (2018). For the model structure it is recommended by Sprong (2019) and Munsters (2019) to use all order fulfillment inventory locations. Whereas the main used indicator is service level.

What is the current state of the order fulfillment system?
According to interviews and data, in contrary with literature borrows are not used in standard operations the same with sending to external repair if full capacity is reached. The main base kit stock level influences the demand class of a request. Where a stock replenishment increases spares availability and a forward exchange reduces spare availability. To be able showing results in a single graph the feedback in the system is measured in the percentage of components send from base and the percentage of forward exchange requests. The performance of the current situation is in Table 27

What is the discrete time-step simulation order fulfillment model?
The model structure can be best described as: the demand originates from contracted components based on the component's last failure (instead of assembly). The physical structure is a triple echelon with a depot as global warehouse, a remote base as continental warehouse and an MBK as local (airport) warehouse. The stock level at the local warehouse determines the demand class. The demand class in combination with if a request is send from a remote base location and the stochastic repair TAT determine how long a component is not available for a single loop.
The eight validations for each of the ten components with a confidence intervals is more elaborate than only validation on failures from Sprong (2019) or the only validation on borrows from Munsters (2019) without confidence intervals. Cornelisse (2018) and Hofman (2017) are respectively validated by experts and not. Table 27 depicts a summary of the validation; the summary must be interpreted within mind summarizing the ten supporting components.

Validation	**Actual**	**Simulation**
Removals	164	164
Service level	72%	80%
From Base	20%	24%
Stock replenishment requests	33%	43%
Forward exchange response	3.8	2.4
Stock replenishment response	1.2 5.3	1.2
MBK stock	10.2	9.3

Table 27: Averages of all validation values

What is the result of varying repair turn around times with expansion?
The results are in table 28.

Scenario/ configuration	Indicator	50	150	400
Theo TAT	Service level	100	100	100
	Forward exchange	63%	41%	40%
	Base shipped	44%	38%	32%
	supporting spares	7%	5%	4%
Theo TAT 70%	Service level	90%	85%	85%
	Forward exchange	65%	46%	53%
	Base shipped	25%	19%	18%
	supporting spares	6%	4%	3%
Actual TAT	Service level	88%	74%	60%
	Forward exchange	64%	53%	75%
	Base shipped	25%	20%	17%
	supporting spares	7%	5%	4%

Table 28: Average indicator average results the theoretic TAT and practical TAT scenario and the theoretic TAT 70% service level configuration.

What is recommended to improve the service level?
To improve service levels KLM should reduce the fleet size to below 50 units and slightly reduce repair TAT or reach the fixed TAT. If the MRO does not want to reduce the service level it should not increase fleet size without TAT reduction or spares should be bought on equal supporting spares per component percentages as the latest added contract. Besides the strategies inventory control has opportunities to improve validity and with that decreasing risks and control overhead. The last point is to research opportunities besides safety stock to buffer variability with for example TAT pull or forecasting. For a more elaborate answer on this sub-question go to Section 6.3.

Main question: What aspects make line replaceable unit maintenance, repair and overhaul not reach service levels in global commercial availability contracts?

This report results in five main aspects why service levels are not reached. Most important are the strategies, the use of a fixed TAT in inventory control results in a reduction of spare part availability. The strategy to grow results in less buffer (more vulnerability) for variability, factors and variability not taken into consideration in inventory control will reduce spare part availability. Both strategies reduce the service level but also offer competitive contract pricing due to lower WAPP values. A reduction down to a service level of 60% at a fleet size of 400 with actual TAT.

Secondly, inventory control does not allow additional control (besides buying circulating spares) when the TAT is not reached. Thus there are no insights in how service levels are reached when repair TAT is not as expected and how to improve order fulfillment performance by adjusting operational procedures.

Thirdly, a lack of documentation on component level operational procedures and processes in order fulfillment cause a simplified inventory control model. The simplification has reduced validity with inadequate circulation spares or stock levels.

Feedback mechanisms in the system can improve and deteriorate the system. Longer response periods on stock replenishment requests will cause lower MBK stock levels and with that more component availability reducing forward exchange requests. Longer response periods on remote base restocking requests cause less stock send from a remote base result in more available components due to less transshipment but also less availability due to longer response period to restock the MBK from the depot.

At last, reducing variability in the system changes the effectiveness of operational procedures. When request demand becomes more constant requests with higher priority will be fulfilled faster and more often than with lower priority. The results show that a forward exchange request has on average the same response period while a stock replenishment request raises from the same average from 5 days at a fleet size of 50 up to 40 days at a fleet size of up to 400 within a the actual TAT scenario.

6.2 Discussion

Research and methods

- *What is the scientific relevance of this research?*
  The scientific relevance is giving the triple echelon structure insights in the area of global LRU availability contract MROs. To the best of the authors' knowledge, the structure is not described in either KLM thesis and scientific literature. KLM does calculate the stock for MBK locations but from the viewpoint of a single warehouse. Those insights overreach operational procedures, a discrete time-step simulation computational model, and component availability feedback.

Simulation conceptual model and validation

- *What would be the impact on the results of implementing a repair capacity?*
  Sherbrooke (2004) and Hofman (2017) use a constant repair period, but if the repair would have a capacity, then it could also benefit from the rule of large numbers. This effect is way smaller because 60% of the 4000 different LRUs go to four internal repair shops. Thus, the effect of large numbers is reaching its limit Driessen (2018). If repairs were more efficient for more fleet units, it would result in a actual TAT scenario with a higher service level, and there would be a rise in service levels in the fixed TAT scenario.
- *What is the influence of using the mean time between: removal-failure or removal-removal?*

The removal-failure period is equal to the time a component is functional in the aircraft, while removal-removal also adds maintenance responsiveness. The responsiveness of maintenance is dependent on the availability of components. Thus improving spares availability while using removal-removal data results in more failures per year. This simulation model does not give insights into this feedback.

- *What is the impact of not validating the sample data to the population?*
  The used sample data in this report is not validated against the population. Thus there are no insights if the sample data is a good representation. For example, component 872318 has a repair time of 100 days with a variation of 100 days; a validation could have shown if the longer repair time was only of a shorts notice. It is recommended to use validation of sample data to the population for order fulfillment decision making.

- *What is the impact of the system boundaries?*
  Excluded in the system are the loan desk and a more detailed repair loop. Both have a substantial influence on the system. The loan desk will add extra demand to the system without altering the number of components. A detailed repair loop gives insights into the repair capacity in duration. Wherein extra repair demand, workforce availability, and second indenture level components will influence the repair time.

Practical applicability

- *Are the most client value-adding problems applied?*
  In practice, the client is most influenced by the service level and response time of AOG requests and components with TBR requests. An AOG request is comparable with a forward exchange request but on extreme priority and requested from a random location in the world. The random location around the world makes it challenging to model. Components with fixed flight hours between removals and requested three months in advance are essential but only applicable to a few components. Because it only represents a minority of components and the operational procedure is different, it is not implemented.

- *Why should an MRO implement the recommendations while there is no business case available?*
  This research lacks a business-case. In the service industry it is difficult to connect direct cost to a service. However, with better asset management and supply chain insights, the MRO can avoid risks and reduce costs. With improved control and insights, a new contract will less likely be underestimated, better asset management will reduce firefighting of supply chain personnel, and better service for the client will make them less likely to terminate the contract or use the penalty clause. If cost are dedicated to not reaching a service level it should be exponentially divided across the difference in service level. Thus the service level improvement from 94% to 95% should have higher cost than from 80% to 81%

6.3 Recommendations

In this section, the starting point of this research is reevaluated with the recommendations from the conclusion. This will answer the fifth sub-question: 'What is recommended to improve the service level?'.

From the conclusion it can be recommended not to increase fleet size without repair TAT reduction or spares should be bought on equal supporting spares per component percentages. Secondly, new contracts should have enough mbk stock for a balance between FE and SR requests or the contract value of airlines with no mbk stock should compensate for more priority requests and less buffer. To increase the service level or response period for forward exchange or AOG requests inventory control could assign two demand classes. Forward exchange requests send from a remote base in the first day, and stock replenishment requests send from the depot in loger response period. Next to that, restocking and circulation stock threshold in the inventory model should align with operations.

Back to the beginning

This research started with the document from Chun et al. (2019). The relations from the causal loop diagram in the report of Chun are all discrete and direct relations except for "pressure on OEM-MRO". If KLM wants to have more insights into "pressure on OEM-MRO" or external repair shops, A recommended first step is to analyze the percentage of work at the OEM-MRO or external shop which is occupied by KLM. After that, define a strategy in consult with the OEM-MRO and external repair to discuss the possibilities to carry out a business relation strategy that is rewarding and mutually beneficial.

For future research into the service of LRU's it is recommended to consult the following works: **Driessen (2018)**, Zijm et al. (2015), Driessen et al. (2015), and Parada Puig & Basten (2015). The mentioned papers are not discussed in this report and almost all consist of a case study at KLM and give a clear literature overview with recommendations(recommendations foremost in Driessen (2018)), even specific recommendations for KLM. The improved order fulfillment model has all the inventory locations as recommended in the research of Sprong (2019) in predictive maintenance and Munsters (2019) in optimal loan desk; thus both research could be executed again with more detail. Next to that the handling of AOG Visintin et al. (2012) and constant time between removal requests by changing operational procedures or forecasting is an interesting topic.
Currently in actual performance is mainly measures in service levels. It is recommended to research for more indicators to measure performance. Since this report points out the complexity of the system while client value is not only dependent on service levels. Examples of this will follow in this Chapter.

Variability and forecasting opportunities

When applying the theory from Hopp & Spearman (2011), variability in the system at an aircraft MRO is currently buffered in inventory. It is interesting to research the reduction of LRU's when the variability of demand is buffered by pulling components from TAT, instead of the TAT currently pushes spares to inventory. Secondly, reduce variability in demand by more strategic loans, longer reactions period, and borrows. Think of daily flexible TAT capacity, TAT period, transshipment periods and allocation to reduce variability in TAT or even up to a point that it can have daily constant output of components buffering variability in returned unserviceable components. But even special adjusted operational procedures to flight schedules and assembly planning per client. The availability of components one repair period in the future is the number of returned components today.

Inventory control

It is recommended to improve inventory control because the model within KLM is relatively basic compared to the proven inventory control already applied to LRU's in published literature. Improvements are the following additions all applicable to KLM CS: local repairs Shekarian (2020), VARI-METRIC shipping between inventory locations Sherbrooke (2004), multiple demand classes Kleijn & Dekker (1999), indenture levels Driessen (2018), lateral transhipment Hofman (2017), AOG demand class Visintin et al. (2012), and variable repair period with workforce planning. Specific to KLM demand from the loan desk and additional repair (repairs not for contracted partners) is advised to implement in the calculation. As a follow-up, the new inventory control model can be used within KLM with the following two functions:

- *Function 1: (circulation stock model) calculate the number of components in circulation*
  It is meant to evaluate the investment of an additional availability or loan desk client as if the fixed TAT is reached. The input is a new client with a specific transshipment time, service level, and MBK stock. The output is the number of circulating spares.
- *Function 2: (transition model) determine the best performing control strategy*
  This function is for the time being that a the fixed repair TAT is not reached. Part of the configuration is the circulation stock calculated in function one. The input is per request type (forward exchange, stock replenishment, base restocking, and loan desk), a percentage of components send from a location (depot, remote base, MBK, or borrow), and in what reaction time and transshipment time. With as output the service level and response period. All the different parameters will have different cost.

Improving the order fulfillment simulation

The validation recommends for better simulation accuracy to use actual MBK stock values instead of an assumption. Secondly the repair period in the simulation is i.i.d. which is not the case in reality, a depended variable based repair time will result in wider confidence intervals which is more realistic. Implement component specific operations since now all the component show the same simulation behaviors but in reality that is not the case.

Authors note: *Collecting, processing, and linking data points is time-consuming at KLM. Note that KLM has its code number and separate part serial numbers sometimes wrongly written over in data-sets. There is a lot of data available but all stored at different: places, people, size(megabytes), and formats. The proposed models would need the following separated complete data files for multiple years: request data clients, request data KLM, repair data, inventory model, MBK circulation stock, TMAS, TMOS, shipment data, borrow data, loand desk requests, and fulfilled loan desk. Followed up by searching components with consistent data for validation. The latest data sets do not show where a component is sent from, the period between finished repair and returning inventory, and what components are shipped only that a package is shipped. Take little steps to improve.*

References

Abrahamson, G. (1979). Terminology for model credibility. SIMULATION. Retrieved from `http://journals.sagepub.com/` doi: 10.1177/003754977903200304

Aisyati, A., Jauhari, W. A., & Rosyidi, C. N. (2013). Determination Inventory Level for Aircraft Spare Parts Using Continuous Review Model. International Journal of Business Research and Management (IJBRM), 4(1), 1–12.

Bosdijk, N. (2019). Aligning logistics with MRO to improve spare parts availability (Master Thesis). TU Delft.

Chen, J., Gusikhin, O., Finkenstaedt, W., & Liu, Y.-N. (2019, jan). Maintenance, Repair, and Operations Parts Inventory Management in the Era of Industry 4.0. IFAC-PapersOnLine, 52(13), 171–176. Retrieved from `https://www.sciencedirect.com/science/article/pii/S2405896319311218` doi: 10.1016/J.IFACOL.2019.11.171

Chun, J.-p., Klundert, T., & Roosmale Nepveu, M. (2019). A digital twin webpage. KLM(June). Retrieved from `www.test.de`

Cornelisse, E. (2018). Modelling business process redesign strategies for improved reverse logistics in an aircraft component (Master Thesis). TU Delft.

Date, D. E., Type, D., Stage, A. P., Type, F. A., Source, O. A., Journal, I., ... Journal, I. (2020). A literature review on closed loop supply chains. Springer Nature Switseland AG 2020, 58(January), 1285–1301. doi: 10.1007/978-3-030-31343-2

Driessen, M. (2018). Integrated Capacity Planning and Inventory Control for Repairable Spare Parts (1st editio ed.). Eindhoven: Eindhoven unversity of technology.

Driessen, M., Arts, J., Van Houtum, G. J., Rustenburg, J. W., & Huisman, B. (2015). Maintenance spare parts planning and control: A framework for control and agenda for future research. Production Planning and Control, 26(5), 407–426. doi: 10.1080/09537287.2014.907586

Ertogral, K., & Öztürk, F. S. (2019, jan). An integrated production scheduling and workforce capacity planning model for the maintenance and repair operations in airline industry. Computers & Industrial Engineering, 127, 832–840. Retrieved from `https://www.sciencedirect.com/science/article/pii/S036083521830562X` doi: 10.1016/J.CIE.2018.11.022

Govindan, K., & Soleimani, H. (2017, jan). A review of reverse logistics and closed-loop supply chains: a Journal of Cleaner Production focus. Journal of Cleaner Production, 142, 371–384. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0959652616301986` doi: 10.1016/J.JCLEPRO.2016.03.126

Gross, D. (1980). On the ample service assumption of Palm's theorem in inventory modeling (Scientific report). The george washington university.

Haak, D. (2019). Designing Repair Processes by Introducing a Return Quality Control Model (Unpublished doctoral dissertation). TU Delft, Delft.

Hockers, J. (2016). Title Page from (Unpublished doctoral dissertation). TU Delft.

Hoed van den, M. (2018). Design of an Advanced Task Planning Model for Aircraft Heavy Maintenance (Master Thesis). TU Delft.

Hofman, I. (2017). The effect of a multi-echelon model on the inventory invest-ment for a supply chain of air-craft rotables (Tech. Rep.). Retrieved from `http://repository.tudelft.nl/`.

