Delft University of Technology

FACULTY MECHANICAL, MARITIME AND MATERIALS ENGINEERING
Department Marine and Transport Technology

Mekelweg 2
2628 CD Delft
the Netherlands
Phone +31 (0)15-2782889
Fax +31 (0)15-2781397
www.mtt.tudelft.nl

Specialization: Transport Engineering and Logistics

Report number: 2016.TEL.8021

Title: **Methodical design of baggage handling system concepts at airports.**

Author: T.J.W. Bentvelsen

Title (in Dutch) Methodisch ontwerpen van bagageafhandelingssysteem concepten voor luchthavens.

Assignment: Master thesis

Confidential: yes (until January 1, 2023)

Initiator (university): Dr. W.W.A. Beelaerts van Blokland

Initiator (company): B.S. ten Berge

Supervisor: Dr. W.W.A. Beelaerts van Blokland

Date: April 31, 2016

Delft University of Technology

FACULTY OF MECHANICAL, MARITIME AND MATERIALS ENGINEERING
Department of Marine and Transport Technology

Mekelweg 2
2628 CD Delft
the Netherlands
Phone +31 (0)15-2782889
Fax +31 (0)15-2781397
www.mtt.tudelft.nl

Student:	T.J.W. Bentvelsen
Supervisor (TUD):	Dr. W.W.A. Beelaerts van Blokland
Supervisor (Company):	B.S. ten Berge
Assignment type:	Master project
Creditpoints (EC):	35
Specialization:	TEL
Report number:	2016.TEL.8021
Confidential:	Yes, until: January 01, 2023

Subject: **Methodical design of airport baggage handling system concepts**

Modern day airports exist in various sizes and types, as is classified by the International Air Transport Association (Bradley, 2010). From local airports to regional airports and intercontinental hub airports, each requiring baggage handling facilities. These facilities may already be existent and need a capacity expansion whilst in other cases a design is made from scratch. In order to understand the complexity of baggage handling systems at airports, such systems may be divided into multiple functions. Several suppliers provide solutions for baggage handling processes. Among these suppliers is Siemens Postal, Parcel & Airport Logistics, the initiator of this assignment.

Prior research on baggage handling system design has primarily focused on the tracking of design requirements in Microsoft Office Excel forms (Lemain, 2002). Manuals have been written on airport terminal design, in which chapters elaborate on baggage handling systems (Bradley, 2010). These manuals are supported by generic design recommendations from the International Air Transport Association. Based on the recommended systems, a tool may be used to optimize system layouts (Grigoraş & Hoede, 2007). A design process for baggage handling systems however has not been found in literature.

Your assignment is to find a design process description for designing baggage handling system concepts at airports and use this to construct a model. An example of a design process is described by Pielage (2005), who has developed a design process for complex automated freight transport systems, a group that baggage handling systems are part of.

Of particular interest are:

- identifying existing baggage handling sub-systems
- defining design process criteria
- comparing the selected design process and current design process in a case study

The report should comply with the guidelines of the section. Details can be found on the website.

The professor,

Prof. dr. ir. G. Lodewijks

Methodical design of baggage handling system concepts at airports

T. J. W. Bentvelsen

Faculty of Mechanical Engineering

TU Delft Delft University of Technology

Challenge the future

Methodical design of baggage handling system concepts at airports

by

T. J. W. Bentvelsen

in partial fulfilment of the requirements for the degree of

Master of Science
in Mechanical Engineering

at the Delft University of Technology,
to be defended publicly on Thursday the 19th of May 2016, at 10:00 AM.

Supervisor:	Dr. W. W. A. Beelaerts van Blokland	
Thesis committee:	Prof. dr. ir. G. Lodewijks,	TU Delft
	Dr. W. W. A. Beelaerts van Blokland,	TU Delft
	B. S. ten Berge,	SPPAL

This thesis is confidential and cannot be made public until the 1st of January 2023.

An electronic version of this thesis is available at http://repository.tudelft.nl/.

Delft University of Technology

PREFACE

I started the master specialisation Transport Engineering and Logistics to gain knowledge on production systems. The master specialisation had just merged with Production Engineering and promised to provide both a systematic view on systems as well as designing mechanical devices in detail. Fuelled by watching numerous "How It's Made" videos on Discovery Channel, this seemed like a good opportunity to expand my knowledge on Mechanical Engineering with automated production system design. A multitude of courses in the master specialisation was available. One course that specifically captured my interest was Advanced Design of Baggage Handling Systems (ME1430). In this course, our group was tasked with designing a baggage retrieval system for passengers.

Several months had passed and it was time to find a graduation subject. When looking back at the baggage handling system design process, something felt out of place. Deciding which sub-systems to use was not based on measurable performance indicators since simulation typically occurs after concepts have been conceived. This leads to simulating several most-promising designs, but it would take far too long to simulate every concept in the entire solution space. This realisation sparked my curiosity and I wondered how this is done in companies. I therefore contacted several baggage handling system suppliers. Some never bothered to respond, some disregarded my notion in favour of their own, but Bart ten Berge almost immediately responded with his view on the subject. My subject had (nearly) been conceived.

The difficulty with such a broad subject is that one could sink a lot of time into the project without getting any real results. This was also emphasized by Prof.dr.ir. G. Lodewijks, who advised me to narrow down parts the assignment. It eventually led to the current research questions and framework. The first several chapters of this thesis are dedicated to a literature survey and design process assessment. The remaining chapters focus on providing a generic model for baggage handling to determine the values of several performance indicators. The companies design process and proposed design process are compared by means of a case study. Conclusions on this are drawn in the final chapter.

I personally would like to thank Bart ten Berge for being there for me every step of the way. Finalising this thesis did not come easy to me and his support is valued greatly. Prof.dr.ir G. Lodewijks and Dr. W.W.A. Beelaerts van Blokland provided critical reviews throughout the project's duration, sometimes to much frustration. I do however like to thank them for their criticism, as it has made this report to what it is now. Furthermore I would like to thank Jaimy and Matthew for checking the report and hopefully reduce the number of misspelt words and broken sentences to zero. Lastly I would like to thank everyone for their aid when I needed it most.

T. J. W. Bentvelsen
Rotterdam, April 2016

Abstract

Although the importance of baggage handling systems is not widely recognised by its many daily users, they are a critical part of airports. A passenger only experiences handing in baggage and reclaiming it, but there is more than meets the eye. In these complex systems, baggage is collected from multiple sources, sorted, stored and redistributed. This is achieved by different devices which are connected to one another.

As population growth and urbanisation are maintained, centralised around Asia and Africa, the demand for air traffic and baggage handling increases. Since no specific design process for baggage handling systems has been found in literature and design choices are based on designer intuition, the following research question is proposed: How may key performance indicators be determined for baggage handling system concept designs based on flight schedule demand? Defining a design process is crucial to this research question.

A literature survey has shown a total of six design methods that may be applicable to baggage handling system concept design. To determine which method is preferred, eleven criteria are defined and ranked by baggage handling system experts in a pairwise manner with use of the analytic hierarchy process. Independently, the design methods are compared to one another in a pairwise manner on each of the eleven criteria. By multiplying both the criteria ranking and the method ranking, it becomes clear that the design method by Pielage (2005) is preferred. This preference is also verified by performing a sensitivity analysis. Results for this analysis are depicted in Table 1.

As a generic model for baggage handling systems is not found in literature, six cases are assessed to determine the system boundaries, functions and interconnections. Defining these properties is part of the first three process steps in Pielage's design method. The remaining two steps, simulation and evaluation, have not been conducted. The generic model for baggage handling systems that follows from applying the selected design method is illustrated in Figure 1. A digitalised version of the model is programmed in Microsoft Office Excel and the functionality of the program has been verified by comparing manually calculated outputs and model outputs. It should be noted that this is only done for manually verifiable inputs.

To assess the practical applicability of the design process and model, two case studies have been performed with data from [redacted]. The first case shows that there is a difference between the model-generated concept and the company-developed concept. However when disregarding the sorting function, both systems are fairly similar. In the second case, both the model-generated and company-developed concepts did not show similarities. A possible explanation for this difference may be found in several missing requirements. In both cases, design time was reduced when using Pielage's design method. The significance of this result cannot be tested as only two case studies have been conducted.

The conclusion that may be drawn from this research is that Pielage's design method, as preferred by experts, is suitable for application in the baggage handling system concept design phase. Assessment of several

Table 1: Final results and sensitivity analysis to the method comparison. Names of criteria and methods have been abbreviated and may be found in chapter 3 of this thesis.

	Standard	AL + 200%	CO + 200%	FL + 200%	IN + 200%	IT + 200%	LC + 200%	PD + 200%	PI + 200%	RA + 200%	SI + 200%	DS + 200%
VDI	0,09	0,11	0,08	0,08	0,09	0,09	0,08	0,12	0,07	0,09	0,09	0,08
R&E	0,06	0,06	0,06	0,07	0,06	0,08	0,06	0,06	0,05	0,06	0,06	0,06
LEM	0,15	0,14	0,20	0,15	0,16	0,13	0,17	0,14	0,14	0,16	0,18	0,14
PIE	0,20	0,21	0,20	0,21	0,18	0,22	0,21	0,21	0,17	0,17	0,18	0,20
G&H	0,15	0,14	0,13	0,14	0,14	0,13	0,14	0,15	0,17	0,18	0,15	0,16
VIA	0,18	0,17	0,15	0,17	0,16	0,15	0,16	0,17	0,23	0,18	0,18	0,21
SIE	0,18	0,17	0,18	0,19	0,20	0,19	0,18	0,16	0,18	0,16	0,16	0,15

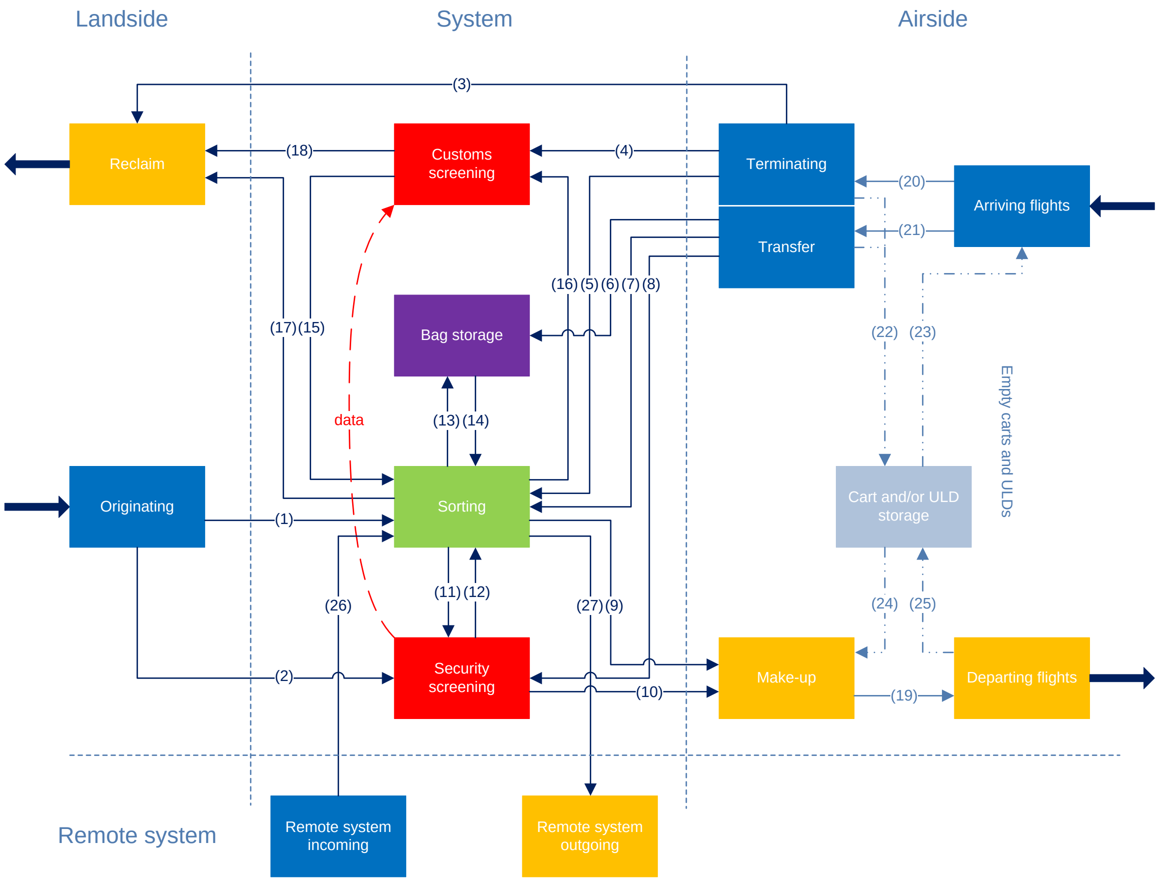


Figure 1: A schematic overview of a generic model for baggage handling system functions and connections. The model may be divided into landside, airside and system. It should also be noted that the sorting system may connect to a remote system that is already present or will be built in a future expansion.

design cases shows recurring functions from which a generic model may be established. Preliminary results on the model's application in two design cases are promising however, more design cases are necessary to either fully validate or reject the model. With this conclusion this research contributes to concept design theory for baggage handling systems. The design method and model may be generalisable for both postal and parcel systems, as these systems show similarities to baggage handling systems on functionality, connectivity and integral parts. Future research may further elaborate on this insight.

Abstract (Dutch)

Ondanks dat het belang van bagageafhandelingssystemen op luchthavens niet altijd herkend wordt door de vele dagelijkse reizigers, zijn deze systemen een cruciaal onderdeel van luchthavens. Hoewel de reiziger alleen het inleveren en ophalen van zijn of haar bagage meemaakt, schuilt er meer achter dit proces. In deze complexe systemen wordt bagage namelijk vanuit meerdere punten verzameld, gesorteerd, opgeslagen en weer herverdeeld. Dit gebeurt door meerdere verschillende apparaten die allen in contact staan met elkaar.

Door de aanhoudende bevolkingsgroei en verstedelijking, met name in Azië en Afrika, neemt de vraag naar luchtverkeer en dus ook bagageafhandeling alleen maar toe. Omdat er voor bagageafhandelingssystemen geen ontwerp proces is gevonden in de literatuur en ontwerpkeuzes op de intuïtie van ontwerpers neerkomen, wordt in deze scriptie ingegaan op de vraag: hoe kunnen performance indicatoren van bagageafhandelingssysteem concept ontwerpen worden bepaald met een vluchtplan als basis? Hierbij moet rekening worden gehouden met het achterliggende ontwerpproces.

In een literatuuronderzoek zijn vervolgens zes verschillende ontwerpmethoden gevonden. Om vast te stellen welk van deze methodes de voorkeur heeft onder bagageafhandelingssysteem experts, zijn een elftal criteria opgesteld. Een aantal experts is gevraagd de criteria op paarsgewijze manier te vergelijken met behulp van het analytisch hiërarchisch proces. Hieruit volgt een rangorde van criteria. Onafhankelijk van deze rangorde zijn de gevonden ontwerpmethoden en huidige ontwerpmethode paarsgewijs vergeleken op ieder van de elf criteria. Door de waardering van de criteria te vermenigvuldigen met de waardering van methoden, wordt duidelijk welke methode de voorkeur heeft. Om het resultaat van deze analyse te verifiëren is een gevoeligheidsanalyse uitgevoerd. Hieruit volgt dat de ontwerpmethode van Pielage (2005) de voorkeur heeft. Dit is aangegeven in Tabel 2.

Omdat er in het literatuur onderzoek geen generiek model is gevonden voor bagageafhandelingssystemen, zijn een zestal casussen geëvalueerd om systeem randvoorwaarden, functies en connecties te bepalen. Het definiëren van deze eigenschappen is onderdeel van de eerste drie stappen in Pielage's ontwerpmethode. De twee resterende stappen, simulatie en evaluatie, zijn niet uitgevoerd voor deze casussen. Het generieke model voor bagageafhandelingssystemen dat volgt uit toepassing van de geselecteerde ontwerpmethode op zes casussen is geïllustreerd in Figuur 2. Een digitale versie van het model is geprogrammeerd in Microsoft Office Excel en de functionaliteit van het programma geverifieerd door het vergelijken van handmatig berekende resultaten en model gegenereerde resultaten. Hierbij moet rekening worden gehouden dat dit alleen mogelijk is voor handmatig berekenbare resultaten.

Om de praktische toepasbaarheid van het ontwerpproces en het model te kunnen beoordelen, zijn twee casussen uitgevoerd met data van [redacted]. Hierbij zijn de ontworpen concepten vergeleken met de in werkelijkheid geconstrueerde systemen. In de eerste casus is er een verschil gevonden in de sorteer functie. Wanneer dit verschil echter buiten beschouwing wordt

Table 2: Resultaten van de methode vergelijking en gevoeligheidsanalyse. Namen van criteria en methodes zijn afgekort en kunnen teruggevonden worden in hoofdstuk 3 van deze these.

	Standard	AL + 200%	CO + 200%	FL + 200%	IN + 200%	IT + 200%	LC + 200%	PD + 200%	PI + 200%	RA + 200%	SI + 200%	DS + 200%
VDI	0,09	0,11	0,08	0,08	0,09	0,09	0,08	0,12	0,07	0,09	0,09	0,08
R&E	0,06	0,06	0,06	0,07	0,06	0,08	0,06	0,06	0,05	0,06	0,06	0,06
LEM	0,15	0,14	0,20	0,15	0,16	0,13	0,17	0,14	0,14	0,16	0,18	0,14
PIE	0,20	0,21	0,20	0,21	0,18	0,22	0,21	0,21	0,17	0,17	0,18	0,20
G&H	0,15	0,14	0,13	0,14	0,14	0,13	0,14	0,15	0,17	0,18	0,15	0,16
VIA	0,18	0,17	0,15	0,17	0,16	0,15	0,16	0,17	0,23	0,18	0,18	0,21
SIE	0,18	0,17	0,18	0,19	0,20	0,19	0,18	0,16	0,18	0,16	0,16	0,15

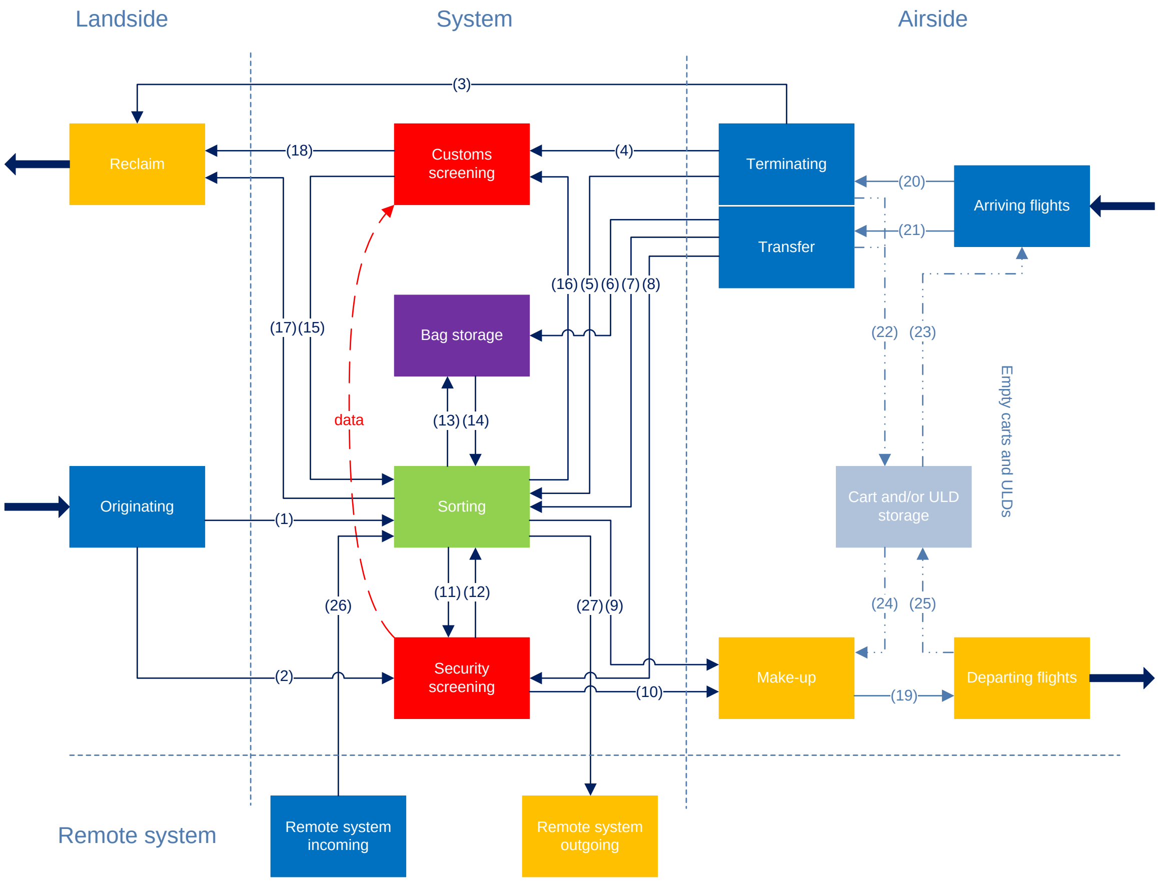


Figure 2: Een schematische weergave van een generiek model voor bagageafhandelingssysteem functies en verbindingen. Het model is onderverdeeld in landzijde, systeemzijde en luchtzijde.

gelaten, lijken beide systemen op elkaar. Het concept ontwerp voor de tweede casus is in alle opzichten anders dan het werkelijke ontwerp. Een mogelijke verklaring hiervoor is dat door het ontbreken van een aantal ontwerpeisen in deze casus geen overeenkomende invoer kon worden bereikt. In beide gevallen is de ontwerp tijd gereduceerd. De significantie van dit resultaat kan echter niet bepaald worden omdat slechts twee casussen zijn uitgevoerd.

De conclusie dan kan worden getrokken uit dit onderzoek is dat Pielage's ontwerpmethode, zoals gepref-ereerd door experts, geschikt is voor toepassing in de bagageafhandelingssysteem concept ontwerp fase. Evaluatie van zes verschillende casussen laat terugkomende functies van waaruit een generiek model kan worden vastgesteld. Voorlopige resultaten over toepassing van het model in twee design casussen zijn veelbelovend, echter zijn er meer casussen nodig om het model volledig te valideren of af te wijzen. Met deze conclusie draagt dit onderzoek bij aan concept ontwerp theorie voor bagageafhandelingssystemen. De ontwerpmethode en het model zijn mogelijk te generaliseren voor post- en pakketsystemen, omdat deze systemen overeenkomsten vertonen met bagageafhandelingssystemen op functioneel niveau, connectiviteit en onderdelen. Mogelijk toekomstig onderzoek in deze richting kan leiden tot dit inzicht.

Contents

Preface iii

Abstract v

Abstract (Dutch) vii

List of Figures xiii

List of Tables xvii

Glossary xix

1 Introduction 1
- 1.1 A short introduction to baggage handling . . . 1
- 1.2 Designing baggage handling systems . . . 2
- 1.3 Research context . . . 3
- 1.4 Motivation and research objective . . . 3
- 1.5 Research framework . . . 5
- 1.6 Research scope . . . 6
  - 1.6.1 Baggage, luggage and items . . . 6
  - 1.6.2 Concepts, systems and sub-systems . . . 7
  - 1.6.3 Loss of items . . . 7
  - 1.6.4 Exception handling . . . 7
  - 1.6.5 Using the analytic hierarchy process . . . 7

2 Research methods 9
- 2.1 Literature surveys . . . 9
  - 2.1.1 Literature on baggage handling systems . . . 10
  - 2.1.2 Literature on design methods . . . 12
- 2.2 Multi-criteria decision making . . . 13
- 2.3 Case study research . . . 14
- 2.4 Model verification and validation . . . 15
  - 2.4.1 Verification . . . 15
  - 2.4.2 Validation . . . 15
- 2.5 Chapter summary . . . 15

3 Design method selection 17
- 3.1 Current design method at SPPAL . . . 17
- 3.2 Criteria definition . . . 17
- 3.3 Design method evaluation . . . 19
  - 3.3.1 Design method by Verein Deutscher Ingenieure (VDI) . . . 19
  - 3.3.2 Design method by Roozenburg & Eekels (R&E) . . . 20
  - 3.3.3 Design method by Lemain (LEM) . . . 20
  - 3.3.4 Design of freight transport systems (PIE) . . . 22
  - 3.3.5 Design of object processing systems (G&H) . . . 22
  - 3.3.6 Simulation-integrated design (VIA) . . . 23
  - 3.3.7 Current design method (SIE) . . . 23
  - 3.3.8 Design method comparisons . . . 24
- 3.4 Applying the analytic hierarchy process . . . 24
  - 3.4.1 Criteria comparisons . . . 24
  - 3.4.2 Analysis results . . . 24
- 3.5 Chapter summary . . . 26

4 Designing a generic model 29
4.1 Case analysis . . . 29
4.2 Problem analysis . . . 31
4.3 System definition . . . 33
4.3.1 System boundaries. . . . 33
4.3.2 System functions. . . . 33
4.3.3 Basic assumptions . . . 34
4.4 System synthesis . . . 35
4.5 Simulation . . . 36
4.5.1 Theoretical formulation . . . 36
4.5.2 Simulation software . . . 40
4.5.3 Process description language . . . 40
4.5.4 Verification . . . 42
4.6 Evaluation and selection . . . 47
4.7 Chapter summary. . . . 47

5 Practical application 49
5.1 Design case C7 . . . 49
5.2 Design case C8 . . . 53
5.3 Chapter summary. . . . 57

6 Concluding remarks 59
6.1 Main research question . . . 59
6.2 Central research questions . . . 59
6.3 Recommendations . . . 61

Bibliography 63

A Scientific paper 69

B Initial research assignment 79

C Brainstorm 85

D Stakeholders 87

E Baggage handling equipment 91
E.1 Baggage handling equipment . . . 91
E.1.1 In-feed solutions. . . . 91
E.1.2 Security measures . . . 92
E.1.3 Transportation . . . 93
E.1.4 System outlets . . . 94
E.1.5 Baggage sorting . . . 95
E.1.6 Storage of baggage . . . 96
E.1.7 Individualization. . . . 96
E.1.8 Identification . . . 97
E.2 Categorization of solutions . . . 97

F Morphological overviews 99

G Design cases 103

H Instructions 113

I Expert criteria rankings 121

J Design case concept rankings 125
J.1 Design case C7 . . . 125
J.2 Design case C8 . . . 125

K Program code **131**
K.1 Declaration of global variables . . . 131
K.2 Main program . . . 133
K.3 Initialize . . . 133
K.4 Randomizer . . . 136
K.5 Flight list . . . 136
K.6 Generate bags . . . 138
K.7 Departure process . . . 146
K.8 Arrival process . . . 147
K.9 Discrete time . . . 149
K.10 Data processing . . . 150
K.11 Calculation 1 . . . 153
K.12 Calculation 2 . . . 163
K.13 Populate charts . . . 171
K.14 Process output . . . 173
K.15 Write file . . . 175
K.16 Write file . . . 178

List of Figures

1 A schematic overview of a generic model for baggage handling system functions and connections. The model may be divided into landside, airside and system. It should also be noted that the sorting system may connect to a remote system that is already present or will be built in a future expansion. vi

2 Een schematische weergave van een generiek model voor bagageafhandelingssysteem functies en verbindingen. Het model is onderverdeeld in landzijde, systeemzijde en luchtzijde. viii

1.1 The flow of baggage and passengers through an airport and baggage handling system. Arrows indicating passengers combined with their baggage are coloured orange, arrows indicating passengers are coloured green and arrows indicating baggage are coloured blue. This process flow chart is a recreation of the original figure [1]. 2
1.2 The manual transportation of luggage from carts to an aircraft's cargo hold [2]. 2
1.3 A belt conveyor based baggage handling system, located at London Heathrow Airport [3]. 2
1.4 An ICS for the handling of baggage at Beijing Capital International Airport [4]. 2
1.5 A collection of steps taken during airport baggage handling design from initial idea to realisation and maintenance contract. Including external consultants is optional in this process and typically happens in Category B and C projects. This figure was constructed after interviews with ten Berge [5] and Deerns [6]. 4
1.6 A comparison of conceptual design and detailed design on inputs, outputs and its transformation process tools. This table was conceived from Pielage [7], Yilmaz and Daly [8] and practice knowledge at SPPAL. 5
1.7 A research framework based around the research objective, constructed as is proposed in Verschuren and Doorewaard [9]. 6

2.1 Flow of baggage through a tiered security system with different scanning capabilities [1, 10, 11]. 11

3.1 A systematic representation of systematic design by VDI [12, 13]. 20
3.2 A systematic design method presented by VDI that includes conceptual design in phase II [12, 13]. 21
3.3 The design method by [14]. 21
3.4 A schematic overview of the design process by Pielage [7] and visualisation. 23

4.1 Flight schedule input list including turnaround time, occupation ratio, baggage ratio, transfer ratio, airline and type as well as start and end date per scheduled flight. 32
4.2 Inputs to the calculation that include check-in, transfer and make-up times as well as distribution and cart or ULD handling per type of flight. 32
4.3 Design parameters such as transport distance, transport time, processing time and function splits as well as formatting of the results. 33
4.4 A schematic overview of a generic model for baggage handling system functions and connections. The model may be divided into landside, airside and system. It should also be noted that the sorting system may connect to a remote system that is already present or will be built in a future expansion. 34
4.5 A representation of a model function block. 36
4.6 A morphological overview for baggage handling systems. 37
4.7 Originating baggage for verification of the departure system propagation. 42
4.8 Security screening baggage for verification of the departure system propagation. 42
4.9 Sorting baggage for verification of the departure system propagation. 43
4.10 Baggage storage for verification of the departure system propagation. 43
4.11 Make-up baggage for verification of the departure system propagation. 43

4.12 Departing baggage for verification of the departure system propagation. 43
4.13 Baggage reclaim for verification of the terminating system propagation. 43
4.14 Arriving baggage for verification of the terminating system propagation. 44
4.15 Terminating baggage for verification of the terminating system propagation. 44
4.16 Transfer baggage for verification of the terminating system propagation. 44
4.17 Customs screening baggage for verification of the terminating system propagation. 44
4.18 Constant distribution for verification of simulation distributions. 44
4.19 Uniform distribution for verification of simulation distributions. 44
4.20 Triangular distribution for verification of simulation distributions. 44
4.21 Normal distribution for verification of simulation distributions. 44
4.22 Originating baggage for verification of the accumulating functions. 45
4.23 Baggage sorting for verification of the accumulating functions. 45
4.24 Bag storage for verification of the accumulating functions. 45
4.25 Baggage make-up for verification of the accumulating functions. 45
4.26 Figure of the originating function for a comprehensive flight schedule. 46
4.27 Figure of the transfer function for a comprehensive flight schedule. 46
4.28 Baggage sorting function for a comprehensive flight schedule. 46
4.29 Security screening function for a comprehensive flight schedule. 46
4.30 Early baggage storage for a comprehensive flight schedule. 46
4.31 Make-up area throughput for a comprehensive flight schedule. 46
4.32 Make-up area baggage accumulation. 46
4.33 Terminating baggage. 46
4.34 Customs screening function for terminating baggage. 46
4.35 Figure of baggage reclaim for verification. 46

5.1 Regional situation of C7. 50
5.2 Originating process of baggage for C7, based on a provided flight schedule. 51
5.3 Process of terminating baggage on a peak day at C7. 51
5.4 A screen capture of the model output for security screening devices. Manual screening is left out as this was not required in this case. 51
5.5 A depiction of the proposed concept for C7 including check-ins, screening machines, belt conveyors, ICS tracks and ICS tilters. 52
5.6 Terminal layout of C8. 54
5.7 Originating flow of baggage for C8, based on a provided flight schedule. 55
5.8 Terminating baggage for C8. 55
5.9 Baggage transfer process for C8, based on a provided flight schedule. 55
5.10 Flow accumulation in storage at C8. 55
5.11 A depiction of the proposed concept for C8 including check-ins, screening machines, belt conveyors, ICS tracks, switches and ICS tilters. 56

6.1 A schematic overview of a generic model for baggage handling system functions and connections. The model may be divided into landside, airside and system. It should also be noted that the sorting system may connect to a remote system that is already present or will be built in a future expansion. 60

C.1 A brainstorm diagram of words that are associated with baggage handling, conceptual design and methods. These may be used in the literature survey for relevant works. 86

E.1 A conventional check-in counter for baggage with a limited size [15]. 92
E.2 The uDrop system at Paris Charles de Gaulle Airport [16]. 92
E.3 A Lufthansa counter for handing in special baggage [17]. 92
E.4 A mixed train consisting of a tractor, three ULD carriers and two carts respectively for transport between make-up and aircraft [18]. 94
E.5 Spiral chute that allows items to drop multiple levels [19]. 94
E.6 Manually loading carts with baggage from laterals [20]. 95
E.7 Loading baggage carts from an inclined conveyor [20]. 95
E.8 Chutes collect baggage for a single destination [20]. 95

E.9 Automated loading robot for loading carts and ULDs [21]. 95
E.10 Wall outlet for special baggage at Salt Lake City Airport. Due to being a popular skiing destination, this airport has a ski-equipment reclaim area [22]. 95
E.11 Lane storage of baggage on an ICS at Bergen Airport [23]. 97
E.12 Storage of baggage in an automated storage and retrieval system [24]. 97

F.1 The morphology for baggage handling systems classified as small by IATA. Solutions that can be created with this diagram may apply to baggage handling systems with a capacity of 0 to 1000 bags per hour, during peak moments at an airport. 100
F.2 The morphology for baggage handling systems classified as medium by IATA. Solutions that can be created with this diagram may apply to baggage handling systems with a capacity of 1000 to 5000 bags per hour, during peak moments at an airport. 101
F.3 The morphology for baggage handling systems classified as large by IATA. Solutions that can be created with this diagram may apply to baggage handling systems with a capacity of more than 5000 bags per hour, during peak moments at an airport. 102

G.1 Design case C1. 104
G.2 Design case C2. 105
G.3 Design case C3. 106
G.4 Design case C4, part 1. 107
G.5 Design case C4, part 2. 108
G.6 Design case C5. 109
G.7 Design case C6. 110
G.8 Design case C7. 111

List of Tables

1 Final results and sensitivity analysis to the method comparison. Names of criteria and methods have been abbreviated and may be found in chapter 3 of this thesis. v

2 Resultaten van de methode vergelijking en gevoeligheidsanalyse. Namen van criteria en methodes zijn afgekort en kunnen teruggevonden worden in hoofdstuk 3 van deze these. vii

2.1 Relevant results of the survey for literature in different search engines and different topics. . . . 9
2.2 Recommended baggage handling solutions per catagory and function according to IATA and Bradley [25]. 10
2.3 The scale of importance for comparing activities. 13
2.4 Available cases for study with given peak flow rate and classification. 14

3.1 A description of the concept design phase at SPPAL as part of an entire design project. 18
3.2 A table containing comparisons for design methods on criterion aggregation layers. The consistency ratio for this comparison is 0,03. 24
3.3 A table containing comparisons for design methods on criterion communication. The consistency ratio for this comparison is 0,02. 24
3.4 A table containing comparisons for design methods on criterion flexibility. The consistency ratio for this comparison is 0,01. 25
3.5 A table containing comparisons for design methods on criterion integration. The consistency ratio for this comparison is 0,05. 25
3.6 A table containing comparisons for design methods on criterion iteration. The consistency ratio for this comparison is 0,09. 25
3.7 A table containing comparisons for design methods on criterion life-cycle. The consistency ratio for this comparison is 0,01. 25
3.8 A table containing comparisons for design methods on criterion parallel development. The consistency ratio for this comparison is 0,02. 25
3.9 A table containing comparisons for design methods on criterion performance indicators. The consistency ratio for this comparison is 0,05. 25
3.10 A table containing comparisons for design methods on criterion rapidity. The consistency ratio for this comparison is 0,03. 25
3.11 A table containing comparisons for design methods on criterion simplicity. The consistency ratio for this comparison is 0,03. 25
3.12 A table containing comparisons for design methods on criterion development scenarios. The consistency ratio for this comparison is 0,01. 25
3.13 A table containing the results to the pairwise comparison of criteria by XXXX. The consistency ratio of this comparison is 0.08. 26
3.14 Final results and sensitivity analysis to the method comparison. 26