Hoogenboom, T. (2019). In-Plant Transport System Design An Integrated Approach for an Aircraft Component MRO Facil (Doctoral dissertation, TU Delft). Retrieved from `https://www.tudelft.nl/3me/`

Hopp, W. J., & Spearman, M. L. (2011). Factory Physics (Second edi ed.). Waveland Press.

Kapoor, R., & Ambekar, S. (2015). Closed Loop Repairable Parts Inventory System: A Literature Review. Indian Journal of Economics and Business, 14(1), 31–47.

Kilpi, J. (2007). Valuation of Rotable Spare Parts (No. December 2007). Helsinki.

Kilpi, J., Töyli, J., & Vepsäläinen, A. (2009, feb). Cooperative strategies for the availability service of repairable aircraft components. International Journal of Production Economics, 117(2), 360–370. doi: 10.1016/j.ijpe.2008.12.001

Kilpi, J., & Vepsäläinen, A. P. (2004). Pooling of spare components between airlines. Journal of Air Transport Management, 10(2), 137–146. doi: 10.1016/j.jairtraman.2003.09.001

Kleijn, M. J., & Dekker, R. (1999). An Overview of Inventory Systems with Several Demand Classes. (September), 253–265. doi: 10.1007/978-3-642-58568-5_13

Meijs, P. (2016). Reducing the Turnaround Time of In House Repairs of Aircraft Engine MRO Services (Master Thesis). TU Delft.

Muhaxheri, D. (2010). Framework for Evaluation of Strategies for Pooling of Repairable Spare Parts (Master Thesis). Lund University.

Munsters, D. G. C. (2019). Evaluating performance improvement strategies in a closed-loop supply chain (Master Thesis). TU Delft.

Palma-Mendoza, J. A. (2014, oct). Analytical hierarchy process and SCOR model to support supply chain re-design. International Journal of Information Management, 34(5), 634–638. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0268401214000644` doi: 10.1016/J.IJINFOMGT.2014.06.002

Palma-Mendoza, J. A., & Neailey, K. (2015). A business process re-design methodology to support supply chain integration: Application in an Airline MRO supply chain. International Journal of Information Management, 35(5), 620–631. Retrieved from `http://dx.doi.org/10.1016/j.ijinfomgt.2015.03.002` doi: 10.1016/j.ijinfomgt.2015.03.002

Panteleev, V., Kamaev, V., & Kizim, A. (2014). Developing a Model of Equipment Maintenance and Repair Process at Service Repair Company Using Agent-based Approach. Procedia Technology, 16, 1072–1079. Retrieved from `http://dx.doi.org/10.1016/j.protcy.2014.10.121` doi: 10.1016/j.protcy.2014.10.121

Papadopoulou, O. (2015). Improving the maintenance process at Component Services, KLM Engineering and Maintenance (Master Thesis). TU Delft.

Parada Puig, J. E., & Basten, R. J. (2015). Defining line replaceable units. European Journal of Operational Research, 247(1), 310–320. doi: 10.1016/j.ejor.2015.05.065

Rezaei Somarin, A., Chen, S., Asian, S., & Wang, D. Z. (2017). A heuristic stock allocation rule for repairable service parts. International Journal of Production Economics, 184(November), 131–140. doi: 10.1016/j.ijpe.2016.11.013

Rozenberg, A. (2016). Designing a comprehensive framework to analyze and improve engine MRO processes from an integra
(Master Thesis). TU Delft.

Shekarian, E. (2020). A review of factors affecting closed-loop supply chain models. doi: 10.1016/j.jclepro.2019.119823

Sherbrooke, C. (2004). OPTIMAL INVENTORY MODELING OF SYSTEMS Multi-Echelon Techniques OPERATIONS R

Soeters, G. (2017). Operational excellence by continuous improvement of the integral engine MRO chain (Master Thesis). TU Delft.

Spaan, T. (2018). Data-driven Decision Support for Component Flow Turnaround Time Reduction in Aircraft Maintenance
(Tech. Rep.). Retrieved from `http://repository.tudelft.nl`.

Sprong, J. (2019). Prognostics-driven supply chain optimization in commercial aviation (Unpublished doctoral dissertation). TU Delft.

Stammes, S. (2018). Quality of the integral aircraft engine MRO chain Engineering & Maintenance Engine Services (Master Thesis). TU Delft.

Sulyman, P. D. (2020). Short term predictive demand model based on transport times for the reverse supply chain A case st
(Tech. Rep.). Retrieved from `http://repository.tudelft.nl/`.

Tako, A. A., & Robinson, S. (2012, mar). The application of discrete event simulation and system dynamics in the logistics and supply chain context. Decision Support Systems, 52(4), 802–815. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0167923611002211` doi: 10.1016/J.DSS.2011.11.015

Thijssens, M. (2019). Reliability Modelling for Aircraft Component Availability Management.

Tokgoz, S., Bulkan, S., Zaim, D., & Delen, N. T. (2017). Journal of Quality in Maintenance Engineering Article information : Modeling airline MRO operations using a systems dynamics approach : a case study of Turkish Airlines. Journal of Quality in Maintenance Engineering.

Tracht, K., von der Hagen, F., & Schneider, D. (2013, jan). Applied Repairable-item Inventory Modeling in the Aviation Industry. Procedia CIRP, 11, 334–339. Retrieved from `https://www.sciencedirect.com/science/article/pii/S2212827113004939` doi: 10.1016/J.PROCIR.2013.07.020

van Rijssel, R. E. (2016). Lowering the Turnaround time for Aircraft component MRO services - A case study at KLM Engi
(Master Thesis). TU Delft.

van Welsenes, J. M. C. (2017). Modelling the impact of controlling E2E turnaround time constraints on the aircraft MRO co
(Master Thesis). TU Delft.

Visintin, F., Porcelli, I., & Ghini, A. (2012, may). Applying discrete event simulation to the design of a service delivery system in the aerospace industry: A case study. In Ifac proceedings volumes (ifac-papersonline) (Vol. 14, pp. 1678–1683). Elsevier. doi: 10.3182/20120523-3-RO-2023.00339

Vlamings, B. (2020). A cost - benefit assessment of condition - based maintenance for different prognostic performance level
(Master Thesis). Delft Unicersity of Technology.

Wibowo, A., & Tjahjono, B. (2017, jan). Designing Contracts for Aero-engine MRO Service Providers: Models and Simulation. Procedia CIRP, 59, 246–251. Retrieved from `https://www.sciencedirect.com/science/article/pii/S2212827116311167` doi: 10.1016/J.PROCIR.2016.10.124

Wibowo, A., Tjahjono, B., & Tomiyama, T. (2016, jan). Towards an Integrated Decision Making Framework for Aero Engine MRO Contract Management in the Productisation Context. Procedia CIRP, 47, 24–29. Retrieved from `https://www.sciencedirect.com/science/article/pii/S2212827116300191` doi: 10.1016/J.PROCIR.2016.03.049

Xie, C. ., & Yao, H. (2016). Study of inventory strategies for recyclable parts of aviation material with limited inventory capa (Tech. Rep.). Shanghai: Shanghai University of engineering science.

Zijm, H., Klumpp, M., Clausen, U., & Ten Hompel, M. (2015). Logistics and supply chain innovation: Bridging the gap between theory and practice. Logistics and Supply Chain Innovation: Bridging the Gap between Theory and Practice(March 2018), 1–312. doi: 10.1007/978-3-319-22288-2

A Research paper

https://repository.tudelft.nl/ 00 (2021) 1–6

https://www.afiklmem.com/en

Simulating order fulfillment for aircraft line replaceable unit strategy: A case study at KLM component services

H.J.M. Kalf[a,*]

[a] *Technical University of Delft the Netherlands*

Abstract

Aircraft LRU (Line Replaceable Unit) MROs (Maintenance, Repair Organization) struggle to meet service levels. To stay competitive, literature describes two strategies; increasing contracted fleet size and not reaching projected repair turn around times. Besides these two strategies, literature does not describe the use of triple echelon order fulfillment for global contracts but it is recommended Sprong (2019) Munsters (2019). This research aims to find relations between the service level and the two strategies from the perspective of expansion rules and operations. A discrete time-step simulation is proposed because the systems component availability changes over time and because of the stochastic behavior of repair time and component failure. The found relations are firstly that send from base operational procedures and secondly that mbk stock influence the theoretic repair time performance. The second relation is that when adding more fleet units (apply a bigger pool), the current performance will deteriorate. *Copyright* © *H.J.M. Kalf*

Keywords: Aircraft MRO, Order fulfillment, triple-echelon, closed-loop supply chain, repairable components, availability contract, discrete time step simulation,

1. Introduction

The MRO signs an availability contract with airlines to swap unserviceable (failed) components for serviceable (functioning) components while paying a flight hour tarif Wibowo et al. (2016). The MRO promises to reach a predetermined service level in the SLA (Service level agreement) for the upcoming 10-15 years Muhaxheri (2010). Worldwide local warehouses as well as short reaction time is needed to ensure the short delivery periods from the SLA.
The problem is that MROs struggle to comply with the agreed service levels Tokgoz et al. (2017), while the number of available contracts is rising faster than the growth of airline transport Wibowo and Tjahjono (2017), Palma-Mendoza (2014). The more complex availability contracts have to deal with high stochastic repair Sulyman (2019) and failures Bosdijk (2019) and a dynamic over time changing environment.
This research aims to give insights into the relationship between the structure, applied strategies, and the service level by performing a discrete-time step simulation. Figure 1 gives an overview of the model. The MROs apply two strategies to stay competitive, first by adding availability contracts Kilpi (2007)and secondly by a fixed repair turn around time Munsters (2019) and Driessen (2018). As 70-80% of capital is in components Tracht et al. (2013), both strategies try to reduce components.

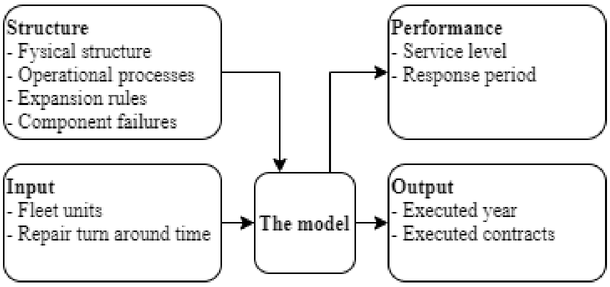


Figure 1: System according to the Delft simulation Approach

2. Literature

A literature review is conducted within KLM for a TU Delft thesis research between 2016-2020 with an MRO or a logistics topic and within component services. After that, the published literature is reviewed. Within both literature types, the focus is on the repair loop. Which is not unusual since the availability contracts using warehouses around the world is relatively new

---

[*] Author and correspondence.

and growing.
Within KLM the following researchers use the order fulfillment from a total perspective in their thesis, the first two with a simulation and the next two with an analytical model. Sprong (2019) uses a single circulating component single echelon constant repair loop discrete time-step simulation to find the optimal business case for predictive maintenance. Munsters (2019) has a failure generating multi-component single echelon multiple processes stochastic repair loop discrete-time step simulation structure to evaluate optimal loan desk factors. The first analytical model is from Hofman (2017). Hofman researches the impact of METRIC and VARI-METRIC double echelon integer programming optimization applied to KLM for differences in results. The other analytical model is Vlamings (2020), who is looking into predictive maintenance with a linear programming optimization to find optimal parameters.
Within published literature the following researchers use the order fulfillment from a total perspective. Sherbrooke (2004) updates his theories and handles a case for the airline industry with an integer programming optimization in a double echelon with optional local repair and indenture level configuration. Visintin et al. (2012) uses a discrete-time step simulation to evaluate how AOG requests are best handled from an OEM perspective. Tracht et al. (2013) describes a double echelon repairable planning model, with service levels focused on the repair loop. Palma-Mendoza (2014) executes business process redesign as a case study for a single echelon component requests situation. All the scientific research use generated failures instead of the goal to use circulating components in this research.
The MROs strategies are two folded; Munsters (2019) cites from an interview with T. Knappers: ”KLM tries to hold as little inventory as possible. The stock sizing calculations are based upon the TAT of a component. At this moment, the TAT in reality of spares is too long.” Secondly, Kilpi (2007) proofs the component reduction advantage of using a pool of components with different fleet sizes in different strategy configurations.
The gap of this research is recommended by Sprong (2019) and Munsters (2019) to build a discrete-time step simulation with all the inventory locations for better accuracy. To the best of the authors' knowledge, a triple echelon model for LRUs is not available in the literature. As found on the website of Lufthansa Technic, AAR, KLM E&M, and Turkish Technic the triple echelon structure is in use. Next to the triple echelon, Kilpi (2007) and Sherbrooke (2004) do not mention the disadvantages or feedback of a reduced buffer by more pooling.

3. Analysis

For this research, a system data analysis and interviews were conducted at KLM. Outside the system boundaries are the market forces on prices and the loan desk. The entire process starts at a single failure. At the same time, all the other processes are connected to a single failure. Figure 2 shows the general overview of the process for a single failure.
The Figure starts with a failure (generated by circulating stock) communicated by the pilot to the airline logistics department. Airline logistics define the type of request according to the component availability in their main base kit (MBK). No stock in the MBK results in a forward exchange request; every other situation will result in a stock replenishment request. A forward exchange request is sent to MRO logistics for handling from the depot or the remote base. The depot is the global stock location to supply the remote base on each continent. After handling the forward exchange from a location, the airline maintenance receives the component and assembles it in the aircraft. When the assembly is done, the unserviceable component is sent to repair. The other type of request is a stock replenishment request. This type of request is directly sent to repair while at the same time handling at the MRO logistics happens. The assembly is executed before requesting thus is not executed in the system. The MRO repair will make a serviceable component from an unserviceable component after which it restocks the base. If a component is shipped from a base location during handling, it is restocked if the depot has enough spares after the duration of a repair.

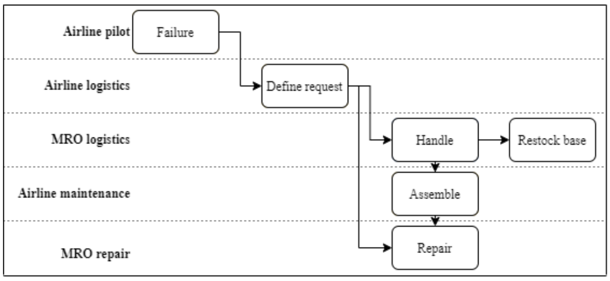


Figure 2: Simplified swimlane of all the overarching processes. Each block has it's own sub-processes.

To evaluate the system, there is a need for new indicators. The most important indicator, the service level, mentioned in every report, occupies the top row of Table 1. The new indicators are percentage of forward exchange requests and send from the remote base location. Both give insights into the system's feedback and are easily compared because all use the same unit, percentage. Forward exchange requests appear when the MBK is empty and take more availability (because transport is longer) of components. An empty MBK is the result of a longer stock replenishment response time. The second feedback is with high availability at the depot. The remote base is more restocked, and this reduces component availability due to less shipment time. Moreover, higher stock at a remote base results in shorter response time for all client requests.