4.1 Occurrence of functions and connections in cases. 30
4.2 Assumed client types and preferences determined in collaboration with ten Berge [26] and Lodewijks [27] 47

5.1 A pairwise comparison according to the AHP. This analysis has a consistency ratio of 0,07. . . . 50
5.2 Multi-criteria analysis for the screening process. 52
5.3 List of components for the departure concept proposal for C7. It should be noted that tray loaders are included in sorting. 52
5.4 List of components for the terminating concept proposal of C7. 53
5.5 List of components for the concept proposal departure system of C8. Merges, switches and unloaders are included in the sorting system. 55

5.6 List of components for the concept proposal terminating system of C8. 57

I.1 A table containing the results to the pairwise comparison of criteria by XXXX. The consistency ratio of this comparison is 0.26. 122
I.2 A table containing the results to the pairwise comparison of criteria by XXXX. The consistency ratio of this comparison is 0.20. 122
I.3 A table containing the results to the pairwise comparison of criteria by XXXX. The consistency ratio of this comparison is 0.13. 122
I.4 A table containing the results to the pairwise comparison of criteria by XXXX. The consistency ratio of this comparison is 0.14. 122
I.5 A table containing the results to the pairwise comparison of criteria by XXXX. The consistency ratio of this comparison is 0.08. 123

J.1 Multi-criteria analysis for the originating process. 125
J.2 Multi-criteria analysis for the screening process. 126
J.3 Multi-criteria analysis for the sorting process. 126
J.4 Multi-criteria analysis for the transport processes. 126
J.5 Multi-criteria analysis for the make-up process. 127
J.6 Multi-criteria decision on terminating systems for design case Dresden. 127
J.7 Multi-criteria analysis for the reclaim process. 127
J.8 Multi-criteria decision on check-ins systems for design case Paris Orly. 128
J.9 Multi-criteria decision on screening systems for design case Paris Orly. 128
J.10 Multi-criteria decision on sorting systems for design case Paris Orly. 128
J.11 Multi-criteria decision on transport systems for design case Paris Orly. 129
J.12 Multi-criteria decision on make-up systems for design case Paris Orly. 129
J.13 Multi-criteria decision on transfer systems for design case Paris Orly. 129
J.14 Multi-criteria decision on storage systems for design case Paris Orly. 130
J.15 Multi-criteria decision on terminating systems for design case Paris Orly. 130
J.16 Multi-criteria decision on reclaim systems for design case Paris Orly. 130

Glossary

AHP Analytic Hierarchy Process; a method applied in decision-making. xvii, 7, 15, 45, 48, 52, 57, 58

Beumer Group Beumer Group GmbH & Co. KG. 89

BHS Baggage Handling System; a system that allows for the transport of baggage between check-in counters and an aircraft. 1, 3, 5, 7, 9, 10, 15, 17, 18, 24, 26, 29, 31, 33, 47, 89

DMU Decision Making Unit; a group of representatives selected by stakeholders to make important decisions for a baggage handling system design. 3, 55, 86, 87

EDS Explosives Detection System; a device capable of detecting explosives in baggage. 10

FAA Federal Aviation Administration; a government administration for airport and airline regulation. 14

FSC Full-Service Carrier; a traditional airline that offers a variety of services to the customer. Due to the increased amount of services compared to low-cost carriers, full-service carriers are typically more expensive to fly with. 2, 3, 10

FTS Freight Transport System. 3

HBS Hold Baggage Screening; the screening of baggage that is placed in the aircraft's hold. 2, 90

IATA International Air Transport Association. xv, xvii, 1, 2, 10, 14, 34, 39, 48, 49, 52, 89, 95, 96, 98–100

ICS Individual Carrier System; a selected amount of carriers operate on predefined pieces of track. The carriers are be loaded with individual baggage items and routed to its destination, where it is offloaded. xiii–xv, 1, 2, 10, 11, 17, 28, 33, 36, 49–51, 53–55, 59, 89, 91, 92, 94, 95

KLM Royal Dutch Airlines; an international aviation company. 12

KPI Key Performance Indicator; are a type of performance measurement. 5, 6, 38, 45, 47, 52, 57

LCC Low-Cost Carrier; an airline that is focussed on gaining market share by providing low-cost flights. The lowered cost are a result of a reduced number of services compared to full-service carriers. 2, 3, 10, 11

MCT Minimum Connecting Time; the smallest amount of time that is required for connecting flights. 11

OCR Optical Character Recognition; a system which is able to recognize folds or tears in baggage labels and is able to derive the correct label from it. 7, 95

OOG Out-Of-Gauge; a type of baggage that exceeds the set dimensions and weight of what is presumed to be standard baggage. 7, 32, 33

OSR On-Screen Resolution; a method of visualizing screened baggage for human assessment. 90

PDL Process Description Language; an early form of programming code that describes processes in code and clarifies its interactions between functions. 34, 38

PPC Passenger Presentation Curve; a mathematical formula representing the arrival pattern of passengers at specific locations, typically a check-in counter. 29

RFID Radio-Frequency Identification; passive or active devices which emit radio waves which allow for the identification of specific items. 7, 95

SPPAL Siemens Postal, Parcel & Airport Logistics GmbH. xiii, xvii, 1, 5, 6, 13, 17, 18, 20, 23, 24, 26, 38, 58, 59, 89

SRS Storage and Retrieval System; a system which stores baggage and retrieves it before flight departure. 10, 94

TU Delft Delft University of Technology. 9, 12, 20

ULD Unit Load Device; a container in that holds multiple luggage items and can be loaded into an aircraft. xiii–xv, 1, 29, 30, 33, 40, 43, 49, 89–94

Vanderlande Industries Vanderlande Industries Holding B.V. 12, 85, 89

VBA Visual Basic for Applications; a programming language supported by Microsoft Office applications. 38, 49

VDI Verein Deutscher Ingenieure; an association for German Engineers that supports engineers in their practise. xiii, 20, 21

1

Introduction

Although the importance of baggage handling systems (BHSs) is not widely known by its many daily users, such systems are an integral part of an airport's logistics system. Passengers will hand in luggage at a check-in counter and may reclaim it at their destination. The entire baggage handling process however, is more complicated and includes transfer, screening, sorting and storage of bags among others. Siemens Postal, Parcel & Airport Logistics GmbH (SPPAL) [28] is one of the suppliers that provides solutions for the baggage handling process at airports. The Airport Logistics division of the company is responsible for designing and developing baggage handling- and cargo handling systems in combination with airport rail link systems, support systems and IT solutions.

Continuing population growth and urbanisation, centring around Asia and Africa [29], is the main driver behind an increased demand for air transport [30–32]. This is also noticed by SPPAL in the increased demand for baggage handling systems and modernisation of existing systems at airports. Although the amount of baggage handling expertise present at airports has diminished, there is still a desire to understand the system concepts that are developed. Therefore, designing a model for such a purpose is the topic of this thesis.

In section 1.1 and section 1.2 the objects of research, baggage handling systems and conceptual design, are explained. The context of this research is described in section 1.3. This is followed by a description of the motivation behind- and research objective of this thesis in section 1.4. In section 1.5, a framework is presented that serves as a guideline to reach the research objective. This is followed by its demarcating the scope in section 1.6.

1.1. A short introduction to baggage handling

Baggage can be subdivided into two types: carry-on baggage and hold baggage. Passengers are allowed to take their carry-on baggage on board of an aircraft, bypassing baggage handling systems. Hold baggage, consisting of standard and odd-sized baggage, is typically processed by baggage handlers on airports. Hold baggage can be handed in at the airport's check-in counters after which it is transported, screened and sorted to make-up carousels or laterals. At make-up areas, baggage handlers load items into carts or unit load devices (ULDs). Carts are taken to aircraft and baggage is manually loaded into an aircraft's hold, whereas ULDs are loaded mechanically as a whole [25, 33]. This process is reversed at an aircraft's destination and baggage may be retrieved by passengers at reclaim areas. In case of a flight transfer, luggage may be transferred by a baggage handler, without intervention of passengers. The above mentioned logistic flows are depicted in Figure 1.1, in which a general schematic overview of the baggage handling process is given. This figure shows which processes are part of the baggage handling system and which processes linked to it.

Several types of equipment exist that allow for variously sized baggage handling systems. These systems are categorized by the International Air Transport Association (IATA) [34] in peak baggage flow rates [25]. Small airports with a flow rate of up to a 1000 bags per hour are recommended to adopt manual or basic automated handling, as depicted in Figure 1.2. Medium airports with a flow rate of 1000 to 5000 bags per hour are advised to use automatic sorting systems based on belt conveyors (Figure 1.3). The final category involves complex baggage handling systems like individual carrier systems (ICSs, Figure 1.4) and large belt conveyor systems, which are recommended for flow rates beyond 5000 bags per hour. Although these recommendations suggest clear boundaries between categories, the mentioned systems may be applied in various

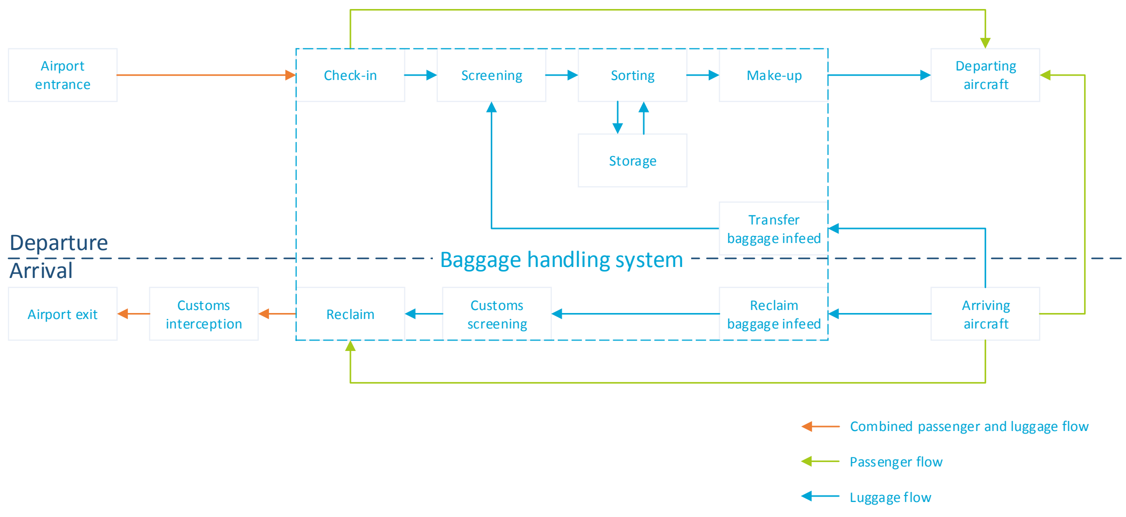


Figure 1.1: The flow of baggage and passengers through an airport and baggage handling system. Arrows indicating passengers combined with their baggage are coloured orange, arrows indicating passengers are coloured green and arrows indicating baggage are coloured blue. This process flow chart is a recreation of the original figure [1].

Figure 1.2: The manual transportation of luggage from carts to an aircraft's cargo hold [2].

Figure 1.3: A belt conveyor based baggage handling system, located at London Heathrow Airport [3].

Figure 1.4: An ICS for the handling of baggage at Beijing Capital International Airport [4].

scenarios. The IATA classification does not make a distinction between regional airports or hub airports.

At present, one of the most important reasons for handling baggage is the requirement of 100% hold baggage screening (HBS) that is mandatory according to international legislation. This requirement can be traced back to three events that took place previously; the Pan Am 103 bombing, the explosion of TWA Flight 800 and the events that occurred on the 11th of September 2001 [35]. The requirement for 100% hold baggage screening is in effect since 2002.

Before this safety regulation was implemented, baggage handling had developed itself into being an additional service towards the passenger. Based on this fact, one could argue that baggage handling is an unnecessary process and wonder why baggage is not carried into the aircraft's hold by the passenger. This comes close to what many low-cost carriers (LCCs) attempt to do in order to reduce handling costs, as opposed to full-service carriers (FSCs) [36–38]. By charging for every hold baggage item, low-cost carriers stimulate their passengers to take carry-on baggage on board instead of hold baggage. This reduces the need for expensive baggage handling charges at airports and has a negative effect on capacity requirements for baggage handling [39].

Future developments in the field of baggage handling are considered to be: *1*) further reducing the amount of mishandled bags, *2*) increasing handling efficiency, *3*) enabling faster innovation and adoption of technologies, *4*) tracking of items, *5*) home-printing of tags, and *6*) baggage deliveries at home [40, 41].

1.2. DESIGNING BAGGAGE HANDLING SYSTEMS

The organisation of a baggage handling process at airports is dependent on several factors, such as airport size and type of passengers. These two factors have been elaborated upon in the previous section. Airport size and passenger type place different demands on the design of the baggage handling process. Therefore, this section elaborates on the design process behind baggage handling from initial idea to realisation, with a focus on conceptual design and its involved parties.

One of the first steps in a design project is identifying the need for expansion. Recognising that future developments may yield an aircraft, passenger or baggage handling capacity shortage may lead to such an expansion of existing systems. In case of an expansion to the baggage handling system, an equipment supplier is always involved. [redacted]
.
[redacted]
. This consultant will then either invite suppliers to submit concept designs or design a brand-neutral design which suppliers may bid on. After receiving several bids, the invitation to tender is closed and all bids are reviewed. This process is repeated until a satisfactory result has been met. Agreement upon the conceptual design will lead to detailing of a concept and manufacturing of the product. Finally, the delivered product is validated for operational readiness and may become part of a service level agreement.

A concise definition of conceptual design is given by Ashby *et al.* [42]: "The translation of design requirements (end-product qualities) into several most-promising designs". These design requirements are dependent on the goal of a design and together with the boundaries define a problem space. When designing BHSs, static calculations are used to transform these inputs into a solution space, system description, sketches and 3D concept models. During detailed design, the preferred concept is taken and sub-systems are further detailed whilst the whole system retains its function. An example of this is the exact placing of transport belts in the building during the detailed design phase, as opposed to a rough estimation during the concept design phase. Figure 1.6 compares these inputs, tools and outputs for concept and detailed design.

In the previous elaboration of concept and detailed design, the airport has been regarded as a single entity. This is however not the case, as an airport entity may consist of multiple stakeholders. [redacted]
. Each of the stakeholders has a specific objective. A representative from each of these stakeholders is present in a decision making unit (DMU). This unit is responsible for conveying requirements of stakeholders (Appendix D) into concept and final designs.

1.3. Research context

When regarding hold baggage, passengers typically do not personally carry their baggage into the aircraft's hold. It is however noticeable that the requirements for BHSs change as strategies differ per airline. Airlines operating according to the LCC strategy typically request less expensive baggage handling systems than FSC [39]. Among others, this introduces the need for a flexible design approach. Several previous studies have addressed the issue of conceptual design for freight transport systems (FTSs), a group of systems which baggage handling systems are part of.

The preceding studies have different points of focus. A dissertation by Pielage [7], focussing on the development of a generic design method for FTSs, investigates the applicability of existing design methods and proposes a new method. Lemain [43] provides a new method for baggage handling design from a business management perspective. In the thesis, much attention is given to the analysis of requirements. Based on this requirements analysis a new concept design method is proposed for Vanderlande Industries, disregarding existing design methods. Yet another research project, performed by Grigoraş and Hoede [44], approaches baggage handling system design from a mathematical viewpoint with the use of graph theory.

In these studies different assumptions are made in respect to the designs. Lemain [43] and Grigoraş and Hoede [44] investigate two specific areas of baggage handling design: an analysis of requirements and route finding, respectively. Which equipment is actually needed is not discussed and leaves a knowledge gap [9] for further research. [redacted]
.

1.4. Motivation and research objective

This research originally focussed on optimising the design process for generating baggage handling concepts (see Appendix B). Several months after starting with the research, it became apparent that the company was not looking for optimisation of the design process but a way to provide a more detailed specification to customers during the conceptual design process. A revision of both the research objective and research questions was thus opted for.

During this time it was also suggested to follow research design as described by Verschuren and Doore-

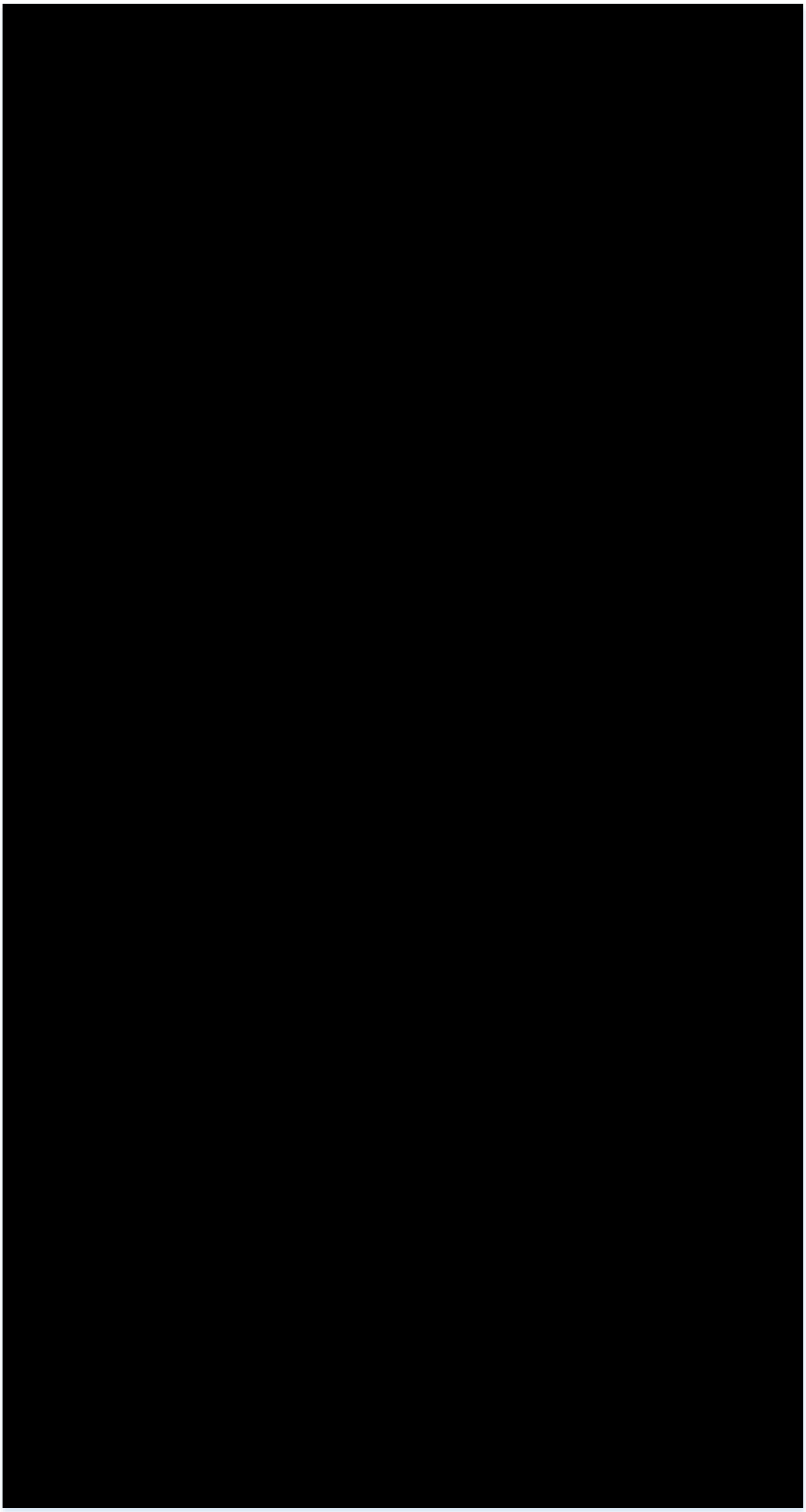


Figure 1.5: A collection of steps taken during airport baggage handling design from initial idea to realisation and maintenance contract. Including external consultants is optional in this process and typically happens in Category B and C projects. This figure was constructed after interviews with ten Berge [5] and Deerns [6].

	Concept design	Detailed design
Input	Design goals Design requirements Design boundaries	Final concept design Design requirements Design boundaries
Tools	Problem space Static calculation Computer aided design	Simulation Computer aided design
Output	Solution space System descriptions KPI estimations Material flow diagram Concept sketches 3D concept model	Final solution Simulation results Simulated KPIs 3D design model 3D modelled parts Exact location and size

Figure 1.6: A comparison of conceptual design and detailed design on inputs, outputs and its transformation process tools. This table was conceived from Pielage [7], Yilmaz and Daly [8] and practice knowledge at SPPAL.

waard [9], in order to systematically determine the research context, research framework and central research questions. The work is regarded as a valuable source in organising research projects. Because of this critical acclaim, the research project will be structured as is recommended in the work by Verschuren and Doorewaard [9]. After much consideration, the following research objective was established: to quantify KPIs for concept designs of airport baggage handling systems by modelling its capacity based on flight schedule demand.

The established research objective leads to the main research question of this thesis: *How may KPIs of BHS concept designs be determined based on flight schedule demand?* Part of this research question is the selection and execution of a design method. This is further elaborated upon in the next section.

1.5. RESEARCH FRAMEWORK

After setting the objective of a research, Verschuren and Doorewaard [9] advise to construct a framework around the objective in order to demarcate the research. This framework consists of the research objective and all intermediate steps to achieve it. Combining information leads to preliminary conclusions which may be used in subsequent steps. Figure 1.7 is a visual representation of the research framework and serves as a guideline to creating the central research questions.

Figure 1.7 shows that the research framework contains four central research questions that are elaborated upon in chapters 2-5. The first central research question is linked to chapter 2 and attempts to find what scientific knowledge is already available on BHSs and concept design. The chapter furthermore describes the research methods by which this is done and which other research methods are used throughout this thesis. The gathered design methods are assessed in chapter 3 and graded on several criteria. Weight factors for the criteria are determined by experts with questionnaires to find the most preferred design method. The design method is used to conceive a generic model for baggage handling systems in chapter 4 by applying the method in several cases. This is tied to the third central research question. Lastly, the fourth central research question is concerned with the application of the generic model in case studies. Two case studies are executed in chapter 5. All combined, these four central research questions yield the conclusion to the main research question presented in section 1.4 and is elaborated upon in chapter 6. The following central research questions and sub-questions are derived from the research framework:

- What knowledge on baggage handling and design methods may be learnt from literature?
  - What knowledge on baggage handling systems is available in literature?
  - What knowledge on design methods is available in literature?
  - How is the analytic hierarchy process defined?

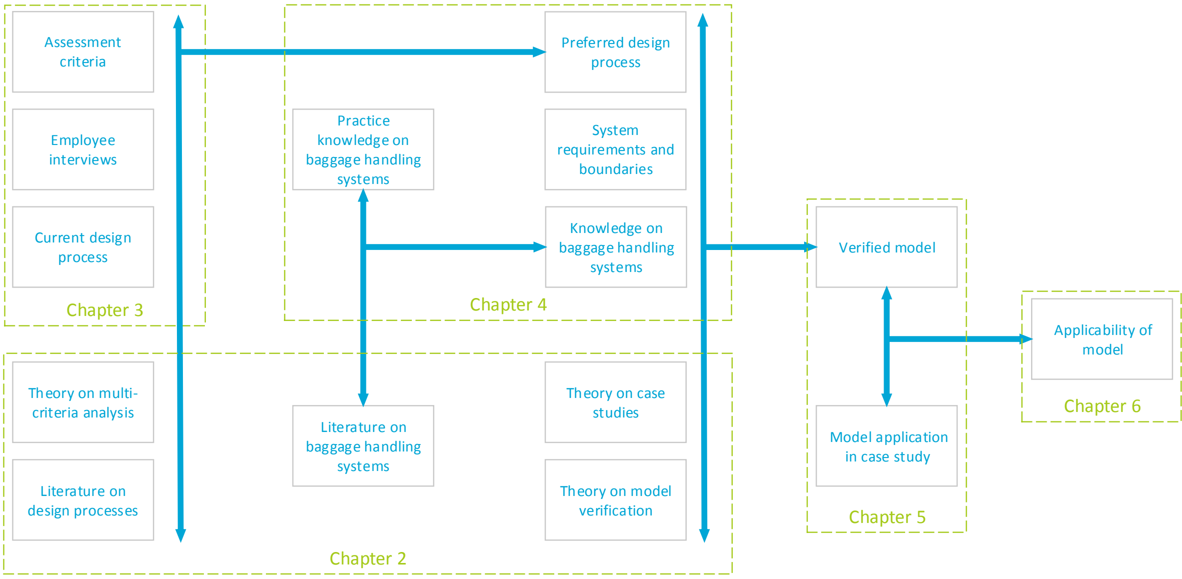


Figure 1.7: A research framework based around the research objective, constructed as is proposed in Verschuren and Doorewaard [9].

  - In what way are case studies executed?
  - How may models be verified and validated?
- Which design method, applicable in baggage handling concept design, is preferred when applying the analytic hierarchy process?
  - Which design method is currently used at SPPAL?
  - What characterises the design methods found in literature?
  - On what criteria should the design methods be evaluated?
  - Which criteria are preferred by baggage handling experts?
- How may a generic model for baggage handling systems be defined?
  - What knowledge on baggage handling system design is learnt from practice experience?
  - Which functions and interconnections define a baggage handling system?
  - Which requirements and boundaries define the model?
  - How may KPI calculation occur?
  - How may concepts be evaluated?
- What may be learnt when the design method and model are applied in case studies?
  - Which cases are used?
  - Which design concepts have been conceived by following the current design method?
  - Which design concepts have been conceived by following the proposed design method?
  - What differences exist in designs using both methods?
  - Which design times are measured for the current and proposed design method?

1.6. RESEARCH SCOPE

In order to achieve a scientifically relevant result, research typically has a specific scope in which is studied. This scope may simplify several parts of a research with the right justification. Several boundaries are given in this section to limit the scope of this research.

1.6.1. BAGGAGE, LUGGAGE AND ITEMS

As is elaborated in section 1.1, baggage or luggage consists of carry-on baggage and hold baggage. Carry-on baggage is carried by the passengers themselves and bypasses the baggage handling process. In the event that overhead baggage bins are full, crews may decide to move bags to an aircraft's hold for safety reasons. As these bags will still bypass the baggage handling system, this type of baggage is therefore not further considered in this study.

Hold baggage, consisting of normal sized and oversized (OOG) baggage, is processed by a baggage handling system. These sizes are set by airlines and airports, and may differ per region. Normal and oversized baggage may be processed on different systems, however concepts exist which are capable of handling both sizes. In the literature survey presented in section 2.1, no knowledge about typical amounts of oversized baggage was obtained although an effort has been made to categorize different shapes of baggage [45]. It is therefore opted to take normal and oversized baggage as a whole and further refer to them as items, as conceptual design does not require the exact sizes of baggage. This is also done as sub-systems are rated in items per hour.

1.6.2. CONCEPTS, SYSTEMS AND SUB-SYSTEMS

The definition of a baggage handling system is given by Lodewijks [46] as: *"A baggage handling system is a type of transport system installed in airports and which transports checked baggage from ticket counters to areas the bags can be loaded onto aircraft. A BHS also transports checked baggage coming from aircraft to baggage claims or to an area where the bag can be loaded onto another aircraft"*. The word "system" in this context is used for the realisation of what was a design "concept".

Taking a baggage handling system as a whole, the sub-system sorting may then be seen as a function. A sorter can also have its own sub-systems, for example a merge. These are the first, second and third aggregation layers respectively [47]. As this research focusses on conceptual design, other aggregation layers that are specified during detailed design of the system are not regarded in this report. Nuts and bolts for example are not regarded.

1.6.3. LOSS OF ITEMS

The mishandling of bags plays an important role in the airport and airline industry [41, 48]. An average of 7.3 per 1000 bags are lost each year which results in a cost of US$ 2.4 billion. These figures show that the loss of items is an important theme within baggage handling. One of the global trends that supports this development is the automation of baggage handling systems [41], which increases handling accuracy.

Although it is important, loss of items should however not be taken into account during the capacity calculation. This would give an incorrect indication of required system capacities. Loss of items is therefore not taken into account during the calculation.

1.6.4. EXCEPTION HANDLING

Exceptions such as passengers being late at the check-in of an airport are not taken into account in the capacity calculation for a baggage handling system. Having a late bag enter the system will result in it not reaching the aircraft in time, and therefore missing the flight. Solving these exceptions may happen but will not involve the baggage handling system. Other exceptions such as a loss of identification strip (barcode or RFID) have to be handled by the system. [redacted]
.

1.6.5. USING THE ANALYTIC HIERARCHY PROCESS

The central research question on deciding which design process should be used already specifies which method is used for decision-making. The field of decision-making contains several methods [49, 50]. One of these classical methods is the analytic hierarchy process (AHP) [51, 52] and is still frequently used. In this process, weights are awarded to several criteria, performances scored and results ranked accordingly. With the AHP it is possible to derive a consistency ratio for a set of preferences, making it useful for checking the correctness of preferences. This method is used and is further elaborated in chapter 2.

2

RESEARCH METHODS

Mentioned in the previous chapter is the use of Verschuren and Doorewaard [9] as general basis for this thesis to construct an objective, research framework and central research questions. The work elaborates that specific methods may be opted for reaching conclusions on the central research questions. This chapter elaborates on these methods for the different parts of this research work. This ensures that the conducted research may be repeated and tested for its validity of the approach.

This chapter describes four methods that are used throughout this research. The first method describes how literature surveys are conducted and the results thereof. This is part of section 2.1. Knowledge on the objects of research is expanded by these literature surveys and design methods are gathered for analysis. The found design methods are graded on several criteria, which in turn are graded by experts. This method for multi-criteria decision making is described in section 2.2. From this analysis follows a preferred design method for BHSs. A generic model for BHSs is constructed by applying the selected design method in six design cases. Case study research is described in section 2.3. After construction, verification and validation of the model should occur. Methods thereto are described in section 2.4. Validation of the model occurs by two case studies for which case study research is reapplied. It should be noted that full validation of the model could not be achieved as it was only possible to perform validation on two cases. A preliminary conclusion is drawn from this but further research is advised. This chapter concludes with a short summary of the chapter in section 2.5.

2.1. LITERATURE SURVEYS

The previous chapter discusses the objective of this research. In order to achieve the stated objective, a literature survey on existing works is performed. The literature is retrieved from the following sources: *a*) TU Delft [53], *b*) Science Direct [54], *c*) Springer Link [55], and *d*) Google Scholar [56]. As all search results of these search engines are ranked by relevance, only the first 50 articles for each search query are taken into account. This is done as articles beyond this limitation only mention the topic once, not being the main topic of an article. It should also be noted that when an article has already been found in a previous search, it is not added to the list again [9].

Baggage handling as a topic is scarce in literature. Although search engines typically show more than a thousand results, only a fraction of these results is relevant to this topic. Table 2.1 shows that of the 200 articles per search query, only 43 were found relevant. Among the search results, no work has been found that is dedicated to describing baggage handling systems. Such descriptions are often a small part of a book on airport terminal design. Knowledge in this research will therefore be partially based on practice experience.

Table 2.1: Relevant results of the survey for literature in different search engines and different topics.

	TU Delft library	Science Direct	Springer Link	Google Scholar
baggage handling	14/50	10/50	9/50	10/50
concept design	0/50	4/50	3/50	12/50

Table 2.2: Recommended baggage handling solutions per catagory and function according to IATA and Bradley [25].

	In-feed	Transport	Outlet	Sorting	Storage
Cat. A	Check-in counters and laterals	Belt conveyors	Laterals or carousels	Manual sorting with racetracks or automatic sorting with pushers, diverters or vertical sorters	
Cat. B	Check-in counters and laterals	Belt conveyors or basic ICS	Laterals, carousels or chutes	Automatic sorting with pushers, diverters, vertical sorters, tilting tray sorters or basic ICS	SRS
Cat. C	Check-in counters, off-site check-in systems, laterals and automated unloading devices	Belt conveyors or top end ICS	Laterals, carousels, chutes or robot	Tilting tray sorters or top end ICS	SRS

The results in Table 2.1 for baggage handling are obtained by searching for the terms *airport*, baggage*, handl*, equipment*, system*, concept** and *transport**. These terms are derived from the brainstorm that is depicted in Appendix C. Similarly, the results for concept design are obtained by searching for *airport*, baggage*, system*, concept*, design*, method** and *approach**. A combination of terms has been used. It should be noted that these works have been supplemented by recommendations from involved thesis committee members and several other referenced works.

2.1.1. LITERATURE ON BAGGAGE HANDLING SYSTEMS

In this section, the results of the literature survey into baggage handling systems are reviewed. A general overview of baggage handling is given by Vickers and Chinn [33], and Bradley [25]. Both argue that check-in, sorting and screening are important functions of BHSs. Baggage storage and off-site check-in may also be included in the baggage handling process of airports. Recommendations by IATA outline a total of three baggage handling system categories based on peak flow rates: *a*) 0 - 999 bags/peak hour, *b*) 1000 - 4999 bags/peak hour and *c*) 5000 and more bags/peak hour. In these recommendations, several sub-systems per function are related to the flow rate categories. Realisations of these sub-systems can be found in Table 2.2. It is stressed that in order to reduce the amount of baggage mishandling further, cart-based systems are preferred over belt-based systems and belt-based systems are preferred over manually handling of bags.

Bradley [25] continues with discussing hold baggage screening processes depicted in Figure 2.1 and options for each airport category. This is confirmed by Leone and Liu [35] who contribute with stating possibilities for hold baggage screening. The article also states that the amount of explosives detection systems (EDSs) can be approached numerically by using Equation 2.1. In this equation, P is the passenger volume per hour, T is the percentage of passengers that have no checked baggage and K is the percentage of passengers that require intense screening. Furthermore, r is the demand scale factor between 1,0 and 1,4 that governs the variability in arrival rate. The service volume of the EDS is given by S and the utilization factor by f. In the article, the numerical equation is used in a simulation to determine its applicability for planning activities. Several other detection techniques for baggage are discussed in Wells and Bradley [57].

$$N_{EDS} = \frac{P(1-T)(1+K)r \times B}{S \times f} \tag{2.1}$$

Out of the search results, Franke [37], de Wit and Zuidberg [38], and Barrett [36] discuss competition, limits and demands of low-cost carriers and full-service carriers respectively. Traditionally, full-service carriers (FSCs) provided a high quality but expensive service in a hub and spoke type network. Franke [37] argues that economic downturn and fear of terrorism led to overcapacity at airlines and a long-term decline in revenue. Low-cost carriers (LCCs) contrarily got boosted by this downturn, as passengers looked to avoid high prices. Namely, LCCs are able to deliver 80% of the service quality at less than 50% of the cost of FSCs. This led to establishing a new equilibrium. de Wit and Zuidberg [38] further elaborate the growth limits of the low-cost carrier model. As LCCs strive for lower costs, it is not feasible to have the high operational complexities and vulnerabilities of large hub airports. These are set in terms of costly baggage handling systems, long

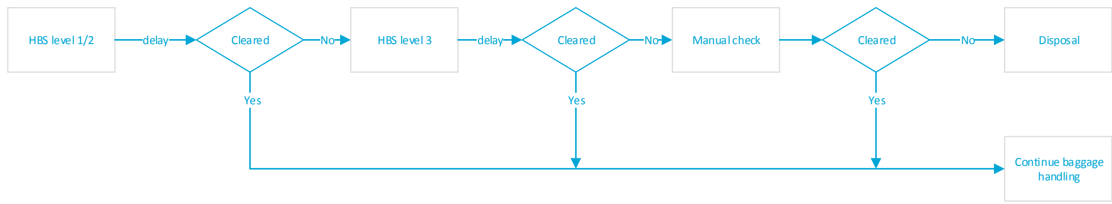


Figure 2.1: Flow of baggage through a tiered security system with different scanning capabilities [1, 10, 11].

turnaround times, high labour costs at hubs and additional peak capacity costs. The costs for handling baggage are circumvented by charging passengers for each item and peak capacity costs by routing to secondary airports. de Wit and Zuidberg [38] therefore conclude that although growth figures for LCCs are impressive, route density will ultimately limit growth perspectives. As is supported by Barrett [36], LCCs will have a larger presence at regional airports, providing a point-to-point model as opposed to the hub and spoke model. By doing so they can reduce turnaround times and waiting times. Additionally, charging passengers for baggage items reduces the need for baggage handling costs for low-cost carriers.