Table 1: Request performance indicators

Indicator	Unit
Service level	[%]
Forward exchange requests	[%]
Send from base	[%]

4. Model

The conceptual model is partly described in the previous section analysis. This section adds assumptions to the conceptual model and checks the models' validity. The model makes use of the circulating stock failures from Sprong (2019) and depot request handling from a combination of Visintin et al. (2012) and Tracht et al. (2013). The mentioned expansion rules are based on Sherbrooke (2004) applied by Hofman (2017) but calculated as separate locations. Additional are the following assumptions:

1. Financial parameters are influenced by the components .
2. Airlines differentiate on flight hours, location, contacted components, and number of aircraft.
3. Removals per component are cumulative exponential distributed based on the total number of failures divided by the fleet size.
4. A total of ten components are selected, of which two components of each MEL and two from each essentiality with the highest capital employed and with at least 20 failures in the past six months.
5. The simulation is started for every component and airline times aircraft and QPA (Quantity Per Aircraft) with a day as time step.
6. Airliners will place a forward exchange request when MBK stock level is below threshold.
7. The handling order on a day is as follows: repair restocking depot, forward exchange, and at last restocking the base.
8. The aggregate repair loop exists of waiting days with a normal distribution.
9. A request is handled the first day from the base, if there is no base stock then at the depot, if there is no depot stock the process started over the day after.
10. A request has reached service level when a component is sent from base or depot on the first day.
11. Initial stock quantity for the base, depot, and mbk are calculated by the expansion rules.
12. Shipping days from depot to base, depot to MBK or base to MBK are constant and based on Hofman (2017) for to the base, and from base or depot to the MBK on request data 2017-2018.
13. The base is restocked at the end of every day when the depot stock is above 1 and takes place for the duration of a repair loop after a request is handled from the base.

To validate the model, both the validation run and each experiment have ten replications with a run time of ten years. Validation parameters are chosen to best represent both feedback loops within the system. The validation is per individual component, whereas Table 2 only shows the averages of all the components compared with the average actual situation; this results in 70 validation parameters with each a confidence interval. This validation is more extensive than research in the past. Munsters (2019), Sprong (2019), and Vlamings (2020) use either number of failures or number of loans to validate the system, respectively 1,1, and 10 (for 10 components) validations. Whereas, no validation or expert validation is done in Hofman (2017), and Cornelisse (2018).

The validation parameters in Table 2 show a slight off from the actual situation. The more detailed validation showed that the MBK assumption would result in lower MBK spares for airlines connected to the depot because of fewer airplanes in the contract and MBK stock is rounded. This results in relatively more stock replenishment requests send from a base connected airline. These requests have a second chance of succeeding at the depot after not succeeding at the remote base. This results in a stock replenishment response time shorter than the actual response time and even faster than a forward exchange. Secondly, the i.i.d. repair time is not a good representation of the actual situation where repair times are dependent on each other, resulting in the broader service level confidence interval and lower service level. The loan desk described in Munsters (2019) is outside the system boundary but could have a significant impact because it will result in more components in repair. Finally, the send from base actual parameter is an average of all components in a year and is unrealistic low because most components are not stocked at a depot.

Table 2: Average component performance of ther indicators as a validation of the model

Validation	**Actual**	**Simulation**
Removals	164	164
Service level	72%	80%
From Base	20%	24%
Stock replenishment requests	33%	43%
Forward exchange response	3.8	2.4
Stock replenishment response	5.3	1.2
MBK stock	10.2	9.3

5. Results

The scenarios in Figure 3 and 5 give insights in a combination of the pooling and not reaching repair turn around time strategies. The configuration in Figure 4 is additional to the theoretical variant by reducing the expansion rule service level from 99% to 70%.

Figure 3 depicts the scenario with reached repair turn around time. The service level is 100% for all the fleet unit sizes, meaning that all the stock replenishment and forward exchange requests are send the first day. With perfect 100% service levels on stock replenishment, the % forward exchange is equal to the protection level interpreted as the base line. As a validation, the 40% forward exchange ratio between 150-400 is equal to the expansion rules' average protection level. The same could be said about the base (but without validation), a 100% service level means that every request could always be fulfilled and assumed that there is stock left at the end of the day to fulfill a base request directly.

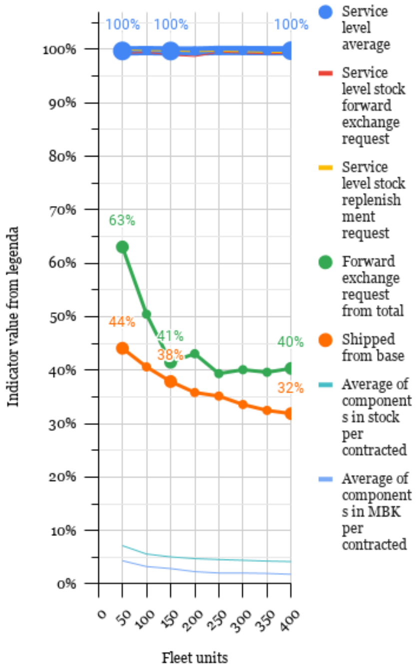


Figure 3: Average component indicator value in the scenario with theoretic constant repair turn around time

The low send from remote base percentage points out that the operational procedure for restocking is not aligned. The expansion rule accounts for always restocking within some shipping days from the depot, while in the actual situation, the component is locally repaired or waited for to be repaired at the depot. The area in the graph between 150-400 is relatively stable and perfect for comparison with the other scenarios and configuration.

Figure 4 is a configuration with expansion rule service level set at 70% instead of 99% to show more influences of a lack in components. The forward exchange requests are comparable to the reached repair time between 50-150 fleet units, meaning that stock replenishment requests are reacted on within short notice. From 150-400, the forward exchange requests increase by 13%, caused by longer response time on stock replenishment requests. For the send from base, a difference of around 20% occurs between 50-150, but up to 400 fleet units it stays approximately constant. Thus the first buffer in the system is the send from base requests without impacting the response time, after the send from remote base reached an equilibrium of around 19%, the number of forward exchanges will rise, caused by longer stock replenishment requests.

The service level slightly reduces, which is unexpected because it should be constant according to the expansion rules. From a dozen of experiments, the protection level of the MBK is the only one impacting it. The component is longer unavailable for a forward exchange request, while the component is faster available than expected with a stock replenishment. Both are not accounted for in the expansion rules.

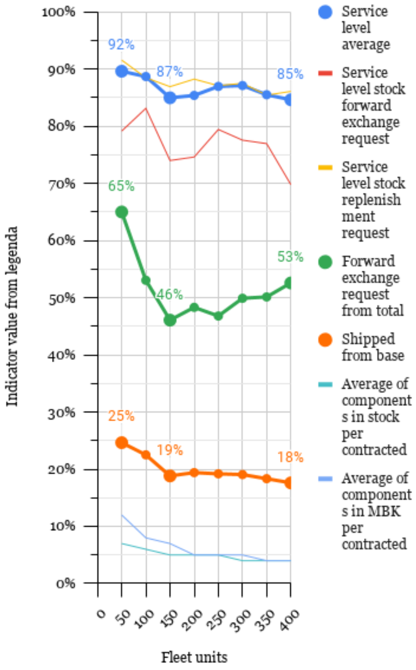


Figure 4: Average component indicator value in the scenario with theoretic constant repair turn around time

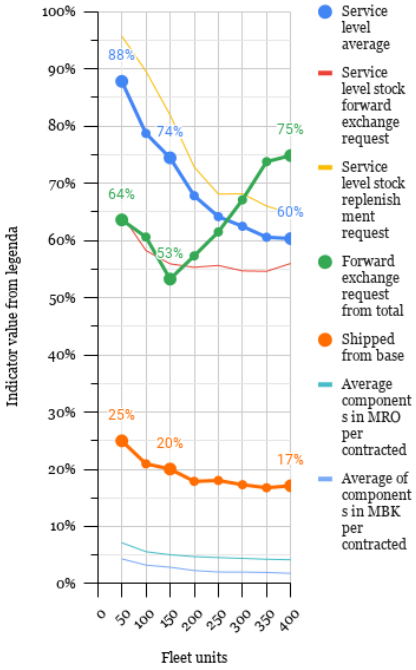


Figure 5: Average component indicator value in the scenario with actual stochastic repair turn around time.

Figure 5 depicts the scenario with a longer repair turn around time and has a reduced service level at 400 fleet units from [100-60%] with reaching the repair TAT. The forward exchanges start with no difference and end with a difference of 35% with the first scenario. The number of sent from the base is approximately the same as the configuration in Figure 5. In this scenario there is a combination of a low service level with high forward exchanges; this is a dangerous combination because then most of the requests are important forward exchange requests and not fulfilled in time.

The financial advantage of components pooling is hard to consider because it is unknown what the cost is of not reaching service levels.

6. Conclusions

The results conclude the following. If repair turn around time is improved up to the threshold used in the expansion rules, a 100% service levels will be reached, but only around 40% of the components will be sent from the base due to misaligned operational procedures with the expansion rule. The scenario with 70% expansion rule service level shows a slight descending service level caused by low MBK stock, from which it is advised to adjust expansion rules or compensate the spares in the depot and base. Within this scenario, an overall decrease of 20% in send from the base and an increase from 150 to 400 fleet units up to 13% in the number of forward exchange requests. With the long repair turnaround time, the service level rapidly drops to 60%. New contracts must only be added after a reduction of repair is accomplished to stop service level reduction. Furthermore, the number of forward exchange requests will stay the same at 50 fleet units but increases to 35% at 400 fleet units. While the number of send from the base is comparable with the reduced components configuration (70% configuration).

No conclusions on capital employed, the cost of "firefighting" should be mapped and dedicated to a service level point. These could compromise: losing a client, penalty clause, borrows, and employee hours spent on client troubleshooting.

7. Recommendations

For reuse of this model, it is crucial to improve the comments mentioned during the simulation model's validation.

From literature, there are still open topics to research. Sprong (2019) and Munsters (2019) recommend both to redo their research on predictive maintenance and an optimal loan desk with the expanded all location simulation. Sherbrooke (2004) names the problem of a component with low failures and wear because failures are not i.i.d, a proposed method to adjust the mathematical formula is a discrete-time step simulation. Sherbrooke also names the use of local repairs and indenture levels for better asset management and service level improvement; both additions are applicable to KLM's expansion rules.

The analysis pointed out that the response time is more important than the service level for a satisfied client. The relative importance and limits of service levels and response time between the client requests (stock replenishment, forward exchange, AOG) need to be analyzed. As a next step, the MRO could determine an operational strategy or procedure to send their requests (stock replenishment, forward exchange, AOG, loandesk, and restock base) from one of the locations (depot, base, MBK, borrow) after a specifically defined reaction time. When the MRO is sending from another airline's MBK, the airline needs to be compensated.

When applying the theory from Hopp and Spearman (2011), variability in the system at an aircraft MRO is currently buffered in inventory. Future research could be to reduce variability by using the loan desk and borrows to achieve more constant demand. Or it could be to optimize the buffer with forecasting or pull in order to catch variability with repair capacity or repair time instead of having more inventory. Think about it, the number of failures today indicates the number of components available in the future after a constant repair time.

At last, the excess buffer for low contracted components could be used in a more efficient matter. It is expected that there is a boundary of buffer in which multiple extra loans would still be beneficial above a single borrow.

References

Bosdijk, N., 2019. Aligning logistics with MRO to improve spare parts availability. Master thesis. TU Delft.

Cornelisse, E., 2018. Modelling business process redesign strategies for improved reverse logistics in an aircraft component supply chain (MSc Thesis). Master thesis. TU Delft.

Driessen, M., 2018. Integrated Capacity Planning and Inventory Control for Repairable Spare Parts. 1st editio ed., Eindhoven unversity of technology, Eindhoven.

Hofman, I., 2017. The effect of a multi-echelon model on the inventory investment for a supply chain of air-craft rotables. Technical Report. TU Delft. URL: `http://repository.tudelft.nl/`.

Hopp, W.J., Spearman, M.L., 2011. Factory Physics. Second edi ed., Waveland Press.

Kilpi, J., 2007. Valuation of Rotable Spare Parts.

Muhaxheri, D., 2010. Framework for Evaluation of Strategies for Pooling of Repairable Spare Parts. Master thesis. Lund University.

Munsters, D.G.C., 2019. Evaluating performance improvement strategies in a closed-loop supply chain. Master thesis. TU Delft.

Palma-Mendoza, J.A., 2014. Analytical hierarchy process and SCOR model to support supply chain re-design. International Journal of Information Management 34, 634–638. doi:10.1016/J.IJINFOMGT.2014.06.002.

Sherbrooke, C., 2004. OPTIMAL INVENTORY MODELING OF SYSTEMS Multi-Echelon Techniques OPERATIONS RESEARCH & MANAGEMENT SCIENCE.

Sprong, J., 2019. Prognostics-driven supply chain optimization in commercial aviation. Ph.D. thesis. TU Delft.

Sulyman, P.D., 2019. Short term predictive demand model based on transport times for the reverse supply chain A case study at KLM Engineering & Maintenance Compo-nent Services 2.0. Technical Report. URL: `http://repository.tudelft.nl/`.

Tokgoz, S., Bulkan, S., Zaim, D., Delen, N.T., 2017. Journal of Quality in Maintenance Engineering Article information : Modeling airline MRO operations using a systems dynamics approach : a case study of Turkish Airlines. Journal of Quality in Maintenance Engineering .

Tracht, K., von der Hagen, F., Schneider, D., 2013. Applied Repairable-item Inventory Modeling in the Aviation Industry. Procedia CIRP 11, 334–339. doi:10.1016/J.PROCIR.2013.07.020.

Visintin, F., Porcelli, I., Ghini, A., 2012. Applying discrete event simulation to the design of a service delivery system in the aerospace industry: A case study, in: IFAC Proceedings Volumes (IFAC-PapersOnline), Elsevier. pp. 1678–1683. doi:10.3182/20120523-3-RO-2023.00339.

Vlamings, B., 2020. A cost - benefit assessment of condition - based maintenance for different prognostic performance levels. Master thesis. Delft Unicersity of Technology.

Wibowo, A., Tjahjono, B., 2017. Designing Contracts for Aero-engine MRO Service Providers: Models and Simulation. Procedia CIRP 59, 246–251. doi:10.1016/J.PROCIR.2016.10.124.

Wibowo, A., Tjahjono, B., Tomiyama, T., 2016. Towards an Integrated Decision Making Framework for Aero Engine MRO Contract Management in the Productisation Context. Procedia CIRP 47, 24–29. doi:10.1016/J.PROCIR.2016.03.049.

B Literature overview

B.1 Recommendations from KLM thesis reports

This Appendix consists of an overview of all TU Delft thesis research in between 2016-2020 within KLM E&M in Table 29 and the recommendations from those reports in Table 30 and Table 31.