Not only airline competition is a large topic within research, routing and scheduling for baggage handling systems is also well represented. Zeinaly *et al.* [58] discuss the development of a model predictive scheme that optimizes baggage handling system routing, line balancing and empty-cart management on efficiency and reliability. Although several system designs exist, the scheme is developed for cart-based systems. Application of the developed model predictive scheme in a simulation model shows that the scheme is inefficient for large prediction horizons due to the scale of the problem. Comparisons with existing solutions do show that the proposed method outperforms other methods. A method to formally verify routing methods in material handling systems has been proposed by Klotz *et al.* [59] and is based on previous work [60]. Applicability of the method has been shown by using real world examples and show appropriate scaling of the approach with network size. Adding to the routing of baggage is the scheduling of facilities on airports by Abdelghany *et al.* [61]. In order to complete this, a model has been proposed. Departing flights are scheduled to various piers on an existing airport and show a trade-off between satisfying operational requirements and achieving a feasible solution. It is recommended to use the model in finding a near optimal solution. Predictive route control for ICSs in baggage handling is performed by Tarău *et al.* [62]. In order to compute (sub)optimal routes efficiently, an alternative approach is proposed. A benchmark shows that the proposed routes are not optimal but do require significantly less calculation time. A similar approach has been applied to travel-time control [63] and route choice control [64, 65]. Subject to the routing and scheduling of baggage is the analysis of merges in baggage handling systems by Johnstone *et al.* [66]. Tested with two different control algorithms, the simulation results show that throughput efficiency depends on geometric design of a merge point. As the precise design of merge points is part of the detailed design phase, this topic is not further considered.

The possibility of saving energy on belt conveyors is discussed in Lodewijks [67]. A theoretical model of belt conveyor behaviour to inter-arrival patterns is described and tested with experimental data from Rotterdam The Hague Airport in the Netherlands. Calculations show possible energy savings in the order of 300 to 900kWh varying per throughput and belt conveyor length. As belt conveyor controls can be easily altered, it makes sense to assess the energy savings per belt.

In a publication by Gediehn [68], the influence of minimum connecting time (MCT) on baggage handling systems is discussed. An analysis of existing systems yields that for an MCT of 45 minutes, baggage handling may take only 13 minutes from transfer input to final sorting. This is the case as unloading an aircraft typically takes 14 minutes and loading an aircraft takes 18 minutes. In order to achieve an MCT of 30 minutes, significant reductions are required. A solution to this is to locate inputs and final sorting closer to aircraft and increasing the maximum velocities reached in baggage handling systems. Scheduling appropriate gates for critical transfers may also yield reductions.

Rijsenbrij and Ottjes [69] propose a concept to reducing the aforementioned loading and unloading times with the use of a baggage truck that replaces conventional carts for narrow-body aircraft. With the use of a simulation program, viability of the concept is evaluated in multiple cases. Determining the expansion capacities for baggage claim carousels is elaborated in Yoon and Jeong [70]. A method is proposed and applied to planning new capacities for Korea's Incheon International Airport which includes allocation of aircraft to different carousels. The result of this method is a planning when to build additional carousels. Another work by Sørensen [71] focusses on the dynamics of loop-sorting-system chains and implementation thereof. It also supports a tool to design loop-sorting-systems based on the findings of the work.

Thomas *et al.* [72], Pikaar and Asselbergs [73] as well as Korkmaz *et al.* [74] discuss ergonomics on belt conveyors intended for human use. Different heights, loads and velocities are discussed. It is found that conveyor belts may use speeds of up to $0.4m/s$ in case of human interaction. With the exception of conveyor belt velocities, such design details are part of detailed design as discussed in chapter 1 and will therefore not be considered further. For belt conveyor safety at airports, Wang and Jia [75] discuss a method for assessment. As safety is also part of detailed design, it will not be discussed further in this thesis.

Jochems [76] has proposed an application by which Royal Dutch Airlines (KLM) is to differentiate itself from its competitors. In this application, customers of KLM are able to track their baggage through various stages of their journey and are able to receive notifications when problems arise. Other research at KLM searches for operational factors that influence baggage handling processes. This is performed by A.M.M. van der Lande [77]. The research concludes that baggage is often mishandled when its minimum connecting time is short.

All articles on baggage handling systems found in literature have been elaborated upon in this section. Several relevant articles will be compared to practice knowledge that is elaborated upon in chapter 4.

2.1.2. LITERATURE ON DESIGN METHODS

As is elaborated in de Neufville [78], some baggage handling system designs realised appear to be failures. An example of this are the events that took place at Denver International Airport. At this airport, newly developed systems were installed by cutting corners, which was advised against by experts. It is therefore important to select a design method as basis for the model that complies with the recommendations given in de Neufville [78]. Several of these recommendations are:

- avoid obvious mechanical problems like misaligned track,
- ensure proper project planning, execution and final testing,
- ensure delivery of systems is reliable, and
- do not implement technological improvements without testing these both off-site and on-site.

In order to design concepts for products, Keinonen and Takala [79] argue that there is no general approach that may be applied to all products. What is possible according to them is to identify a general set of activities that concepting processes have to follow to solve design problems and deliver acceptable results. They define three layers, namely: *a*) background research, *b*) concept generation and *c*) concept evaluation. After evaluation of concepts, an acceptable result may be selected. Central to these activities is having an overview of the whole product, defining user groups, implementation-related evaluation criteria and test results. The latter is often overlooked, but may be performed by using simulation. Although no specific design method is given, the work does give a guideline by which other design processes can be evaluated.

For educating design thinking, the method by Eekels and Roozenburg [14] is a commonly used method at the TU Delft. The work considers the presented steps as part of the basic design cycle for both concept and detailed design. Instead of generating multiple concepts simultaneously, this method approaches design from an iterative perspective. It assumes trial-and-error will eventually lead to a superior design concept. Candalino [80] researches the design of baggage screening activities. It is stated that current design methods merely evaluate screening strategies rather than design an optimal strategy. The work uses simulated annealing, a cost function driven algorithm, that minimizes expected annual total costs of baggage screening machines. Different ways of finding the best method are evaluated for efficiency. The method however does not consider operational costs of devices, and is therefore not considered further.

Two studies at the TU Delft have researched design methods on different levels. The first is a design method by Pielage [7], which assesses multiple high level design methods and ranks them according to their applicability for designing complex freight handling systems. It is concluded that none of the assessed methods suffice and a new method is conceived. With this newly developed method, several cases have been studied to show strengths and weaknesses of the newly formed method. A more detailed approach was taken by van Vianen [81] in using simulation-integrated design for bulk terminals. Although not specifying the design process, such an approach may prove useful for baggage handling systems.

Furthermore, research at Vanderlande Industries has yielded two design methods for baggage handling systems. The design method by Lemain [43] focusses on documenting all requirements from stakeholders and their changes from a business management perspective. The method however focusses less on other parts such as design rules for selecting appropriate equipment. This is left to designers to specify. Grigoraş

Table 2.3: The scale of importance for comparing activities.

Intensity of importance	Definition	Explanation
1	Equal importance	Two activities are of equal importance
3	Moderate importance	Experience and judgement favor one activity
5	Strong importance	Experience and judgement strongly favor one activity
7	Very strong importance	The dominance of one activity can be demonstrated in practice
9	Extreme importance	The evidence favoring one activity is of the highest possible order of affirmation
2, 4, 6, 8	Intermediate values	Use only when a compromise is needed

and Hoede [44] continue on from equipment selection to automatically defining topologies for given systems with a graph-based method. This method is derived from object processing systems and produces different layouts [82].

In this section, several articles on concept design have been discussed. Methods such as Pielage [7], Lemain [43], Grigoraş and Hoede [44] and van Vianen [81] will be taken into account at the design process comparison in chapter 3.

2.2. MULTI-CRITERIA DECISION MAKING

Continuing on from the literature study on design approaches, this thesis will compare gathered design methods. This is done by regarding the obtained design methods in literature and comparing according to Pielage [7]. As time has passed since that comparison, new methods could have emerged. The current design method at SPPAL can then also be included as a reference. From the literature survey discussed in the previous section, several articles describing design methods are found. Suggestions have been made for additional articles, among which Pielage [7] and van Vianen [81].

In this method, requirements for the design process are stated and ranked according to the analytic hierarchy process by Saaty [51], [52]. Criteria matrix c, containing $n \times n$ elements i and j, is filled with values that compare the importance of element i to element j. The scale of importance for the comparison is given in Table 2.3. Naturally, the importance of element j to i is the reciprocal of the importance of element i to element j. The ranking of individual criteria is obtained by finding the normalized eigenvector of the maximum eigenvalue (λ_{max}).

With this method, it is possible to check the consistency of each comparison. This is performed by first calculating the consistency index (CI) with Equation 2.2 and then calculating the consistency ratio (CR) with Equation 2.3. The random index (RI) may be obtained from Donegan and Dodd [83] for the appropriate matrix size. When the calculated consistency ratio is smaller than 0,2, the matrix is considered consistent. As an exception for people unfamiliar with this technique of ranking, this boundary may be stretched to 0,3.

$$CI = \frac{\lambda_{max} - n}{n-1} \tag{2.2}$$

$$CR = \frac{CI}{RI} \tag{2.3}$$

Measurable criteria, for example energy consumption, may be multiplied by the normalized importance of that criterion. In the case of non-measurable criteria, the alternatives need to be judged on importance in alternatives matrix a_i, with being the i^{th} element to be judged. When the scores for each criterion are denoted in matrix b, the final ranking r may be obtained with Equation 2.4. This result then indicates which solution is preferred according to the ranking.

$$r_{ij} = \sum_{k=1}^{n} b_{ik} c_{kj} \tag{2.4}$$

In the subsequent chapter, chapter 3, the analytic hierarchy process is used to find which design process is preferred based on expert opinion. Experts will compare criteria in a pairwise manner and this ranking

Table 2.4: Available cases for study with given peak flow rate and classification.

Code	Airport name	Peak flow rate [bags/hour]	IATA category	FAA classification
C1	[redacted]	7.000	Category C	International hub
C2	[redacted]	11.500	Category C	International hub
C3	[redacted]	4.800	Category B	International hub
C4	[redacted]	3.700	Category B	International hub
C5	[redacted]	8.300	Category C	International hub
C6	[redacted]	2.100	Category B	Regional hub
C7	[redacted]	2.400	Category B	Regional hub
C8	[redacted]	11.000	Category C	International hub

is multiplied by each design process's performance on each criterion. This results in one preferred design processes that will be recommended for designing baggage handling concepts.

2.3. CASE STUDY RESEARCH

The method by which cases are studied and incorporated in this thesis is described by Dul and Hak [84]. It is argued that a distinction should be made between practice-oriented case studies and theory-oriented case studies. In this thesis, two variants of case study research are used. The first is theory-building research, part of the theory-oriented case studies. The majority of case studies however are practice-oriented. The aim of practice-oriented research is to use theory in practice.

In order to contribute to theory-building research, Dul and Hak [84] argue that exploration of practice cases may be necessary. The objective of theory-building research is to contribute to the development of theory by formulating new propositions based on evidence drawn from observation of instances of the studied object. A proposed method is by discovering concepts and their relation drawn from observation. In the case of baggage handling systems, the object of research, designs and specifications may be studied to build theory on which elements are featured in designs and how these elements may be connected. In most cases, it is assumed that up to this point, nothing on relevant aspects of the subject is known. In practice however this is usually not the case and relevant strategies may be:

- gathering information from general media such as newspapers, television, and the internet;
- reading professional literature, such as managerial, professional and trade literature regarding the object of study;
- communicating with practitioners with experience regarding the object of study;
- visiting places where the object of study occurs and observing it;
- participating in situations in which the object of study occurs.

Applying this generic method to this thesis would mean that several designs may be studied as cases, and a generic baggage handling model may be conceived. A total of eight cases were made available for study. These cases are listed in Table 2.4 with peak flow rate, IATA category and Federal Aviation Administration (FAA) classification. The FAA gives a classification on airports as regional or international, hub or non-hub airports. Whether an airport is deemed international depends on the destinations of airlines at that airport. Classifying an airport as a hub requires it to have more than a 10.000 passenger boardings per year [85]. All airports listed in Table 2.4 are thus considered to be hub airports. Drawings of each airport's baggage handling layout are given in [redacted], with the exception of case C8 since no design was available.

When studying the table it may be observed that among these cases, no airport of category A exist, as no designs were available. This means that small airports may not be represented well by the model. It should also be noted that IATA recommendations on equipment do not hold for every airport. Examples are cases C4 and C6. In cases C4 and C6, the peak flow rate would indicate a category B airport, with up to 5.000 bags per hour. The concept design drawings however only shows equipment from category A.

As cases C1 to C6 only provide information on the layout, which functions are present and how the functions are connected, these cases will be used in chapter 4 to construct a generic model for baggage handling systems. Cases C7 and C8 will be used in chapter 5, to distinct differences in the current design process and newly proposed design process.

2.4. MODEL VERIFICATION AND VALIDATION

A part of assessing the practical applicability of models is to formally verify and validate a model. This is described in both Rykiel [86] and Hillston [87]. It is argued that models are created to represent different virtual or physical objects. A model typically is a simplification of reality to allow for easier computing. To demonstrate that the model is formulated correctly one asks whether the proposed model is right. How this is done is elaborated in subsection 2.4.1. After verification has taken place, a model is often validated. Validation tests whether the proposed model is the right model, as it should represent a part of reality. This is further elaborated upon in subsection 2.4.2.

2.4.1. VERIFICATION

A typical method to verify models is to provide simple inputs and assess whether the outputs correspond to the expected output [87]. This may be performed in several ways. One way is to include additional checks and outputs in a model that may capture remaining bugs. This may already happen during the first program versions and is expanded upon in newer versions. Another possibility is to simplify the model and manually check the results. This may amount to checking every step the model takes by hand. It is recommended to do this for small models. When a model consists of a simulation, step by step checking becomes problematic it is advised to resort to tracing or animation. Finally, one may test the consistency of the output by providing slightly differing inputs. As the generic baggage handling system model in chapter 4 is a relatively simple model, it will be verified by manually relating simplified inputs to expected outputs of the model.

2.4.2. VALIDATION

In order to validate a model, one needs to demonstrate that a model is a reasonable representation of the actual system and that it reproduces system behaviour well enough that it allows for study. Three aspects should be considered during validation: *a*) assumptions, *b*) input parameter values and distribution, and *c*) output values and conclusions. Hillston [87] notes that in practice it may be difficult to achieve a full validation of the model, especially if the modelled system does not exist yet. Initial validation thus relies on expert intuition and real system measurements.

As six cases are used to construct the model, only two cases remains for model validation. As Hillston [87] argues, two cases are insufficient to fully validate a model. Preliminary conclusions on model validation may be drawn from these two design cases studies. Formal validation of the model is recommended for future research.

2.5. CHAPTER SUMMARY

In this chapter, the results to the literature survey on baggage handling systems and concept design have been elaborated. Literature on BHSs primarily focuses on providing equipment recommendations and security screening system descriptions. Literature on design methods intended for BHSs is scarce however, design methods for freight transport systems are available. A collection of methods are elaborated upon in chapter 3. Their applicability in baggage handling system concept design is assessed by experts with the use of the analytic hierarchy process. A preferred design method results from the AHP analysis.

The preferred method is used to construct a generic model for BHSs in chapter 4 with the use of case study research [84]. Cases are assessed, reported on and linked to the preferred design method. The conceived model is verified by comparing manually calculated results with model outputs for given inputs. Case study research is reapplied to draw preliminary conclusions on the model's validation in two final case studies. This is elaborated upon in chapter 5.

3

Design method selection

After elaborating on the methods used for this research in the previous chapter, the current chapter will describe which design method is preferred by experts in baggage handling. This corresponds with the second central research question presented in chapter 1. Part of this central question are several sub-questions. The first sub-question requires a description of the current design method used by BHS experts SPPAL. This is elaborated upon in section 3.1. Several design methods that were found in literature are characterised in section 3.3. This coincides with the second sub-question for this chapter.

In order to evaluate the described design methods, criteria are described in section 3.2. Each method is assessed for relevance per criterion in section 3.3 and a grade is awarded to each method by performing a pairwise comparison prescribed by the analytic hierarchy process. Criteria are ranked in similar fashion by experts with the use of questionnaires and a final method ranking is obtained. This is described in section 3.4. A summary of this chapter is available in section 3.5.

3.1. Current design method at SPPAL

A schematic overview of a typical BHS design project and interactions between parties has been given in Figure 1.5. This can be elaborated upon by a detailed description of the design method by SPPAL which focuses on the supplier. Table 3.1 indicates which activities are performed to successfully deliver concept designs. The described activities are accompanied by their inputs and outputs respectively. The design phase starts with the approval of [redacted] and finalises with [redacted] and [redacted]. Execution of activities happens sequentially from top to bottom and includes: *a*) [redacted], *b*) [redacted], *c*) [redacted], *d*) [redacted], and *e*) [redacted]. During the [redacted] to adopt a [redacted], a choice is made for which [redacted] will be used. This can be performed by either [redacted] or a combination of the aforementioned types. Afterwards the [redacted] are determined and the [redacted] is then designed. [redacted] include [redacted] which are described in [redacted]. Documentation of [redacted] will lead to the [redacted], which describes how the [redacted]. This part is then concluded with a [redacted]. The planning includes milestones for the completion of [redacted]. These milestones are integrated with ongoing projects. Although the project execution steps are described sequentially, projects may retrace steps due to changing customer requirements, also referred to as design iterations. These design iterations stretch design time but may ultimately be beneficial to customer satisfaction.

The current design method will be used as a reference and will be compared to the design methods found in the previously described literature survey. To assess the design methods, criteria are required. These criteria are elaborated upon in section 3.2. Each design method is assessed in section 3.3.

3.2. Criteria definition

Before giving a detailed description and evaluation of the design methods found during literature survey, criteria for assessing these methods will be described in this section. A basis for these criteria may be found

Table 3.1: A description of the concept design phase at SPPAL as part of an entire design project.

System Design Phase		
Entry Conditions: [redacted]		
Inputs	**Activities**	**Outputs**
[redacted]	[redacted]	
[redacted]		
[redacted]	[redacted]	[redacted]
	[redacted]	[redacted]
[redacted]	[redacted]	
	[redacted]	[redacted]
[redacted]		[redacted]
	[redacted]	[redacted]
Exit conditions: [redacted]		

in Keinonen and Takala [79] and Pielage [7] which is expanded upon. A possible set of evaluation criteria may be described as follows:

Aggregation layers The existence of different layers of a system is referred to as aggregation layers. Each layer increases the amount of details that are taken into account. For example, on the first aggregation layer, an entire BHS may be seen as a whole. Its internal features are not visible from this first aggregation layer. Opening the black box reveals the internal systems and interactions between these systems. This may then be considered as the second aggregation layer. A third aggregation layer will then reveal the sub-systems and their properties. Changing properties of a sub-system in turn influences the performance of the whole BHS. Having at least three aggregation layers as trait for a design method is desirable, whilst having less than three aggregation layers is considered unfavourable.

Communication During conceptual design, communication and interaction with a customer is considered important. Design methods that stimulate customer interaction are considered more valuable than those that do not. The scoring for this criterion is determined by how much attention a design process has for a stakeholder analysis, establishment of requirements and tracking changing requirements.

Flexibility The flexibility of a method indicates how easily it can cope with both small and large projects. Not every BHS is of equal size and it is desirable for a method to be applicable on large international airports that handle thousands of passengers each day as well as rural airports with a few aircraft per day. Flexibility also covers the ability of designing new systems as well as revision older systems. Methods being more flexible will be graded higher in the analytic hierarchy process.

Integration The current conceptual design method is part of an encompassing design project. It is thus necessary for a conceptual design method to integrate well with the project phases. It may be argued that this induces a bias towards the grading of the current design method. In order to negate the influence of this bias on the end result, a sensitivity analysis will be performed after grading. As with all criteria in this analytic hierarchy process, grading cannot be performed quantitatively. In order to maintain the replicable nature of such grading processes, grading will be performed by comparing the steps of the current conceptual design method with others.

Iteration At some point in time, it becomes necessary to retrace previously made steps. This is the iterative nature of designing. Causes like unforeseen events, changing requirements or design errors may trigger these iterations and are considered an inherent part of designing. A conceptual design method should indicate the possibility of retracing steps, since retracing is regarded as refining the design. Methods indicating iterations are graded higher than methods that do not.

Life-cycle Regarding the entire life-cycle of a baggage handling system is important since it allows designers to include every aspect in the design. When regarding the retirement of a system for example, options may be considered which allow for easy removal of the build systems.

Parallel development Design methods may conceptualise by amending a previous design or develop multiple concepts simultaneously. As amendments do not explore the widely differing options, investigation of multiple parallel concepts will offer a more informed overview for solutions. Methods defining the parallel development of concepts are graded higher than methods that do not.

Performance indicators In section 1.6, an overview is given on the performance indicators that are of importance to different stakeholders. A design method should be able to determine these indicators with quantitative calculations. Design methods will be ranked according to the possibility of calculating these indicators.

Rapidity The design time for baggage handling systems is regarded as service to the customer. Reducing the designing time improves the level of service towards a customer and can be achieved in two ways. The first option reduces the amount of phases and steps in the conceptual design phase, whilst the second way consists of using faster tools for the design. Reducing the amount of phases and steps in a design process is the only course of action considered for this criterion.

Simplicity The application of this method is easily understandable and it is conveyable to other baggage handling experts. A more clearly defined and properly explained method is considered better. This is indicated by having clear steps in the method, as opposed to having no approach at all.

Development scenarios Estimation of future events is typically done by assessing different scenarios. In order to assess the applicability of a concept in an airport's environment, it must be able to project different scenarios on the concept. The ability to assess different scenarios is not implied by any of the methods and is therefore assessed by the mentioning of calculating a variety of scenarios.

These evaluation criteria will be used to compare various design methods. These methods consist of methods found in literature and the current design method. An evaluation of each method on these criteria is given in the subsequent section.

3.3. DESIGN METHOD EVALUATION

The literature survey presented in the previous chapter has yielded several works on design methods. This section evaluates these design methods, with the exception of negligible ones. An overlapping evaluation on design methods has been performed by Pielage [7] and Cross and Roozenburg [88]. As Pielage [7] has conceived a new method based on the evaluation, adding a method from that research will suffice as dummy method. This dummy method is added in this evaluation as to ensure that the results are non-biased. In the case that the dummy method would rank higher than Pielage [7], the evaluation has not been performed correctly. The method described by Verein Deutscher Ingenieure [12] and Verein Deutscher Ingenieure [13] is selected as dummy. An overview of the methods will then consist of:

- Design method by Verein Deutscher Ingenieure
- Design method by Roozenburg & Eekels
- Design method by Lemain
- Design of freight transport systems
- Design of object processing systems
- Simulation-integrated design
- Current design method

These methods are examined in the following pages. Several beneficial aspects of each design method are mentioned and other disadvantageous aspects as well. The aspects correspond to the design requirements presented in section 3.2.

3.3.1. DESIGN METHOD BY VEREIN DEUTSCHER INGENIEURE (VDI)

The first design method that will be described is the method by Verein Deutscher Ingenieure (VDI). As was mentioned before, this method has been selected as dummy to include into the analysis. The method is frequently referred to for systematically designing complex systems and is conceived by Verein Deutscher Ingenieure [12, 13]. This method is generally applicable in multiple fields of engineering and has been an inspiration for other design methods, confirming its value. As is depicted in Figure 3.2, the design method consists of four phases, of which phase II is denoted as conceptual design. It includes the interpretation of a

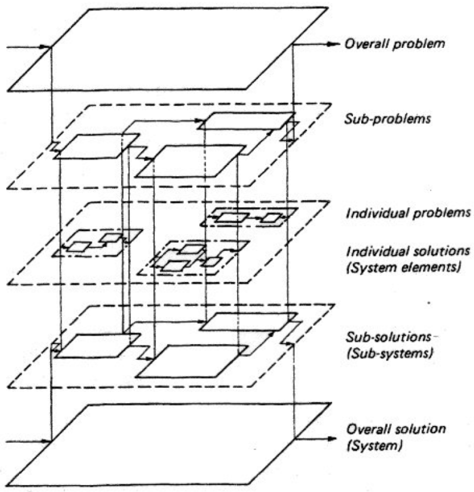


Figure 3.1: A systematic representation of systematic design by VDI [12, 13].

design objective, determining functions, solutions and design structure into layouts. A schematic representation of the modular design is given in Figure 3.1. In Veeke *et al.* [47], this is referred to as aggregation layers. Complex problems are recommended to be separated into different aggregation layers. Each individual problem on the lowest aggregation layer is presented with multiple solutions. Combining different solutions per problem then yields multiple concepts. This process is referred to as 'diverting and converting'. A decision for which concept should be continued to detailed design is then made.

Advantageous aspects of this method are: *1*) it uses at least three aggregation layers, *2*) the method is applicable on variously sized problems, *3*) the method may be integrated at SPPAL, *4*) process steps can be retraced when desired, *5*) the method distinguishes project phases and steps, *6*) requirements are considered during each step, and *7*) diverting and converting assists the solution space. Several disadvantageous aspects are that: *1*) the method does not mention communication, *2*) the life-cycle of the subject is not considered, *3*) not all fundamentally correct steps are performed, and *4*) the method does not indicate the existence of development scenarios.

3.3.2. DESIGN METHOD BY ROOZENBURG & EEKELS (R&E)

In order to teach design thinking in mechanical engineering, the TU Delft teaches how to apply the design method by Eekels and Roozenburg [14]. The authors regard this as a basic design cycle that every design project goes through. Notable steps in the design method are analysis, synthesis, simulation and evaluation, as may be seen in Figure 3.3. The analysis step describes the initial forming of an idea for a new product and its objective. During synthesis, ideas are combined to form a collection of solutions. An image of its function is formed during simulation and functions evaluated. It is also stated that design is an iterative process and decisions to continue with detailing the design may lead to re-doing the problem analysis or synthesis of ideas. The method however disregards the proposed phases. Instead of generating multiple concepts simultaneously, this method approaches design from an iterative perspective, as it assumes that trial-and-error will eventually lead to a superior concept.

Several advantageous aspects are: *1*) being a basic design cycle, the project is applicable in different situations, *2*) as a basic design cycle, it can easily be integrated in any project, *3*) when requirements are not met, the process should be repeated, *4*) steps are clearly defined, *5*) requirements are considered during each step, *6*) diverting and converting assesses solution options, *7*) the fundamentally correct steps are performed, and *8*) simulation allows for assessing different scenarios. Several disadvantageous aspects are that: *1*) the method does not mention aggregation layers, *2*) it does not regard communication, *3*) project phases are not represented in the model, *4*) parallel developments do not take place in this method, *5*) the method does not indicate the existence of scenarios, and *6*) with many iterations, the method can turn out to be lengthy.

3.3.3. DESIGN METHOD BY LEMAIN (LEM)

Lemain [43] describes a method to track requirements for baggage handling systems in spreadsheets. The

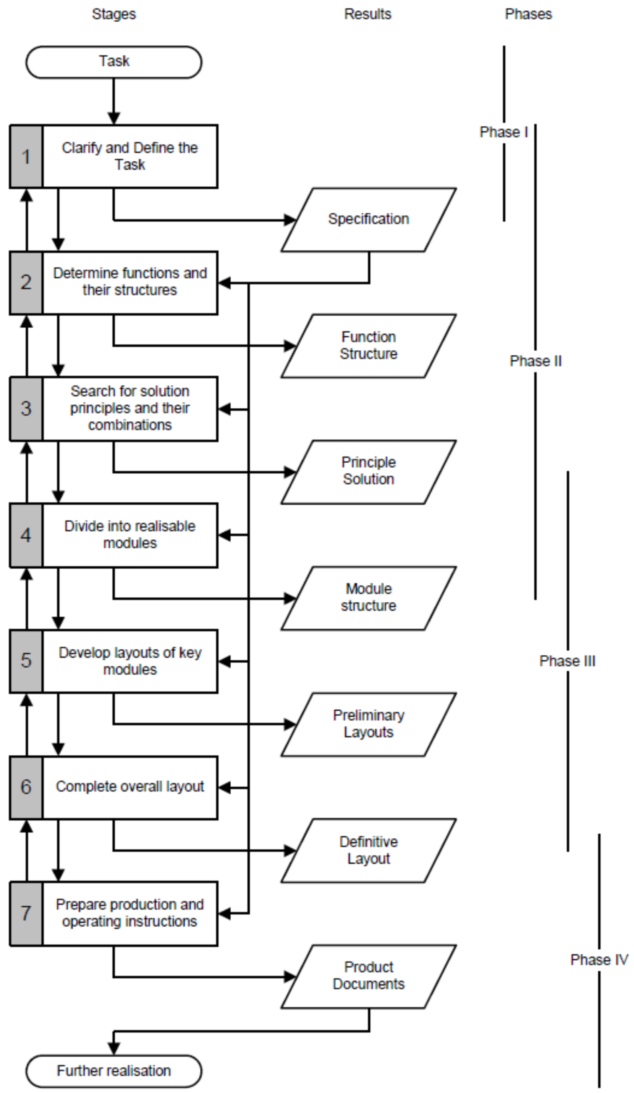


Figure 3.2: A systematic design method presented by VDI that includes conceptual design in phase II [12, 13].

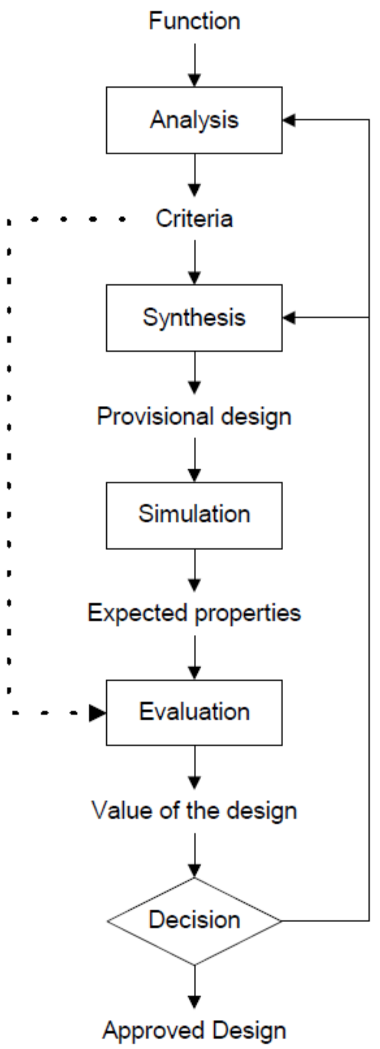


Figure 3.3: The design method by [14].

method is oriented towards tracking of requirements instead of transforming these requirements into one or multiple designs. Not only the requirements themselves are tracked, also the owner, date and stakeholder are denoted. The method has thus focussed less on other parts of the method. In the method, a proposal is made for the use of a specific type of software that allows for simulation in three dimensions, including moving objects. From this simulation it would be possible to define performance indicators for designs, this is however not performed. Instead, typical costs and additional percentages are given for a project and timespan, respectively. From this, an expenditure is calculated.

Several advantageous aspects are: *1*) a thorough mapping of stakeholders is performed, *2*) the life-cycle of a baggage handling system is regarded, *3*) requirements are mapped extensively, enforcing communication, *4*) the method is applicable to differently sized baggage handling systems, *5*) the method would integrate well, and *6*) the method is straight forward and does not require adjusting each case. Several disadvantageous aspects are that: *1*) the method does not mention different aggregation layers, *2*) iterations in the design or requirements mapping are not mentioned, *3*) the method focusses on developing a single concept, *4*) performance indicators are defined in little detail, and *5*) the design is not tested with different scenarios.

3.3.4. Design of freight transport systems (PIE)

As was mentioned before, Pielage [7] evaluates multiple design methods on their applicability in complex freight handling system design problems. After evaluation it was concluded that none of the methods sufficed, and a new method was conceived. The method includes defining a project structure, a design process and its multi-x aspects. Multi-x aspects focus on providing the project layers, disciplines, types of people and types of stakeholders involved. As this thesis focuses on providing conceptual designs, the latter will not be considered further. The design process is depicted in Figure 3.4. The process describes the relation between the problem and solution, as well as intermediary steps. These steps are problem analysis, system definition, system synthesis, simulation and evaluation. During problem analysis, project objectives, stakeholders, requirements and environment are mapped. After this, system boundaries and function criteria are documented. These will lead to the forming of a solution space by diverting and converting during the system synthesis. Contrary to other methods, this method argues for a performance and cost comparison by simulation of several concepts. Based on this comparison, an objective evaluation may occur. A well-defined and comprehensive model can however be regarded as too extensive, such that one might lose track.

Several advantageous aspects are: *1*) multiple aggregation layers are regarded in the method, *2*) communication is well represented by evaluating different disciplines, *3*) the method is designed for large scale freight handling systems, but may be applied on smaller systems as well, *4*) the method clearly states that iterations are an inherent part of design, *5*) steps of the design method are well represented, *6*) phases of a project are well represented, *7*) diverting and converting is taken into account when solving problems, and *8*) simulation allows for the calculation of scenarios. Several disadvantageous aspects are that: *1*) with the many steps, this method is lengthy, *2*) multiple aspects are considered in this method, which reduces simplicity, and *3*) performance indicators are defined in little detail.

3.3.5. Design of object processing systems (G&H)

For the design of object processing systems, Grigoraş and Hoede [82] apply graph theory to determine the layout of transportation flows in buildings. An application for this is determining where conveyors need to be placed in Grigoraş and Hoede [44]. The following steps are described. First, a process flow diagram of baggage is established. It should be noted that in this scenario, equipment such as check-ins, screening, sorters, make-up and reclaims are already preselected. The flows are then mapped on a geometrical constraint graph. These routes are multiplied for their demand and processors are placed at the corresponding locations. This generates a layout, which can be compacted. A constraint of this method is that processors are already predetermined, but may influence the design of a concept.

Several advantageous aspects are: *1*) the method is applicable on both large and small systems, *2*) the method would fit well within design projects, *3*) results are quickly achieved with this method, *4*) performance indicators are calculated, and *5*) different scenarios can be calculated with this method. Several disadvantageous aspects are that: *1*) system is observed from one aggregation layer, *2*) no requirements or stakeholder analysis beforehand, *3*) the method does not mention iteration but could be repeated, *4*) steps and phases are not defined, *5*) the life-cycle of the product is not regarded, and *6*) the method does require knowledge of graph theory.

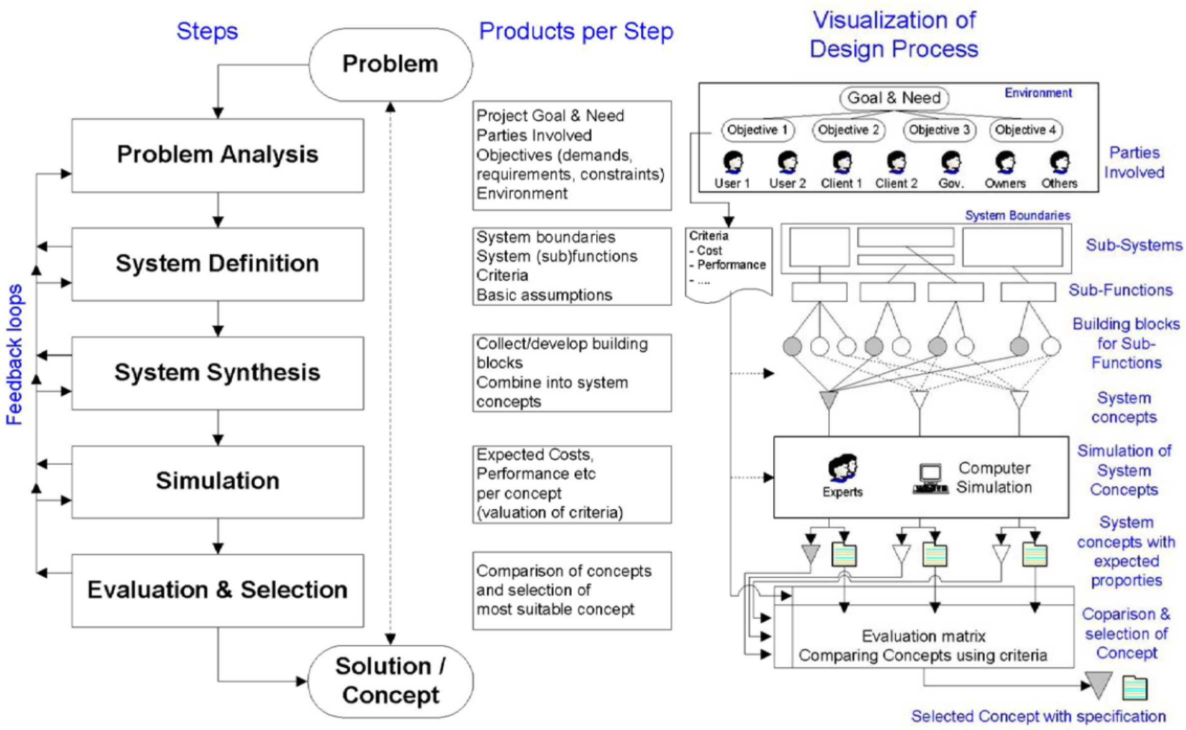


Figure 3.4: A schematic overview of the design process by Pielage [7] and visualisation.

3.3.6. SIMULATION-INTEGRATED DESIGN (VIA)

In van Vianen [81], a simulation model for a dry bulk terminal is created in order to perform simulation-integrated design. Based on measurable variables, an accurate representation of real world scenarios can be given by this simulation model. This may then lead to a well balanced design. As this dissertation focuses on the creation of a simulation model, no actual design process is given. It should be noted however, that a similar approach may be taken for baggage handling systems and could prove beneficial during concept design. Also important to mention is that although no design process is given, elements of previously discussed methods tend to appear in this simulation-integrated design method. An example of this is the existence of aggregation layers in the form of seaside modelling, landside modelling, stockyard design and belt conveyor network design. These are combined into a total terminal in a later stage. Such an approach is closely related to having aggregation layers.

Several advantageous aspects are: *1)* the method uses several aggregation layers, *2)* the method can be used for both small and large systems, *3)* multiple solutions are regarded during design and a substantiated decision is made, *4)* simulation allows for performance indicator calculation, *5)* the method contains few steps and is therefore rapid, *6)* the method is straightforward and therefore simple, and *7)* scenarios can be assessed with simulation. Several disadvantageous aspects are that: *1)* there is no explicit stakeholder analysis and requirements analysis with the exception of characteristics, *2)* integrating this method is difficult as it is designed for bulk terminals, *3)* iteration of the design is not explicitly mentioned, and *4)* the life-cycle of a bulk terminal and its equipment is not regarded.

3.3.7. CURRENT DESIGN METHOD (SIE)

The current design method at SPPAL has already been discussed in section 3.1 and is shown in Table 3.1. The method however has not been evaluated yet. Several advantageous aspects are: *1)* [redacted] , *2)* [redacted] , *3)* [redacted] , *4)* [redacted] , and *5)* [redacted] . Several disadvantageous aspects are that: *1)* [redacted] , *2)* [redacted] , *3)* [redacted] , *4)* [redacted] , and *5)* [redacted] .

3.3.8. DESIGN METHOD COMPARISONS

Comparing the advantages and disadvantages described in the previous sections will lead to establishing comparison tables as proposed by Saaty [51]. The method that is used for comparison is described in section 2.2. There is a table for each criterion which indicates the ranking of each method compared to the other methods. Table 3.2-3.12 represent the comparison per criterion that is based on the methods analysis in previous sub-sections. The tables are placed in alphabetical order from left to right, up to down. A consistency ratio is given for each table and is required to be less than 0.20 in order to be consistent. As may be observed, all tables are considered consistent. It should also be noted that the abbreviations are used to streamline the tables. These abbreviations correspond to the methods discussed in the previous sub-sections. In the subsequent section, these tables are used to find which design method is preferred among baggage handling experts.

With this evaluation of design methods, a ranking of design method per criterion may be obtained. This ranking is then multiplied with the score of each criterion that is provided by experts.

3.4. APPLYING THE ANALYTIC HIERARCHY PROCESS

For the analytic hierarchy process, the previously elaborated evaluation of design methods is combined with expert criteria preferences in order to establish which of the design methods is preferred. An expert criteria comparison is made in subsection 3.4.1, describing the execution of the analysis. Afterwards, subsection 3.4.2 describes the results to the analysis in detail and which of the design methods is preferred.

3.4.1. CRITERIA COMPARISONS

To establish which of the criteria described in section 3.2 are important to BHS experts, a pairwise comparison between criteria is made by five experts in questionnaires. These comparisons are presented in Table I.1-I.5 of Appendix I and follow the method described in section 2.2. To allow for smaller tables, the following abbreviations are used for criteria: *1*) Aggregation layers = AL, *2*) Communication = CO, *3*) Flexibility = FL, *4*) Integration = IN, *5*) Iteration = IT, *6*) Life-cycle = LC, *7*) Parallel development = PD, *8*) Performance indicators = PI, *9*) Rapidity = RA, *10*) Simplicity = SI, and *11*) Development scenarios = DS. Combining these tables with the method comparison tables leads to a preferred design method. This is described in the next subsection.

3.4.2. ANALYSIS RESULTS

The previous sections have presented several tables in which pairwise comparisons are made for criteria and methods. Table I.1-I.5 are combined with equal weights to form an average of the five tables. This averaged score is taken as to minimise outliers among the entries. The averaged table is then used as a weight function for Table 3.2-3.12 as is described in section 2.2. The consistency of this averaged table is 0.11, well within the set boundary of 0.00 – 0.20, and may therefore be applied.

Table 3.14 shows the results after matrix multiplication has taken place. The "standard" column represents the results of the analysis. It is however useful to perform a sensitivity analysis on the criteria to show the robustness of the results. These are depicted in the subsequent columns of the table. In each of these subsequent columns a criterion is taken and its importance is increased by 200%, after which all criteria are normalised. The table shows that in the normal case and 7 sensitivity cases, the design method by Pielage [7] is regarded as most preferred. Simulation-integrated design by van Vianen [81] is favoured in three of the sensitivity cases, making it the second most favoured design. The current design method at SPPAL is only favoured in the integration case. This was to be expected as this method scores high on integration as

Table 3.2: A table containing comparisons for design methods on criterion aggregation layers. The consistency ratio for this comparison is 0,03.

	VDI	R&E	LEM	PIE	G&H	VIA	SIE
VDI	1,00	7,00	7,00	1,00	5,00	3,00	3,00
R&E	0,14	1,00	1,00	0,14	0,33	0,20	0,20
LEM	0,14	1,00	1,00	0,14	0,33	0,20	0,20
PIE	1,00	7,00	7,00	1,00	5,00	3,00	3,00
G&H	0,20	3,00	3,00	0,20	1,00	0,33	0,33
VIA	0,33	5,00	5,00	0,33	3,00	1,00	1,00
SIE	0,33	5,00	5,00	0,33	3,00	1,00	1,00

Table 3.3: A table containing comparisons for design methods on criterion communication. The consistency ratio for this comparison is 0,02.

	VDI	R&E	LEM	PIE	G&H	VIA	SIE
VDI	1,00	1,00	0,11	0,14	1,00	1,00	0,14
R&E	1,00	1,00	0,11	0,14	1,00	1,00	0,14
LEM	9,00	9,00	1,00	3,00	9,00	9,00	3,00
PIE	7,00	7,00	0,33	1,00	5,00	5,00	1,00
G&H	1,00	1,00	0,11	0,20	1,00	1,00	0,14
VIA	1,00	1,00	0,11	0,20	1,00	1,00	0,14
SIE	7,00	7,00	0,33	1,00	7,00	7,00	1,00

Table 3.4: A table containing comparisons for design methods on criterion flexibility. The consistency ratio for this comparison is 0,01.

	VDI	R&E	LEM	PIE	G&H	VIA	SIE
VDI	1,00	0,33	0,33	0,20	0,33	0,33	0,20
R&E	3,00	1,00	1,00	0,33	1,00	1,00	0,33
LEM	3,00	1,00	1,00	0,33	1,00	1,00	0,33
PIE	5,00	3,00	3,00	1,00	3,00	3,00	1,00
G&H	3,00	1,00	1,00	0,33	1,00	1,00	0,33
VIA	3,00	1,00	1,00	0,33	1,00	1,00	0,33
SIE	5,00	3,00	3,00	1,00	3,00	3,00	1,00

Table 3.5: A table containing comparisons for design methods on criterion integration. The consistency ratio for this comparison is 0,05.

	VDI	R&E	LEM	PIE	G&H	VIA	SIE
VDI	1,00	1,00	0,33	5,00	1,00	7,00	0,20
R&E	1,00	1,00	0,33	5,00	1,00	7,00	0,20
LEM	3,00	3,00	1,00	7,00	3,00	9,00	0,50
PIE	0,20	0,20	0,14	1,00	0,20	3,00	0,14
G&H	1,00	1,00	0,33	5,00	1,00	7,00	0,20
VIA	0,14	0,14	0,11	0,33	0,14	1,00	0,11
SIE	5,00	5,00	2,00	7,00	5,00	9,00	1,00

Table 3.6: A table containing comparisons for design methods on criterion iteration. The consistency ratio for this comparison is 0,09.

	VDI	R&E	LEM	PIE	G&H	VIA	SIE
VDI	1,00	0,20	5,00	0,14	3,00	3,00	0,14
R&E	5,00	1,00	7,00	0,33	5,00	5,00	0,33
LEM	0,20	0,14	1,00	0,11	0,33	0,33	0,11
PIE	7,00	3,00	9,00	1,00	5,00	5,00	3,00
G&H	0,33	0,20	3,00	0,20	1,00	1,00	0,20
VIA	0,33	0,20	3,00	0,20	1,00	1,00	0,20
SIE	7,00	3,00	9,00	0,33	5,00	5,00	1,00

Table 3.7: A table containing comparisons for design methods on criterion life-cycle. The consistency ratio for this comparison is 0,01.

	VDI	R&E	LEM	PIE	G&H	VIA	SIE
VDI	1,00	1,00	0,14	0,14	1,00	1,00	0,20
R&E	1,00	1,00	0,14	0,14	1,00	1,00	0,20
LEM	7,00	7,00	1,00	1,00	7,00	7,00	3,00
PIE	7,00	7,00	1,00	1,00	7,00	7,00	3,00
G&H	1,00	1,00	0,14	0,14	1,00	1,00	0,20
VIA	1,00	1,00	0,14	0,14	1,00	1,00	0,20
SIE	5,00	5,00	0,33	0,33	5,00	5,00	1,00

Table 3.8: A table containing comparisons for design methods on criterion parallel development. The consistency ratio for this comparison is 0,02.

	VDI	R&E	LEM	PIE	G&H	VIA	SIE
VDI	1.00	7,00	7,00	1,00	3,00	3,00	5,00
R&E	0,14	1,00	1,00	0,14	0,20	0,25	0,33
LEM	0,14	1,00	1,00	0,14	0,20	0,25	0,33
PIE	1,00	7,00	7,00	1,00	3,00	3,00	5,00
G&H	0,33	5,00	5,00	0,33	1,00	1,00	2,00
VIA	0,33	4,00	4,00	0,33	1,00	1,00	2,00
SIE	0,20	3,00	3,00	0,20	0,50	0,50	1,00

Table 3.9: A table containing comparisons for design methods on criterion performance indicators. The consistency ratio for this comparison is 0,05.

	VDI	R&E	LEM	PIE	G&H	VIA	SIE
VDI	1,00	1,00	0,20	0,20	0,14	0,14	0,14
R&E	1,00	1,00	0,20	0,20	0,14	0,14	0,14
LEM	5,00	5,00	1,00	1,00	0,33	0,33	0,33
PIE	5,00	5,00	1,00	1,00	0,33	0,33	0,33
G&H	7,00	7,00	3,00	3,00	1,00	0,33	1,00
VIA	7,00	7,00	3,00	3,00	3,00	1,00	3,00
SIE	7,00	7,00	3,00	3,00	1,00	0,33	1,00

Table 3.10: A table containing comparisons for design methods on criterion rapidity. The consistency ratio for this comparison is 0,03.

	VDI	R&E	LEM	PIE	G&H	VIA	SIE
VDI	1,00	3,00	0,33	3,00	0,20	0,33	1,00
R&E	0,33	1,00	0,20	1,00	0,14	0,20	0,33
LEM	3,00	5,00	1,00	5,00	0,33	1,00	3,00
PIE	0,33	1,00	0,20	1,00	0,14	0,33	0,33
G&H	5,00	7,00	3,00	7,00	1,00	3,00	5,00
VIA	3,00	5,00	1,00	3,00	0,33	1,00	3,00
SIE	1,00	3,00	0,33	3,00	0,20	0,33	1,00

Table 3.11: A table containing comparisons for design methods on criterion simplicity. The consistency ratio for this comparison is 0,03.

	VDI	R&E	LEM	PIE	G&H	VIA	SIE
VDI	1,00	1,00	0,20	3,00	0,33	0,33	1,00
R&E	1,00	1,00	0,20	3,00	0,33	0,33	1,00
LEM	5,00	5,00	1,00	5,00	3,00	3,00	5,00
PIE	0,33	0,33	0,20	1,00	0,20	0,20	0,33
G&H	3,00	3,00	0,33	5,00	1,00	1,00	3,00
VIA	3,00	3,00	0,33	5,00	1,00	1,00	3,00
SIE	1,00	1,00	0,20	3,00	0,33	0,33	1,00

Table 3.12: A table containing comparisons for design methods on criterion development scenarios. The consistency ratio for this comparison is 0,01.

	VDI	R&E	LEM	PIE	G&H	VIA	SIE
VDI	1,00	1,00	1,00	0,20	0,20	0,14	1,00
R&E	1,00	1,00	1,00	0,20	0,20	0,14	1,00
LEM	1,00	1,00	1,00	0,20	0,20	0,14	1,00
PIE	5,00	5,00	5,00	1,00	1,00	1,00	5,00
G&H	5,00	5,00	5,00	1,00	1,00	0,33	5,00
VIA	7,00	7,00	7,00	1,00	3,00	1,00	7,00
SIE	1,00	1,00	1,00	0,20	0,20	0,14	1,00

Table 3.13: A table containing the results to the pairwise comparison of criteria by XXXX. The consistency ratio of this comparison is 0.08.

	AL	CO	FL	IN	IT	LC	PD	PI	RA	SI	DS
AL	1,00	3,00	1,00	3,00	1,00	5,00	0,33	0,20	1,00	3,00	0,50
CO	0,33	1,00	0,20	0,33	0,33	3,00	0,25	0,20	0,33	3,00	0,33
FL	1,00	5,00	1,00	5,00	1,00	5,00	1,00	0,33	3,00	1,00	1,00
IN	0,33	3,00	0,20	1,00	0,33	5,00	0,20	0,14	0,20	1,00	0,20
IT	1,00	3,00	1,00	3,00	1,00	5,00	0,33	0,33	3,00	3,00	0,50
LC	0,20	0,33	0,20	0,20	0,20	1,00	0,20	0,14	0,20	1,00	0,20
PD	3,00	4,00	1,00	5,00	3,00	5,00	1,00	1,00	3,00	3,00	1,00
PI	5,00	5,00	3,00	7,00	3,00	7,00	1,00	1,00	5,00	5,00	3,00
RA	1,00	3,00	0,33	5,00	0,33	5,00	0,33	0,20	1,00	3,00	0,33
SI	0,33	0,33	1,00	1,00	0,33	1,00	0,33	0,20	0,33	1,00	0,33
DS	2,00	3,00	1,00	5,00	2,00	5,00	1,00	0,33	3,00	3,00	1,00

Table 3.14: Final results and sensitivity analysis to the method comparison.

	Standard	AL + 200%	CO + 200%	FL + 200%	IN + 200%	IT + 200%	LC + 200%	PD + 200%	PI + 200%	RA + 200%	SI + 200%	DS + 200%
VDI	0,09	0,11	0,08	0,08	0,09	0,09	0,08	0,12	0,07	0,09	0,09	0,08
R&E	0,06	0,06	0,06	0,07	0,06	0,08	0,06	0,06	0,05	0,06	0,06	0,06
LEM	0,15	0,14	0,20	0,15	0,16	0,13	0,17	0,14	0,14	0,16	0,18	0,14
PIE	0,20	0,21	0,20	0,21	0,18	0,22	0,21	0,21	0,17	0,17	0,18	0,20
G&H	0,15	0,14	0,13	0,14	0,14	0,13	0,14	0,15	0,17	0,18	0,15	0,16
VIA	0,18	0,17	0,15	0,17	0,16	0,15	0,16	0,17	0,23	0,18	0,18	0,21
SIE	0,18	0,17	0,18	0,19	0,20	0,19	0,18	0,16	0,18	0,16	0,16	0,15

opposed to the other methods. Lastly, it should be noted that the design method by Verein Deutscher Ingenieure [12] scores considerably lower than the method by Pielage [7]. This was also expected since Pielage [7] has based his method on the method by Verein Deutscher Ingenieure [12] among others. This further confirms the correctness in executing the analytic hierarchy process.

Following the analysis of methods and results thereto, as presented in this chapter, it may be concluded that when following the analytic hierarchy process for decision making, the design method by Pielage [7] is preferred among baggage handling experts. This is based on an evaluation of design methods on several criteria and ranking these criteria. The workings of this method have been described in subsection 3.3.4.

3.5. CHAPTER SUMMARY

SPPAL currently uses the design method that is described in section 3.1. A general indication to what steps should be taken is given but no clear design method is presented. Design methods found in literature may be characterised in several ways:

- Verein Deutscher Ingenieure [12, 13] define design from a systems perspective,
- Eekels and Roozenburg [14] define design steps that may be applied in every engineering project,
- Lemain [43] focuses on tracking requirements,
- Pielage [7] introduces a design method for freight handling systems,
- Grigoraş and Hoede [44] use a system definition to find routes between functions, and
- van Vianen [81] describes simulation integrated design for bulk terminals.

A set of criteria, consisting of: *1*) aggregation layers, *2*) communication, *3*) flexibility, *4*) integration, *5*) iteration, *6*) life-cycle, *7*) parallel development, *8*) performance indicators, *9*) rapidity, *10*) simplicity, and *11*) development scenarios has been provided by which each method is evaluated. The criteria have been ranked by BHS experts and results in a preference towards the ability of calculating performance indicators, showing several scenarios and having the possibility of iterations in a design process. Pairwise multiplication and summation of methods per criterion yields that the design of freight transport systems method conceived by Pielage [7] is preferred. This method proposes five design steps for conceptual design: *1*) problem analysis, *2*) system definition, *3*) system synthesis, *4*) simulation, and *5*) evaluation. The design problem for baggage handling systems may be reduced by finding common traits in design cases. Several design cases are therefore analysed in chapter 4. From this analysis a system definition and system synthesis follows. These cases also allow for the construction of a generic model for baggage handling system simulation during concept development.

4

DESIGNING A GENERIC MODEL

Previous chapters of this report have elaborated on the motivation for this research, available literature on the subject, research methods and selecting a preferred design method as basis for concept design. The current chapter will continue by using the selected design method to construct a generic model for baggage handling systems. This coincides with the third central research question of the research framework in chapter 1.

The design process for the selected method follows five consecutive steps for the concept phase of design projects. These steps are: *1*) defining goals and stakeholders in the problem analysis step, *2*) defining system boundaries and criteria in the system definition step, *3*) defining system concepts by combining function solutions in the system synthesis step, *4*) simulating different concepts, and *5*) evaluating the simulated concepts on performance and costs. These steps are used as a framework for this chapter and section 4.2 to section 4.6 elaborate on them respectively. Before these steps are described however, section 4.1 describes the cases on which the generic model will be based. This chapter is concluded with a summary discussing the sub-questions to this chapter in section 4.7.

4.1. CASE ANALYSIS

As was mentioned in the introduction to this chapter, several cases will be analysed to establish a generic model for baggage handling systems. This is done by first studying these cases and then comparing this practice knowledge to the knowledge obtained from literature. The case selection is described in section 2.3. These cases correspond to the layouts depicted in Appendix G.

C1 The following functions are observed in case C1: *1*) infeed with two-belt check-ins in line, *2*) security screening units, *3*) sorting by diverters, and *4*) make-up with carousels. These functions are interconnected by belt conveyors and baggage is transported with tractor pulled carts between make-up and aircraft. Several additional security screening units are stand-alone devices to screen transfer baggage.

C2 The following functions are observed in case C2: *1*) infeed with three-belt check-ins in several groups, *2*) transfer inputs for transferring baggage, *3*) sorting with the use of a tray system, *4*) security screening in several levels, *5*) storage in several storage lanes, and *6*) make-up with carousels. The functions are connected in order, the check-ins and transfer infeeds connect to loading stations for the tray system. The tray system allows for sorting to screening, storage and make-up carousels. Before storage and make-up occur however, it is required to pass security screening first.

C3 The following functions are observed for baggage reclaim in case C3: *1*) infeed takes place with laterals, *2*) transportation is performed with belt conveyors, and *3*) baggage reclaim is done with baggage carousels. For departures, the following functions are observed: *1*) infeed happens with two-belt check--ins, *2*) security screening is done in several levels, *3*) sorting is done with diverters, and *4*) make-up is performed with both laterals and carousels. Temporary storage may happen on the baggage carousels and laterals at the make-up. The functions described are connected in order with the use of belt conveyors.

C4 The following functions are observed for baggage reclaim in case C4: *1*) infeed takes place with laterals, *2*) transportation is performed with belt conveyors, and *3*) baggage reclaim is done with baggage carousels. For departures, the following functions are observed: *1*) infeed occurs with two-belt check--ins, *2*) security screening takes place in levels, *3*) sorting occurs with diverters, and *4*) make-up occurs with two carousels. The functions are connected in order with belt conveyors. Check-in will lead to security screening, after which diverters sort the bags to different carousels.

C5 The following functions are observed in case C5: *1*) infeed with three-belt check-ins in several islands, *2*) transfer inputs for transferring baggage, *3*) sorting with the use of a tray system and diverters, *4*) security screening in several levels, *5*) storage in several storage lanes, and *6*) make-up with carousels and laterals. The functions are connected in order, the check-ins and transfer infeeds connect to loading stations for the tray system. The tray system allows for sorting to screening, storage and make-up carousels. Before storage and make-up occur however, it is required to pass security screening first.

C6 The following functions are observed in case C6: *1*) infeed takes place with two-belt check-ins, *2*) security screening takes place after infeed, *3*) sorting is performed with diverters, and *4*) make-up takes place with laterals and carousels. The functions are connected in order with belt conveyors.

Of the previously analysed cases, an overview of relations is presented in Table 4.1. In literature, few articles have been written on conceptual and detailed design of baggage handling systems. Several articles focus on the design of security systems such as security screening. These are described by Bradley [25], Leone and Liu [35], and Wells and Bradley [57]. The proposed models and described systems correspond to what has been found in the described practice cases. Security screening is constructed in a layered manner from level 1 to level 4. Each level has a rejection ratio of 30%, 15%, 4% and 2% respectively. This is also elaborated on in subsection 4.5.1. Furthermore, Lodewijks [46] describes several functions for baggage handling systems. These correspond to the functions analysed in practice cases, with the exception of individualisation and identification as these functions are part of sorting systems to recognise items.

Table 4.1: Occurrence of functions and connections in cases.

Function	Type	Occurrence					
		C1	C2	C3	C4	C5	C6
In-feed	Check-in	x	x	x	x	x	x
	Transfer		x			x	x
	Termination		x		x	x	x
Screening	Security screening	x	x	x	x	x	x
	Customs screening	x	x			x	x
Transportation	Manual	x	x	x	x	x	x
	Belt conveyor, ICS	x	x	x	x	x	x
	ICS return track		x			x	
Sorting	Sorting systems	x	x	x	x	x	x
	Required trays		x			x	
Storage	Positions		x			x	
Outlet	Make-up systems	x	x	x	x	x	x
	Reclaim systems		x		x	x	x
	Conveyor length	x	x	x	x	x	x

4.2. PROBLEM ANALYSIS

In the problem analysis step of a project, a project's objective is defined as well as the involved parties. This is described in the design method by Pielage [7] as the first step. A unified goal of baggage handling systems at airports is: *"To transport baggage from ticket counters to areas where the bags can be loaded onto aircraft. A BHS also transports bags from an aircraft to a reclaim area or other aircraft"* [46]. Several parties that are involved in this process are: *1*) airlines, *2*) airport holding group, *3*) architects, *4*) consultants, *5*) contractors & suppliers, *6*) customs & state police, *7*) governments, *8*) ground handlers, *9*) investors, and *10*) municipalities [39, 43]. These parties may also be referred to as stakeholders. A detailed description of these stakeholders is given in Appendix D. Each stakeholder may have influence on specific design traits of a baggage handling system. It is therefore denoted on which design traits stakeholders are considered to have influence.

Several of the traits that stakeholders have influence on may be determined quantitatively, based on system performance. These traits have been described as performance indicators in chapter 3 and may be approximated during the conceptual design phase. A list of performance indicators that will be used to quantify baggage handling systems:

- in-system time
- system throughput
- storage capacities
- required space per sub-system
- energy consumption per sub-system
- capital expenditure
- operational expenditure

It should be noted that these performance indicators are not a definitive set and may be expanded upon in further research.

Other design traits such as a flight schedule, opening times of an airport terminal, screening equipment, screening regulations and desired processing time may be used as inputs to the model. Additionally, non-measurable traits such as work and safety regulations as well as ergonomics are considered to be part of the detailed design phase of projects. Determining where safety railings are placed is an example of that and these traits will not be considered hereafter. Government influences such as airspace access and aircraft movements are considered to be taken into account in the flight schedule and will not be considered further. Stakeholders may also be able to express demands in the form of transport distances, processing times, check-in opening times, make-up opening times, transfer windows and the way carts and ULDs are processed. These are also considered as inputs to the model.

These inputs and parameters have been included in the model, as the following paragraph will elaborate upon. As is depicted in Figure 4.1, the flight schedule consists of an airline and type of flight, turnaround time, occupation ratio, baggage ratio and transfer ratio. These values are used to generate flights, passengers and bags at the times and days that an aircraft is supposed to arrive and depart. Generating passengers and baggage happens stochastically, by which variations are taken into account. It is then possible to simulate patterns of real life passengers, referred to as passenger presentation curves (PPCs). Figure 4.2 details this information with check-in opening times, a transfer window and make-up opening times for different flight characters. The ULD train or cart train process for departing and arriving flights is also modelled to achieve a realistic processing pattern. Finally, several design and result formatting parameters may be set in the 'design parameters' tab that is depicted in Figure 4.3. These include transport times and distances, which in turn govern the average transport speed. The processing time for functions is also adjustable in this chart.

The information in these input tables will be retrieved by the program to generate passengers per aircraft and assign baggage to every passenger. The baggage is simulated to determine the required capacity. This is described in the subsequent sections.

As is cited by van Doorne [89], airports experience different demands on daily, weekly, monthly and annual basis. Several sources therefore do not recommend the use of the absolute peak demand, as this would lead to over-designing the system. Reichmuth *et al.* [90] describes several methods to determine the maximum demand for a system. A common approach taken is described by Wang and Pitfield [91], in which the 30^{th} peak hour of a year is taken. Accordingly, this corresponds to the 95% certainty of providing appropriate service [92, 93]. Although using the capacity that is in line with the 95% certainty of providing appropriate service is well founded, this model will also provide a value for the maximum and average design capacity. This is done to allow for the ability to make decisions on sub-system level.

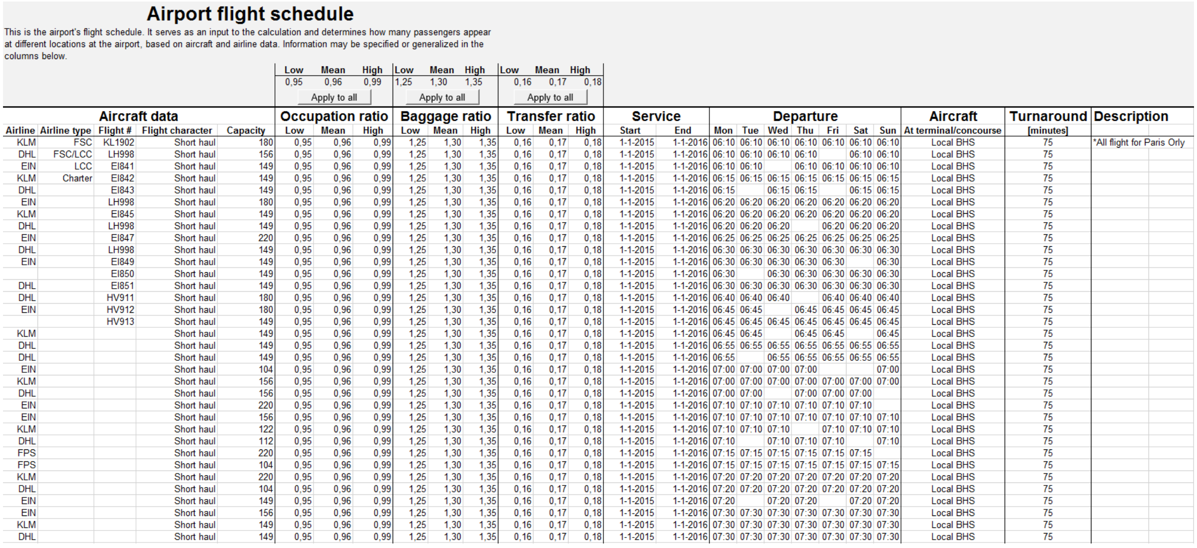


Figure 4.1: Flight schedule input list including turnaround time, occupation ratio, baggage ratio, transfer ratio, airline and type as well as start and end date per scheduled flight.

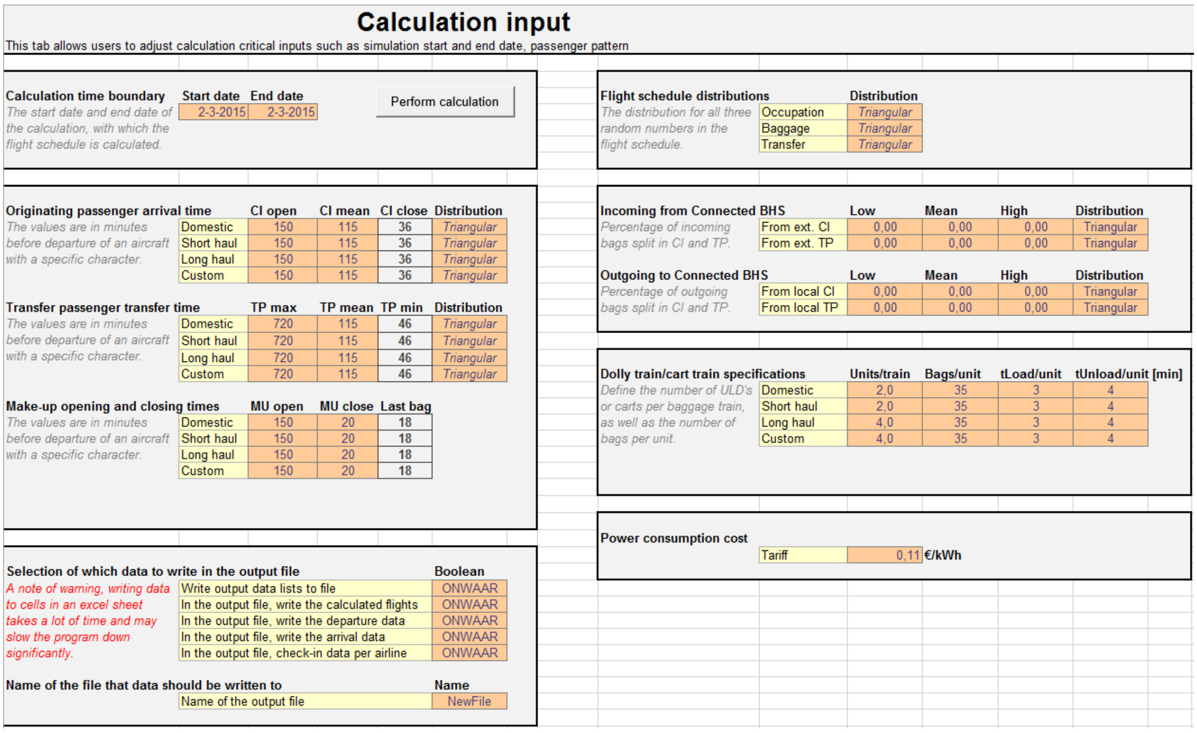
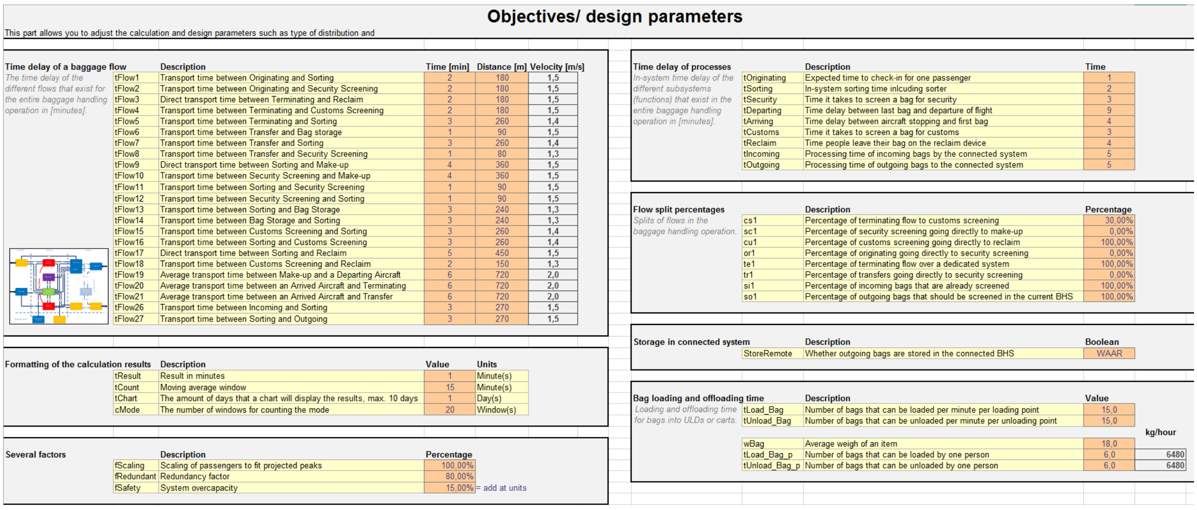


Figure 4.2: Inputs to the calculation that include check-in, transfer and make-up times as well as distribution and cart or ULD handling per type of flight.