Author	Model	Component	Area's	Stock locations	priority requests	Stock management	Stochastic fail	OF quality	comp. spec. Shops	Work station	Workforce	Quality	MLC	TAT Repair loop	Overhaul	Main. planning
KLM TU Delft thesis																
Hofman (2017)	IP	LRU	R,O	D,B		+										
Sprong (2019)	DTSS	LRU	R,O	D			+							+		
Munsters (2019)	DTSS	LRU	R,O	D			+						+	+		
Vlammings (2020)	IP	LRU		D												
Cornelisse (2018)	OE, DTSS	LRU	R, O	D			+						+	+		
Papadopoulou (2015)	OE	LRU	R						+	+	+	+				
Sulyman (2020)	OE	LRU	R										+	+		
Bosdijk (2019)	DTSS	LRU	R						+				+	+		
Thijsens (2019)	N.A.	LRU	R													
Hoogenboom (2019)	OE	LRU	R						+	+	+					
van Rijssel (2016)	OE, DTSS	LRU	R						+	+	+					
Spaan (2018)	MCA, LP	LRU	R							++	++	++				
Soeters (2017)	OE	Engine	R							+	+	+				
Meijs (2016)	OE	Engine	R							+	+	+				
van Welsenes (2017)	OE	Engine	R							+	+	+				
Haak (2019)	OE	Engine	R						+	+		+	+	+		
Stammes (2018)	OE	Engine	R									+				
Mogendorff(2016)	N.A.	Engine	R													
Rozenberg (2016)	OE	Engine	R							+	+	+				
Hoed van den (2018)	OE	Airframe	R												+	+
Hockers (2016)	OE	Airframe	M												+	+

Table 29: LP = Linear programming, DTSS = Discrete time step simulation, R = repair loop, I = inventory, M = airframes, LRU is within component services, engine is whithin engine services, airframe is within airframes, main = maintenance, D = Depot, B = Base, K = MBK, OF = order fulfillment (all related master thesis documents at KLM from the TU Delft 2016-2020)

Master Thesis TU Delft	Object	Subject	Scientific recommendation
Soeters (2017)	Engines, internal repair and time	Evaluation of the continuous improvement framework with performance measurements to support management decision making	Application of the continuous improvement framework and the performance measurement model are also suitable to apply in other industries.
Papadopoulou (2015)	Avionics, repair and time	A framework for the identification of bottlenecks within component services with the use of Value of constraints, LEAN and sixsigma	No scientific recommendation
van Welsenes (2017)	Rotables, logistics, repairs and time	A business planning and control system is designed to measure the process performance and identify the turnaround time constraints. The identification of these constraints gives managers insight on how to efficiently and effectively control the constraints and thereby improve component availability performance.	It is useful to test the proposed BPCS at other industries to see if this design helps other MRO businesses in defining the processes, measuring the E2E operational performance and identifying the constraints.
van Rijssel (2016)	Repair shop time	Flow improvements to lower the component repair turnaround time by designing a framework for within the MRO repair shop. Evaluated with a discrete event simulation on avionics components	Link the benefits of repair time reduction with the total cost of the supply chain. Secondly the use of double t-test for the spread of variation. Thirdly the use of stochastic data instead of deterministic data for more realistic results.
Haak (2019)	Engines, internal repair and quality	Presenting a method for sustainable initial repair process design.	Further application is recommended.
Hoed van den (2018)	Heavy Maintenance	This model combines the task elements of manpower, work order, and aircraft zone to find the minimum TAT given the task complexity-based heuristic by redesign of the task planning model	Using the model to study the effects of different distributions available technicians on simultaneously executed maintenance turn around time. Secondly including work order and aircraft zone in the non-routine prediction for the effects on maintenance repair time.

Table 30: Overview of all current TU Delft master thesis literature at the Royal KLM between 2016-2020 on the subject of MRO with logistics and components

Master Thesis TU Delft	Object	Subject	Scientific recommendation
Stammes (2018)	Engine Repair Shop Quality	A model for the quality performance in terms of the Exhaust Gas Temperature Margin of an Engine MRO process chain using six sigma	Verify the framework on other engine types. Secondly develop a framework for quality prediction of the total engine performance of all repair steps.
Cornelisse (2018)	Rotables and reverse Logistics	A methodology with practical design criteria on how to redesign the reverse logistical processes of a component MRO provider in order to improve the performance from an integral supply chain perspective. A new automated logistic handling area is introduced by separating the physical and administrative handling operations.	Further research in MRO should focus on acquiring more accurate data from airline customers by i.e. rewarding programs or better collaborating platforms.
Rozenberg (2016)	Engines, repair	Desiging a framework for improving cost, quality and turnaround time.	From a scientific aspect, it is useful to fit previously developed frameworks for aircraft MRO into the comprehensive seven-step framework developed in this research. Examples of these frameworks are given by Meijs (2016), (Mogendorff, 2016) and van Rijssel (2016).
Sulyman (2019)	Logistics and time	Short term predictive demand model based on transport times for the reverse supply chain	(yet unavailable)
Thijssens (2019)		Reliability modelling for aircraft component availability	(yet unavailable)
Mogendorff (2016)		proposed a method to decrease the TAT of combustor maintenance through process improvement and simulation.	(yet unavailable)
Hockers (2017)	Airframes, maintenance and time	Redesign of airframe MRO processes from an up-time perspective with the use of theory of constraints (incomplete report)	It is advised to use the theoretical model at different airframe MRO processes to test if it can improve the output performance
Hogenboom (2019)			(yet unavailable)

Table 31: Overview of all current TU Delft master thesis literature at the Royal KLM between 2016-2020 on the subject of MRO with logistics and components

C Data

C.1 Model parameters

The data used in the simulation specific from the inventory control is depicted in Table 32.

Table 33 shows a validation from when the simulation did had a borrowing assumption. None of the actual borrowed numbers fit within the 95% confidence interval of the 29 years model running. The assumption is that if there is no components coming from repair in the upcoming 7 days or 7 days passed then to fulfill the request the component is borrowed. According to more recent information from Gaston [6], components are borrowed if and only when there are problems in the supply of component from the repair shop. In the simulation it is unknown if there are supply problems thus the total number of borrows should be zero.

Table 34 is an overview per component from the 2017-2018 request set and depicts the distribution of times in several parts of the closed loop in practice.

For the shipping time in Table 35 from depot and base to MBK a .CSV file from the KLM drive was uploaded with request data shared by S.Zeedijk. The shipment times above 30 days are eliminated by assumption that the data is incorrect or an outlier. According to Zeedijk a component could also be send from another base if an aircraft of the airliner would have an unexpected failure at a different location than the MBK base location. Send from another location data is eliminated by only selecting the shortest shipment time location, the connected bases are not validated with the effectivity in the 787 inventory model.
The cost of sending a package is received from Vennink in the form of data sheets with all send packages at Bollore for KLM CS. The huge differences in naming a location made it impossible to gather use full data according to the transport location. But it can be concluded that a normal shipment costs an average of around 100$ and a priority sending if possible only slightly more. The impact of 100$ on the new price of a VFSG costing 550.000$ is nihil and also why it is not taken into the cost calculation.

CN	QPA	Actual MTBR	ESS	CRIT	Turnaround time	EFF total	Data rows
870056	1	7044	2	D	40	129	64
870087	2	23915	2	B	27	199	73
870180	2	17933	1	-	32	220	77
870261	1	21353	1	-	32	203	23
871010	8	31335	2	C	27	205	23
871754	2	28091	2	B	27	210	20
871915	1	12325	2	A	32	211	34
872318	36	64037	2	D	27	211	34
872517	4	15046	2	C	40	205	52
888003	4	21904	2	A	26	216	160

Table 32: Table showing the selected components for simulation

---

[6] Supply chain engineer at KLM component service

Component	**Interval failures**	**Actual failures**	**Interval borrowed**	**borrowed**
870056	57.4-64.7	62	0.9-2.7	0
870087	54.4-59.6	68	0	0
870180	83.5-90	86	0-0.2	2
870261	29.7-34.2	35	0	0
871010	182-194.5	197	0	2
871754	49.1-54.3	53	0-0.6	3
871915	56.2-62.5	64	4.4-8.2	0
872318	407.9-422.7	430	0	0
872517	184.1-193.9	191	0-0.1	0
888003	133.8-143.6	140	-0.1-0.3	0

Table 33: Confidence T test with a sample of 29 years with failures per year per component and number of leased components per year per component. All green actual values are within the confidence interval

			Forward Exchange			**Exchange**			**Stock replenishment**		
	rem-depot	**Std dev**	**req-ship**	**Shipping**	**Arrival - rem**	**req-ship**	**Shipping**	**Arrival - rem**	**req-ship**	**Shipping**	**req**
870180	54	68	1	5	2	0	1	38	3	4	-3
870261	46	30	20	4	10	9	3	#N/A	1	3	-3
871915	100	33	3	6	4	20	7	#N/A	28	9	-2
888003	48	42	1	17	1	1	#N/A	#N/A	5	10	-24
870087	70	36	3	13	7	0	#N/A	0	1	5	-3
871754	48	24	0	1	5	3	5	0	3	3	-3
870056	61	29	5	4	3	7	2	30	7	6	-3
872318	49	29	3	5	4	0	#N/A	#N/A	1	5	-5
871010	53	34	1	14	5	2	#N/A	#N/A	4	5	-5
872517	57	19	2	9	7	4	3	15	3	8	-3
AVG	55	28	3	8	5	3	3	23	4	6	-4

Table 34: Table with the specification per request type per component

Customer	**Depot to MBK**	**RJA to MBK**	**RMI to MBK**
Air Austral	3,7		
Air Canada	10,5		
Air France	5,1		10,5
Air New Zealand	1,8	2,0	8,1
Jetstar	10,7	4,0	9,7
Kenya Airways	3,7	5,0	7,0
LATAM Airline Group	2,7	4,1	1,8
LOT Polish Airlines	3,4	0,0	1,5
Qantas	5,7	3,6	3,9
Royal Air Maroc	5,6		
Royal Brunei Airlines	6,6	1,5	7,5
Saudi Arabian Airlines	4,7		
Thai Airways	5,2	2,0	6,4
Vietnam Airlines	7,7	3,0	12,3
Virgin Atlantic Airways Ltd	4,5	1,5	2,8
Xiamen Airlines	11,6	9,6	14,0
Average per location	4.7	3.5	2.1

Table 35: Table with the corresponding sending times from a depot or base to an MBK

D Results

D.1 Extra results theoretic scenario scenario

	870056	870087	870180	870261	871010	871754	871915	872318	872517	888003
50	99.8%-100.2%	99.7%-100.1%	99.5%-100.3%	99.3%-100.1%	99.8%-100.2%	99.3%-100.1%	98.8%-100.1%	98.2%-99.4%	99.8%-100.1%	99.5%-100.2%
100	99.7%-100%	99.3%-100%	99.8%-100.2%	99.3%-100.3%	99.3%-100.2%	100%	99.8%-100%	98.8%-99.5%	99.7%-100.1%	99.7%-100.1%
150	99.9%-100.1%	99.7%-100.1%	99.7%-100.1%	98.9%-100.2%	99.9%-100.1%	99.2%-100.1%	99.5%-100.1%	97.5%-99.1%	99.8%-100.1%	99.8%-100.1%
200	99.9%-100.1%	99.8%-100.1%	99.8%-100.2%	99.8%-100.1%	99.9%-100.1%	99.5%-100.1%	99.3%-100.1%	96.9%-98.4%	99.9%-100.1%	99.8%-100.1%
250	99.9%-100.1%	99.7%-100%	99.8%-100.2%	99.7%-100.2%	99.9%-100.1%	99.8%-100.1%	99.4%-100%	98.4%-99.4%	99.9%-100.1%	99.9%-100.1%
300	99.7%-100.1%	99.9%-100.1%	99.8%-100.1%	99.8%-100.1%	99.9%-100.1%	99.8%-100.1%	99.8%-100.1%	98%-99%	99.8%-100%	99.9%-100.1%
350	99.9%-100%	99.9%-100.1%	99.9%-100.1%	99.8%-100.1%	99.9%-100%	99.8%-100.1%	99.8%-100.1%	97.3%-99%	99.7%-100%	99.9%-100.1%
400	99.9%-100.1%	99.8%-100.1%	99.9%-100.1%	99.6%-100%	99.9%-100.1%	99.9%-100.1%	99.9%-100.1%	97.5%-98.7%	99.8%-100.1%	99.9%-100.1%

Table 36: Experimental plan scenario 1. theoretic repair turnaround time depicting the service level 95% confidence interval per component.

	870056	870087	870180	870261	871010	871754	871915	872318	872517	888003
50	47.8%-51.3%	51.9%-56.7%	53.8%-58.1%	48%-55.6%	44.1%-46.5%	45%-50.8%	39.1%-44%	37.7%-39.6%	38.7%-41.2%	47.8%-50.9%
100	39.9%-42.6%	48%-51.4%	44.6%-47.8%	47.9%-52.7%	40.4%-42.3%	51.9%-56.4%	45.6%-48.8%	33.9%-35%	32.7%-34.3%	49.5%-52.2%
150	41%-43%	47.9%-50.1%	41.6%-44.1%	47.5%-51.3%	40.4%-42%	47.8%-50.9%	42.2%-44.6%	30.4%-31.4%	32.2%-33.4%	39.8%-41.1%
200	33.5%-35.2%	45.6%-47.7%	40.6%-42.8%	48.2%-50.6%	36.9%-38%	41.9%-44%	42.5%-45.3%	29.8%-30.5%	28.3%-29%	42.8%-44.1%
250	35%-36.5%	40.7%-42.5%	35.8%-37.4%	43.5%-46.2%	37.5%-38.6%	48.7%-50.4%	41.3%-43.2%	28.9%-29.5%	29.4%-30.4%	40%-41.6%
300	33.4%-34.7%	43.6%-45.2%	40.6%-42.3%	39.2%-41.6%	34.9%-35.9%	43.4%-45.8%	37.1%-38.7%	27.8%-28.3%	26.4%-27.1%	38.4%-39.4%
350	30%-31.1%	42.2%-44.2%	36.6%-37.9%	41.3%-43.8%	33.9%-34.7%	40.2%-42%	38.1%-39.6%	26.8%-27.3%	26%-26.7%	38.6%-39.8%
400	29.8%-31%	40.2%-41.5%	36.6%-37.9%	43.3%-45.1%	34.2%-35%	41.1%-42.5%	40.3%-41.7%	25.6%-26.1%	25.2%-25.9%	36.6%-37.7%

Table 37: Experimental plan scenario 1. theoretic repair turnaround time depicting the send from base 95% confidence interval per component.

	870056	870087	870180	870261	871010	871754	871915	872318	872517	888003
50	62.9%-68.4%	60.4%-65.6%	1.4%-3.5%	100%	41.6%-45.4%	100%	100%	100%	39.2%-42.7%	2.5%-3.9%
100	38.7%-42.4%	65.3%-68.7%	4%-5.5%	61.6%-66.6%	37.4%-39.9%	62.9%-67.6%	100%	72.4%-73.7%	38.3%-40.2%	5.2%-7%
150	35.4%-37.5%	41.4%-44%	6.5%-8.2%	64.3%-68.6%	12.9%-14.4%	63.9%-67.2%	64%-67.2%	75.4%-76.5%	13.7%-14.9%	9.9%-11.2%
200	37.5%-39.3%	36.9%-39%	7.8%-9.4%	63%-67.2%	17.7%-18.9%	43.9%-46.3%	63.9%-66.7%	77.8%-78.5%	18.7%-19.9%	12%-13.6%
250	38.9%-40.9%	37.3%-39%	9.4%-10.9%	40.2%-44.3%	20.7%-22%	35.6%-38%	66.2%-68.4%	64.2%-65%	21.3%-22.6%	15.7%-17.3%
300	12.8%-14.2%	38.1%-40.4%	11.6%-13.1%	43.4%-46.3%	24.6%-25.4%	37.1%-39.8%	67%-69.4%	67.3%-68.1%	25.5%-26.5%	17.4%-18.9%
350	15.1%-16.3%	39.2%-41%	13.4%-14.6%	33.8%-36.6%	27.5%-28.6%	37.9%-39.8%	46.1%-48.1%	64.7%-65.8%	28.2%-29.4%	20.3%-21.7%
400	16.8%-18.2%	13.1%-14.7%	15.1%-16.6%	36.1%-38.4%	29.8%-30.8%	38.3%-40%	47.5%-49.3%	67.3%-68%	31.8%-32.9%	23.6%-24.9%

Table 38: Experimental plan scenario 1. theoretic repair turnaround time depicting the forward exchange percentage 95% confidence interval per component.

D.2 Extra results actual scenario

	870056	870087	870180	870261	871010	871754	871915	872318	872517	888003
50	62.5%-67.6%	63.4%-70.3%	1.3%-3.3%	100%	42.7%-46.3%	100%	100%	100%	40.8%-44%	2.9%-4.8%
100	42.7%-45.9%	66.3%-69.7%	4.6%-7%	61.8%-66.4%	41.5%-45%	65.3%-69%	100%	99.9%-100%	38.5%-40.8%	7.1%-9.1%
150	36.3%-39.3%	46.1%-50%	11%-15.6%	65.7%-69.4%	19.1%-23.4%	65.8%-68.7%	88.4%-94.4%	100%	14.9%-16.6%	14.7%-17.9%
200	38.5%-41.6%	47.5%-55.4%	14.9%-19.8%	63.2%-66.5%	36.9%-43.8%	43.3%-46.7%	99.7%-100%	99.9%-100%	20.6%-23.1%	18.2%-23.3%
250	41.5%-44.2%	64%-71.3%	21.6%-29.9%	39.5%-43.1%	46.5%-56.8%	36%-38.8%	100%	99.9%-100%	25.1%-27.7%	28%-34.4%
300	14.2%-15.8%	70.1%-79.9%	25.9%-31.4%	42.1%-45.9%	82.8%-91.4%	38.5%-41.3%	100%	99.9%-100%	29.7%-34%	39.2%-46.7%
350	18%-20.4%	81.7%-89.9%	38.5%-49.6%	36%-39.4%	97.2%-99.3%	39.6%-41.9%	100%	100%-100%	37.5%-44.2%	55.7%-66.9%
400	20.9%-25.7%	59%-71.3%	49.9%-62.1%	37%-39.6%	99.1%-99.5%	41.9%-44.9%	100%	99.9%-100%	39.3%-43.7%	63.4%-76.1%

Table 39: Experimental plan scenario 2. actual repair turnaround time depicting the forward exchange percentage 95% confidence per component

	870056	870087	870180	870261	871010	871754	871915	872318	872517	888003
50	40.8%-44.6%	29.4%-34.8%	37.6%-42.2%	40.2%-47.2%	28.9%-32.4%	34.1%-39.7%	17.5%-23.4%	4.1%-7.9%	31.2%-33.6%	33.4%-37.6%
100	29%-31.8%	21.3%-25.8%	25.7%-28.7%	41.5%-46.7%	19.9%-23.6%	40.5%-44.9%	2.7%-7.2%	6.6%-7.6%	25.3%-27.1%	34.1%-37.2%
150	30.3%-33%	18.8%-23.2%	18.6%-23.1%	41.7%-45.9%	20.2%-23.2%	34.9%-38.6%	1.6%-3.7%	10.8%-11.7%	25%-26.4%	19.3%-22.4%
200	22.6%-25.3%	11.7%-17.2%	15.3%-19.4%	40.6%-44.5%	12.3%-14.8%	28%-31.4%	0.5%-1.1%	15.1%-15.8%	20.2%-21.7%	19.7%-23.8%
250	23.9%-26.1%	4.1%-6.8%	11.9%-15.4%	36%-39.5%	12.8%-15%	34.5%-37.2%	0.6%-1.2%	18.9%-19.4%	20.7%-21.9%	15.8%-19%
300	24%-25.6%	4.4%-7.2%	12.6%-15.9%	30.7%-33.3%	12.7%-13.9%	28.3%-31.5%	1.1%-1.8%	20.2%-20.8%	18.7%-19.7%	12.1%-14.7%
350	19.8%-21.2%	4.2%-5.9%	8.7%-11.2%	33.1%-36.1%	14.4%-15.1%	26%-28.3%	1.4%-2.2%	20.9%-21.3%	18%-18.9%	12%-13.9%
400	18.6%-20.5%	5.9%-7.5%	8.2%-11%	35.1%-37.4%	16.7%-17.2%	23%-26.5%	1.9%-2.7%	20.7%-21.1%	18.4%-19.1%	12.5%-14.4%

Table 40: Experimental plan scenario 2. actual repair turnaround time depicting the send from base percentage 95% confidence per component

	870056	870087	870180	870261	871010	871754	871915	872318	872517	888003
50	98.8%-100.1%	87.3%-94%	98.5%-100%	97.8%-99.8%	88.9%-93.9%	94.4%-98.8%	59.3%-74.8%	31.7%-40.6%	97.1%-99.1%	98.6%-99.8%
100	94.2%-98%	74.1%-82.2%	96.3%-99.2%	99%-99.9%	78.8%-85.4%	96.6%-99.1%	13.9%-24%	42.9%-44.7%	94%-97.5%	94.6%-98%
150	95.4%-98.3%	66.9%-77.6%	93.5%-97.3%	98.5%-100.1%	74.3%-83%	92.2%-96.9%	16%-19.5%	47.5%-48.7%	94.7%-97.8%	86.5%-91.4%
200	89.6%-94.8%	44.9%-54.1%	91.5%-96.3%	97.9%-99.7%	56.8%-65.1%	87.4%-93.2%	17.5%-19.8%	49.3%-50.3%	88.8%-94.1%	86.9%-92%
250	89%-94.2%	30.5%-36.8%	90.2%-94.7%	98%-99.5%	50.6%-57.6%	93.5%-97.3%	19.5%-21.6%	51.9%-53.1%	86.2%-92.4%	73.9%-81.5%
300	92.6%-97.3%	31.3%-35%	88%-92.8%	96.2%-98.7%	46.9%-49.8%	90.3%-94.1%	20.8%-23%	48.5%-50.8%	83%-89.2%	65.9%-74.3%
350	84.4%-90.6%	34%-36.6%	86.1%-92%	96.8%-99.1%	53.3%-55.1%	86.1%-91%	24%-26%	44.3%-46%	77.5%-84.1%	61.7%-69.1%
400	80.9%-88.2%	30.1%-33.3%	77.6%-85.8%	97.1%-99.5%	56.1%-57.1%	83.4%-89%	25.8%-27.7%	42.3%-44.2%	71.3%-79.7%	56.3%-63.6%

Table 41: Experimental plan scenario 2.a. actual constant repair turnaround time depicting the service level percentage 95% confidence interval per component

	870056	870087	870180	870261	871010	871754	871915	872318	872517	888003
50	59.4%-64.3%	63.8%-69%	0.9%-2.2%	100%	40.7%-44.6%	100%	100%	100%	42.5%-46.4%	3.5%-5.1%
100	42.5%-46.2%	65.4%-69.7%	4.1%-5.9%	62%-67.7%	40.6%-44%	64.6%-69.7%	100%	99.9%-100%	38%-40.8%	6.4%-8.2%
150	36%-38.7%	46.3%-50.6%	6.5%-8.3%	61.7%-66.1%	17.4%-20.8%	64.9%-68.4%	96.1%-99.1%	100%	14.5%-16.3%	11.9%-13.7%
200	38.5%-41%	47.8%-54.4%	9.7%-11.3%	64.8%-68.5%	28.8%-34.5%	44.6%-47.6%	98.5%-100.2%	99.9%-100%	20.4%-22.3%	14.6%-16.9%
250	41.4%-43.8%	64.9%-73.9%	12%-13.9%	41.4%-44.6%	39.2%-47.6%	36.3%-38.8%	99.9%-100%	99.8%-100%	24.6%-27.5%	19.9%-23%
300	13.8%-16.3%	73.3%-81.4%	13.4%-15.6%	42.4%-45.5%	69.4%-79.6%	38.3%-40.8%	100%	99.7%-100%	28.8%-32.1%	25.3%-29.8%
350	18.3%-20.8%	81.7%-89.9%	15.3%-17.9%	35.8%-38.2%	89.2%-97%	41%-43.1%	100%-100%	100%-100%	35%-39.1%	32.1%-38%
400	19.8%-23%	61.1%-69.6%	19.4%-22.6%	35.9%-38.8%	98.1%-99%	39.9%-42.2%	100%	100%-100%	41.1%-48.8%	40.2%-47.7%

Table 42: Experimental plan scenario 2.a. actual constant repair turnaround time depicting the forward exchange percentage 95% confidence interval per component

	870056	870087	870180	870261	871010	871754	871915	872318	872517	888003
50	42.8%-46.6%	30.2%-36.1%	45.1%-50.3%	44.1%-51.9%	30%-33.3%	36.7%-42.3%	14.6%-20.9%	2.3%-6.1%	31.5%-33.8%	37.6%-40.5%
100	29.7%-32.1%	23.2%-28.3%	34.2%-37.2%	43.6%-47.6%	22.3%-25.1%	40.6%-45%	0.8%-4%	5.9%-6.8%	25.3%-27%	36.1%-38.7%
150	30.9%-33%	19%-23.6%	30.2%-32.6%	44.1%-47%	21.4%-24.3%	34.3%-37.8%	0.3%-1.3%	9.5%-10.3%	24.6%-26.1%	24.8%-26.8%
200	24.3%-26.3%	10.4%-14.2%	27.9%-30.1%	40.5%-42.9%	15.3%-17.7%	27.8%-30.9%	0.4%-1%	15.6%-16.7%	20.7%-21.8%	27.5%-29.7%
250	24.2%-26.2%	3.7%-6.4%	23.7%-25%	36.4%-39.1%	14.3%-16.5%	35.3%-37.7%	0.7%-1.2%	20.2%-20.7%	20.6%-21.9%	21.9%-24.4%
300	24.2%-25.9%	3.3%-5.4%	26.6%-28.6%	31.7%-34.3%	12.8%-14%	29.7%-32%	1.1%-2%	21%-21.2%	19.1%-20%	18.9%-21.4%
350	19.8%-21.4%	3.1%-4.6%	23.6%-25.6%	34.5%-36.5%	13.5%-14.6%	25.2%-27.8%	1.6%-2.4%	20.5%-20.6%	18.6%-19.4%	18%-20.3%
400	19.1%-20.9%	4.7%-6%	21.3%-23.8%	35.4%-37.7%	15.8%-16.6%	24.7%-27.3%	1.6%-2.3%	19.8%-19.9%	18.2%-18.9%	16.8%-18.9%

Table 43: Experimental plan scenario 2.a. actual constant repair turnaround time depicting the from base percentage 95% confidence interval per component

	870056	870087	870180	870261	871010	871754	871915	872318	872517	888003
50	63.8%-68.5%	64.2%-69.3%	1.5%-3.4%	100%	42.7%-46.3%	100%	100%	100%	42.3%-45.5%	3.3%-5.2%
100	42.2%-45.8%	64%-67.6%	3.8%-5.7%	62.4%-67.2%	39.6%-42.2%	64%-67.8%	100%	76.2%-79.4%	39.7%-42.5%	8.4%-10%
150	38.3%-41.7%	44.4%-47.5%	7.4%-9.3%	63.1%-67.6%	15.5%-17.5%	66.6%-69.7%	65.9%-68.8%	86.4%-90.4%	16.5%-18.6%	10.6%-11.9%
200	40.1%-42.7%	37.1%-39.9%	10.1%-11.9%	63.9%-67.6%	19.8%-21.5%	43%-46.6%	69.3%-72.2%	90.3%-93.2%	20.4%-22.5%	15.1%-17%
250	43%-45.1%	38.9%-41%	10.7%-12.2%	41.3%-44.3%	23.4%-25.4%	37.5%-40.2%	70.4%-72.9%	79.8%-85%	24%-26%	17.9%-19.5%
300	15.2%-17.3%	41.8%-44.3%	14%-16.1%	44%-47.2%	27.5%-29.3%	37.9%-40.8%	68.9%-71.1%	87.5%-92.5%	29.1%-31.6%	20%-21.8%
350	17.3%-19.1%	43.5%-46.1%	16.4%-18.4%	37.9%-40.7%	31.4%-33.6%	39.4%-41.8%	48.9%-51.4%	88.6%-93.6%	32%-34%	23.3%-24.9%
400	19.5%-21.2%	16.1%-17.9%	18.1%-19.8%	39.4%-42.2%	33.9%-35.9%	40.7%-42.7%	52.2%-54.6%	97.4%-98.4%	36.4%-39%	26.9%-28.5%

Table 44: Experimental plan configuration 1.a. fixed constant repair turnaround with equal MBK protection levels and 70% service level depicting the forward exchange percentage 95% confidence interval per component.

	Service level	Forward exchange %	Base shipped %
50	91.2%-89.1%	31.9% -30%	66.6%-65.1%
100	89.4%-87.5%	25%-24.4%	51.1%-49.8%
150	86.9%-85.3%	21%-20.2%	41.8%-40.9%
200	89.4%-87.5%	21.6%-21%	36.8%-35.9%
250	89.5%-87.8%	20.7%-20.2%	36.4%-35.4%
300	89.2%-87.8%	20.1%-19.6%	35.9%-35.1%
350	88.8%-87.4%	19.4%-19%	34.6%-33.6%
400	88.8%-87.1%	19%-18.5%	34.7%-33.5%

Table 45: Service level average per component for configuration 1.b

	870056	870087	870180	870261	871010	871754	871915	872318	872517	888003
50	26.8%-31.7%	34.5%-39.9%	34%-38.8%	36.5%-45.2%	22.2%-24.8%	37.1%-44.2%	34.6%-40.5%	22.6%-24.9%	18.6%-21.1%	25.6%-29.3%
100	17.6%-19.8%	26.2%-29.6%	21.2%-24.2%	34.3%-38.2%	25.8%-27.8%	27.3%-31.3%	26.4%-29.8%	23%-24%	18.8%-20.9%	18.1%-20.1%
150	13.9%-15.7%	19.6%-22%	17.5%-19.4%	27%-32.3%	19.1%-20.5%	21.8%-25%	21.8%-24.4%	18.2%-18.8%	14.3%-15.2%	24.2%-26.3%
200	19.4%-21.2%	23.8%-25.9%	19.5%-21.6%	24.6%-27.8%	20.8%-21.8%	20.1%-22.4%	16.2%-18.4%	19.1%-19.5%	16.3%-17.1%	20.6%-21.7%
250	16.6%-18.3%	26%-27.5%	19.7%-21.7%	21.6%-24.1%	20.4%-21.3%	18.9%-21%	19.8%-21.9%	18.6%-19%	15.2%-15.8%	20.5%-21.5%
300	15.2%-16.1%	22.8%-24.4%	17.2%-19%	19.1%-21.6%	20%-20.8%	21%-22.9%	23.5%-25.2%	19%-19.4%	14.6%-15.1%	24.4%-25.4%
350	15.8%-16.8%	19%-20.8%	16.9%-17.9%	17.3%-19.5%	19.9%-20.5%	24.9%-27%	20.2%-21.8%	18.5%-18.8%	14.4%-14.9%	20.7%-21.7%
400	16.7%-17.6%	21.9%-23.2%	17.9%-19%	15.4%-17.7%	19.2%-19.8%	22.1%-23.9%	18.3%-19.8%	17.8%-18.2%	14.1%-14.5%	18.8%-19.6%

Table 46: Experimental plan configuration 1.a. fixed constant repair turnaround with equal MBK protection levels and 70% service level depicting the from base percentage 95% confidence interval per component.