Objectives/ design parameters

This part allows you to adjust the calculation and design parameters such as type of distribution and

Time delay of a baggage flow — *The time delay of the different flows that exist for the entire baggage handling operation in [minutes].*

	Description	Time [min]	Distance [m]	Velocity [m/s]
tFlow1	Transport time between Originating and Sorting	2	180	1,5
tFlow2	Transport time between Originating and Security Screening	2	180	1,5
tFlow3	Direct transport time between Terminating and Reclaim	2	180	1,5
tFlow4	Transport time between Terminating and Customs Screening	2	180	1,5
tFlow5	Transport time between Terminating and Sorting	3	260	1,4
tFlow6	Transport time between Transfer and Bag storage	1	90	1,5
tFlow7	Transport time between Transfer and Sorting	3	260	1,4
tFlow8	Transport time between Transfer and Security Screening	1	80	1,3
tFlow9	Direct transport time between Sorting and Make-up	4	360	1,5
tFlow10	Transport time between Security Screening and Make-up	4	360	1,5
tFlow11	Transport time between Sorting and Security Screening	1	90	1,5
tFlow12	Transport time between Security Screening and Sorting	1	90	1,5
tFlow13	Transport time between Sorting and Bag Storage	3	240	1,3
tFlow14	Transport time between Bag Storage and Sorting	3	240	1,3
tFlow15	Transport time between Customs Screening and Sorting	3	260	1,4
tFlow16	Transport time between Sorting and Customs Screening	3	260	1,4
tFlow17	Direct transport time between Sorting and Reclaim	5	450	1,5
tFlow18	Transport time between Customs Screening and Reclaim	2	150	1,3
tFlow19	Average transport time between Make-up and a Departing Aircraft	6	720	2,0
tFlow20	Average transport time between an Arrived Aircraft and Terminating	6	720	2,0
tFlow21	Average transport time between an Arrived Aircraft and Transfer	6	720	2,0
tFlow26	Transport time between Incoming and Sorting	3	270	1,5
tFlow27	Transport time between Sorting and Outgoing	3	270	1,5

Formatting of the calculation results

	Description	Value	Units
tResult	Result in minutes	1	Minute(s)
tCount	Moving average window	15	Minute(s)
tChart	The amount of days that a chart will display the results, max. 10 days	1	Day(s)
cMode	The number of windows for counting the mode	20	Window(s)

Several factors

	Description	Percentage	
fScaling	Scaling of passengers to fit projected peaks	100,00%	
fRedundant	Redundancy factor	80,00%	
fSafety	System overcapacity	15,00%	= add at units

Time delay of processes — *In-system time delay of the different subsystems (functions) that exist in the entire baggage handling operation in [minutes].*

	Description	Time
tOriginating	Expected time to check-in for one passenger	1
tSorting	In-system sorting time inlcuding sorter	2
tSecurity	Time it takes to screen a bag for security	3
tDeparting	Time delay between last bag and departure of flight	9
tArriving	Time delay between aircraft stopping and first bag	4
tCustoms	Time it takes to screen a bag for customs	3
tReclaim	Time people leave their bag on the reclaim device	4
tIncoming	Processing time of incoming bags by the connected system	5
tOutgoing	Processing time of outgoing bags to the connected system	5

Flow split percentages — *Splits of flows in the baggage handling operation.*

	Description	Percentage
cs1	Percentage of terminating flow to customs screening	30,00%
sc1	Percentage of security screening going directly to make-up	0,00%
cu1	Percentage of customs screening going directly to reclaim	100,00%
or1	Percentage of originating going directly to security screening	0,00%
te1	Percentage of terminating flow over a dedicated system	100,00%
tr1	Percentage of transfers going directly to security screening	0,00%
si1	Percentage of incoming bags that are already screened	100,00%
so1	Percentage of outgoing bags that should be screened in the current BHS	100,00%

Storage in connected system

	Description	Boolean
StoreRemote	Whether outgoing bags are stored in the connected BHS	WAAR

Bag loading and offloading time — *Loading and offloading time for bags into ULDs or carts.*

	Description	Value	kg/hour
tLoad_Bag	Number of bags that can be loaded per minute per loading point	15,0	
tUnload_Bag	Number of bags that can be unloaded per minute per unloading point	15,0	
wBag	Average weigh of an item	18,0	
tLoad_Bag_p	Number of bags that can be loaded by one person	6,0	6480
tUnload_Bag_p	Number of bags that can be unloaded by one person	6,0	6480

Figure 4.3: Design parameters such as transport distance, transport time, processing time and function splits as well as formatting of the results.

4.3. SYSTEM DEFINITION

As second step in concept design, Pielage [7] proposes to define system boundaries, define functions and make assumptions on how to model the problem. The design criteria by which designs are evaluated are already mentioned in the previous section as performance indicators. Based on these indicators, performances are compared with a multi-criteria analysis. This is further detailed in following sections.

4.3.1. SYSTEM BOUNDARIES

The system boundaries will give a clear indication of which matters will be taken into account in the design. Several inputs and outputs to the model have been described in section 4.2. This section will go into detail on the actual baggage handling system itself. The functionality of BHSs has been described in section 1.1 and depicted in Figure 1.1. Boundaries to such systems are: *1*) originating input, *2*) transfer input, *3*) terminating input, *4*) incoming connections from other systems, *5*) outgoing connections to other systems, *6*) aircraft loading, and *7*) aircraft unloading. Additionally, conceptual designs typically do not design past the third aggregation layer. The first aggregation layer consists of the entire baggage handling system of an airport. The second aggregation layer describes the functions inside the first layer. The third aggregation layer will detail which sub-systems are needed to meet the required demand. This is thus also considered a boundary of the system.

Matters that are not included in the system boundaries are the detailed simulation of passenger movements throughout the terminal, check-in area and reclaim area. As passenger movements at check-in and reclaim areas may influence throughput and capacity of such systems, an average dwell-time is used for passengers and baggage in these areas [94].

4.3.2. SYSTEM FUNCTIONS

A historical way of categorising baggage handling systems is by size and complexity according to the throughput of a system [25]. This is however not an ideal categorisation since many solutions cover a wide throughput. Another possibility of partitioning BHS equipment is to make this division on a functional level. The reason for doing this is that missing parts of a system can be found easily, since every desired function needs to be covered by the baggage handling system [95]. A typical handling system is depicted in Figure 1.1 and shows some of the functions that are required. From practical and theoretical knowlegde, the following basic functions are required in baggage handling systems: *a*) in-feed, *b*) security screening, *c*) transport, and *d*) outlet.

Additional to the basic functions, it may be decided to integrate other functions into the baggage handling system [95]: *a*) sorting, and *b*) storage. Sorting involves diverting and merging of baggage items with different destinations and can also be performed for the balancing of loads. The necessity for sorting exists because baggage items on a single conveyor may have different destinations. Storage on the other hand is used to store departure and transfer baggage that is too early for loading. Due to the working of sorting and storing,

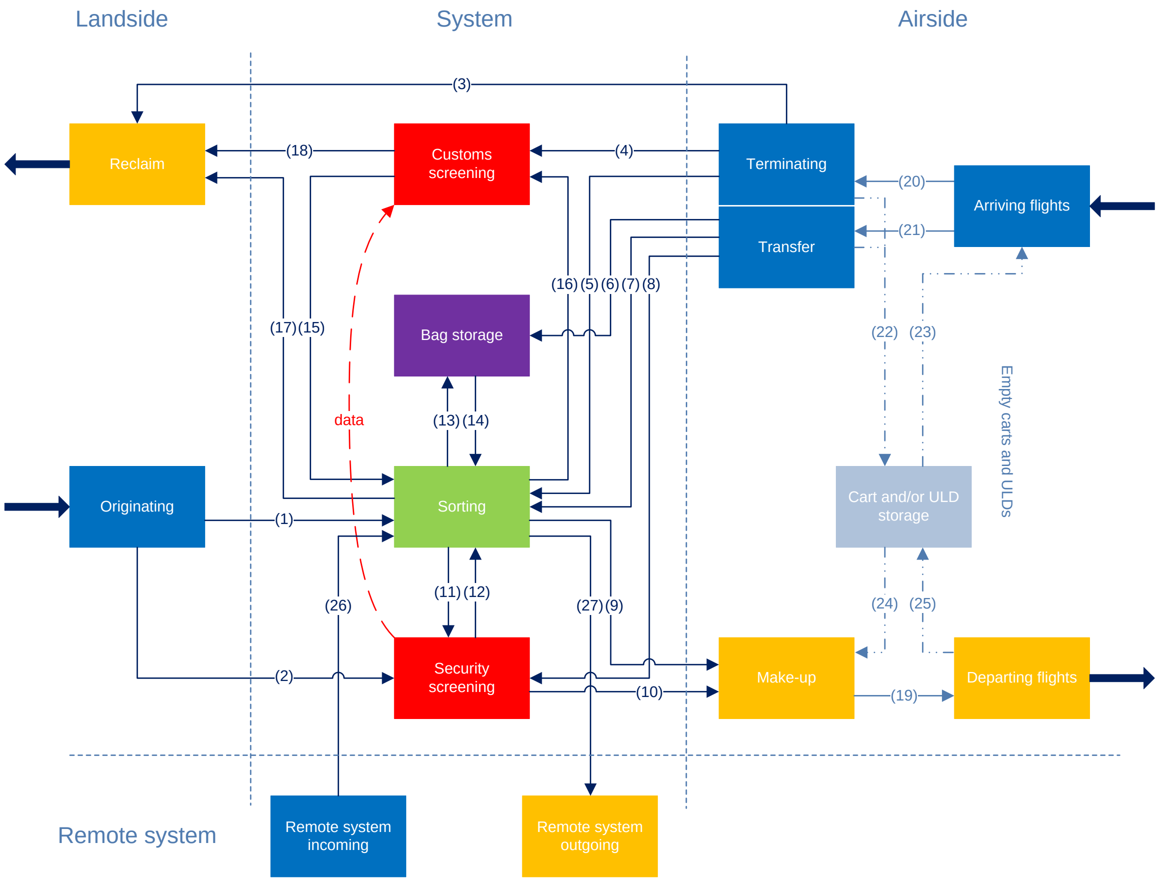


Figure 4.4: A schematic overview of a generic model for baggage handling system functions and connections. The model may be divided into landside, airside and system. It should also be noted that the sorting system may connect to a remote system that is already present or will be built in a future expansion.

item individualisation and identification becomes necessary. The order of these additional functions may change in the system, depending on national and international regulations as well as preferences. A generic model containing these baggage handling system functions is depicted in Figure 4.4.

A simple depiction of connections between several baggage handling system functions is given in Figure 1.1. Practical and theoretical knowledge however have given more insight in additional connections. Examples are: *a*) check-ins may directly connect to either sorting or security screening, *b*) security screening may connect to sorting or directly go to make-up, *c*) sorting is connected to storage, security screening, customs screening, make-up and baggage reclaim, *d*) transfer infeed may connect to sorting and security screening, *e*) terminating infeed may connect to sorting, customs screening or baggage reclaim, and *f*) customs screening may connect to sorting and reclaim. These connections represent the transport function, connecting all other functions. In the construction of a model, it is required to incorporate the mentioned connections. A model proposal is depicted in Figure 4.4.

4.3.3. BASIC ASSUMPTIONS

In order to capture and model the behaviour of baggage handling systems, basic assumptions are to be made. As this part of the design is concerned with concepts, details such as work and safety regulations will not be regarded. The following assumptions are made for the designs and generic model:

- Flights in the flight schedule will not be delayed nor cancelled.
- A year contains 365 days.
- Standard and OOG sized baggage are considered unified items as several baggage handling systems are capable of handling both sizes.
- When check-ins are opened before make-up carousels are assigned to a flight, baggage is rerouted to an early bag storage system.

- When transfer inputs are opened before make-up carousels are assigned to a flight, baggage is rerouted to an early bag storage system.
- The described six cases give an accurate representation of various baggage handling systems.
- Passenger movement simulation may be approximated by an average baggage dwell-time.
- System failures are not considered in the simulation, instead a redundancy factor is taken into account.
- When loading carts and ULDs at a make-up area, fully loaded cart and ULD trains are immediately removed.
- When unloading carts and ULDs at a transfer or terminating area, empty cart and ULD trains are immediately removed.
- System start-up times may be neglected as average energy consumption accounts for this fact.
- It is assumed that the shape of required space may still be adjusted as this model is applied in the concept phase of design projects, allowing for building adjustments.

These system boundaries, functions and assumptions define the solutions space of concepts. This solution space is explored in the next section.

4.4. SYSTEM SYNTHESIS

An analysis of baggage handling equipment from several suppliers has been made in Appendix E. Equipment is sorted by type into different functions. These functions correspond to the functions described in subsection 4.3.2. This is what Pielage [7] describes as a system synthesis, finding concept solutions to the functions defined in the previous steps. The following equipment exists to fulfil these functions:

In-feed Equipment used at the in-feed of baggage handling systems can be divided into two groups. The first group contains landside in-feed equipment and the second group contains airside in-feed equipment. Among the first group are: *1*) check-in counters, *2*) baggage drop-off counters, *3*) OOG check-in counters, *4*) car park check-in counters, and *5*) railway station check-in. The second group consists of: *1*) terminating unloading piers, *2*) transfer unloading piers, and *3*) automated unloaders for carts and ULDs. This type of equipment is indicated with blue coloured process blocks in Figure 4.4. In order to maintain consensus, these sub-systems are also indicated with a blue colour in Figure 4.6.

Screening As has been described in several articles, security screening is an important part of a baggage handling system. Due to past incidents, there is a requirement for screening all baggage that passes baggage handling systems. Screening systems are: *1*) standalone screening unit, and *2*) in-line screening unit. It should be noted that both screening units are available with different belt velocities, among which with $0,3; 0,34$ and $0,5^{m}/_{s}$. It goes without saying that an increased velocity offers additional throughput capacity. This type of equipment is indicated with red coloured process blocks in Figure 4.4. In order to maintain consensus, these sub-systems are also indicated with this colour in Figure 4.6.

Transport Transportation of items occurs between functions. It may be fulfilled by the following equipment: *1*) manual transport, *2*) belt conveyors, *3*) chutes and elevators, and *4*) ICSs. An ICS typically operates in the range of $6-10^{m}/_{s}$, whereas belt conveyors are available with velocities of $1,5$ and $6^{m}/_{s}$, standard- and high-speed conveyors respectively. Manual transport may occur by hand or carts and its average velocity is dependant on the transport distance. This type of equipment is indicated with dark blue coloured lines between process blocks in Figure 4.4. In order to maintain consensus, these sub-systems are also indicated with this colour in Figure 4.6.

Sorting Advanced BHSs may include a sorting process. This may range from: *1*) manual sorting and *2*) in-line conveyor sorting, to *3*) dedicated sorting equipment and *4*) in-line ICS sorting. In-line conveyor sorting occurs with pushers, diverters and vertical sorters. Dedicated sorting equipment comprises of tilting tray sorters and cross-belt sorters. Finally, in-line ICS sorting may occur with switches or tilting track. The sorting process is indicated with a green coloured block in Figure 4.4. To maintain consensus between figures, these sub-systems are also indicated with a green colour in Figure 4.6.

Storage In the case that baggage may arrive at the airport before a departing aircraft is being loaded, baggage is stored. This may be fulfilled by: *1*) manual storage, *2*) belt conveyor buffer lines, *3*) ICS carrier lines, *4*) storage and retrieval systems, and *5*) virtual looping. Virtual looping occurs when baggage is kept in a sorting loop. In order to achieve this, the sorting system needs to consist of a loop. The sorting process is indicated with purple blocks in Figure 4.4. To maintain consensus between figures, these sub-systems are also indicated with this colour in Figure 4.6.

Outlet The system outlet consists of: *1*) bins, *2*) laterals, *3*) carousels, and *4*) automatic loaders. It should be noted that these outlets, with the exception of an automatic loader, may be used in both the make-up and baggage reclaim areas. Yellow blocks indicate system outlets in Figure 4.4. The same colour is used for outlets in Figure 4.6.

A morphological overview of the sub-systems for described functions is depicted in Figure 4.6. Each subsystem is connected to other systems in order to indicate functional compatibility. The figure depicts the morphology for all categories of systems, but may also be presented for IATA categories A to C. These can be found in Appendix F.

With the solution space, system boundaries and model assumptions defined, the next concept design process step may be performed. This includes simulation of the system. The next section will elaborate on this.

4.5. SIMULATION

After describing the system synthesis as step three of the design process, simulation of concepts occurs. This is step four in the design method by Pielage [7]. In the previous sections an effort has been made to construct a generic model for baggage handling systems. This model is depicted in Figure 4.4 and baggage items may be simulated within this model. Based on this simulation, performance indicators may be determined for each process. Which performance indicators are taken into account is described in section 4.2.

The calculations used in simulation and to determine values for performance indicators are described in subsection 4.5.1. What simulation software to use is selected in subsection 4.5.2. The theoretical formulation is crafted into a process description language (PDL) in subsection 4.5.3. This PDL is expanded into fully working software. The functionality of this software is verified to guarantee proper functioning. Model verification is described in subsection 4.5.4.

4.5.1. THEORETICAL FORMULATION

In the system definition, boundaries to the baggage handling system have been proposed and system functions defined. These functions include transport, screening and storage among others. Transport and screening are typical functions that, when the system functions properly, do not accumulate items. The storage function on the other hand does accumulate items throughout the day and dispenses them when a make-up device is designated to an aircraft. The behaviour of both function types is described by Equation 4.1 and Equation 4.2 respectively. Figure 4.5 depicts a generic example for a function in support of both equations. In these equations n represents the current time step and t represents the time delay a function induces.

$$c_n + d_n = a_{n-t} + b_{n-t} \tag{4.1}$$

$$F_n = F_{n-1} + a_n + b_n - c_n - d_n \tag{4.2}$$

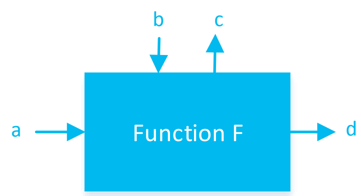


Figure 4.5: A representation of a model function block.

It should be noted that several functions, such as sorting, make-up and baggage reclaim may accumulate items for a short time before dispensing them to other functions. During sorting this is the case as the process requires several minutes to complete. At make-up and baggage reclaim however, items are held until they are loaded into trains or picked up by passengers respectively. In this model, an adjustable average dwell time for baggage at the baggage reclaim area is used whereas items at make-up wait until a baggage train arrives. With this function description, the generic model is able to determine required throughputs at each function based on given inputs such as flight schedule, function times and distances. Splits in flows may also be added.

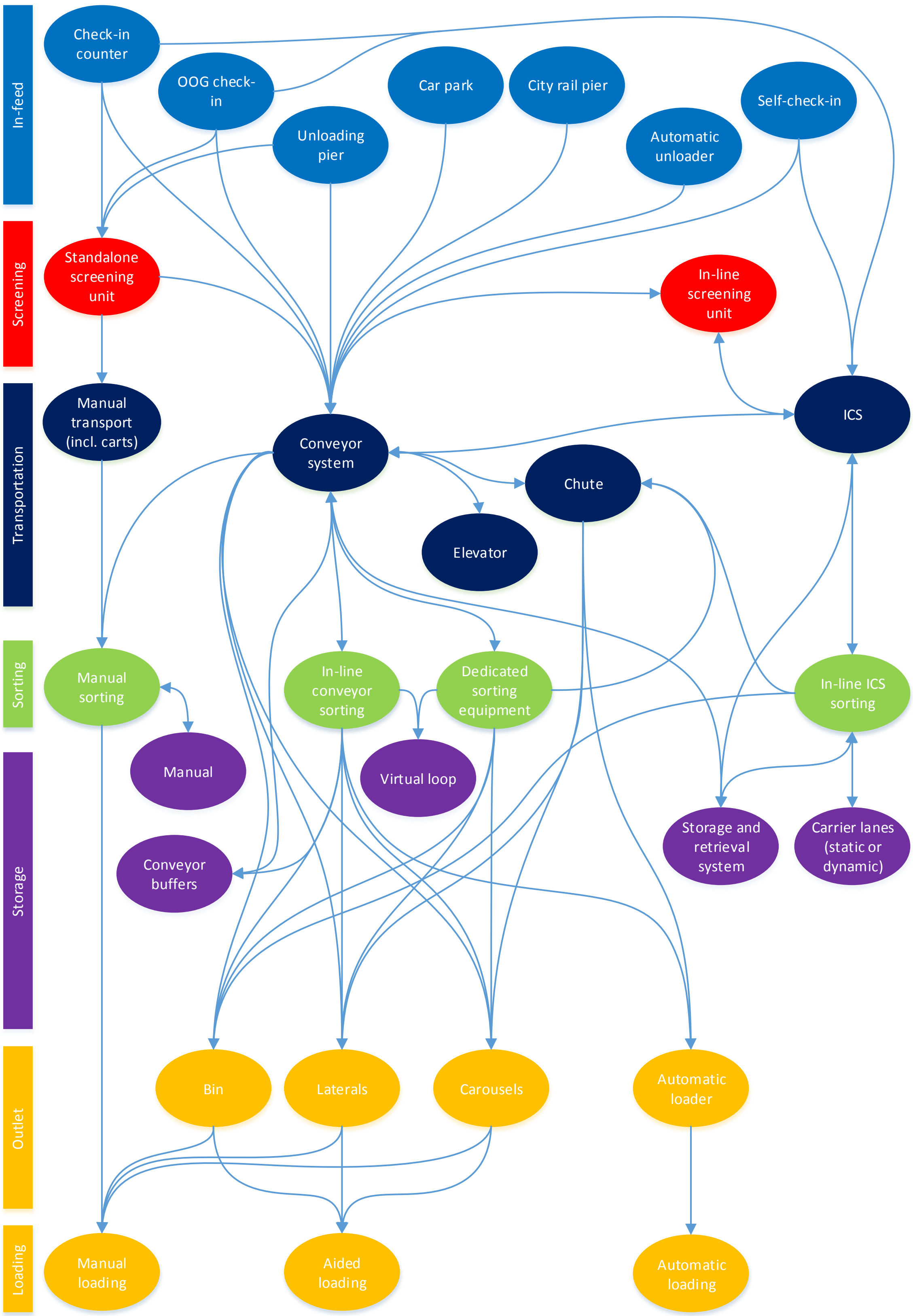


Figure 4.6: A morphological overview for baggage handling systems.

The required throughputs can then be used to determine the amount of sub-systems that are needed to fulfil the demand. These calculations may be captured in design rules, presented in the following equations and are based on the analysed cases. The first design rule is used for determining how many check-ins, transfer inputs and terminating inputs are needed. In this equation and following equations, n represents the amount and f_s represents the safety factor by which required devices are to be increased.

$$n_{devices} \geq (1+f_s)\cdot n_{requiredservers} \tag{4.3}$$

Equation 2.1, the mathematical description for determining how many screening devices are needed, is not used as the described factors are already taken into account by the simulation. As security screening occurs in several rejection levels, the following typical rejection percentages are used in the calculation:

- 30% of items passing level 1 security screening is rejected
- 15% of items passing level 2 security screening is rejected
- 4% of items passing level 3 security screening is rejected
- 2% of items passing level 4 security screening is rejected

These rejection rates may be used to determine how many security screening devices are needed with Equation 4.4. Customs screening does not use such rejection rates and may be calculated with Equation 4.5. In these equations, the unit C represents capacity in bags per hour.

$$n_{devices} \geq \frac{C_{peak}\cdot(1+f_s)\cdot(f_{security,L1}+f_{security,L2}+f_{security,L3}+f_{security,L4})}{C_{device}} \tag{4.4}$$

$$n_{devices} \geq \frac{C_{peak}\cdot(1+f_s)\cdot f_{customs}}{C_{device}} \tag{4.5}$$

Transportation may occur manually with baggage trains, with the use of belt conveyors or the use of ICS tracks. For these equations, loading and unloading times of baggage trains are represented by t in seconds. Distance and average velocity are represented by l and v respectively. This is captured in Equation 4.6. Belt conveyors and ICS tracks may be determined by using Equation 4.7. The required ICS return tracks may be determined by Equation 4.8 and practice experience learns that these tracks typically require only a 25% of the capacity.

$$n_{devices} \geq (1+f_s)\cdot\frac{C_{peak}\cdot(t_{load}+\frac{2\cdot l_{drive}}{v_{average}}+t_{unload})}{3600\cdot C_{train}} \tag{4.6}$$

$$n_{devices} \geq (1+f_s)\cdot\frac{C_{peak}}{C_{device}} \tag{4.7}$$

$$n_{devices,return} \geq n_{devices}\cdot 0,25 \tag{4.8}$$

Sorting in baggage handling systems may occur by merging and diverting baggage. This is done by identifying separated bags and directing them to the correct destination. In the sorting function of the model, as presented in Figure 4.4, items from each flow are redirected to other flows. This may be done with the use of Equation 4.9. It should be noted that the flows to the make-up positions should be included in this calculation and added to the total amount of diverting devices. This equation may also be used to determine the amount of merges. For tilting tray type sorters, it should be noted that the number of merges and diverts may be infinite, but the amount of sorting loops depends on the throughput and positioning of merges and diverts. A typical tilting tray sorter is able to process 6.000 bags per hour. In the case of two successive merge and divert zones, a recirculation of 25% may be expected and cannot be used. This increases the sorting systems capacity to an estimated 9.000 bags per hour. The amount of available trays for tilting tray sorters and ICS systems may be determined with Equation 4.10 and Equation 4.11 respectively. In these equations, P represents the number of occupied positions in a system.

$$n_{devices} \geq (1+f_s)\cdot\frac{C_{peak}}{C_{device}} \tag{4.9}$$

$$n_{trays,tilting} \geq P_{peak,in-system}\cdot(1+f_s) \tag{4.10}$$

$$n_{trays,ICS} \geq (P_{peak,in-system} + P_{peak,storage}) \cdot (1 + f_s) \tag{4.11}$$

For baggage make-up, several devices are available. In the simulation it is determined how many baggage trains are being loaded simultaneously. Each device can have multiple loading positions. To determine how many devices are needed, Equation 4.12 may be used. The amount of reclaims may be determined with the use of Equation 4.13. In the simulation, dwell time of baggage at reclaim is already accounted for. To determine the length of make-up and reclaim devices, Equation 4.14 may be used.

$$n_{devices,make-up} \geq (1 + f_s) \cdot \frac{n_{loads}}{n_{positions}} \tag{4.12}$$

$$n_{devices} \geq (1 + f_s) \cdot \frac{C_{peak}}{C_{device}} \tag{4.13}$$

$$l_{device} \geq \frac{P_{average} \cdot (1 + f_s)}{P_{permeter}} \tag{4.14}$$

With these equations, sub-systems may be determined. These sub-systems are depicted in Figure 4.6. It should be noted that system redundancy should still be taken into account. The redundancy represents the amount of system capacity that should still be available when one of the sub-systems becomes inoperable. Using redundancy factor f_r, having a value between 0 and 1 where 1 represents a fully redundant system, n as the number of required sub-systems and C as the maximum throughput capacity in bags per hour would result in Equation 4.15 by:

$$f_r \leq \frac{\sum_{i=1}^{n-1} C_i}{\sum_{i=1}^{n} C_i} \tag{4.15}$$

This means that when a belt conveyor line is calculated at 6.000 bags per hour and a redundancy of 80% is desired, the available system capacity should still be at least 4.800 bags per hour after one belt conveyor has stopped. When a single belt conveyor line has a capacity of 2.500 bags per hour, a total of three lines should be installed.

After simulating the flow of baggage through the model and determining the required sub-systems, usage of these sub-systems may be calculated by rerunning parts of the saved simulation data. This is done to simulate line balancing, when not all equipment is used during off-peak times. Using $f_{u,p}$ as usage factor for process p, t as simulation time in minutes, t_{end} as final simulation time and n as number of sub-systems allows for calculating $f_{u,p}$ with Equation 4.16. It goes without saying that a longer simulation time would lead to a more accurate representation of the usage factor.

$$f_{u,p} = \frac{\sum_{t=0}^{t_{end}} n_{used,p}}{\sum_{t=0}^{t_{end}} n_{available,p}} \tag{4.16}$$

The capital expenditure is given by the number of purchased sub-systems and may be determined with Equation 4.17, in which c_c represents the total capital costs for the functions, n the number of devices to achieve the designed capacity and c_j the cost of a single sub-system j in process i.

$$c_c = \sum_{i=1}^{p} \sum_{j=1}^{n} c_j \tag{4.17}$$

The usage factor of sub-systems may be taken to determine its energy consumption and operational expenditure. To calculate the energy consumption of the entire system, Equation 4.18 may be used. In this calculation, E is the total energy used per year in kilowatt hour. The calculation adds the results of processes p and P_n represents the power of a sub-system in kilowatt. For each sub-system, these values are known. For sub-systems that have the power per meter, Equation 4.19 is used which incorporates sub-system length l.

$$E = \sum_{i=1}^{p} 365 \cdot 24 \cdot f_{u,p} \cdot n \cdot P_n \tag{4.18}$$

$$E = \sum_{i=1}^{p} 365 \cdot 24 \cdot f_{u,p} \cdot n \cdot P_n \cdot l_n \tag{4.19}$$

To determine the operational expenditure c_o, Equation 4.20 is used. Operational expenditure consists of the following costs per hour: *1*) average cost per operator per hour, *2*) average cost of maintenance per hour, and *3*) costs of energy consumption. This equation uses c_e for the energy costs in €per kWh, $c_{om,n}$ as the maintenance costs per system per hour and $c_{oo,n}$ as the operator costs per operator, per sub-system, per hour.

$$c_o = E \cdot c_e + \sum_{i=1}^{p} 365 \cdot 24 \cdot f_{u,p} \cdot n \cdot (c_{oo,n} + c_{om,n}) \tag{4.20}$$

Finally, the amount of space required is approximated with Equation 4.21 by multiplying the length l and width w for every device in the baggage handling system.

$$A = \sum_{i=1}^{p} \sum_{j=1}^{n} l_j \cdot w_j \tag{4.21}$$

For reclaim or make-up laterals and carousels specifically, the number of required systems is calculated by taking the amount of simultaneously loading or unloading baggage trains. The belt conveyor length of each sub-system is determined by taking the calculated short term storage capacity and dividing that by the average amount of items per meter of belt.

In this section, equations have been described by which simulation and KPI calculation may occur. These equations will be used in the upcoming sections.

4.5.2. SIMULATION SOFTWARE

The proposed model both in visual and theoretical form that has been described in previous sections, has to be translated into a Process Description Language (PDL) for simulation. However, before the PDL is constructed, appropriate simulation software is to be selected. A short survey on available simulation software results in the following software packs:

- Matlab - Simulink [96],
- Python [97] - SimPy [98],
- Delphi [99] - Tomas [100], and
- Excel - Visual Basic for Applications [101, 102].

As the aim of this thesis is to provide a theoretical model for system architects, the decision has been made to implement the model in Microsoft Excel with the use of Visual Basic for Applications (VBA). This is done because of three main reasons. The first being that system architects at SPPAL have access to Microsoft Office Excel and is pre-installed on all laptops within the company. This makes applying the model easy. Secondly, all architects have knowledge of Excel and how it functions, this ensures quick adaptability to the model. Lastly, the high cost for purchasing either Matlab and Delphi is a severe disadvantage to these software programs.

It should be noted however that during programming of the code in Excel VBA, an error was encountered when the a subroutine exceeded the size of 64 kilobytes. This limitation was not mentioned in any of the articles at the time of the decision. A workaround to this problem has been found in that subroutines could be divided into smaller parts. This results in a program that deviates from the ideal pseudo code proposed in the subsequent section. It is not known which influence this change has on the required processing power of the subroutines.

4.5.3. PROCESS DESCRIPTION LANGUAGE

Before programming the created model in code, it is recommended by Veeke [103] to first write the code in a PDL and gradually move towards the complete program. Before creating pseudo code for the model however, it is necessary to determine what type of program is created. Several types of simulation for production systems are described by Kouikoglou and Phillis [104]. Veeke [103] describes three generic formalisms for simulation: *1*) differential equation systems, *2*) discrete time systems, and *3*) discrete event systems. As differential equations describe continuous systems, they are not fit for simulating baggage items in the system. This leaves the comparison between discrete time and discrete event simulation. A similar problem is researched by Özgün and Barlas [105]. Several advantages and disadvantages of both methods are given in the article. As discrete event simulation would require tracking the states of every item in the system, costing calculation power, it is opted to use discrete time simulation to perform the calculations as described in previous sections of this chapter.

The discrete time simulation would allow for varying inter arrival times of baggage based on given distributions. Different flows from several sources may merge and divert within the baggage handling system, to several sinks. Processes and flows have fixed processing times. This is done since internal systems still need to be defined in the conceptual design phase. Breakdowns of systems are not accounted for in the simulation but a redundancy calculation as proposed by IATA negates this by adding several additional sub-systems. A summation of high level program functions would result in: *1*) reading and sorting flight schedule, *2*) generating passengers baggage and arrival times, *3*) performing discrete time simulation of baggage, *4*) calculating process and flow rates, and *5*) writing output files. The discrete time simulation can be detailed further by stating the following internal processes: *1*) baggage train departure process, *2*) baggage train transfer process, *3*) baggage train terminating process, and *4*) function process as described in subsection 4.5.1.

It should be noted that as flight schedules typically have a resolution of 5 minutes, time steps of 1 minute are sufficient for discrete time simulation. Relating to the knowledge described above, the following subroutines in pseudo code can be described:

```
Sub FlightSchedule
   For i = FirstEntry To LastEntry
      Copy data to dynamic array
      Sort entries from first to last departing aircraft
      Add current entry to position
      Store data in new dynamic array
   Next i
End Sub
```

```
Sub GenerateBaggage
   For j = FirstEntry of FlightSchedule To LastEntry of FlightSchedule
      Generate number of passengers on an aircraft
      For each passenger
         Generate passenger arrival time
         Generate baggage items
      Next passenger
   Next j
End Sub
```

```
Sub DiscreteTimeSimulation
   For time = 0 To SimulationEnd
      Move items through the system

      When make-up is ready, call DepartureProcessTrain
   Next time
End Sub

Sub ArrivalProcessTrain
   Wait until Aircraft is unloaded into train
      Drive to Terminating
      Unload into Terminating
      Free position
   End
End Sub

Sub TransferProcessTrain
   Wait until Aircraft is unloaded into train
      Drive to Transfer
      Unload into Transfer
      Free position
   End
End Sub

Sub DepartureProcessTrain
   Wait until carts are full or make-up closes
      Free make-up position
      Drive to Aircraft
      If Aircraft is Empty
         Start Loading
      Else
         Position train on apron
         Wait
      End If
```

```
    End

    Wait until Aircraft is loaded
       Return drive to make-up
    End
End Sub
```

```
Sub SubsystemCalculation
   For each Process and Flow
      Calculate maximum throughput
      Calculate 95% throughput throughput
      Calculate average throughput
   Next Process and Flow
End Sub
```

```
Sub PrintToFile
   For each Array when print is true
      Print each array in a new excel tab
   Next Array
End Sub
```

The proposed pseudo code has been used as a basis for the program code. The code is presented in Appendix K. Each section of the appendix describes a different subroutine, with the first section describing the main routine. A layout is created in Mircosoft Office Excel in which inputs may be given. A manual with instructions to operate the program is presented in Appendix H. These instructions contain figures of the program with indications.

4.5.4. VERIFICATION

After coding Excel VBA in the previously proposed way, it is necessary to verify the functionality of the coded model. This will be tested with five different configurations. It should be noted that this concerns the verification of the simulation model only and not the calculations, as these calculations are common practice. Simple inputs are used to be able to test the model.

Departure system propagation The first verification takes place by having one aircraft in the flight schedule and letting all passengers arrive with one baggage item as soon as the check-in counter opens. This should result in a peak of 180 baggage items in the system, propagating through several functions. As is depicted in Figure 4.7, a total of 180 baggage items may be observed at check-in simultaneously. The aircraft is scheduled to depart at 12:00 PM and Figure 4.12 does indeed show a departure of 180 bags at 720 minutes into the simulation, corresponding to 12:00 PM. As may be seen from the other figures, Figure 4.8, Figure 4.9 and Figure 4.11, the peak propagates through the system with each function after another. No storage was allowed in this verification and Figure 4.10 does indeed show no signs of storage occurring during this system simulation.

Terminating system propagation As has been done for the departure system, the terminating system in which passengers reclaim their baggage must also be verified. This system is the reverse of the departure system and the flow influenced by the unloading speed of carts and ULDs. For this verification, the unloading speed of baggage trains has been set at 15 bags per minute. The percentage of bags that will be screened by customs is taken as 12%. In Figure 4.14 it may be seen that an aircraft containing

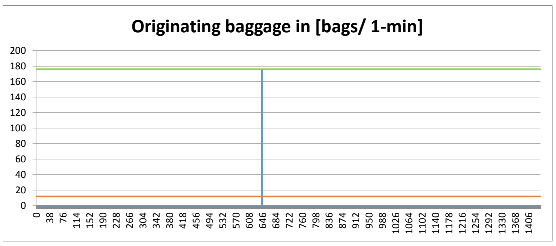


Figure 4.7: Originating baggage for verification of the departure system propagation.

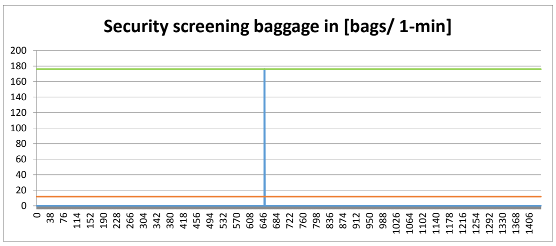


Figure 4.8: Security screening baggage for verification of the departure system propagation.

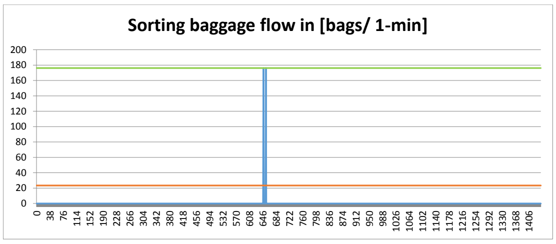


Figure 4.9: Sorting baggage for verification of the departure system propagation.

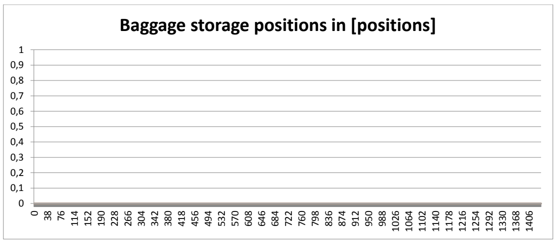


Figure 4.10: Baggage storage for verification of the departure system propagation.

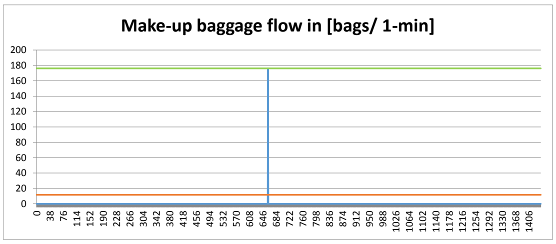


Figure 4.11: Make-up baggage for verification of the departure system propagation.

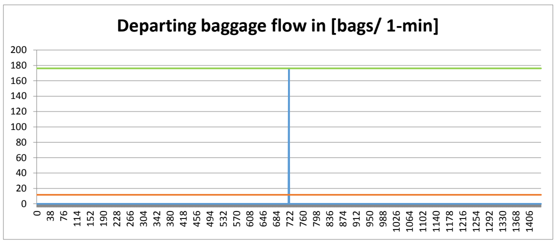


Figure 4.12: Departing baggage for verification of the departure system propagation.

180 items lands at 60 minutes before the departure flight leaves. This is at 11:00 AM, as was intended with a turnaround time of 60 minutes. The baggage is unloaded from the aircraft, transported to an unloading pier and unloaded into the system. It enters the terminating system at a rate of 15 bags per minutes. This is depicted in Figure 4.15. There are no transfers set for this flight and Figure 4.16 does indeed show no transfers occurring during this simulation. The baggage directed to customs screening amounts to 1,8 bags per minute as may be seen in Figure 4.17. Baggage throughput for reclaim, depicted in Figure 4.13, leaves the system after a dwell time of 3 minutes.

System input distributions A third verification of the model concerns different input distributions. These consist of a constant, uniform, triangular and normal distribution and their function is described in the instructions presented in Appendix H. A constant distribution with one aircraft containing 180 baggage items should produce a peak in the system with 180 items. This is seen in Figure 4.18. When selecting a uniform distribution, the probability of occurrence is equal for each moment. Variations may occur however due to the nature of a random number generator. This is seen in Figure 4.19. An average of 1,5 items is received each minute, however moments pass in which no items arrive and a few moments pass in which four items are received. A triangular distribution of baggage is shown in Figure 4.20. Such distribution place the highest occurrence of items towards the mean, decreasing towards the minimum and maximum. The mean in this verification is positioned at 115 minutes before departure, as may be observed in the figure. The last distribution is the normal distribution. A normal distribution should produce a typical bell shape. This is seen in Figure 4.21. The shape is narrow as the variance is small compared to the amount of bags.

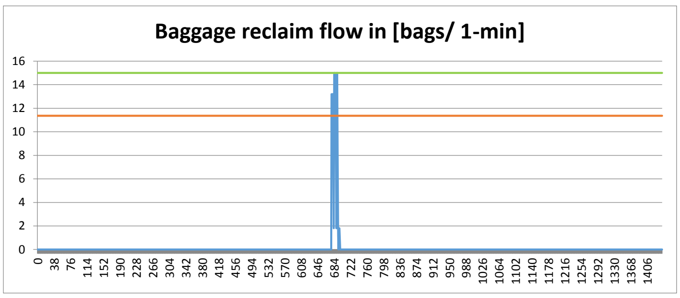


Figure 4.13: Baggage reclaim for verification of the terminating system propagation.

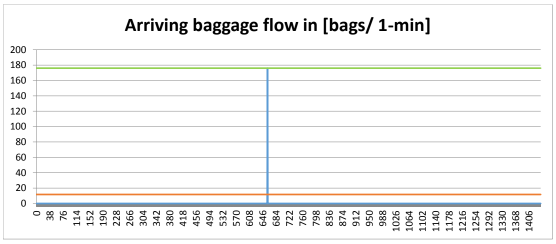


Figure 4.14: Arriving baggage for verification of the terminating system propagation.

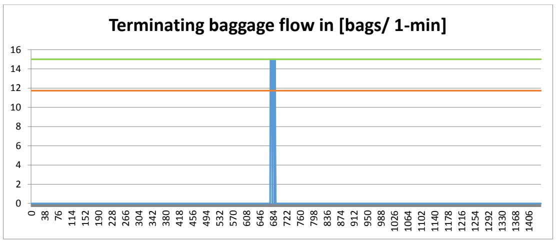


Figure 4.15: Terminating baggage for verification of the terminating system propagation.

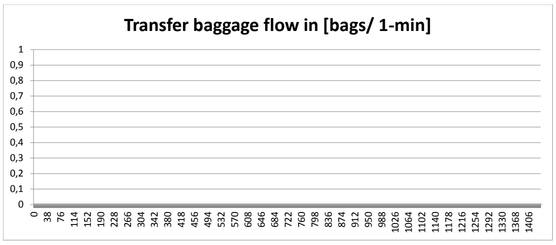


Figure 4.16: Transfer baggage for verification of the terminating system propagation.

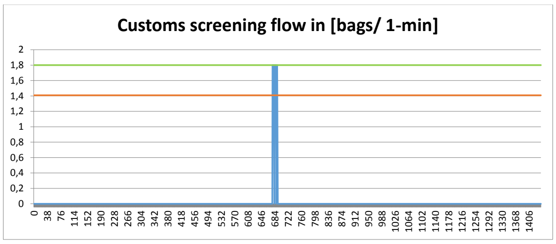


Figure 4.17: Customs screening baggage for verification of the terminating system propagation.

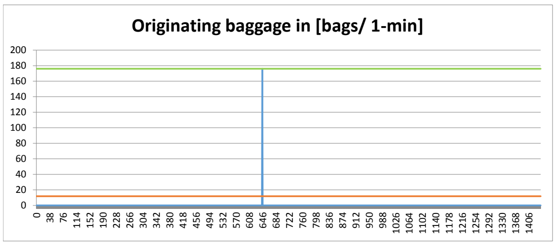


Figure 4.18: Constant distribution for verification of simulation distributions.

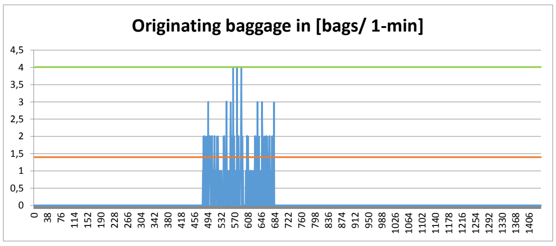


Figure 4.19: Uniform distribution for verification of simulation distributions.

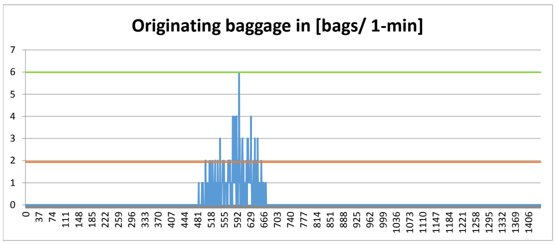


Figure 4.20: Triangular distribution for verification of simulation distributions.

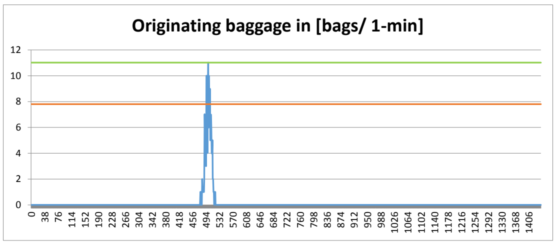


Figure 4.21: Normal distribution for verification of simulation distributions.

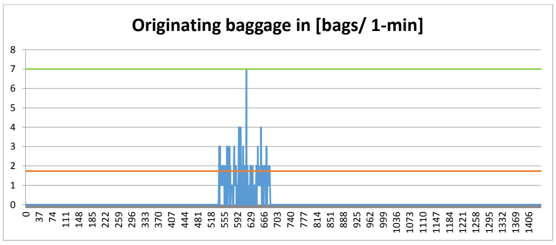


Figure 4.22: Originating baggage for verification of the accumulating functions.

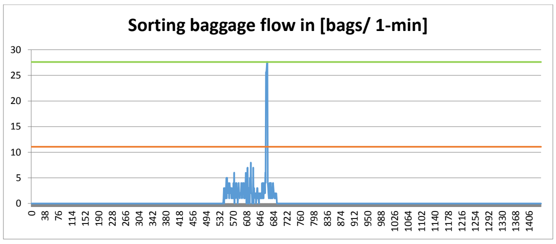


Figure 4.23: Baggage sorting for verification of the accumulating functions.

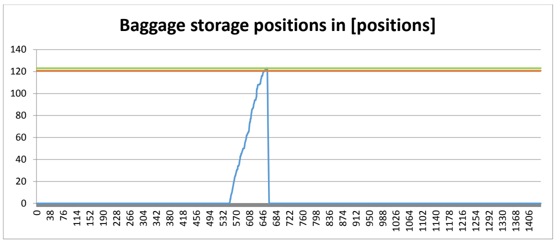


Figure 4.24: Bag storage for verification of the accumulating functions.

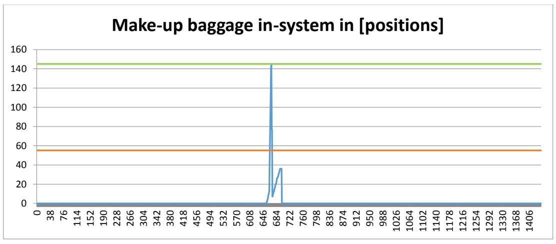


Figure 4.25: Baggage make-up for verification of the accumulating functions.

Storage accumulation Now that propagation and baggage generation have been verified, the accumulation of baggage at the storage and make-up functions will be discussed. Again, an aircraft with 180 passengers having 1 bag each is used. The flight will depart at 12:00 PM. Passengers will arrive according to a uniform distribution between 180 and 36 minutes before departure. Their baggage may be seen in Figure 4.22. These baggage items propagate through the system via security screening to sorting, depicted in Figure 4.23. Bags are sorted to the storage system, the accumulation of this may be seen in Figure 4.24. The make-up carousel will open 60 minutes before flight departure and all bags that are in the system at the moment of opening will be redirected to make-up. This clarifies why a peak demand occurs at sorting 60 mintes before flight departure. The final figure, Figure 4.25, shows these bags accumulating at the function and leave after baggage trains have been filled. The figure also shows that after the peak has been processed and baggage is transported to the apron, additional bags from passengers occur at make-up and are loaded into the last baggage train. This train leaves 20 minutes before flight departure, as was required.

Comprehensive flight schedule In the large system verification, the working of transfers will be verified. Transfer bags are bags that arrive on aircraft that have landed earlier at the airport and intend to leave with another aircraft. These transfers are transported in carts or ULDs to transfer piers where they are loaded into the baggage handling system. A total of 62 aircraft are simulated for one day between 6:00 AM and 9:00 AM. Aircraft start arriving from 6:45 AM. As can be seen in Figure 4.27, transfers start occurring from that moment. As a reference, the arriving passengers are depicted in Figure 4.26. Make-up will open 100 minutes before flight departure, so the storage function is used.

Studying these system verifications shows that the model is able to correctly adjust to both constant, uniform and triangular distribution cases and is able to store baggage when make-up is not yet available. It also produces the propagation of baggage through the system. This part of the model functions as intended based on these verification calculations. The ability to determine and simulate transfer baggage is shown as well as further confirming the transfer function in Figure 4.26 and Figure 4.27. The propagation of baggage from check-in and transfer to make-up is shown as well as baggage from terminating to baggage reclaim. This is demonstrated by tracking a peak flow in the model. After origination and transfer, baggage is processed by security and sorting functions depicted in Figure 4.28 and Figure 4.29 respectively. Baggage is then stored as is shown in Figure 4.30. Figure 4.31 and Figure 4.32 show baggage make-up throughput and accumulation. The flow of terminating baggage may be seen in Figure 4.33 through Figure 4.35, depicting termination, customs screening and baggage reclaim respectively. These figures show that peaks propagate through the model, as should be the case.

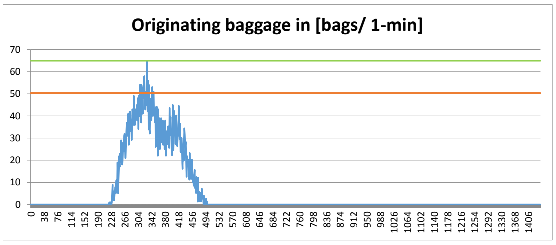


Figure 4.26: Figure of the originating function for a comprehensive flight schedule.

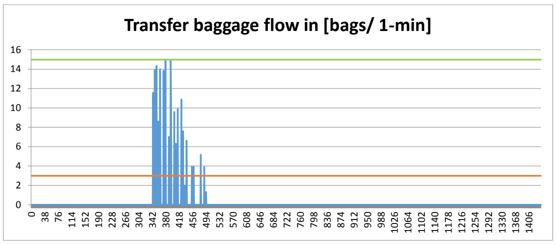


Figure 4.27: Figure of the transfer function for a comprehensive flight schedule.

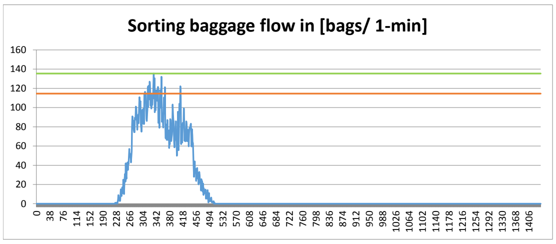


Figure 4.28: Baggage sorting function for a comprehensive flight schedule.

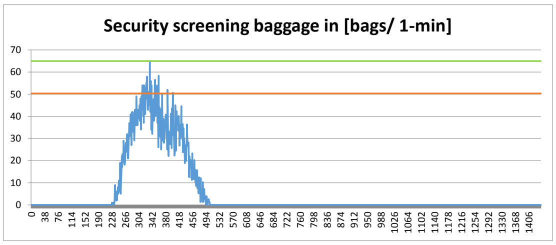


Figure 4.29: Security screening function for a comprehensive flight schedule.

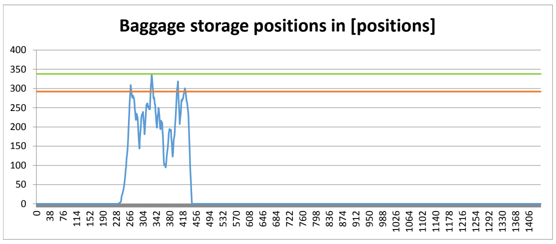


Figure 4.30: Early baggage storage for a comprehensive flight schedule.

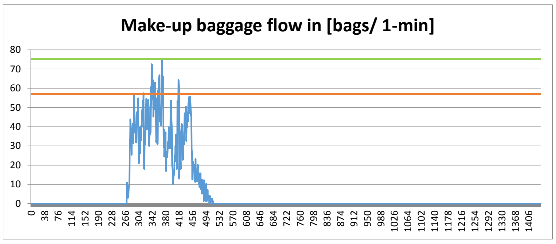


Figure 4.31: Make-up area throughput for a comprehensive flight schedule.

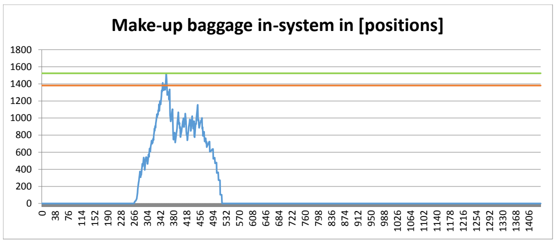


Figure 4.32: Make-up area baggage accumulation.

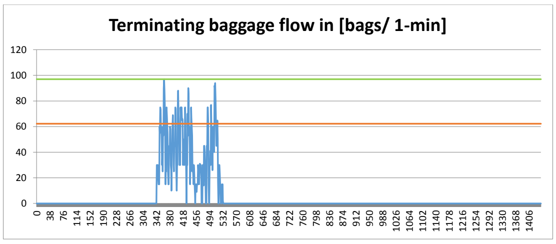


Figure 4.33: Terminating baggage.

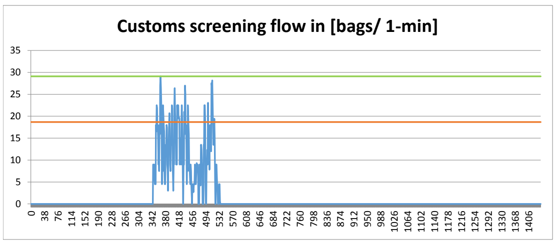


Figure 4.34: Customs screening function for terminating baggage.

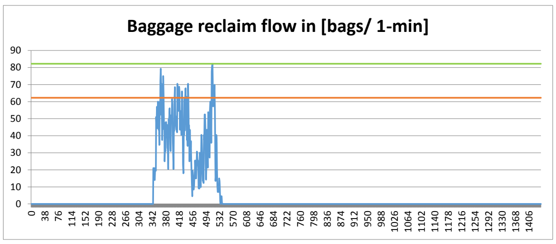


Figure 4.35: Figure of baggage reclaim for verification.

Table 4.2: Assumed client types and preferences determined in collaboration with ten Berge [26] and Lodewijks [27]

Region	Labour costs	Security	Reclaim	Customs
Europe	High	Grouped	Private	Partial
North-American	High	Grouped	Public	Partial
South-American	Low	Grouped	Private	Partial
Asian	Low	Every CI	Private	Partial
Middle-eastern	Low	Grouped	Private	Partial
African	Low	Grouped	Private	Partial
Australian	High	Grouped	Private	All

Based on these manual verification calculations, it could be concluded that the programmed model functions as is proposed in subsection 4.5.1. A note of warning should however be present, as the model includes multiple functionalities, not all different functionalities have been tested with one another exclusively. Although the described mentioned verification suggests otherwise, different combinations of functionalities could in theory conflict and produce unwanted behaviour. It would be safe to say that the model is plausibly verified to work as intended, but it has not been fully confirmed yet.

4.6. EVALUATION AND SELECTION

The final step in the design of freight transport systems method by Pielage [7] is to evaluate concepts based on previously defined criteria. It is described how a multi-criteria analysis like the AHP may be used for this. The application of the AHP method has been described in section 2.2. In section 4.2, criteria have been proposed, based on research by Lemain [43] and ten Berge [39]. These are:

- in-system time
- system throughput
- storage capacities
- required space per sub-system
- energy consumption per sub-system
- capital expenditure
- operational expenditure

The actual ranking of these criteria to importance should be done in accordance with the customer. For now however, several possible typical client types are described in Table 4.2. These client types may govern which criteria are more important than others to base design decisions on. Unique requests are presented in this table, such as North-American airports typically having publicly accessible baggage reclaim areas or Australian airports requiring 100% customs screening. These should be accounted for in baggage handling system designs. Several clients may be less interested in capital costs of a system since more expensive technologies may bring greater prestige to an airport. It should also be noted that this establishment of client types is merely an assumption and other types may exist.

Final evaluation of concepts may thus occur by using the AHP. To determine the ranking of criteria, typical client types may be used as basis. The weight of criteria however, may change depending on a client.

4.7. CHAPTER SUMMARY

In this chapter, the design process steps proposed by Pielage [7] have been used to construct a generic model for baggage handling systems. This design method consists of steps: *1*) problem analysis, *2*) system definition, *3*) system synthesis, *4*) simulation, and *5*) evaluation. Based on six cases, the model is elaborated upon with a general objective, inputs, outputs, boundaries and a synthesis. From the cases, functions and connections have been found. A generic model for baggage handling systems may be defined by the following functions: *1*) originating, *2*) transfer, *3*) terminating, *4*) customs and security screening, *5*) sorting, *6*) storage, *7*) baggage make-up, and *8*) baggage reclaim. Connections between these functions have been presented in Figure 4.4. The proposed model has been expanded upon with equations for calculating KPIs and is implemented in Microsoft Office Excel. The model has been verified. Final evaluation of designs may be based on assessing concept performances with a AHP, as is proposed by Pielage [7].

5

PRACTICAL APPLICATION

The previous two chapters have proposed a design method and generic model for defining baggage handling system concepts. Although the model has been verified, practical applicability in real world cases should still be tested for validation. Two design cases are described and assessed in this chapter, as follows from the research framework in presented section 1.5.

Execution of the first case, [redacted], is described in section 5.1. In this description, the problem analysis, system definition and system synthesis are discussed according to the design method by Pielage [7]. This is continued by comparing the constructed system drawings and the concept drawings that follows from the two design methods. The second case, [redacted], contains less comprehensive information than the first case, so assumptions have to be made. It is described in section 5.2. The second case is also assessed by following the design method by Pielage [7]. To conclude this chapter, a summary containing the answers to several sub-questions from the research framework is given in section 5.3.

5.1. DESIGN CASE C7

In this section, the design case of [redacted] has been used to test the practical applicability of the proposed design method and model. Out of the eight cases, this case was chosen because it contained the most comprehensive information on it. The airport is located about [redacted]. It shares a flight schedule with [redacted], about [redacted]. The flight plan is shared as both airports belong to the same holding company. The geographic position of [redacted] is depicted in Figure 5.1. As may be observed from the figure, the airport consists of a single runway with length of 2500 metres and a single passenger terminal.

Problem definition In the problem definition, the objective, involved parties and requirements are defined. The objective for this case is to design a BHS that is capable of processing 6 million passengers per year by 2020. The available case information did not specify which parties were involved. As the airport is located in [redacted], the typical client set of [redacted] is used to determine which KPIs are preferred. As labour costs are relatively high [redacted], decision makers would value lower operational costs above lower capital costs. It should be noted however that the airport may provide labour in the region and is subsidised. This is not known at the time and thus not accounted for. From Table 4.2 may be seen that security screening occurs centrally, customs screening occurs partially and the reclaim area is not publicly accessible.

The flight schedule that is provided for this airport contains the operation of a 100 flights, with each flight landing at least once a week. A list of approximated aircraft capacities has also been provided. The schedule mainly consists of flights to and from the [redacted]. A few flights however have the [redacted] as their destination, mainly for vacation purposes. The character of flights is therefore set to domestic or short haul. Little to no transferring passengers are expected and the transfer ratio is therefore set to 0. Currently, a total of 3,5 million passengers use this airport each year. Passengers are expected to have between 1,1 and 1,3 bags per person with them. Each of these bags is required to comply to the 100% security screening requirement and oversized baggage should not be accounted for. Due to home check-ins, baggage check-in and drop-off may be taken as 1 minute. This airport consists of one terminal, therefore no incoming and outgoing connections have to be designed.

[redacted]

Figure 5.1: Regional situation of C7.

Table 5.1: A pairwise comparison according to the AHP. This analysis has a consistency ratio of 0,07.

	In-system time	System throughput	Storage capacity	Required space	Energy consumption	Operational expenditure	Capital expenditure
In-system time	1,00	0,20	1,00	3,00	0,20	0,25	0,20
System throughput	5,00	1,00	5,00	3,00	1,00	2,00	0,33
Storage capacity	1,00	0,20	1,00	0,33	0,20	0,20	0,14
Required space	0,33	0,33	3,00	1,00	0,33	0,33	0,25
Energy consumption	5,00	1,00	5,00	3,00	1,00	1,00	0,33
Operational expenditure	4,00	0,50	5,00	3,00	1,00	1,00	0,33
Capital expenditure	5,00	3,00	7,00	4,00	3,00	3,00	1,00

System definition The system definition step elaborates on system boundaries, functions, criteria and basic assumptions for the design. Boundaries and functions for baggage handling systems have already been described in the system definition in section 4.3. To determine which criteria are important, a pairwise comparison of performance indicators is made in Table 5.1. These are based on the assumption of typical clients presented in the previous section. The consistency ratio of 0,07 for the table is well within the proposed boundaries of the AHP and may thus be accepted. It is assumed that this ranking represents the wishes of the clients well. From this pairwise comparison, the following normalised ranking is retrieved with the AHP: *1*) in-system time: 0,059, *2*) system throughput: 0,187, *3*) storage system capacity: 0,035, *4*) required space: 0,058, *5*) energy consumption: 0,167, *6*) operational expenditure: 0,147, and *7*) capital expenditure: 0,348. Summation of these factors leads to a total of 1.

Following IATA guidelines, the baggage handling system of this category B airport should be designed for less than 75% redundancy. A redundancy of 50% is assumed to be sufficient in this case. Other assumptions include an occupation ratio of 1, representing fully occupied aircraft, as well as baggage trains having two carts. Each cart has space for 35 bags and the carts are loaded and unloaded at a rate of 15 bags per minute. Distances, travel times and processing times are based on Gediehn [68]. Baggage passes the following functions in order: originating, security screening, sorting and make-up before it is loaded into carts.

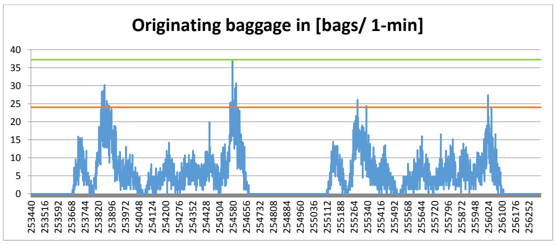


Figure 5.2: Originating process of baggage for C7, based on a provided flight schedule.

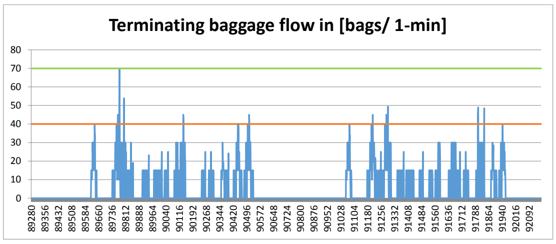


Figure 5.3: Process of terminating baggage on a peak day at C7.

Low speed screening machine (0,34 m/s)					
	Amount	Surface area [m2]	Power [kWh]	CAPEX [€]	OPEX [€/year]
Average	4	108	312206,4	€ 5.800.000,00	€ 31.220,64
95%	4	108	312206,4	€ 5.800.000,00	€ 31.220,64
Maximum	5	135	390258	€ 7.250.000,00	€ 39.025,80

Medium speed screening machine (0,4 m/s)					
	Amount	Surface area [m2]	Power [kWh]	CAPEX [€]	OPEX [€/year]
Average	3	81	234154,8	€ 5.010.000,00	€ 23.415,48
95%	3	81	234154,8	€ 5.010.000,00	€ 23.415,48
Maximum	4	108	312206,4	€ 6.680.000,00	€ 31.220,64

High speed screening machine (0,5 m/s)					
	Amount	Surface area [m2]	Power [kWh]	CAPEX [€]	OPEX [€/year]
Average	2	54	156103,2	€ 3.400.000,00	€ 15.610,32
95%	2	54	156103,2	€ 3.400.000,00	€ 15.610,32
Maximum	3	81	234154,8	€ 5.100.000,00	€ 23.415,48

Figure 5.4: A screen capture of the model output for security screening devices. Manual screening is left out as this was not required in this case.

System synthesis Functions that have been defined in the system definition step may be translated into building blocks from which concepts may be combined. As an IATA category B airport, Figure F.2 may be used to determine the system synthesis. This excludes manual handling of baggage in the system, with the exception of cart and ULD loading, as check-ins are not connected to manually operable machines.

Simulation The simulation step is used to determine expected costs and performance of concepts. This is done for performance indicators proposed in section 4.2. From several design cases, a generic model for baggage handling systems is constructed and is used for this step. VBA code for this macro may be found in Appendix K and instructions on how to use the model are presented in Appendix H. Graphs of the originating and terminating process are shown in Figure 5.2 and Figure 5.3 respectively. An output example for security screening machines is given in Figure 5.4.

Evaluation Concepts are evaluated in this step on the previously proposed criteria of step two. This evaluation should lead to a most suitable concept. An example for such an evaluation is shown in Table J.2. This evaluation is based on Figure 5.4. A low-speed, medium-speed and high-speed security screening machine may be selected. The normalised weight is obtained from Table 5.1 and multiplied by each concepts performance. A maximum score of 5 is obtainable. It follows that the three machines score 3,3; 3,9 and 4,5 respectively. From this evaluation would follow that a high-speed security screening machine ($0,5^m/_s$) is most favourable in the design. It is also shown that a total of 2 devices are required to reach 95% of peak capacity.

A similar exercise has been performed for the other functions of the baggage handling system. Tables may be found in Appendix J. This results in the use of ICS tracks and ICS sorting systems since they are both preferred for transport and sorting. For originating, a total of 26 3-belt check-in counters are favoured. Make-up should occur with chutes as these score slightly higher in the comparison. These chutes also connect to the 12 ICS tilting system to offload bags.

For the terminating baggage process, the same method is used. This results in a simple system where baggage is loaded from carts or ULDs onto a belt conveyor, as the belt conveyor scores higher in the comparison. This pier belt transports baggage to a reclaim carousel that is inclined. A total of 5 pier belts and 5 inclined carousels are required during peak hours. Each of the carousels should have a length of at least 35 metres to store baggage that has not yet been reclaimed by passengers. This is presented in Table 5.4.

The final system concept bill of quantities is presented in Table 5.3. A total of 26 check-in, 2 screening machines, 12 chutes, 12 tilting ICS tracks and 630 meters of ICS track are needed. The 12 ICS tilters include metering conveyors. In total, the system would require $1461m^2$ of surface area, will consume $7TWh$ of power each year, costs €5,2 million to build and €4,6 million to operate yearly. These figures include peripheral equipment such as controllers. The operational expenditure for the reclaim carousels and terminating laterals are estimated at €1,8 million. Dividing the yearly operational expenditure by the 7,2 million processed items results in a cost per bag of €0,63 for the departure system

Table 5.2: Multi-criteria analysis for the screening process.

	Weight	low speed	medium speed	high speed
System time	0,059	3	4	5
System throughput	0,187	5	4	3
Storage capacity	0,035	0	0	0
Required space	0,058	3	4	5
Energy consumption	0,167	3	4	5
Operational expenditure	0,147	3	4	5
Capital expenditure	0,348	3	4	5

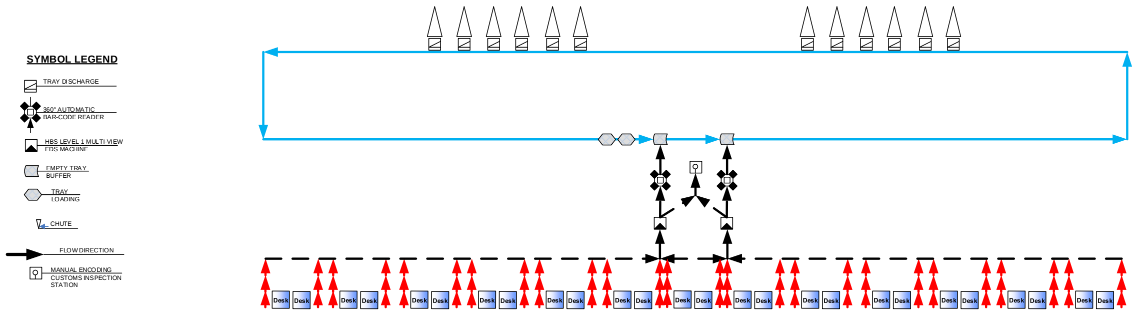


Figure 5.5: A depiction of the proposed concept for C7 including check-ins, screening machines, belt conveyors, ICS tracks and ICS tilters.

and €0,25 for the terminating system.

A concept proposal for the departure system is visualised in Figure 5.5, geometry of the layout is not accounted for. The concept is designed for a peak of approximately 2.400 bags per hour. The ICS tracks are rated for 4.000 bags per hour and would allow for an increase of 1.600 bags per hour. Limiting factors however are the check-in counters, security screening machines and make-up chutes. If further expansion is desired, an extra line of check-ins needs to be built and connected to the ICS with a tray loader. Extra dynamic tilters also need to be added. In the case that transfers become important on this airport, a stand-alone screening unit may be placed. Further expansion of transfer capabilities can be done by adding an additional tray loader to the system.