E Computational model

```python
import itertools

import random

import simpy

import numpy

import pandas

import math

from scipy.stats import norm

RANDOM_SEED = 42

#end_total_stock = 0
# DATA_FILE = 'c:\\simulationdata.xlsx'
DATA_FILE = 'simulationdata.xlsx'
data_locations = pandas.read_excel(DATA_FILE, sheet_name='locations')
LOCATIONS_MATRIX = data_locations.to_numpy()
#LOCATIONS_MATRIX = pandas.DataFrame(data_locations)
LOCATIONS_DATA = pandas.DataFrame(data_locations)
# Column
# 0 = Row number
# 1 = Name location [AMS, RMI, RJK, SHA]
# 2 = Transport time

data_parameters = pandas.read_excel(DATA_FILE, sheet_name='parameters')
#SETTINGS = pandas.DataFrame(data_parameters)
SETTINGS = data_parameters.to_numpy()[0]
# Column
# 0 = Total simulation time in days
# 1 = TRANSPORT_COST = 1 # UNKNOWN [sprong,2019].......................
# 2 = 0.075 #Lease cost of new price [sprong,2019]
# 3 = 0.20 # Percentage of new price to hold a componenent [sprong,2019]
# 4 = WACC percent of new price, investment cost for spare components [sprong,2019]
# 5 = 2.18 * TRANSPORT_COST #Emgency transport cost[sprong,2019]
# 6 = AOG repsonse time in days
# 7 = MEL-A repsonse time in days
# 8 = MEL-B repsonse time in days
# 9 = MEL-C repsonse time in days
# 10 = MEL-D repsonse time in days
# 11 = Stock Replenishment repsonse time in days
# Rebuild in the simulation data sheet
# 13 = Random seed


data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners')
AIRLINERS_MATRIX = data_airliners.to_numpy()

# Column
# 0 = row number [airliner]
```

```
# 1 = Name airliner
# 2 = Flight hours
# 3 = Travel time from the depot to the cliens MBK
# 4 = Travel time from the base 1 to the cliens MBK
# 5 = Travel time from the base 2 to the cliens MBK
# 6 = Travel time from the base 3 to the cliens MBK
# 7 = Number of contracted to component 870056
# 8 = Number of contracted to component 870087
# 9 = Number of contracted to component 870180
# 10 = Number of contracted to component 870261
# 11 = Number of contracted to component 871010
# 12 = Number of contracted to component 871754
# 13 = Number of contracted to component 871915
# 14 = Number of contracted to component 872318
# 15 = Number of contracted to component 872517
# 16 = Number of contracted to component 888003

data_components = pandas.read_excel(DATA_FILE, sheet_name='components')
data_component = data_components.to_numpy()
# COMPONENTS_MATRIX = data_components.to_numpy()
# Column
# 0 = Name component
# 1 = Code name component
# 2 = Price thousand Euro's
# 3 = QPA (Quantity Per Aircraft) in number
# 4 = MTBR (Mean Time Between Repairs) in hours
# 5 = repair cost in thousand Euro's
# 6 = ess
# 7 = mel
# 8 = SL
# 9 = Theoretical TAT
# 10 = 12 month removals
# 11 = average repair days
# 12 = repair time deviation
# 13 = % forward exchange
# 14 = Wating days forward exchange
# 15 = Stock relative to calculation
# 16 = Real MTBR
# 17 = eff 2018
# 18 = eff 2019
# 19 = eff 2020

results_header = {'component':[],
                    'configuration':[], 'scenario':[],

                    'fe#':[], 'aog#':[], 'sr#':[],

                    'base#':[], 'depot#':[], 'borrow#':[],

                    'borrow day#':[],

                    'sr day#':[],'fe day#':[],

                    'removals#':[], 'sl%':[],

                    'cost':[] ,'capital':[],

```

```
                    'total stock':[], 'depot stock':[], 'base stock':[], 'airline stock':[],
                    'AIP':[], 'Planning':[],
                    'AIP_RMI':[], 'OLV_RMI':[], 'planning_RMI':[] ,
                    'AIP_RMI':[], 'OLV_RMI':[], 'planning_RMI':[] ,
                    'in ship base rep':[], 'in ship base':[], 'in ship depot':[], 'in ship lease':[],
                    'reserved forward':[], 'in lease':[], 'in repair':[], 'last repair days':[],

                 }
results = pandas.DataFrame(results_header)

# requests_results_header = {'component':[],
#                       'Airline':[],
#                       'Request':[],
#                       'Send from':[],
#                       'Days':[]
#                  }
# request_results = pandas.DataFrame(requests_results_header)


print(data_component)
print(AIRLINERS_MATRIX)
print(LOCATIONS_MATRIX)
print(SETTINGS)
print(results)
```

scripts/data_import.py

```
def failure_generator(component, airline, env):
    while True:
        for failure_count in itertools.count():
            yield env.timeout(exponential_failure_distribution(airline, component, env))
            component.failures.put(1)
            extra_day = 0

            if airline.stock.level == 0:
                "AOG TYPE!"
                component.exchange.put(1)
                request = "AOG"

                env.process(request_handling(failure_count, airline, component, request, extra_day, env))

            elif airline.stock.level < airline.initial_stock_level_mbk():

                component.forward_exchange.put(1)
                request = "forward_exchange"

                env.process(request_handling(failure_count, airline, component, request, extra_day, env))

            else:
                if random_number() < 0.3:

                    component.forward_exchange.put(1)
                    request = "forward_exchange"
```

```

                    env.process(request_handling(failure_count, airline, component, request, extra_day, env))
                else:

                    component.stock_replenishment.put(1)
                    request = "exchange_stock_replenishment"

                    airline.stock_get()
                    component.in_repair.put(1)

                    env.process(repair_loop(failure_count, airline, component, request, env))
                    yield env.timeout(0.01) # Prioritize second
                    env.process(request_handling(failure_count, airline, component, request, extra_day, env))

def random_number():
    return round(random.random(),0)

def exponential_failure_distribution(airline, component, env):
    scale   = component.mtbr() / airline.flight_hours_airline()
    number  = numpy.random.exponential(scale=scale, size=None)
    rounded = round(number, 0)
    return rounded

def request_handling(failure_count, airline, component, request, extra_day, env):

    if airline.connected_base and airline.connected_base.stock_available():
        if extra_day == 0:
            component.on_time_handled.put(1)
        if extra_day != 0:
            component.to_late_time_handled.put(1)
        if extra_day > 0:
            if request == "forward_exchange" or request == "AOG":
                component.response_time_forward_exchange.put(extra_day)
            if request == "exchange_stock_replenishment":
                component.response_time_stock_replenishment.put(extra_day)
#             component.extra_day_handled.put(extra_day)
        env.process(handled_from_base(failure_count, airline, component, request, env))

        from_ = 'base'
#         env.process(append_request(failure_count, airline, component, request, env, from_, extra_day))
    elif DEPOT_BASE.stock.level > 0:
        if extra_day == 0:
            component.on_time_handled.put(1)
        if extra_day != 0:
            component.to_late_time_handled.put(1)

        if extra_day > 0:
            if request == "forward_exchange" or request == "AOG":
                component.response_time_forward_exchange.put(extra_day)
            if request == "exchange_stock_replenishment":
                component.response_time_stock_replenishment.put(extra_day)
#             component.extra_day_handled.put(extra_day)


```

```
        env.process(handled_from_depot(failure_count, airline, component, request, env))
        from_ = 'depot'
#         env.process(append_request(failure_count, airline, component, request, env, from_ , extra_day))
    else:
#         if component.last_repair_days.level > 0 and extra_day < 7 :
        extra_day += 1

        yield env.timeout(1)
        env.process(request_handling(failure_count, airline, component, request, extra_day, env))

#         else:
#             if extra_day <= 2:
#                 component.on_time_handled.put(1)
#             if extra_day > 2:
#                 component.to_late_time_handled.put(1)

#             if extra_day > 0:
#                 component.extra_day_handled.put(extra_day)
#             env.process(handled_from_lease(failure_count, airline, component, request, env))
#             from_ = 'lease'
#             env.process(append_request(failure_count, airline, component, request, env, from_,  extra_day))
#     print(f'Component: {component.name()} failure: {failure_count} times at {airline.name()} on time moment {env.now} extra day {extra_day}')


# def append_request(failure_count, airline, component, request, env, from_, extra_day):
#     global request_results
#     row_request_results = {
#         'component':round(component.name()),
#         'Airline':airline.name(),
#         'Request':request,
#         'Send from': from_ ,
#         'Days': extra_day
#         }
#     global request_results
#     request_results = request_results.append(row_request_results, ignore_index=True)
#     yield env.timeout(1)

def handled_from_depot(failure_count, airline, component, request, env):

    DEPOT_BASE.get()
    component.in_shipping_depot_to_airline.put(1)

    component.handled_from_depot.put(1)
    yield env.timeout(airline.shipping_depot_airline())

    component.in_shipping_depot_to_airline.get(1)

#     if request == "exchange":
#         component.in_repair.put(1)

#         env.process(repair_loop(failure_count, airline, component, request, env))

    if request == "forward_exchange" or request == "AOG":
```

```
        component.forward_reserved_stock.put(1)

        yield env.timeout(component.forward_exchange_until_repair())

        component.forward_reserved_stock.get(1)
        component.in_repair.put(1)

        env.process(repair_loop(failure_count, airline, component, request, env))

    elif request == "exchange_stock_replenishment":
        airline.stock.put(1)
    #print(f'Component: {component.name()} failure: {failure_count} , request type: {request} from depot arrived at MBK: {airline.name()} on time moment {env.now}')



def handled_from_base(failure_count, airline, component, request, env):
    airline.connected_base.stock.get(1)
    component.in_shipping_base_to_airline.put(1)

    env.process(stock_check_base_1(failure_count, airline, component, request, env))
#     print(f'Component: {component.name()} failure: {failure_count}, request type: {request} from base arrived at MBK: {airline.name()} on time moment {env.now}')

    component.handled_from_base.put(1)
    yield env.timeout(airline.base_shipping_time)

    component.in_shipping_base_to_airline.get(1)

#     if request == "exchange":
#         component.in_repair.put(1)

#         env.process(repair_loop(failure_count, airline, component, request, env))

    if request == "forward_exchange" or request == "AOG":

        component.forward_reserved_stock.put(1)

        yield env.timeout(component.forward_exchange_until_repair())

        component.forward_reserved_stock.get(1)
        component.in_repair.put(1)

        env.process(repair_loop(failure_count, airline, component, request, env))

    elif request == "exchange_stock_replenishment":
        airline.stock.put(1)


def repair_loop(failure_count, airline, component, request, env):
    if component.on_lease.level > 0:

        if component.in_repair.level == 0 :
            print('error: no stock in repair')
        component.in_repair.get(1)

        if component.on_lease.level == 0 :
```

```
            print('error: no stock in lease')
        component.on_lease.get(1)

        #print(f'Component: {component.name()} failure: {failure_count} arived back to the leasing company on time moment {env.now}')
    else:

        yield env.timeout(component.repair_waiting_days()-7)
        component.last_repair_days.put(1)
        yield env.timeout(7)
        component.last_repair_days.get(1)

        component.in_repair.get(1)
        yield env.timeout(0.2) # Prioritize first for the next day
        DEPOT_BASE.stock.put(1)

        #print(f'Component: {component.name()} failure: {failure_count} from internal repair arrived at depot on time moment {env.now}')

def stock_check_base_1(failure_count, airline, component, request, env):
#     print('restocking: start waiting for a component')
    restocking_fulfilled = False
    yield env.timeout(component.repair_waiting_days())
    yield env.timeout(0.1) # Prioritize fourth
    while restocking_fulfilled == False :
#         print('restocking: start requesting a component')
        if DEPOT_BASE.stock.level > 2 :
#         if DEPOT_BASE.enough_inventory_depot:
#             print('error: restocking depot stock level:',DEPOT_BASE.stock.level)

            DEPOT_BASE.get()
            component.in_shipping_base_replenishment.put(1)

            yield env.timeout(airline.connected_base.restock_time())

            component.in_shipping_base_replenishment.get(1)
            airline.connected_base.stock.put(1)

            restocking_fulfilled = True
        else:
            yield env.timeout(1)



def results_components(component, env, configuration, scenario):
    running_year = 0
    while True:
        yield env.timeout(365)
        running_year += 1
        if running_year == 10 or running_year == 20:
            print('year:',running_year)
        if env.now == 364 or env.now == 365 or env.now == 366:
                if component.forward_exchange.level > 0:
                    component.forward_exchange.get(component.forward_exchange.level)
                if component.exchange.level > 0:
                    component.exchange.get(component.exchange.level)
                if component.stock_replenishment.level > 0:
```

```
                component.stock_replenishment.get(component.stock_replenishment.level)
            if component.leased.level > 0:
                component.leased.get(component.leased.level)
            if component.failures.level > 0:
                component.failures.get(component.failures.level)
            if component.on_time_handled.level > 0:
                component.on_time_handled.get(component.on_time_handled.level)
            if component.response_time_forward_exchange.level > 0:
                component.response_time_forward_exchange.get(component.response_time_forward_exchange.level)
            if component.response_time_stock_replenishment.level > 0:
                component.response_time_stock_replenishment.get(component.response_time_stock_replenishment.level)
            if component.handled_from_base.level > 0 :
                component.handled_from_base.get(component.handled_from_base.level)
            if component.handled_from_depot.level > 0:
                component.handled_from_depot.get(component.handled_from_depot.level)
            if component.handled_from_lease.level > 0 :
                component.handled_from_lease.get(component.handled_from_lease.level)
            if component.to_late_time_handled.level > 0 :
                component.to_late_time_handled.get(component.to_late_time_handled.level)
            if component.lease_days.level > 0 :
                component.lease_days.get(component.lease_days.level)
    if env.now > 400:
        year_stock = 0
        year_ROCE = 0
        year_ROS = 0
        year_operation_profit = 0
        initial_total_stock = 0
        cost = 0
        revenue = 0
        lease_cost = 0
        holding_cost = 0
        repair_cost = 0
        transport_cost = 0
        MBK_stock_level = 0
        initial_total_stock += DEPOT_BASE.initial_stock_level_depot()
        for base in BASES:
            year_stock += base.stock.level
            if base.is_base():
                initial_total_stock += base.initial_stock_level_base()
        for airline in AIRLINES:
            year_stock += airline.stock.level
        for component in COMPONENTS:
            year_stock += component.in_repair.level
            year_stock += component.in_shipping_base_replenishment.level
            year_stock += component.in_shipping_base_to_airline.level
            year_stock += component.in_shipping_depot_to_airline.level
            year_stock += component.forward_reserved_stock.level
            year_stock -= component.on_lease.level

            lease_cost += component.handled_from_lease.level * SETTINGS[2] * component.part_price()
            holding_cost += component.part_price() * initial_total_stock * SETTINGS[3]
            repair_cost += component.repair_price() * component.failures.level
            transport_cost += 272 * component.handled_from_depot.level + 272 * component.handled_from_base.level
```

```

#                 print('lease cost', lease_cost)
#                 print('holding cost', holding_cost)
#                 print('repair cost', repair_cost)
#                 print('transport cost', transport_cost)
                cost += lease_cost + holding_cost + repair_cost + transport_cost
                revenue = 0 #(component.part_price() * initial_total_stock /   ) * Revue per $
                year_operation_profit += revenue - cost
#                 print('cost', cost)
#                 print('revenue', revenue)
#                 print('operation profit', year_operation_profit)


                cost_total = lease_cost + holding_cost + repair_cost + transport_cost
                capital = initial_total_stock * component.part_price()
#                 print('ROS?', year_ROS)

                global results
                row_results = {'component':round(component.name()),
                'configuration':configuration,
                'scenario':scenario,
                'fe#':component.forward_exchange.level,
                'aog#':component.exchange.level,
                'sr#':component.stock_replenishment.level,
                'base#':component.handled_from_base.level,
                'depot#':component.handled_from_depot.level,
                'borrow#':component.handled_from_lease.level,
                'borrow day#':component.lease_days.level,
                'sr day#':component.response_time_stock_replenishment.level,
                'fe day#':component.response_time_forward_exchange.level,
                'removals#':component.failures.level,
                'sl%':round(component.on_time_handled.level / component.failures.level,4),
                'cost': round(cost_total,0),
                'capital':round(capital,0),
                'configuration':configuration,
                'scenario':scenario,

                'in lease':round(component.on_lease.level),
                'in repair':round(component.in_repair.level),
                'in ship base rep':round(component.in_shipping_base_replenishment.level),
                'in ship base':round(component.in_shipping_base_to_airline.level),
                'in ship depot':round(component.in_shipping_depot_to_airline.level),
                'in ship lease':round(component.in_shipping_leased_to_airline.level),
                'last repair days':round(component.last_repair_days.level),
                'reserved forward': component.forward_reserved_stock.level,
                'total stock': year_stock
                              }


                global results
                results = results.append(row_results, ignore_index=True)
                if component.forward_exchange.level > 0:
                    component.forward_exchange.get(component.forward_exchange.level)
                if component.exchange.level > 0:
                    component.exchange.get(component.exchange.level)
                if component.stock_replenishment.level > 0:
                    component.stock_replenishment.get(component.stock_replenishment.level)
```

```
                if component.leased.level > 0:
                    component.leased.get(component.leased.level)
                if component.failures.level > 0:
                    component.failures.get(component.failures.level)
                if component.on_time_handled.level > 0:
                    component.on_time_handled.get(component.on_time_handled.level)

#                 if component.extra_day_handled.level > 0:
#                     component.extra_day_handled.get(component.extra_day_handled.level)

                if component.response_time_forward_exchange.level > 0:
                    component.response_time_forward_exchange.get(component.response_time_forward_exchange.level)
                if component.response_time_stock_replenishment.level > 0:
                    component.response_time_stock_replenishment.get(component.response_time_stock_replenishment.level)



                if component.handled_from_base.level > 0 :
                    component.handled_from_base.get(component.handled_from_base.level)
                if component.handled_from_depot.level > 0:
                    component.handled_from_depot.get(component.handled_from_depot.level)
                if component.handled_from_lease.level > 0 :
                    component.handled_from_lease.get(component.handled_from_lease.level)
                if component.to_late_time_handled.level > 0 :
                    component.to_late_time_handled.get(component.to_late_time_handled.level)
                if component.lease_days.level > 0 :
                    component.lease_days.get(component.lease_days.level)

def on_lease_days(component, env):
    while True:
        if component.on_lease.level > 0:
            on_lease = component.on_lease.level
            component.lease_days.put(on_lease)
        yield env.timeout(1)
```

scripts/processes.py

```


configurations = ['klm_pooling', 'perfect_mtbur_pooling',
                  'perfect_repair_pooling', 'imperfect_removals', 'repair_dev',
                 'contracted_location', 'protection_levels', 'adding_components', 'inventory_change','inventory_config', 'inventory_sl']

for seperature_runs in range(0, 5, 1):
    for configuration in configurations:
        for scenario in range(1,9,1):
            print('test scenario', scenario)
            print('test configuration', configuration)
            if configuration == 'klm_pooling':
                "for validation"
                if scenario == 1:
                    run = True
                    data_components = pandas.read_excel(DATA_FILE, sheet_name='components')
                    data_component = data_components.to_numpy()
```

```
                    for column in range(0,10,1):
                        removals_per_components = ( data_component[column][10] / ( data_component[column][19] * data_component[column][3]))
                        data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                    data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners')
                    AIRLINERS_MATRIX = data_airliners.to_numpy()
                    scenario = 787
                else:
                    run = False
                run = False



            elif configuration == 'perfect_mtbur_pooling':
                if scenario <= 8:
                    run = True
                    data_components = pandas.read_excel(DATA_FILE, sheet_name='components_50')
                    data_component = data_components.to_numpy()
                    for column in [10, 17, 18, 19, 20, 21, 22]:
                        data_component[:,column] *= scenario
                    for column in range(0,10,1):
                        removals_per_components = ( data_component[column][10] / ( data_component[column][19] * data_component[column][3]))
                        data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                    data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners_50')
                    AIRLINERS_MATRIX = data_airliners.to_numpy()
                    for column in [7,8,9,10,11,12,13,14,15,16]:
                        AIRLINERS_MATRIX[:,column] *= scenario
                    scenario = scenario * 50
                else:
                    run = False
                run = False


            elif configuration == 'perfect_repair_pooling':
                if scenario <= 8:
                    run = True
                    data_components = pandas.read_excel(DATA_FILE, sheet_name='components_50')
                    data_component = data_components.to_numpy()
                    for column in [10, 17, 18, 19, 20, 21, 22]:
                        data_component[:,column] *= scenario

                    for column in range(0,10,1):
                        removals_per_components = ( data_component[column][10] / ( data_component[column][19] * data_component[column][3]))
                        data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                        data_component[column][11] = data_component[column][9]
                        data_component[column][12] = 1

                    data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners_50')
                    AIRLINERS_MATRIX = data_airliners.to_numpy()
                    for column in [7,8,9,10,11,12,13,14,15,16]:
                        AIRLINERS_MATRIX[:,column] *= scenario
                    scenario = scenario * 50

                else:
                    run = False
```

```
            run = False

        elif configuration == 'imperfect_removals':
            print('scenario test', scenario)
            if scenario == 1:
                run = True
                data_components = pandas.read_excel(DATA_FILE, sheet_name='components')
                data_component = data_components.to_numpy()
                for column in range(0,10,1):
                    removals_per_components = ( (1.05 * data_component[column][10]) / ( data_component[column][19] * data_component[column][3]))
                    data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners')
                AIRLINERS_MATRIX = data_airliners.to_numpy()
                scenario = 1.05
            elif scenario == 2:
                run = True
                data_components = pandas.read_excel(DATA_FILE, sheet_name='components')
                data_component = data_components.to_numpy()
                for column in range(0,10,1):
                    removals_per_components = ( data_component[column][10] / ( data_component[column][19] * data_component[column][3]))
                    data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners')
                AIRLINERS_MATRIX = data_airliners.to_numpy()
                scenario = 1.0
            elif scenario == 3:
                run = True
                data_components = pandas.read_excel(DATA_FILE, sheet_name='components')
                data_component = data_components.to_numpy()
                for column in range(0,10,1):
                    removals_per_components = ( (0.95 * data_component[column][10]) / ( data_component[column][19] * data_component[column][3]))
                    data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners')
                AIRLINERS_MATRIX = data_airliners.to_numpy()
                scenario = 0.95
            else:
                run = False
            run = False


        elif configuration == 'protection_levels':
            "Make the protection level adjustable"
            if scenario == 1:
                run = True
                data_components = pandas.read_excel(DATA_FILE, sheet_name='components')
                data_component = data_components.to_numpy()
                for column in range(0,10,1):
                    removals_per_components = ( data_component[column][10] / ( data_component[column][19] * data_component[column][3]))
                    data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners')
                AIRLINERS_MATRIX = data_airliners.to_numpy()
                scenario = 0.5
            elif scenario == 2:
                run = True
```

```python
                data_components = pandas.read_excel(DATA_FILE, sheet_name='components')
                data_component = data_components.to_numpy()
                for column in range(0,10,1):
                    removals_per_components = ( data_component[column][10] / ( data_component[column][19] * data_component[column][3]))
                    data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners')
                AIRLINERS_MATRIX = data_airliners.to_numpy()
                scenario = 1
            elif scenario == 3:
                run = True
                data_components = pandas.read_excel(DATA_FILE, sheet_name='components')
                data_component = data_components.to_numpy()
                for column in range(0,10,1):
                    removals_per_components = ( data_component[column][10] / ( data_component[column][19] * data_component[column][3]))
                    data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners')
                AIRLINERS_MATRIX = data_airliners.to_numpy()
                scenario = 1.5
            else:
                run = False
            run = False

        elif configuration == 'contracted_location':
            if scenario == 1:
                run = True
                data_components = pandas.read_excel(DATA_FILE, sheet_name='components_depot')
                data_component = data_components.to_numpy()
                for column in range(0,10,1):
                    removals_per_components = ( data_component[column][10] / ( data_component[column][19] * data_component[column][3]))
                    data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners_depot')
                AIRLINERS_MATRIX = data_airliners.to_numpy()
                scenario = -15
            elif scenario == 2:
                run = True
                data_components = pandas.read_excel(DATA_FILE, sheet_name='components')
                data_component = data_components.to_numpy()
                for column in range(0,10,1):
                    removals_per_components = ( data_component[column][10] / ( data_component[column][19] * data_component[column][3]))
                    data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners')
                AIRLINERS_MATRIX = data_airliners.to_numpy()
                scenario = 0.15
            elif scenario == 3:
                run = True
                data_components = pandas.read_excel(DATA_FILE, sheet_name='components_base')
                data_component = data_components.to_numpy()
                for column in range(0,10,1):
                    removals_per_components = ( data_component[column][10] / ( data_component[column][19] * data_component[column][3]))
```

```
                        data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                    data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners_base')
                    AIRLINERS_MATRIX = data_airliners.to_numpy()
                    scenario = 15
                else:
                    run = False
                run = False


            elif configuration == 'adding_components':
                print('scenario:', scenario)
                if scenario <= 5:
                    adding_components = [0, -2, -1, 0 , 1 , 2]
                    run = True
                    data_components = pandas.read_excel(DATA_FILE, sheet_name='components')
                    data_component = data_components.to_numpy()
                    for column in range(0,10,1):
                        removals_per_components = ( (1 * data_component[column][10]) / ( data_component[column][19] * data_component[column][3]))
                        data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                    data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners')
                    AIRLINERS_MATRIX = data_airliners.to_numpy()
                    scenario = adding_components[scenario]
                else:
                    run = False
                run = False

            elif configuration == 'inventory_change':
                """Changing the square root of the inventory management model"""
                if scenario <= 6:
                    run = True
                    data_components = pandas.read_excel(DATA_FILE, sheet_name='components_50')
                    data_component = data_components.to_numpy()
                    for column in [10, 17, 18, 19, 20, 21, 22]:
                        data_component[:,column] *= scenario

                    for column in range(0,10,1):
                        removals_per_components = ( data_component[column][10] / ( data_component[column][19] * data_component[column][3]))
                        data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                        data_component[column][11] = data_component[column][9]
                        data_component[column][12] = 1

                    data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners_50')
                    AIRLINERS_MATRIX = data_airliners.to_numpy()
                    for column in [7,8,9,10,11,12,13,14,15,16]:
                        AIRLINERS_MATRIX[:,column] *= scenario
                    scenario = scenario * 50

                else:
                    run = False
                run = False

            elif configuration == 'inventory_config':

#                  if scenario == 1:
#                      run = True
```

```
#                     data_components = pandas.read_excel(DATA_FILE, sheet_name='components_config')
#                     data_component = data_components.to_numpy()
#                     for column in range(0,10,1):
#                         removals_per_components = ( data_component[column][10] / ( data_component[column][19] * data_component[column][3]))
#                         data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
#                         data_component[column][11] = data_component[column][9]
#                         data_component[column][12] = 1
#                     data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners')
#                     AIRLINERS_MATRIX = data_airliners.to_numpy()
#                     scenario = 999
#                 else:
#                     run = False
#                 run = False
                if scenario <= 8:
                    run = True
                    data_components = pandas.read_excel(DATA_FILE, sheet_name='components_50')
                    data_component = data_components.to_numpy()
                    for column in [10, 17, 18, 19, 20, 21, 22]:
                        data_component[:,column] *= scenario

                    for column in range(0,10,1):
#                         data_component[column][10] = 4 * data_component[column][10]
                        removals_per_components = ( data_component[column][10] / ( data_component[column][19] * data_component[column][3]))
                        data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                        data_component[column][11] = data_component[column][9]
                        data_component[column][12] = 1
                        data_component[column][8] = 0.7

                    data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners_50')
                    AIRLINERS_MATRIX = data_airliners.to_numpy()
                    for column in [7,8,9,10,11,12,13,14,15,16]:
                        AIRLINERS_MATRIX[:,column] *= scenario
                    scenario = scenario * 50
                else:
                    run = False
                run = False


            elif configuration == 'inventory_sl':
                if scenario <= 8:
                    run = True
                    data_components = pandas.read_excel(DATA_FILE, sheet_name='components_50')
                    data_component = data_components.to_numpy()
                    for column in [10, 17, 18, 19, 20, 21, 22]:
                        data_component[:,column] *= scenario

                    for column in range(0,10,1):
#                         data_component[column][10] = 4 * data_component[column][10]
                        removals_per_components = ( data_component[column][10] / ( data_component[column][19] * data_component[column][3]))
                        data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                        data_component[column][11] = data_component[column][9]
                        data_component[column][12] = 1
```

```
                        data_component[column][8] = 0.7

                    data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners_50')
                    AIRLINERS_MATRIX = data_airliners.to_numpy()
                    for column in [7,8,9,10,11,12,13,14,15,16]:
                        AIRLINERS_MATRIX[:,column] *= scenario
                    scenario = scenario * 50

                else:
                    run = False
                run = False



            elif configuration == 'repair_dev':
                if scenario <= 8:
                    run = True
                    data_components = pandas.read_excel(DATA_FILE, sheet_name='components_50')
                    data_component = data_components.to_numpy()
                    for column in [10, 17, 18, 19, 20, 21, 22]:
                        data_component[:,column] *= scenario
                    for column in range(0,10,1):
                        removals_per_components = ( data_component[column][10] / ( data_component[column][19] * data_component[column][3]))
                        data_component[column][4] = ( 1 / removals_per_components ) * 365 * 12
                        data_component[column][12] = 1
                    data_airliners = pandas.read_excel(DATA_FILE, sheet_name='airliners_50')
                    AIRLINERS_MATRIX = data_airliners.to_numpy()
                    for column in [7,8,9,10,11,12,13,14,15,16]:
                        AIRLINERS_MATRIX[:,column] *= scenario
                    scenario = scenario * 50
                else:
                    run = False
#                 run = False


            else:
                run = False

    # COMPONENTS_MATRIX = data_components.to_numpy()
    # Column
    # 0 = Name component
    # 1 = Code name component
    # 2 = Price thousand Euro's
    # 3 = QPA (Quantity Per Aircraft) in number
    # 4 = MTBR (Mean Time Between Repairs) in hours
    # 5 = repair cost in thousand Euro's
    # 6 = ess
    # 7 = mel
    # 8 = SL
    # 9 = Theoretical TAT
    # 10 = 12 month removals
    # 11 = average repair days
    # 12 = repair time deviation
    # 13 = % forward exchange
    # 14 = Wating days forward exchange
    # 15 = Stock relative to calculation
```

```
    # 16 = Real MTBR
    # 17 = eff 2018
    # 18 = eff 2019
    # 19 = eff 2020




            for row_component in range(10):
                COMPONENTS_MATRIX = [data_component[row_component]]

                class Component_spec:
                    def __init__(self, row, configuration, scenario):
                        env.process(results_components(self, env, configuration, scenario))
#                         env.process(on_lease_days(self, env))

                        self.row = row
                        self.failures = simpy.Container(env, init = 0)
                        self.forward_exchange = simpy.Container(env, init = 0)
                        self.exchange = simpy.Container(env, init = 0)
                        self.stock_replenishment = simpy.Container(env, init = 0)
                        self.response_time_forward_exchange = simpy.Container(env, init = 0)
                        self.response_time_stock_replenishment = simpy.Container(env, init = 0)
                        self.handled_from_depot = simpy.Container(env, init = 0)
                        self.handled_from_lease = simpy.Container(env, init = 0)
                        self.handled_from_base = simpy.Container(env, init = 0)
                        self.on_lease = simpy.Container(env, init = 0)
                        self.leased = simpy.Container(env, init = 0)
                        self.lease_days = simpy.Container(env, init = 0)
                        self.in_repair = simpy.Container(env, init= 0)
                        self.in_shipping_base_replenishment = simpy.Container(env, init= 0)
                        self.in_shipping_base_to_airline = simpy.Container(env, init= 0)
                        self.in_shipping_depot_to_airline = simpy.Container(env, init= 0)
                        self.in_shipping_leased_to_airline = simpy.Container(env, init= 0)
                        self.last_repair_days = simpy.Container(env, init= 0)
                        self.on_time_handled = simpy.Container(env, init= 0)
                        self.to_late_time_handled = simpy.Container(env, init= 0)
                        self.forward_reserved_stock = simpy.Container(env, init = 0)

                        self.total_response_time_forward_exchange = simpy.Container(env, init = 0)
                        self.total_response_time_stock_replenishment = simpy.Container(env, init = 0)

                        self.on_time_forward_exchange = simpy.Container(env, init = 0)
                        self.on_time_stock_replenishment = simpy.Container(env, init = 0)

                        self.list_response_forward_exchange = []
                        self.list_response_stock_replenishment = []

                    def name_real(self):
                        return self.row[0]


                    def name(self):
                        return self.row[1]
```

```

                def part_price(self):
                    return self.row[2]