As a reference, the currently constructed baggage handling system is described. For this developed concept, a similar problem analysis and system definition have been conceived by designers. This concept is depicted in Figure G.8. It consists of 38 two-belt check-in counters, 1 high speed security screening machine, a total of 220 meters of belt conveyor, 1 tilting-tray sorter, 2 sorter inductions, 22 chutes and 1 make-up

Table 5.3: List of components for the departure concept proposal for C7. It should be noted that tray loaders are included in sorting.

Process	**Amount**	**Surface area [m2]**	**Power [kWh]**	**CAPEX [€]**	**OPEX [€/year]**
Originating	26 units	53,0	185.852,2	€405.600,00	€2.296.185,22
Screening	2 units	54,0	156.103,2	€3.400.000,00	€15.610,32
Sorting	14 units	25,4	351.041,7	€300.000,00	€35.104,17
Make-up	12 chutes	383,5	0	€184.095,80	€1.576.800,00
Transport	630 meters ICS	945,0	6.374.214,0	€882.000,00	€637.421,40
Total		1460,9	7.067.211,0	€5.171.695,80	€4.561.121,10

Table 5.4: List of components for the terminating concept proposal of C7.

	Amount	Surface area [m2]	Power [kWh]	CAPEX [€]	OPEX [€/year]
Terminating	5 units	112,5	65.043,0	€236.250,00	€663.504,30
Reclaim	5 units	460,2	11.702.003,5	€310.661,66	€1.170.200,35
Total		572,7	11.767.046,5	€546.911,66	€1.833.704,65

carousel. For baggage reclaim, 2 in-feed belts and 2 carousels are designed. The carousels have a length of 35 metres to temporarily store baggage on. A glance at both baggage handling system concepts indicates that the designs for this case differ in various ways. These are summarised per process below.

Originating The developed system includes 38 two-belt check-in counters as opposed to the 26 three-belt check-in counters that are included in the concept following the method by Pielage [7]. The additional two-belt check-ins would however compensate for the faster processing speed of three-belt check-ins.

Security screening The developed system includes 1 high-speed security screening machine as opposed to the 2 high speed security screening machines that are calculated in the concept. Having one screening device would not suffice during peak hours and also not comply to having at least 50% redundancy in the system.

Transport In the developed system, 220 meters of belt conveyor are used to transport belts between check-in and sorting as well as between sorting and a single make-up carousel. The concept indicates that the use of ICS track is preferred based on the design criteria ranking. The total length of ICS tracks is longer than the belt conveyor, mainly due to the required return loop for carriers. A total of 70 carriers would suffice to handle peak capacity as a return journey only takes 60-100 seconds.

Sorting In the developed system, a tilting tray sorter is used to sort baggage to the make-up chutes and carousels. The ICS system itself sorts with track switches and tilters. In this case only tilters are used since the loop is short.

Make-up Make-up is completed with 22 chutes and 1 baggage carousel in the developed system. The proposed concept requires a total of 12 chutes to comply to the make-up capacity.

Termination and reclaim The baggage termination process is in both versions equal, but the concept calculation shows that 4 carousels are needed whereas the developed system only has 2 carousels.

From this description may be seen that the proposed system and developed system differ in certain ways. However, when the sorter and belts of the developed system are replaced by ICS tracks, the material flow diagrams are similar. The choice for using ICS tracks in the concept proposal is based on listed typical clients and it could occur that real world criteria would have been ranked differently.

For the developed system, the total concept and detailed design time were estimated at [redacted]. The conceptual design is estimated to have lasted [redacted] and detailed design [redacted]. For the concept design, it was required to enter the flight schedule into the model, which took a total of 4 hours. Denoting requirements and defining the system took another 3 additional hours. The system synthesis has already been presented in this report and did not have to be updated. Running the simulation and evaluating system concepts then took 6 hours. Reporting on these concepts, also part of the conceptual design phase, took an additional 4 hours. In total this comes down to 17 hours.

Completing the first case and its discussion, a second case may be studied. This design case concerns [redacted] and will be analysed in the next section.

5.2. DESIGN CASE C8

The second design case on which the design method and model will be tested is the case of [redacted]. This airport is situated in the outskirts of [redacted]. It is one of the main airports serving the city. The airport is depicted in Figure 5.6. [redacted] is part of the west terminal of the airport.

Problem definition In the problem definition, the objective, involved parties and requirements are defined. The objective for this design case is to design a baggage handling system that is able to serve a total

[redacted]

Figure 5.6: Terminal layout of C8.

of 12,5 million passengers at this terminal hall each year. A flight schedule is provided for this design case and consists of a single day containing 233 flights, domestic, short haul and long haul. As [redacted] has a dedicated low budget airport, it is expected that passengers carry between 1,2 and 1,55 bags on them. The transfer ratio for passengers is considered to be about 17% but may deviate. Both transfer and originating traffic have to comply to the required 100% hold baggage screening regulations.

As the airport is located in [redacted], the typical client set of [redacted] airports is used to determine which KPIs are preferred. From Table 4.2 may be seen that security screening occurs centrally, customs screening occurs partially and the reclaim area is not publicly accessible. Further information on this case is not available and assumptions have to be made. Since details of the externally connected system are not known, an external connection is disregarded. It should also be noted that there is no design drawing available for this case. A list of components is available that will be mentioned shortly.

System definition The system definition step elaborates on system boundaries, functions, criteria and basic assumptions for the design. Several boundaries, functions and assumptions have already been presented in section 4.3. The criteria are also defined and may be ranked with the AHP. A possible ranking of performance indicators may be: *1*) in-system time: 0,076, *2*) system throughput: 0,164, *3*) storage system capacity: 0,164, *4*) required space: 0,049, *5*) energy consumption: 0,150, *6*) operational expenditure: 0,109, and *7*) capital expenditure: 0,287. This is achieved by pairwise comparison of criteria.

Following IATA guidelines, a system redundancy of 75% is assumed. Aircraft are also assumed to be fully occupied. Baggage trains for aircraft of a domestic or short haul type are assumed to have a length of two carts. Long haul aircraft types are assumed to have four carts. A total amount of 35 bags are able to fit into a cart. Both loading and unloading of the baggage trains takes 1 minute per 15 bags. As no details were given on the layout similar distances, travel times and processing times are used as in the previous case.

System synthesis Functions that have been defined in the system definition step may be translated into building blocks from which concepts may be combined. The peak hour capacity of the baggage handling system is sufficiently high to be in IATA category C and Figure F.3 may thus be used for the system synthesis.

Simulation The simulation step is used to determine expected costs and performance of concepts, as described by Pielage [7]. A generic model is constructed and used to simulate the flight schedule that was provided for this case. Simulation of the model shows required capacities for several functions in the model. Four of these, originating, terminating, transfer and storage, are depicted in Figure 5.7 to Figure 5.10 respectively. Instruction on how to use the simulation model are presented in Appendix H.

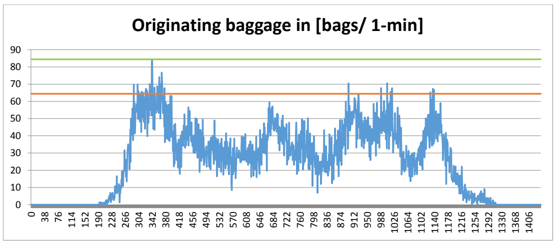


Figure 5.7: Originating flow of baggage for C8, based on a provided flight schedule.

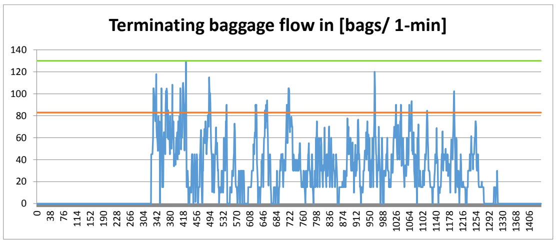


Figure 5.8: Terminating baggage for C8.

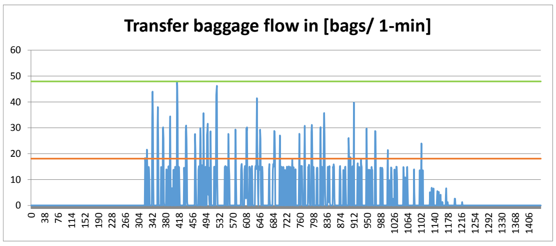


Figure 5.9: Baggage transfer process for C8, based on a provided flight schedule.

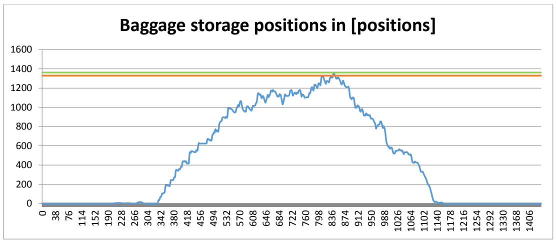


Figure 5.10: Flow accumulation in storage at C8.

Evaluation Concepts are evaluated in this step on the previously proposed criteria of step two. This evaluation should lead to a most suitable concept. Evaluation of design criteria happens in a similar manner as described in design case [redacted]. The data to this is presented in Appendix J. It yields that the following devices may be designed: *1*) 80 two-stage baggage drop-offs, *2*) 4 transfer in-feed laterals, *3*) 7 high speed security screening machines, *4*) 2000 meters of ICS track, *5*) 12 tray loaders, *6*) 10 tray tilters, *7*) 12 track switches, *8*) 12 track merges, *9*) 1500 storage rack positions, and *10*) 10 make-up carousels.

The terminating and reclaim system should consist of 16 reclaim carousels with an average length of 35 metres. To unload baggage trains, a similar amount of unloading piers is required. The reclaim carousels and unloading piers are connected by a short belt. Customs screening can occur after passengers have reclaimed their baggage. This may be seen in Table 5.6.

Table 5.6 shows the total amount of devices, required surface area, power, capital expenditure and operational expenditure. Dividing the operational expenditure of €$9,1 million$ by the $16,9 million$ processed bags gives a total cost of €$0,54$ per bag. For the terminating baggage, this results in a cost of €$0,43$ per bag. A concept proposal has been depicted in Figure 5.11. It should be noted that this does not include geometry of the layout.

Although a list of components was available for this case, information on this case however was incomplete since design drawings were not available. The list of components consists of: *1*) 120 two-belt check-ins arranged in 6 islands, *2*) 3 transfer in-feed belt conveyors, *3*) 1 conveyor line from an external system, *4*) 6 belt

Table 5.5: List of components for the concept proposal departure system of C8. Merges, switches and unloaders are included in the sorting system.

	Amount	Surface area [m2]	Power [kWh]	CAPEX [€]	OPEX [€/year]
Originating	80 units	212,2	418.167,4	€2.464.800,00	€212.636,74
Transfer	4 units	90,0	5.2034,4	€189.000,00	€662.203,44
Screening	7 units	189,0	546.361,2	€11.900.000,00	€54.636,12
Transport	2000 meters	3.000,0	418.167,4	€2.464.800,00	€212.636,74
Sorting	46 units	292,1	4.036.979,2	€3.068.200,00	€403.697,92
Storage	1495 positions	1.495,0	471.463,2	€493.350,00	€47.146,32
Make-up	10 units	1.866,8	47.466.245,1	€1.260.121,19	€7.538.874,51
Total		7.145,1	53.409.417,9	€21.840.271,19	€9.131.831,79

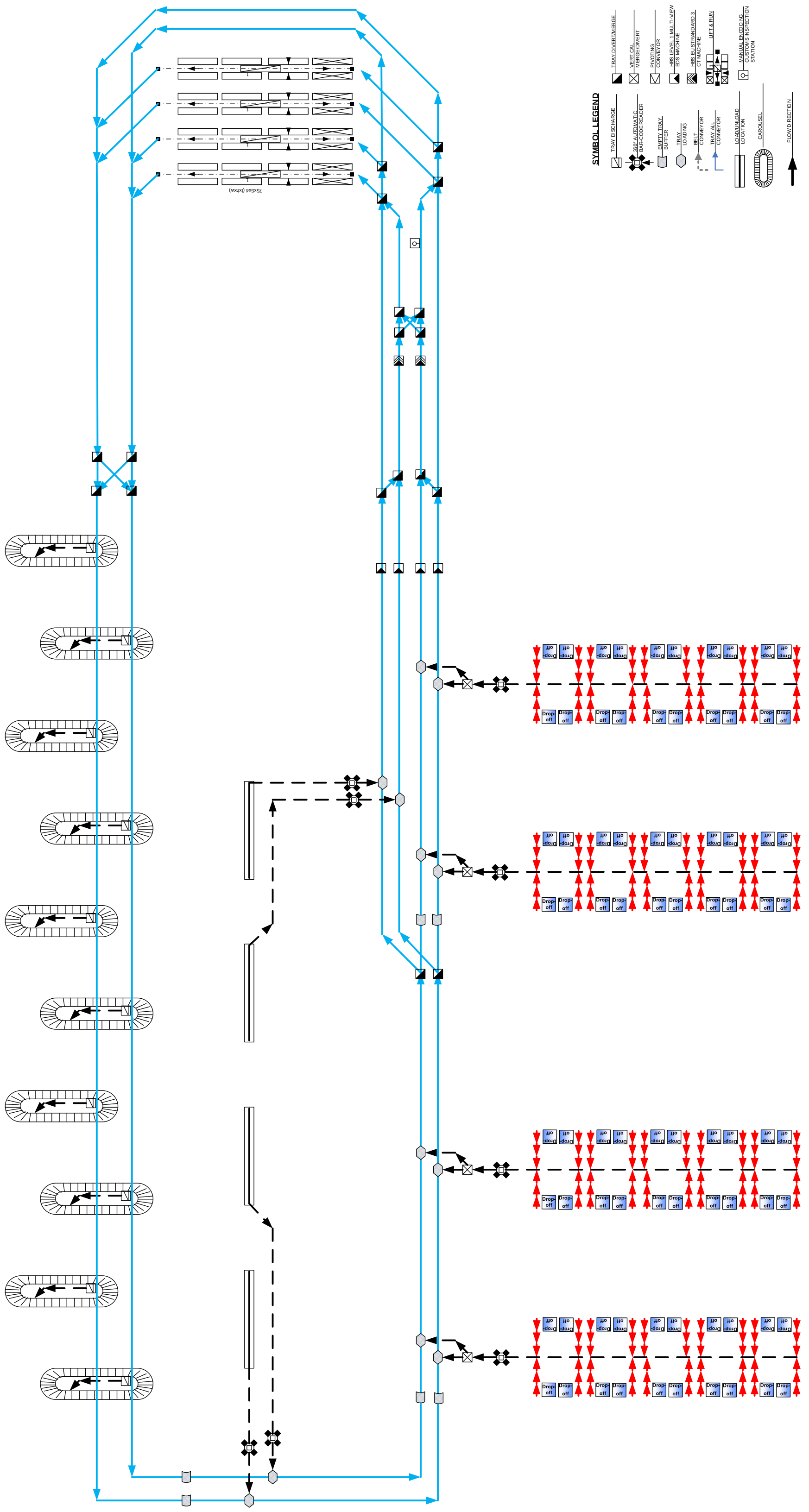


Figure 5.11: A depiction of the proposed concept for C8 including check-ins, screening machines, belt conveyors, ICS tracks, switches and ICS tilters.

Table 5.6: List of components for the concept proposal terminating system of C8.

	Amount	Surface area [m2]	Power [kWh]	CAPEX [€]	OPEX [€/year]
Terminating	16	360	208.137,6	€756.000,00	€2.648.813,76
Reclaim	16	1.787,2	45.440.259,9	€1.206.335,88	€4.544.025,99
Total		2.147,2	45.648.397,5	€1.962.335,88	€7.192.839,753

conveyor lines from check-in to sorting, *5)* 2 tilting tray sorters, *6)* 6 security screening devices, *7)* 6 conveyor lines between sorting and screening, *8)* 6 conveyor lines between screening and sorting, and *9)* 10 make-up carousels. Several parts are not specified in the list of components such as the existence of early baggage storage and the details to the external system.

A comparison between the realised system and proposed concept for [redacted] shows less similarities than for design case [redacted]. The amount of transfer inputs, make-up devices and terminating system calculated by the model and present in the developed system correspond but besides these similarities, no others remain. A short description of these differences:

Originating The concept proposes 80 self check-in and baggage drop-off counters, whereas the developed system has a total of 120 manned 2-belt check-ins. The throughput of both the realised and concept is similar. This difference may be explained by the proposed criteria ranking for this design case, as information was scarce.

Security screening The realised system has a total of 6 security screening devices. The concept on the other hand proposes a total of 7 security screening devices.

Transport In the developed system, transportation occurs with belt conveyors. Following the design method by Pielage [7] shows favour to ICS systems.

Sorting Sorting in the design concept is done with the same medium as by which transport occurs, namely an ICS. The developed system is built with two tilting tray sorters.

The developed system was estimated to have taken a total [redacted] in conceptual design time. Applying the design method by Pielage [7] has taken 2 hours to denote the problem analysis, system definition and system synthesis. Furthermore, the flight schedule was already in the right format. It therefore only cost 30 minutes to enter in the model. Running the simulation, evaluating the results and reporting on this required an additional 9 hours. Finally, a material flow diagram is made, resulting in a total conceptual design time of 13 hours and 30 minutes. This is significantly less than the [redacted] for the developed concept. The available information for this case was less comprehensive and could therefore be a cause of the shortened design time compared to [redacted].

5.3. CHAPTER SUMMARY

In this chapter, the design method by Pielage [7] has been applied on design case [redacted] and design case [redacted]. The first case was selected as it contained data on belt conveyor lengths, amount of check-ins and make-up systems, whereas the other available cases did not. The second design case was selected as it represented a larger system and a possibility of assessing how the model copes with such information.

For each design case, the real world developed system and conceptual design by following Pielage [7] have been presented. These are depicted in Figure 5.5 and Figure 5.11. The currently realised design concepts have also been presented. The proposed concepts and developed systems differ in several ways for both cases. However when the tilting tray sorter of case [redacted] is substituted by an ICS system, the concepts are almost identical. For design case [redacted], a dissimilar result has been obtained. A cause may be found in assumptions due to the absence of information. Another reason may be the existence of different design preferences for the DMU. It is however notable that although there was little information, a concept could be proposed based on several assumptions and the generic model.

The total conceptual design time for the developed system of case [redacted] amounts to [redacted], whereas designing the concept with the proposed design method took 17 hours. Although the design time in case [redacted] was 13 hours and 30 minutes compared to the estimated design time of the developed system of [redacted], the resulting designs showed much differences. Further tests with the method and model on baggage handling system design should occur before the model can be adopted in real world scenarios.

6

Concluding remarks

After selecting a conceptual design method, using it to construct a generic model for baggage handling systems and assessing two design cases, this chapter concludes on the findings of this research. A research framework has been proposed in section 1.5 as a guideline to this research. The final part of the research framework concerns the conclusion to the main research question. The conclusion is presented in section 6.1. Building up to the main research question are the central research questions that structure each chapter. These are synthesized in section 6.2. Finally, several recommendations for further research are made in section 6.3.

6.1. Main research question

At the start of this thesis, the motivation and main research question have been stated. This was expanded upon by the research framework and central research questions that have been answered sequentially in the subsequent chapters. The main research question is: *How may KPIs of baggage handling system concept designs be determined based on flight schedule demand?*

The proposed method by which KPIs may be determined is the design method by Pielage [7]. This design method has been selected by determining which method was preferred by baggage handling experts with the AHP, since literature on concept design for baggage handling systems is scarce. The design method proposes simulation by which key performance indicators may be determined. To assist in the calculation, a generic baggage handling system model has been constructed, based on six individual cases.

By following the five design steps: *1*) defining goals and stakeholders in the problem analysis step, *2*) defining system boundaries and criteria in the system definition step, *3*) defining system concepts by combining function solutions in the system synthesis step, *4*) simulating concepts, and *5*) evaluating the simulated concepts on performance and costs, practice knowledge on cases has been obtained. This knowledge has been used to create the generic model depicted in Figure 6.1. This model was implemented in Visual Basic for Applications and its functionality verified. To validate the model, two cases have been performed with the design method and model as a means of validation. It should be noted however that the model is not fully validated with two cases and further validation is required, this will be discussed in the recommendations.

The applicability of the design method by Pielage [7] and the generic model for baggage handling systems has been shown in two design cases of different size. The results on these two cases show that the design method and model function as intended, relating the concepts to the realised systems. Design time for both designs was reduced when using the proposed design method. The significance of this result can however not be tested, as more results are needed. The method and model may be expandable to the postal and parcel industry, since functions and sub-systems are similar. It is however advisable to first focus on refining the model for baggage handling systems, as this is the intention of the model.

6.2. Central research questions

It was possible to answer the main research question by following the research framework presented in section 1.5. From the research framework, a total of four central research questions could be derived. Each central research questions represents a chapter in this thesis and has several sub-questions. These sub-questions have been answered in the appropriate chapters. The following four central research questions are:

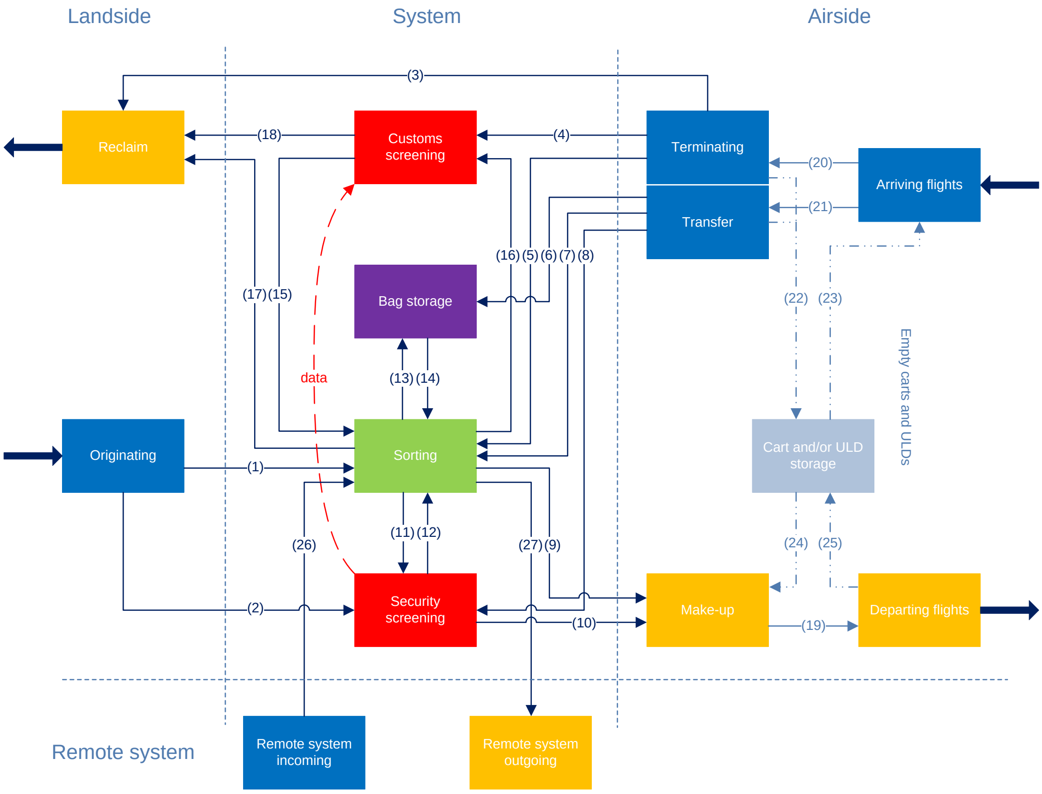


Figure 6.1: A schematic overview of a generic model for baggage handling system functions and connections. The model may be divided into landside, airside and system. It should also be noted that the sorting system may connect to a remote system that is already present or will be built in a future expansion.

What knowledge on baggage handling design may be learnt from literature? There is a surprising scarcity of literature on baggage handling systems. The available literature focusses on providing equipment recommendations and descriptions for security screening systems. None of these articles however describes a design method for baggage handling systems. A general search for design methods has been performed and several methods were found. These methods have been evaluated in chapter 3 and are linked to the second central research question.

Which design method is preferred when applying the AHP? SPPAL currently uses the design method that is described in section 3.1. A general indication to what steps should be taken is given but no clear design method is presented. To compare the found design methods, a set of criteria consisting of: *1*) aggregation layers, *2*) communication, *3*) flexibility, *4*) integration, *5*) iteration, *6*) life-cycle, *7*) parallel development, *8*) performance indicators, *9*) rapidity, *10*) simplicity, and *11*) development scenarios has been provided. From application of the AHP follows that the design of freight transport systems method conceived by Pielage [7] is preferred among experts. In this method, five design steps are proposed for design: *1*) problem analysis, *2*) system definition, *3*) system synthesis, *4*) simulation, and *5*) evaluation.

How may a generic model for baggage handling systems be defined? The design problem for baggage handling systems may be reduced by finding common traits in design cases. Several design cases are therefore analysed in chapter 4. From this analysis a system definition and system synthesis follow. These cases also allow for the construction of a generic model for baggage handling system simulation during concept development. Based on six practise cases, the model is elaborated upon with a general objective, inputs, outputs, boundaries and a synthesis. From the cases, functions and connections have been found. A generic model for baggage handling systems may be defined by the following functions: *1*) originating, *2*) transfer, *3*) terminating, *4*) customs and security screening, *5*) sorting, *6*) storage, *7*) baggage make-up, and *8*) baggage reclaim. Connections between these functions have been presented in Figure 6.1.

What may be learnt when the design method and model are applied in case studies? The design method by Pielage [7] has been applied on design case [redacted] and design case [redacted]. The proposed concepts and developed systems differ in several ways for both cases. However when the tilting tray sorter of case [redacted] is substituted by an ICS system, the concepts are almost identical. For design case [redacted], a dissimilar result has been obtained. It is however notable that although there was little information, a concept could be proposed based on several assumptions and the generic model.

The total conceptual design time for the developed system of case [redacted] amounts to [redacted], whereas designing the concept with the proposed design method took 17 hours. The design time needed in case [redacted] was 13 hours and 30 minutes compared to the estimated design time of the developed system of [redacted], the resulting designs however showed much differences.

6.3. RECOMMENDATIONS

During research and the writing of this thesis, assumptions have been made. They may have been made to simplify the objects of research or because information was scarce. The latter mostly occurred during the study of two design cases in chapter 5. From these assumptions, recommendations for further research may be established. These recommendations are:

Validation It is important to state that validation of a complex model cannot occur by applying it in two cases. Although proper methods have been found, described and used, a set of two cases only allows for preliminary conclusions. Whether the project should be proceeded or halted may be based on these preliminary conclusions. As the results of the two cases look promising, it is advisable to continue this research with validation.

Design rules The current model is based on design rules captured from six studied cases. It is not said whether there are more design rules. Another way of improving this model may be by critically assessing the design rules and adding intelligence to the model.

Limitations As design rules may be added, the current model also has its limitations. A small airport case, category A, has not been studied as none were available. This limits the applicability of the model to category B and C airports. Another limitation is the way sub-systems are calculated. Currently it is all or nothing, meaning that a concept contains either all make-up carousels or all make-up laterals. A combination is not yet possible. Future research could investigate ways to add design rules that allow for different optimisations. In the case of make-up for example, the amount of selected carousels and laterals may be optimised for a certain amount of available space.

Excel VBA Although several sources have written on the application of simulation in Excel VBA, practice experience learns that the program is limited by a subroutine size of 64kb. This is sufficient for most code but when data is to be printed in Excel cells, numerous lines of code are required and will sometimes span several subroutines as the maximum size does not allow for only one subroutine. A workaround has been achieved in for the macro, but the impact on calculation time is not known. Future research may continue with this model but it should be investigated whether application in a different program is more beneficial.

Geometric design In literature, the design of object processing systems method has been found that allows for geometric design of paths in buildings. This method may be applicable during detailed design, when a final concept has been chosen and building limitations become available. As sub-systems have already been selected in detailed design, several path designs may be assessed. SPPAL may want to look into this.

Visualisation The model currently functions by inputting lengths and processing times into tables. This is a valid way of providing information in programs but as the aesthetics of programs becomes more important, a different way of providing input may be conceived. A possibility for program designers is to change the model's inputs to a layout where process blocks may be dragged and dropped, lines may be connected and a simulation may be done. The basic calculations and simulation of processes may remain as is.

Bibliography

[1] B. S. ten Berge, *Airport of the future - baggage handling innovations and new trends,* (2013), a guest lecture at the Delft University of Technology on Thursday 5th of December, 2013.

[2] All the Rage, *It ain't heavy, it's part of the job,* First accessed the 6th of July, 2015.

[3] Royal Haskoning DHV, *World class baggage network for london heathrow,* First accessed the 6th of July, 2015.

[4] Siemens AG, *Baggage handling at beijing airport,* First accessed the 6th of July, 2015.

[5] B. S. ten Berge, Personal correspondence (2015), a personal conversation on Monday 24th of August, 2015.

[6] Deerns, Personal correspondence (2015), a personal conversation with Anke Matijssen on Wednesday 23rd of September, 2015.

[7] B.-J. A. Pielage, *Conceptual Design of Automated Freight Transport Systems*, Ph.D. thesis, Technische Universiteit Delft, Delft, The Netherlands (2005).

[8] S. Yilmaz and S. R. Daly, *Feedback in concept development: Comparing design disciplines,* Design Studies , 1 (2016).

[9] P. J. M. Verschuren and J. A. C. M. Doorewaard, *Designing a research project*, 2nd ed. (Eleven International Publishing, 2010).

[10] J. Skorupski and P. Uchronńki, *Fuzzy inference system for the efficiency assessment of hold baggage security control at the airport,* Safety Science **79**, 314 (2015).

[11] Q. Feng, H. Sahin, and K. C. Kapur, *Designing airport checked-baggage-screening strategies considering system capability and reliability,* Reliability engineering and system safety **94**, 618 (2008).

[12] Verein Deutscher Ingenieure, *Methodik zum entwickeln und konstruieren technischer systeme und produkte,* (1993), vDI 2221.

[13] Verein Deutscher Ingenieure, *Konstruktionsmethodiek: Konzipieren technischer produkte,* (1977), vDI 2222.

[14] J. Eekels and N. F. M. Roozenburg, *Productntwerpen, structuur en methoden*, 2nd ed. (Boom Lemma Uitgevers, 2007).

[15] Star Alliance, *Update: Suvarnabhumi closed by airports of thailand,* First accessed the 7th of August, 2015.

[16] Wikipedia, *File:udrop bagdrop kiosk.jpg,* First accessed the 7th of August, 2015.

[17] Lufthansa, *Excess and special baggage,* First accessed the 7th of August, 2015.

[18] 2DEAL BAG EXPERT, *Airport luggage carts 2,* First accessed the 7th of August, 2015.

[19] Safeglide, *Safeglide material handling chutes,* First accessed the 7th of August, 2015.

[20] Vanderlande Industries, *Manual loading with carousel and lateral conveyor-based solutions,* First accessed the 9th of August, 2015.

[21] Chris Lo, *Airport baggage systems go high-tech: handling with care,* First accessed the 9th of August, 2015.

[22] Alliance Airport Advertising, *Airport | sal lake city, utah,* First accessed the 9th of August, 2015.

[23] Passenger Terminal Today.com, *Crisplant wins baggage handling contracts in norway and canada,* First accessed the 9th of August, 2015.

[24] Copybook, *Oslo airport awards contract for new terminal 2 bhs project to vanderlande industries,* First accessed the 9th of August, 2015.

[25] A. L. W. Bradley, *The Independent Airport Planning Manual*, 1st ed. (Woodhead Publishing, 2010).

[26] B. S. ten Berge, Personal correspondence (2016), a personal conversation on Thursday 4th of Februari, 2016.

[27] G. Lodewijks, Personal correspondence (2016), a personal conversation on Thursday 5th of Februari, 2016.

[28] Siemens Postal, Parcel & Airport Logistics GmbH, *About siemens postal, parcel & airport logistics,* First accessed the 3th of June, 2015.

[29] United Nations - Department of Economic and Social Affairs, *World urbanization prospects,* First accessed the 11th of June, 2015.

[30] Boeing, *Long-term market - demand by size,* First accessed the 11th of June, 2015.

[31] Airbus, *Flying on demand - global market forecast,* First accessed the 11th of June, 2015.

[32] Markets & Markets, Online publication (2015), first accessed on the 15th of September, 2015.

[33] K. Vickers and R. W. Chinn, *Passenger terminal baggage handling systems,* The Institution of Electrical Engineers , 1 (1998).

[34] International Air Transport Association, First accessed the 21st of July, 2015.

[35] K. Leone and R. Liu, *The key design parameters of checked baggage security screening systems in airports,* Journal of Air Transport Management **11**, 69 (2005).

[36] S. D. Barrett, *How do the demands for airport services differ between full-service carriers and low-cost carriers?* Journal of Air Transport Management **10**, 33 (2004).

[37] M. Franke, *Competition between network carriers and low-cost carriers - retreat battle or breakthrough to a new level of efficiency?* Journal of Air Transport Management **10**, 15 (2004).

[38] J. G. de Wit and J. Zuidberg, *The growth limits of the low cost carrier model,* Journal of Air Transport Management **21**, 17 (2012).

[39] B. S. ten Berge, Personal correspondence (2015), a personal conversation on Monday 20th of July, 2015.

[40] International Air Transport Association, *Inbag general presentation,* Presentation slides (2013), first accessed on the 13th of June, 2015.

[41] Société Internationale de Télécommunications Aéronautiques, *The baggage report 2015 - air transport industry insights,* (2015).

[42] M. F. Ashby, H. Shercliff, and D. Cebon, *Materials: engineering, science, processing and design*, 3rd ed. (Butterworth-Heinemanng, 2013).

[43] T. R. Lemain, *Get the bag on the right track! 'Development of a new methodology for baggage handling system design'*, Ph.D. thesis, Technische Universiteit Delft, Delft, The Netherlands (2002).

[44] R. Grigoraş and C. Hoede, *Design of a baggage handling system,* Memorandum 1835, Department of Applied Mathematics, University of Twente (2007).

[45] International Air Transport Association, *Baggage identification chart,* Online or printed chart (2007), first accessed on the 18th of September, 2015.

[46] G. Lodewijks, *Advanced design of baggage handling systems - lecture 1, introduction,* (2014), Delft University of Technology, Delft, The Netherlands.

[47] H. P. M. Veeke, J. A. Ottjes, and G. Lodewijks, *The Delft Systems Approach*, 1st ed. (Springer London, 2008).

[48] I. van den Berg, *Help, bagage kwijt! wie gaat er dokken?* Metro , 5 (2015).

[49] R. Dittrich, A. Wreford, and D. Moran, *A survey of decision-making approaches for climate change adaptation: Are robust methods the way forward?* Ecological Economics **122**, 79 (2016).

[50] Z. Xu and N. Zhao, *Information fusion for intuitionistic fuzzy decision making: An overview,* Information Fusion **28**, 10 (2016).

[51] R. W. Saaty, *The analytic hierarchy process - what it is and how it is used,* Mathematical Modelling **9**, 161 (1987).

[52] R. W. Saaty and J. M. Katz, *How to make a decision: The analytic hierarchy process,* European Journal of Operational Research **48**, 9 (1990).

[53] TU Delft Library, *TU Delft Library,* First accessed the 24th of June, 2015.

[54] Science Direct, *Science Direct,* First accessed the 24th of June, 2015.

[55] SpringerLink, *SpringerLink,* First accessed the 24th of June, 2015.

[56] Google, *Google Scholar,* First accessed the 24th of June, 2015.

[57] K. Wells and D. Bradley, *A review of x-ray explosives detection techniques for checked baggage,* Applied Radiation and Isotopes **70**, 1729 (2012).

[58] Y. Zeinaly, B. de Schutter, and H. Hellendoorn, *An integrated model predictive scheme for baggage handling systems: Routing, line balancing, and empty cart management,* IEEE Transactions on control systems technology **23**, 1536 (2015).