                def repair_price(self):
                    return self.row[5]

                def code(self):
                    return self.row[2]

                def chance_forward_exchange(self):
                    return self.row[13]

                def forward_exchange_waiting(self):
                    return round(self.row[14], 0)

                def forward_exchange_until_repair(self):
                    return 5

                def qpa(self):
                    return self.row[3]

                def mtbr(self):
                    return self.row[4]

                def repair_waiting_days(self):
                    return abs(round(numpy.random.normal(loc=self.row[11], scale=self.row[12], size=None)))

                def stock_replenishment_waiting_days(self):
                    return SETTINGS[11]

                def ess(self):
                    return self.row[6]

                def mel(self):
                    return self.row[7]

                def sl(self):
                    return self.row[8]

                def sl_mbk(self, configuration, scenario):
                    if self.name() == 870056:
                        protectionlevel = 0.4
                    if self.name() == 870087:
                        protectionlevel = 0.4
                    if self.name() == 870180:
                        protectionlevel = 0.8
                    if self.name() == 870261:
                        protectionlevel = 0.4
                    if self.name() == 871010:
                        protectionlevel = 0.4
                    if self.name() == 871754:
                        protectionlevel = 0.4
                    if self.name() == 871915:
                        protectionlevel = 0.35
                    if self.name() == 872318:
```

```
            protectionlevel = 0.2
        if self.name() == 872517:
            protectionlevel = 0.4
        if self.name() == 888003:
            protectionlevel = 0.6

        if configuration == 'protection_levels':
            protectionlevel = scenario * protectionlevel

        if configuration == 'inventory_sl':
            protectionlevel = 0.4

        if protectionlevel > 0.99:
            protectionlevel = 0.99

        return protectionlevel





    def theoretic_tat(self):
        return self.row[9]

    def removal_last_12mon(self):
        return self.row[10]

    def real_stock_diff(self):
        return self.row[15]

    def eff_2018(self):
        return self.row[17]

    def eff_2019(self):
        return self.row[18]

    def eff_2020(self):
        return self.row[19]

    def eff_kul(self):
        return self.row[20]

    def eff_rmia(self):
        return self.row[21]

    def eff_sha(self):
        return self.row[22]

    def eff_total(self):
        return self.row[10]

    def ess_waiting_days(self):
        if self.ess() == 1:
            return SETTINGS[6]
        elif self.ess() == 2:
            if self.mel() == "A":
                return SETTINGS[7]
```

```
            elif self.mel() == "B":
                return SETTINGS[8]
            elif self.mel() == "C":
                return SETTINGS[9]
            else:
                return SETTINGS[10]
        else:
            return SETTINGS[11]




class Base:
    def __init__(self, row, components, configuration, scenario):
        self.row = row
        self.component = components[0]

    def initial_stock_level_depot(self):
        stock_diff = 0
        if configuration == 'klm_pooling':
            stock_diff = stock_diff = round(self.component.real_stock_diff(), 0)
        if configuration == 'adding_components':
            stock_diff = scenario
        init_value = max(math.ceil(self.OLV_depot()),0) + stock_diff
        return init_value

    def OLV_depot(self):
        sl = self.component.sl()
        aip_dep = self.average_in_process_depot()
        if configuration == 'inventory_change':
            sqr = 56
            return norm.ppf(sl, loc= aip_dep, scale = numpy.power(aip_dep, sqr / 100))
        else:
#             print('AIP',aip_dep)
            return norm.ppf(sl, loc= aip_dep, scale = math.sqrt(aip_dep))

    def average_in_process_depot(self):
        value = ( self.component.eff_total() * self.component.theoretic_tat() / 365)
#         print('AIP',value)
        return value

    def initial_stock(self):
        return max(math.ceil(self.OLV_base()),0)

    def base_failures(self):
        base_airline_ratio = self.connected_airlines / self.component.eff_total()
        total_failures = self.component.removal_last_12mon()
        base_failures = base_airline_ratio * total_failures
        return base_failures

    def check_AOG(self):
        is_AOG = self.component.ess() == 1
        return is_AOG and self.base_failures() > 2
```

```

                def check_MEL_AB(self):
                    is_AB = self.component.mel() == "A" or self.component.mel() == "B"
                    is_ESS2 = self.component.ess() == 2
                    return is_AB and is_ESS2 and self.base_failures() > 2

                def check_MEL_CD(self):
                    is_CD = self.component.mel() == "C" or self.component.mel() == "D"
                    is_ESS2 = self.component.ess() == 2
                    return is_CD and is_ESS2 and self.base_failures() > 5

                def check_ESS3(self):
                    return self.component.ess() == 3

                def initial_stock_level_base(self):
                    if self.check_AOG():
                        return self.initial_stock()
                    elif self.check_MEL_AB():
                        return self.initial_stock()
                    elif self.check_MEL_CD():
                        return self.initial_stock()
                    elif self.check_ESS3:
                        return max(round(self.OLV_base(),0) ,0)
                    else:
                        print('Error')

                def OLV_base(self):
                    sl = self.component.sl()
                    aip_bas = self.average_in_process_base()
                    if configuration == 'inventory_change':
                        sqr = 56
                        return norm.ppf(sl, loc= aip_bas, scale = numpy.power(aip_bas, sqr / 100))
                    else:
                        return norm.ppf(sl, loc= aip_bas, scale = math.sqrt(aip_bas))

                def average_in_process_base(self):
                    if self.name() == "RMI":
                        self.base_effectivity = self.component.eff_rmia()
                    if self.name() == "RKL":
                        self.base_effectivity = self.component.eff_kul()
                    if self.name() == "SHA":
                        self.base_effectivity = self.component.eff_sha()
                    a = ( self.base_effectivity / self.component.eff_2020() )
                    b = a * self.stock_calculated_base_travel_time() * self.component.eff_total()
                    c = b / 365
                    self.average_processing_base = c
                    return c


                def stock_calculated_base_travel_time(self):
                    return 7

                def set_connected_airlines(self, airlines):
                    self.connected_airlines = 0
                    # TODO: refactor
```

```
                    self.total_airlines = airlines[0].eff_total()
                    for airline in airlines:
                        if airline.connected_base and airline.connected_base.name() ==
self.name():
                            self.connected_airlines += airline.num_aircrafts()

                def get(self):
                    if DEPOT_BASE.stock.level == 0:
                        print('Error: no stock in depot')
                    DEPOT_BASE.stock.get(1)

                def enough_inventory_base(self):
                    return self.stock.level > self.initial_stock_level_base()

                def enough_inventory_depot(self):
                    return DEPOT_BASE.stock.level > 1

                def name(self):
                    return self.row[1]

                def is_base(self):
                    return self.row[1] != DEPOT

                def restock_time(self):
                    return self.row[2]

                def stock_available(self):
                    return self.stock.level > 0

                def init_stock(self):
                    if self.is_base():
                        self.stock = simpy.Container(env, init = self.
initial_stock_level_base())
                    else:
                        self.stock = simpy.Container(env, init = self.
initial_stock_level_depot())











            class Airline:
                def __init__(self, row, bases, components, configuration, scenario):
                    self.component = components[0]
                    self.row = row
                    self.base_shipping_time, self.connected_base = self._calculate_base(
bases)


                def initial_stock_level_mbk(self):
#                     init_value_mbk = max(math.ceil(self.olv_mbk()), 0)
                    init_value_mbk = round(math.ceil(self.olv_mbk()), 0)
```

```
                    return init_value_mbk

                def olv_mbk(self):
                    a = norm.ppf(self.component.sl_mbk(configuration, scenario),
                                 loc= self.average_in_process_mbk() , scale = math.sqrt(self.average_in_process_mbk()))
                    return a

                def average_in_process_mbk(self):

                    if self.connected_base:
                        airplanes = self.eff_base()
                    else:
                        airplanes = self.eff_total()

                    a = self.component.removal_last_12mon() * self.stock_calc_mbk_travel_time()
                    b = a * self.num_aircrafts()
                    c = b / (365 * airplanes)
                    return c

                def stock_calc_mbk_travel_time(self):
                    #TODO: check if the travel time from the base to the airline 2 days is
                    return 2

                def eff_base(self):
                    return self.connected_base.connected_airlines

                def eff_total(self):
                    eff_total = 0
                    if self.component.name() == 870056:
                        for airline in AIRLINERS_MATRIX:
                            eff_total += self.row[7]
                        return eff_total
                    if self.component.name() == 870087:
                        for airline in AIRLINERS_MATRIX:
                            eff_total += self.row[8]
                        return eff_total
                    if self.component.name() == 870180:
                        for airline in AIRLINERS_MATRIX:
                            eff_total += self.row[9]
                        return eff_total
                    if self.component.name() == 870261:
                        for airline in AIRLINERS_MATRIX:
                            eff_total += self.row[10]
                        return eff_total
                    if self.component.name() == 871010:
                        for airline in AIRLINERS_MATRIX:
                            eff_total += self.row[11]
                        return eff_total
                    if self.component.name() == 871754:
                        for airline in AIRLINERS_MATRIX:
                            eff_total += self.row[12]
                        return eff_total
                    if self.component.name() == 871915:
                        for airline in AIRLINERS_MATRIX:
                            eff_total += self.row[13]
```

```
            return eff_total
        if self.component.name() == 872318:
            for airline in AIRLINERS_MATRIX:
                eff_total += self.row[14]
            return eff_total
        if self.component.name() == 872517:
            for airline in AIRLINERS_MATRIX:
                eff_total += self.row[15]
            return eff_total
        if self.component.name() == 888003:
            for airline in AIRLINERS_MATRIX:
                eff_total += self.row[16]
            return eff_total

    def init_stock(self):
        init = 0
        if self.num_aircrafts() != 0:
            init = self.initial_stock_level_mbk()
        self.stock = simpy.Container(env, init=init)

    def stock_get(self):
        if self.stock == 0:
            print('Error: no stock in MBK')
        self.stock.get(1)

    def _calculate_base(self, bases):
        base_name = ""
        if self.row[4] > 0 :
            base_name = "RMI"
            base_shipping_time = self.row[4]
        if self.row[5] > 0 :
            base_name = "RKL"
            base_shipping_time = self.row[5]
        if self.row[6] > 0 :
            base_name = "SHA"
            base_shipping_time = self.row[6]

        if not base_name:
            return None, None

        for base in bases:
            if base.name() == base_name:
                return base_shipping_time, base

        print(base_shipping_time)
        return None, None

    def num_aircrafts(self):
        if self.component.name() == 870056:
            return self.row[7]
        if self.component.name() == 870087:
            return self.row[8]
        if self.component.name() == 870180:
            return self.row[9]
        if self.component.name() == 870261:
            return self.row[10]
        if self.component.name() == 871010:
```

```
                        return self.row[11]
                    if self.component.name() == 871754:
                        return self.row[12]
                    if self.component.name() == 871915:
                        return self.row[13]
                    if self.component.name() == 872318:
                        return self.row[14]
                    if self.component.name() == 872517:
                        return self.row[15]
                    if self.component.name() == 888003:
                        return self.row[16]
                    else:
                        print('Error: No corresponding code')

                def flight_hours_airline(self):
                    return self.row[2]

                def shipping_depot_airline(self):
                    return self.row[3]

                def name(self):
                    return self.row[1]


            if run == True:
                print('Start order fulfillment at component services')
                random.seed(RANDOM_SEED)

                env = simpy.Environment()
                DEPOT = "AMS"

                COMPONENTS = [ Component_spec(row, configuration, scenario) for row in COMPONENTS_MATRIX ]
                BASES = [ Base(row, COMPONENTS, configuration, scenario) for row in LOCATIONS_MATRIX ]
                DEPOT_BASE = BASES[0]
                AIRLINES = [ Airline(row, BASES, COMPONENTS, configuration, scenario) for row in AIRLINERS_MATRIX ]
                for base in BASES:
                    base.set_connected_airlines(AIRLINES)

                    base.init_stock()

                for Airline in AIRLINES:
                    Airline.init_stock()


                for component in COMPONENTS:
                    for airline in AIRLINES:
                        for times in range( airline.num_aircrafts() * component.qpa()):
                            env.process(failure_generator(component, airline, env))


                year_stock = 0
                base_stock = 0
                depot_stock = 0
                depot_stock = DEPOT_BASE.stock.level
```

```
                airline_stock = 0
                for base in BASES:
                    if base.name() == 'RMI':
                        RMI_OLV = base.OLV_base()
                        RMI_AIP = base.average_in_process_base()
                        RMI_planning = base.initial_stock_level_base()
                    if base.name() == 'RKL':
                        RKL_OLV = base.OLV_base()
                        RKL_AIP = base.average_in_process_base()
                        RKL_planning = base.initial_stock_level_base()
                    year_stock += base.stock.level
                    base_stock += base.stock.level
                base_stock -= DEPOT_BASE.stock.level
                for airline in AIRLINES:
                    year_stock += airline.stock.level
                    airline_stock += airline.stock.level
                for component in COMPONENTS:
                    year_stock += component.in_repair.level
                    year_stock += component.in_shipping_base_replenishment.level
                    year_stock += component.in_shipping_base_to_airline.level
                    year_stock += component.in_shipping_depot_to_airline.level
                    year_stock += component.forward_reserved_stock.level
                    year_stock -= component.on_lease.level
                    global results
                    row_results = {'component':round(component.name()),

                    'in lease':round(component.on_lease.level),
                    'leased':round(component.leased.level),
                    'lease days':round(component.lease_days.level),
                    'in repair':round(component.in_repair.level),
                    'in ship base rep':round(component.in_shipping_base_replenishment.
level),
                    'in ship base':round(component.in_shipping_base_to_airline.level),
                    'in ship depot':round(component.in_shipping_depot_to_airline.level),
                    'in ship lease':round(component.in_shipping_leased_to_airline.level),
                    'last repair days':round(component.last_repair_days.level),
                    'reached requests':round(component.on_time_handled.level),
                    'not reached requests': component.to_late_time_handled.level,
                    'reserved forward': component.forward_reserved_stock.level,
                    'depot stock': depot_stock,
                    'base stock': base_stock,
                    'airline stock': airline_stock,
                    'total stock': year_stock,



                    'AIP': DEPOT_BASE.average_in_process_depot(),
                    'Planning':DEPOT_BASE.OLV_depot(),
                    'AIP_RMI':RMI_AIP,
                    'OLV_RMI':RMI_OLV,
                    'planning_RMI': RMI_planning,
                    'AIP_RKL': RKL_AIP ,
                    'OLV_RKL': RKL_OLV ,
                    'planning_RKL': RKL_planning,

                    'scenario': scenario,
                    'configuration': configuration,
```

```python
                        'name':component.name_real()

                            }
                    results = results.append(row_results, ignore_index=True)
                    env.run( until = 10*365+1)
                    print('Simulation finished')
                    end_total_stock = 0

                    for component in COMPONENTS:
                        end_total_stock += component.in_repair.level
                        end_total_stock += component.in_shipping_base_replenishment.level
                        end_total_stock += component.in_shipping_base_to_airline.level
                        end_total_stock += component.in_shipping_depot_to_airline.level
                        end_total_stock += component.in_shipping_leased_to_airline.level
                        end_total_stock -= component.on_lease.level

                    for base in BASES:
                        end_total_stock += base.stock.level
                    for airline in AIRLINES:

                        end_total_stock += airline.stock.level

                    end_total_stock = 0
                    print('configuration:', configuration)
                    print('Scenario:', scenario)
                    print('Component:', component.name())
        print('Simulation finished')
    print(results)
results.to_excel("results.xlsx", sheet_name='trial')
```

scripts/Initiate_classes.py