[59] T. Klotz, J. Schönherr, N. Seßler, B. Straube, and K. Turek, *Automated formal verification of routing in material handling systems,* IEEE Transactions on automation science and engineering **10**, 900 (2013).

[60] T. Klotz, J. Schönherr, N. Seßler, B. Straube, and K. Turek, *Compositional verification of material handling systems,* Proceedings of the 17th IEEE International Conference for Emerging Technologies in Factory Automation **10**, 1 (2012).

[61] A. Abdelghany, K. Abdelghany, and R. Narasimhan, *Scheduling baggage-handling facilities in congested airports,* Journal of Air Transport Management **12**, 76 (2006).

[62] A. N. Tarău, B. de Schutter, and J. Hellendoorn, *Predictive route control for automated baggage handling systems using mixed-integer linear programming,* Transportation Research Part C **19**, 424 (2011).

[63] A. N. Tarău, B. de Schutter, and J. Hellendoorn, *Travel time control of destination coded vehicles in baggage handling systems,* Proceedings of the 17th IEEE International Conference on Control Applications , 293 (2008).

[64] A. N. Tarău, B. de Schutter, and J. Hellendoorn, *Hierarchical route choice control for baggage handling systems,* Proceedings of the 12th IEEE International Conference on Intelligent Transport Systems , 679 (2009).

[65] A. N. Tarău, B. de Schutter, and J. Hellendoorn, *Model-based control for route choice in automated baggage handling systems,* IEEE Transactions on Systems, Man and Cybernetics - Part C: Applications and Reviews **40**, 341 (2010).

[66] M. Johnstone, D. Creighton, and S. Nahavandi, *Simulation-based baggage handling system merge analysis,* Simulation modelling practise and theory **53**, 45 (2015).

[67] G. Lodewijks, *Energy efficient use of belt conveyors in baggage handling systems,* Networking, Sensing and Control, 2012 9th IEEE International Conference on , 97 (2012).

[68] A. Gediehn, *Der einfluss der mct (minimum connecting time) auf baggageanlagen,* , 1 (2002).

[69] J. C. Rijsenbrij and J. A. Ottjes, *New developments in airport baggage handling systems,* Transportation Planning and Technology **30**, 417 (2007).

[70] S. W. Yoon and S. J. Jeong, *An alternative methodology for planning baggage carousel capacity expension: A case study of incheon internationl airport,* Journal of Air Transport Management **42**, 63 (2014).

[71] S. E. Sørensen, *Automatic Design of Loop-Sorting-Systems*, Ph.D. thesis, Aalborg University, Aalborg, Denmark (2011).

[72] R. Thomas, C. van Baar, and M. van der Stee, *Baggage handling postures and the design of conveyors,* Applied Ergonimics **26**, 123 (1995).

[73] R. Pikaar and F. Asselbergs, *Systems engineering - innovation in airport baggage handling,* Human Factors In Organizational Design And Management **10**, 1 (2010).

[74] S. V. Korkmaz, J. A. Hoyle, G. G. Knapik, R. E. Splittstoesser, G. Yang, D. R. Trippany, P. Lahoti, C. M. Sommerich, S. A. Lavender, and W. S. Marras, *Baggage handling in an airplane cargo hold: An ergonomic intervention study,* Internatonal Journal of Industrial Ergonimics **36**, 301 (2006).

[75] Q. Wang and H. Jia, *Method of safety assessment for airport baggage conveyor belt,* Procedia Engineering **43**, 561 (2012).

[76] S. Jochems, *Improving KLM Customer Ground Handling's Competitive Market Postition*, Ph.D. thesis, Technische Universiteit Delft, Delft, The Netherlands (2015).

[77] A.M.M. van der Lande, *An analysis of the influences on the operational performance of KLM's Baggage Turnaround Services*, Ph.D. thesis, Technische Universiteit Delft, Delft, The Netherlands (2011).

[78] R. de Neufville, *The baggage system at denver: Prospects and lessons,* Journal of Air Transport Management **1**, 229 (1994).

[79] T. Keinonen and R. Takala, *Product Concept Design*, 1st ed. (Springer, 2006).

[80] T. J. Candalino, *Designing optimal aviation baggage screening strategies using simulated annealing,* Computers & Operations Research **31**, 1753 (2004).

[81] T. van Vianen, *Simulation-integrated Design of Dry Bulk Terminals*, Ph.D. thesis, Technische Universiteit Delft, Delft, The Netherlands (2015).

[82] R. Grigoraş and C. Hoede, *Design of object processing systems,* Memorandum 1830, Department of Applied Mathematics, University of Twente (2007).

[83] H. A. Donegan and F. J. Dodd, *A note on saaty's random indexes,* Mathematical Computational Modelling **15**, 135 (1991).

[84] J. Dul and T. Hak, *Case study methodology in business research*, 1st ed. (Butterwirth-Heinemann, 2008).

[85] Federal Aviation Administration, *Airport categories,* .

[86] E. J. Rykiel, *Testing ecological models: the meaning of validation,* Ecological Modelling **90**, 229 (1996).

[87] J. E. Hillston, *Model validation and verification,* (2003).

[88] N. Cross and N. F. M. Roozenburg, *Modelling the design process in engineering and in architecture,* Journal of Engineering Design **3**, 325 (1992).

[89] M. P. van Doorne, *Adapting Automated People Mover Capacity to Real-Time Demand via Model-Based Predictive Control*, Ph.D. thesis, Technische Universiteit Delft, Delft, The Netherlands (2015).

[90] J. Reichmuth, P. Berster, and M. C. Gelhausen, *Airport capacity constraints: future avenues for growth of global traffic,* CEAS Aeronautical Journal **2**, 21 (2011).

[91] P. Wang and D. Pitfield, *The derivation and analysis of the passenger peak hour: an emperical application to brazil,* Journal of Air Transport Management **5**, 135 (1999).

[92] ACRP, *Guidebook for measuring performance of automated people mover systems at airports,* Transportation Research Board **37A** (2012).

[93] B. W. Sloboda, *Transportation statistics* (J. Ross Publishing, 2009).

[94] J. Brand, S. Knepper, W. Mogendorff, L. Schouten, and M. Wammes, *A new arrival hall for Rotterdam The Hague Airport*, Ph.D. thesis, Technische Universiteit Delft, Delft, The Netherlands (2015).

[95] A. T. Wells and S. Young, *Airport Planning & Management*, 6th ed. (McGraw-Hill Education, 2011).

[96] MathWorks, *Simulink - simulation and model-based design,* First accessed on the 5th of November, 2015.

[97] Python, *Python overview,* First accessed on the 5th of November, 2015.

[98] SimPy, *Welcome to simpy,* First accessed on the 5th of November, 2015.

[99] Embarcadero, *Delphi 10 overview,* First accessed on the 5th of November, 2015.

[100] TomasWeb, *Discrete process simulation,* First accessed on the 5th of November, 2015.

[101] M. A. de Mesquita and A. E. Hernandez, *Discrete-event simulation of queues with spreadsheets: a teaching case,* Proceedings of the 2006 Winter Simulation Conference , 2277 (2006), iEEE.

[102] Z. Cekerevac and D. Zivković, *Manufacturing discrete events simulation using ms excel and vba,* First accessed on the 5th of November, 2015.

[103] H. P. M. Veeke, *Simulation Integrated Design for Logistics*, Ph.D. thesis, Technische Universiteit Delft, Delft, The Netherlands (2003).

[104] V. Kouikoglou and Y. Phillis, *Hybrid Simulation Models of Production Networks*, 1st ed. (Springer US, 2001).

[105] O. Özgün and Y. Barlas, *Discrete vs. continuous simulation: When does it matter,* Proceedings of the 27th International Conference of The System Dynamics Society (2009).

[106] Flughafen Dresden, *Homepage,* First accessed the 5th of January, 2016.

[107] Flughafen Leipzig/Halle, *Homepage,* First accessed the 20th of January, 2016.

A

SCIENTIFIC PAPER

Methodical modelling and design of baggage handling systems at airports

T.J.W. Bentvelsen*, Dr. W.W.A. Beelaerts van Blokland, Prof.dr.ir. G. Lodewijks
Transport Engineering and Logistics, University of Technology, Mekelweg 2, Delft, The Netherlands
B. S. ten Berge
Siemens Postal, Parcel and Airport Logistics, Lyoner straße 27, Frankfurt am Main, Germany

Abstract

A literature survey on concept design of baggage handling systems has revealed that there exists a knowledge gap for this topic. This research therefore investigates which generic design methods are available and assesses their performance on several criteria. The importance of these criteria is given by experts. Hereupon a design method has been selected and applied to construct a generic simulation model for baggage handling systems with several cases. The applicability of the model is tested by conducting two case studies. Although the initial results are promising, their significance cannot be determined until further validation of the model occurs.

I. Introduction

Although the importance of a baggage handling systems (BHS) is not widely known by its many daily users, such systems are an integral part of an airport's logistics system. Passengers will hand in luggage at a check-in counter and may reclaim it at their destination. The entire baggage handling process however, is more complicated and includes functions such as transfer, security screening, sorting and storage of bags among others. At present, one of the most important reasons for handling baggage is the requirement of 100% hold baggage screening (HBS) that is mandatory according to international legislation. This requirement may be traced back to three past events; the Pan Am 103 bombing, the explosion of TWA flight 800 and the shocking events that occurred on the 11th of September 2001 [15]. Since then, strategies and descriptions for hold baggage screening have been developed [2,15,32].

As this baggage screening requirement introduces an extra demand on baggage handling systems, efficient routing and scheduling of existing facilities becomes important. A feasible way of optimizing carrier systems is with model predictive schemes [33], predictive route control and route choice [24–27]. This optimization may include routing, line balancing and empty-cart management to both efficiency and reliability. The model predictive scheme becomes inefficient for large prediction horizons due to the scale of systems, but still outperforms existing methods. Routing of carriers may be tested with route verification methods [11,12]. With optimized routing, scheduling of aircraft to facilities becomes important [1]. Detailed simulation of baggage merge points is also described [10].

When optimizing carrier system routing principles, handling demand should be regarded. This demand is directly linked to the amount and type of passengers. An example of this is that passengers of low-cost carriers may carry less baggage. As low-cost carriers

*Corresponding author. Email adresses: tedbent@hotmail.com (T.J.W. Bentvelsen), g.lodewijks@tudelft.nl (G. Lodewijks), w.w.a.beelaertsvanblokland@tudelft.nl (W.W.A. Beelaerts van Blokland), balthazar-simon.ten-berge@siemens.com (B.S. ten Berge).

provide roughly 80% of the service quality for less than 50% of the cost of full-service carriers [5]. It is however expected that the spectacular growths of low-cost carriers will be limited due to route densities [3].

Furthermore, several articles report on saving energy on belt conveyor characteristics [16], the influences of minimum connecting time on baggage handling systems [6], a concept for handling baggage between make-up and aircraft [19], ergonomics of belt conveyors [13, 18, 28] and dynamics in loop-sorting-systems such as tilting tray sorters [23].

In literature on baggage handling systems, design requirements tracking [14] and geometry of layouts [7] have been described. A recommendation on several systems for three airport categories is made by the International Air Transport Association (IATA) [2]. None of these articles however mention the existence of a method for designing airport baggage handling system concepts. As this leaves a knowledge gap for the topic, this research investigates which design method may be applied for baggage handling system concept design. Selection of a design method is described in section III. The selected design method will be used to construct a generic model for baggage handling simulation in section IV. The applicability of the method and model in design cases is tested in section V. Methods by which this research is conducted are described in section II. Conclusions are drawn in section VI.

II. Research methods

Several methods have been used during this research. Among these methods are a method for surveying literature, a method for multi-criteria decision making, case study research and model verification and validation. In orde to conduct a literature survey, a brainstorm on relevant terms has been made. These terms have been searched for in the TU Delft Library, ScienceDirect, SpringerLink and Google Scholar. Relevant articles to the topic of baggage handling are described in section I. Concept design is described in section III.

Comparison and selection of design methods is conducted in accordance with the analytic hierarchy process [21, 22]. In this decision making process, criteria are ranked in a pairwise manner and alternatives are also ranked i a pairwise manner, per criterion. Matrix multiplication of both rankings is done with Equation 1, in which final ranking r is based on alternatives ranking matrix a and criteria ranking matrix b. The highest ranking alternative in matrix r is considered to be preferred. This result may be verified by a sensitivity analysis.

$$r_{ij} = \sum_{k=1}^{n} b_{ik} c_{kj} \quad (1)$$

The preferred design method is then used to construct a generic model for baggage handling systems by assessing several cases. These cases yield knowledge on system boundaries, functions, connections and solutions. Cases C1 to C6, presented in Table 1, will be used in the model's construction. Verification of the generic model occurs by evaluating the results of manual throughput calculation and model output [9, 20]. A full validation of the model's likeness to real world systems could not be completed as too few cases were available. The remaining two cases, C7 and C8, are thus used to test the applicability of the design method and model.

III. Design method selection

A literature survey on design methods has been conducted, in which several design methods have been found. Additionally, the current design method at Siemens Postal, Parcel and Airport Logistics is also taken into account. This design method sequentially described the steps that need to be taken in order to complete a design, but do not describe specific methods by which concept generation can be achieved.

Verein Deutscher Ingenieure (1977, 1993) also does not describe concept generation but

Table 1: Available cases for study with given peak flow rate and classification of airports on IATA and FAA standards.

Code	Airport name	Peak flow rate [bags/hour]	IATA category	FAA classification
C1	[redacted]	7.000	Category C	International hub
C2	[redacted]	11.500	Category C	International hub
C3	[redacted]	4.800	Category B	International hub
C4	[redacted]	2.700	Category B	International hub
C5	[redacted]	8.300	Category C	International hub
C6	[redacted]	2.100	Category B	Regional hub
C7	[redacted]	4.100	Category B	Regional hub
C8	[redacted]	11.000	Category C	International hub

does however show how complex systems can be separated into sub-systems [30,31]. Roozenburg & Eekels (1998) [4] describe the steps every designer takes to complete a design from initiation to final product. Mentioned in section I is the design method by Lemain (2002) [14], in which design requirements for baggage handling systems are denoted and updated. A calculation is presented by which the profit on baggage handling systems may be calculated. Pielage (2005) [17] presents a design method for freight transport systems and includes phases, steps and activities. The design method also describes simulation as being part of concept design to assess performance of concepts. Van Vianen (2015) [29] further integrates simulation in design for bulk handling terminals. The final design method is by Grigoraş and Hoede (2007) [8].

To evaluate which design method had preference, several baggage handling system experts have been interviewed. In this interview they were asked to compare the importance of eleven criteria in a pairwise manner, according to the analytic hierarchy process [21,22]. The criteria are:

1. Having aggregation layers
2. Client communication
3. Method flexibility
4. Method integration
5. Process iteration
6. Product life-cycle
7. Parallel development of concepts
8. Indicating concept performance
9. Method rapidity
10. Method simplicity
11. Design scenarios

The performance of design methods was evaluated per criterion and incorporated in the comparison. A final result was obtained by multiplying the weight of each criterion with the methods performance on that criterion. The results of this step have been verified with a sensitivity analysis, in which the effect of increasing and decreasing criterion weights has been evaluated. Both the final ranking (standard) and sensitivity analysis results are presented in Table 2. From this evaluation follows that the 'Design of freight transport systems'-method [17] was preferred most. The design method consists of problem analysis, system definition, system synthesis, simulation and evaluation of concepts. Implementation of this method is described in section IV.

IV. A generic BHS model

The selected design method has been used as a basis for the construction of a generic baggage handling system model. By assessing six design cases, C1 to C6, a system boundaries, functions, connections and synthesis may be determined. Analysing various stakeholders yields the following list of performance indicators and outputs of the model:

- In-system time
- System throughput

Table 2: Final results and sensitivity analysis to the method comparison.

	Standard	AL + 200%	CO + 200%	FL + 200%	IN + 200%	IT + 200%	LC + 200%	PD + 200%	PI + 200%	RA + 200%	SI + 200%	DS + 200%
VDI	0,09	0,11	0,08	0,08	0,09	0,09	0,08	0,12	0,07	0,09	0,09	0,08
R&E	0,06	0,06	0,06	0,07	0,06	0,08	0,06	0,06	0,05	0,06	0,06	0,06
LEM	0,15	0,14	0,20	0,15	0,16	0,13	0,17	0,14	0,14	0,16	0,18	0,14
PIE	0,20	0,21	0,20	0,21	0,18	0,22	0,21	0,21	0,17	0,17	0,18	0,2
G&H	0,15	0,14	0,13	0,14	0,14	0,13	0,14	0,15	0,17	0,18	0,15	0,16
VIA	0,18	0,17	0,15	0,17	0,16	0,15	0,16	0,17	0,23	0,18	0,18	0,21
SIE	0,18	0,17	0,18	0,19	0,2	0,19	0,18	0,16	0,18	0,16	0,16	0,15

- Storage capacities
- Required space per sub-system
- Energy consumption per sub-system
- Operational expenditure
- Capital expenditure

The available information is perceived to be a flight schedule, approximated distances, processing times as well as check-in, transfer and make-up opening windows. Work and safety regulations are part of design detailing, after a final concept has been selected and will therefore not be taken into account.

Baggage handling systems are defined as the transportation occurring between baggage drop-off and aircraft loading. Several other functions may however be included, among which in-feed, security screening, sorting, storage and outlet. The functions, connections between them, and system boundaries are depicted in Figure 1. Each function can be fulfilled with a number of different devices. A morphological overview of these devices per function has been established for system synthesis.

A simulation for the proposed model has been programmed in Microsoft Office Excel and Visual Basic for Applications. The simulation provides data on function capacities on which the amount of required systems is based. Assessing the use of such systems yields information on the proposed performance indicators. A formal verification of the model has been performed [9,20]. In this verification, simple inputs are provided to the model and results are compared to manual calculations. This is done to assess the proper functioning of the model. Several functionalities have been tested and approved but the model consists of numerous options and can therefore not be verified to the fullest extent. Validation of the model could not occur in this research as it required testing multiple design cases to compare the model's behaviour to reality. Instead, section V elaborates on two design cases.

Calculated performance indicators may be used to evaluate which sub-system is preferred. The analytic hierarchy process [21,22] is used to assess which performance indicators are prevail and how concepts perform on these indicators. A typical client list may be the basis for ranking performance indicators. A final design is proposed by linking all solutions to functions and adapting the layout to building drawings.

V. Case study

A total of two design cases have been studied to demonstrate the applicability of both the design method and model. The available cases of which flight plans and design parameters were available are case C7 and C8, [redacted] and [redacted] . The first case is a category B airport with

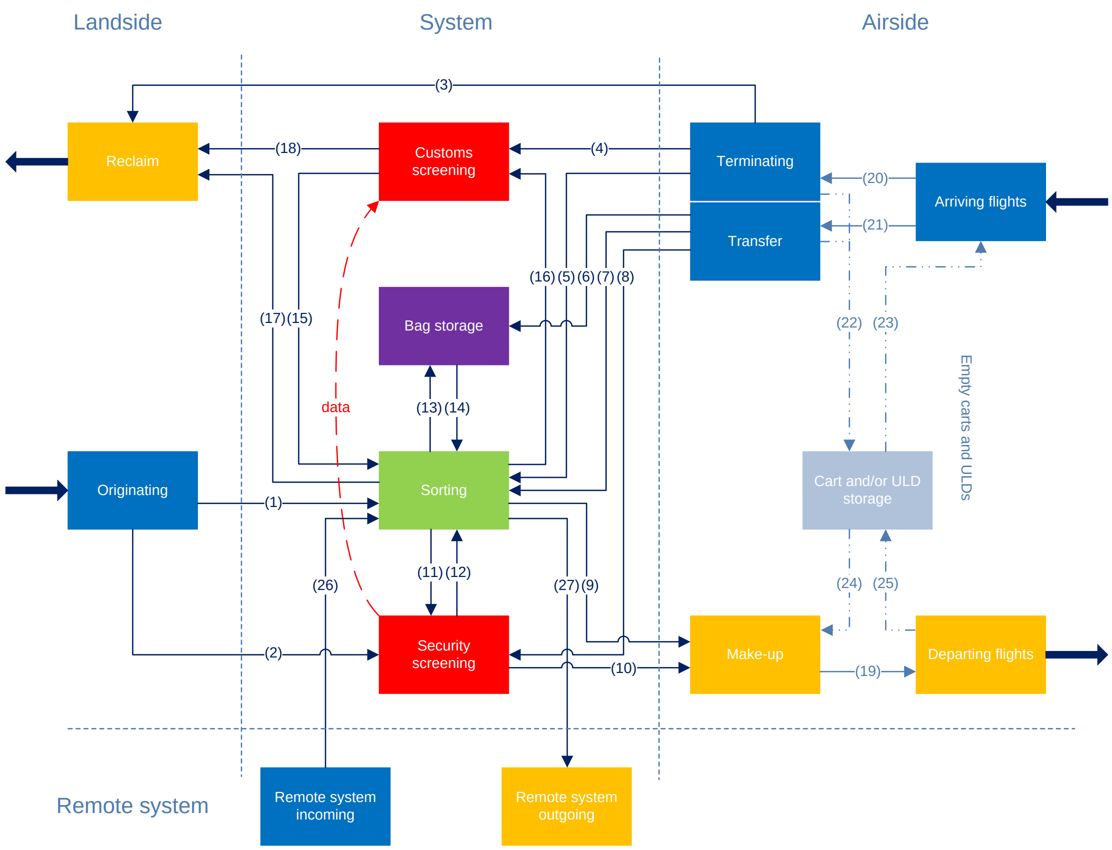


Figure 1: An schematic overview of a generic model for baggage handling system functions and connections. The model may be divided into landside, airside and system. It should also be noted that the sorting system may connect to a remote system that is already present or will be built in future.

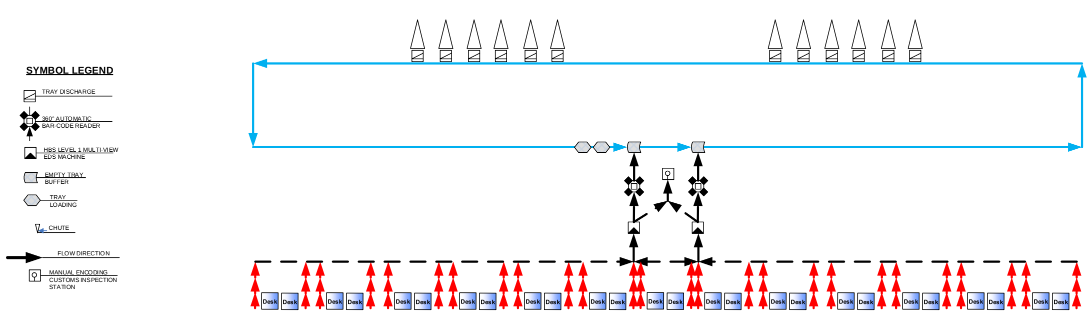


Figure 2: A material flow diagram of the concept for case C7. Belt conveyor transport lines are coloured black, carrier system transport lines are coloured blue.

a peak hour capacity of approximately 4.100 bags per hour. The second case has a peak hour capacity of roughly 11.000 bags per hour and is regarded as category C airport.

Based on the provided data in both cases, final design concepts have been established. For case C8, the concept consists of a list of components. For case C7, a material flow diagram has been constructed. This diagram is depicted in Figure 2. Comparing both concepts with their real world counterparts shows that in case C7, the systems are similar and conceptual design time has been reduced from [redacted] to 17 hours. This includes reporting on the design concept.

For case C8 however, less information was available on the design and its requirements. Comparison of the concept and developed system yields that both systems are different. Several causes may be a basis for this, such as having insufficient design rules or because of differences between proposed and real client preferences. The conceptual design time however has been reduced from an estimated [redacted] to 13 hours and 30 minutes. The significance of these results cannot be determined for these design times as only two cases have been studied.

VI. Conclusion

As no previous studies have been conducted on this topic, this research is a first step towards conceptual design of baggage handling systems with a generic simulation model and contributes to design theory. It has been shown that the design method by Pielage (2005) is preferred by experts and is applicable in concept design for baggage handling systems. The proposed simulation model has been tested in two cases of various size but still requires further validation. It is therefore recommended that future research should first focus on validating and possibly improving the proposed model.

As baggage handling systems are closely related to postal and parcel systems on function and sub-system level, this model may be generalisable in these fields. Possible future research could provide insight into this matter.

References

[1] Ahmed Abdelghany, Khaled Abdelghany, and Ram Narasimhan. Scheduling baggage-handling facilities in congested airports. *Journal of Air Transport Management*, 12:76–81, October 2006.

[2] Alexandre L. W. Bradley. *The Independent Airport Planning Manual*. Woodhead Publishing, 1st edition edition, September 2010.

[3] Jaap G. de Wit and Joost Zuidberg. The growth limits of the low cost carrier model. *Journal of Air Transport Management*, 21:17–23, July 2012.

[4] J. Eekels and Norbert F. M. Roozenburg. *Productntwerpen, structuur en methoden*. Boom Lemma Uitgevers, 2nd edition edition, June 2007.

[5] Markus Franke. Competition between network carriers and low-cost carriers - retreat battle or breakthrough to a new level of efficiency? *Journal of Air Transport Management*, 10:15–21, January 2004.

[6] Alexander Gediehn. Der einfluss der mct (minimum connecting time) auf baggageanlagen. pages 1–14, November 2002.

[7] Roxana Grigoraş and Cornelis Hoede. Design of a baggage handling system. *Memorandum 1835, Department of Applied Mathematics, University of Twente*, April 2007.

[8] Roxana Grigoraş and Cornelis Hoede. Design of object processing systems. *Memorandum 1830, Department of Applied Mathematics, University of Twente*, March 2007.

[9] Jane E. Hillston. Model validation and verification. , September 2003.

[10] Michael Johnstone, Doug Creighton, and Saeid Nahavandi. Simulation-based baggage handling system merge analysis. *Simulation modelling practise and theory*, 53:45–59, August 2015.

[11] Thomas Klotz, Jens Schönherr, Norman Seßler, Bernd Straube, and Karste Turek. Compositional verification of material handling systems. *Proceedings of the 17th IEEE International Conference for Emerging Technologies in Factory Automation*, 10:1–8, 2012.

[12] Thomas Klotz, Jens Schönherr, Norman Seßler, Bernd Straube, and Karste Turek. Automated formal verification of routing in material handling systems. *IEEE Transactions on automation science and engineering*, 10:900–915, October 2013.

[13] Sahika Vatan Korkmaz, Jeff A. Hoyle, Greg G. Knapik, Riley E. Splittstoesser, Gang Yang, David R. Trippany, Parul Lahoti, Carolyn M. Sommerich, Steven A. Lavender, and William S. Marras. Baggage handling in an airplane cargo hold: An ergonomic intervention study. *Internatonal Journal of Industrial Ergonimics*, 36:301–312, January 2006.

[14] T.R̃. Lemain. *Get the bag on the right track! 'Development of a new methodology for baggage handling system design'*. PhD thesis, Technische Universiteit Delft, Delft, The Netherlands, June 2002.

[15] Kelly Leone and Rachel Liu. The key design parameters of checked baggage security screening systems in airports. *Journal of Air Transport Management*, 11:69–78, March 2005.

[16] Gabriel Lodewijks. Energy efficient use of belt conveyors in baggage handling systems. *Networking, Sensing and Control, 2012 9th IEEE International Conference on*, pages 97–102, 2012.

[17] Ben-Jaap A. Pielage. *Conceptual Design of Automated Freight Transport Systems*. PhD thesis, Technische Universiteit Delft, Delft, The Netherlands, November 2005.

[18] Ruurd Pikaar and Floris Asselbergs. Systems engineering - innovation in airport baggage handling. *Human Factors In Organizational Design And Management*, 10:1–6, 2010.

[19] Joan C. Rijsenbrij and Jaap A. Ottjes. New developments in airport baggage handling systems. *Transportation Planning and Technology*, 30(4):417–430, August 2007.

[20] Edward J. Rykiel. Testing ecological models: the meaning of validation. *Ecological Modelling*, 90:229–244, August 1996.

[21] R.W̃. Saaty. The analytic hierarchy process - what it is and how it is used. *Mathematical Modelling*, 9(3-5):161–176, 1987.

[22] R.W̃. Saaty and Joseph M. Katz. How to make a decision: The analytic hierarchy process. *European Journal of Operational Research*, 48:9–26, November 1990.

[23] Søren Emil Sørensen. *Automatic Design of Loop-Sorting-Systems*. PhD thesis, Aalborg University, Aalborg, Denmark, May 2011.

[24] Alina N. Tarău, B. de Schutter, and J. Hellendoorn. Travel time control of destination coded vehicles in baggage handling systems. *Proceedings of the 17th IEEE International Conference on Control Applications*, pages 293–298, September 2008.

[25] Alina N. Tarău, B. de Schutter, and J. Hellendoorn. Hierarchical route choice control for baggage handling systems. *Proceedings of the 12th IEEE International Conference on Intelligent Transport Systems*, pages 679–684, October 2009.

[26] Alina N. Tarău, B. de Schutter, and J. Hellendoorn. Model-based control for route choice in automated baggage handling systems. *IEEE Transactions on Systems, Man and Cybernetics - Part C: Applications and Reviews*, 40(3):341–351, May 2010.

[27] Alina N. Tarău, B. de Schutter, and J. Hellendoorn. Predictive route control for automated baggage handling systems using mixed-integer linear programming. *Transportation Research Part C*, 19:424–439, 2011.

[28] R.G. Thomas, C.E. van Baar, and M.J. van der Stee. Baggage handling postures and the design of conveyors. *Applied Ergonimics*, 26(2):123–127, 1995.

[29] Teus van Vianen. *Simulation-integrated Design of Dry Bulk Terminals*. PhD thesis, Technische Universiteit Delft, Delft, The Netherlands, January 2015.

[30] Verein Deutscher Ingenieure. Konstruktionsmethodiek: Konzipieren technischer produkte. 1977. VDI 2222.

[31] Verein Deutscher Ingenieure. Methodik zum entwickeln und konstruieren technischer systeme und produkte. May 1993. VDI 2221.

[32] K. Wells and D.A. Bradley. A review of x-ray explosives detection techniques for checked baggage. *Applied Radiation and Isotopes*, 70:1729–1746, February 2012.

[33] Yashar Zeinaly, Bart de Schutter, and Hans Hellendoorn. An integrated model predictive scheme for baggage handling systems: Routing, line balancing, and empty cart management. *IEEE Transactions on control systems technology*, 23:1536–1545, July 2015.

B

Initial research assignment

Faculty of Mechanical, Marine and Materials Engineering
Section: Transport Engineering and Logistics

Thesis Entrance Permit MSc Transport Engineering and Logistics

Part 1. General information

Last Name:	Bentvelsen
Initials:	T.J.W.
Email:	tedbent@hotmail.com; t.j.w.bentvelsen@student.tudelft.nl
Student Number:	1534645
Start MSc programme (month/year):	09/2013
Date:	2-4-2015

Completed BSc	☒ yes ☐ no	Completed Bridging Programme (when applicable)	☐ yes ☐ no
Completed Internship	☐ yes ☒ no		
Completed MSc courses	☒ yes ☐ no	Agreed with Chair	☐ yes ☐ no

Graduating Professor:	Prof. dr. ir. Gabriel Lodewijks
First supervisor:	Dr. Wouter W.A. Beelaerts van Blokland
Second supervisor:	

Deliverables end of project:

☐ Literature review report and thesis report
☒ Research paper and thesis report

Part 2. Project information (to be filled in by the supervisor)

Graduation in a company/institution	☐ no ☒ yes if yes, please fill in the contact details below[1]
Adress	Lyoner Str. 27 60528 Frankfurt am Main, Germany
Email adress	balthazar-simon.ten-berge@siemens.com

[1] Please note that if you are going abroad as part of your studies at TU Delft, then you are required to register your emergency contact information in OSIRIS. In case of emergency, the university will then be able to get in touch with family or with your host institution. Log in to OSIRIS through Blackboard to register your emergency contact information

Research description (complete to maximum of 2 page):

Modern day airports exist in various sizes and types. From local airports to regional airports and intercontinental hub airports, each requires a specific approach for the design of a baggage handling system. In some cases the design is made from scratch, whilst in other cases an expansion of the existing system is expected.
To accommodate for these various requirements, this master graduation project intends to improve the method for developing baggage handling system concepts. By reducing the design process complexity a shorter design period may be realised. This issue is addressed by incorporating the relationship of subsystems and their dynamic interaction, referred to as *systemic approach*.

Title: Improving baggage handling system design through simplification of the concept development process.

Company
Siemens Logistics and Airport Solutions is a leading provider in the field of mail and parcel, baggage and cargo sorting solutions headquartered in Constance, Germany. With an installed basis in more than 50 countries worldwide, Logistics and Airport Solutions essentially serves two large groups of customers: airports and airlines as well as courier, express, parcel and postal services.
This puts Logistics and Airport Solutions in the advantageous position of enjoying the solid financial basis provided by Siemens while adopting lean and flexible structures which will improve customer proximity and guarantee timely project execution.

Reason for the assignment
The creation of design concepts for baggage handling systems is affected by a large quantity of known and unknown requirements. These requirements may contain both measurable variables and subjective aspects. The amount of interactions between subsystems (and components) – based on the properties of the components – determines the complexity of the design of baggage handling system concepts.

Aim of the assignment
The aim of this project is to develop a systemic approach that enables a quick baggage handling system design by simplifying the concept development process.
The following objectives are of importance:

- Identify the existing types of baggage handling systems and sub-systems.
- Define classification criteria and design criteria.
- Match the classification criteria with design criteria.
- Develop a systemic design method to enable quick concept development and system evaluation.

The method may be applicable to the various types of systems. It is expected that simplification of the design process will reduce the concept design time and eliminate several errors.

Approach

In the assignment, a systemic approach will be used for the analysis and development of the concept design process. Several methods which may be of use during this assignment:

- Morphological method
- Multi-criteria analysis

- The Delft systems approach
- Black box approach

- Lean manufacturing
- Value stream mapping
- Six sigma approach
- 5S methodology

- Theory of constraints
- Material flow analysis
- Control theory

This will be further clarified during the kick-off meeting 4 weeks after the start of the project.

Project deliverables

At the end of the project a full report will be delivered to both the Delft University of Technology and Siemens Logistics and Airport Solutions. The report will describe how to methodically improve the design process of baggage handling systems with a *systemic approach*. Additionally, a paper will be written on this subject.

Supervision

Company supervisor: Balthazar Simon ten Berge
TU Delft supervisor: Wouter Beelaerts van Blokland

Additional information

For this project, the main research question, main literature and research methodology are still to be determined. This will take place in the first month of the internship, to be referred to as the "orientation" phase.

The project addresses the design of baggage handling systems, which play a key-role in the processing and transport of baggage on airports. Further advancement in the development of system concepts could confirm or falsify established ideas on systemic approaches for engineering design practise, affecting both academic knowledge and industrial relevance.

Keywords: baggage handling system, concept, design, development, systemic approach

Planning

Milestones	**Week (approx)**	**Fill in the (planned) dates**
Start of project	Week 23	**June 1st, 2015**
Kick off session	Week 27	**June 29th, 2015**
Mid term review (optional)	Week 36	**September 1st, 2015**
Greenlight review (optional)	Week 48	**November 23rd, 2015**
Expected Graduation	Week 51	**December 15th, 2015**

Comments

Student	**First supervisor**	**Graduating Supervisor**
Date	Date	Date
Signature	Signature	Signature

C
BRAINSTORM

For the literature survey, a brainstorm has been made to identify words that are relevant for the literature survey. The brainstorm is depicted in Figure C.1.

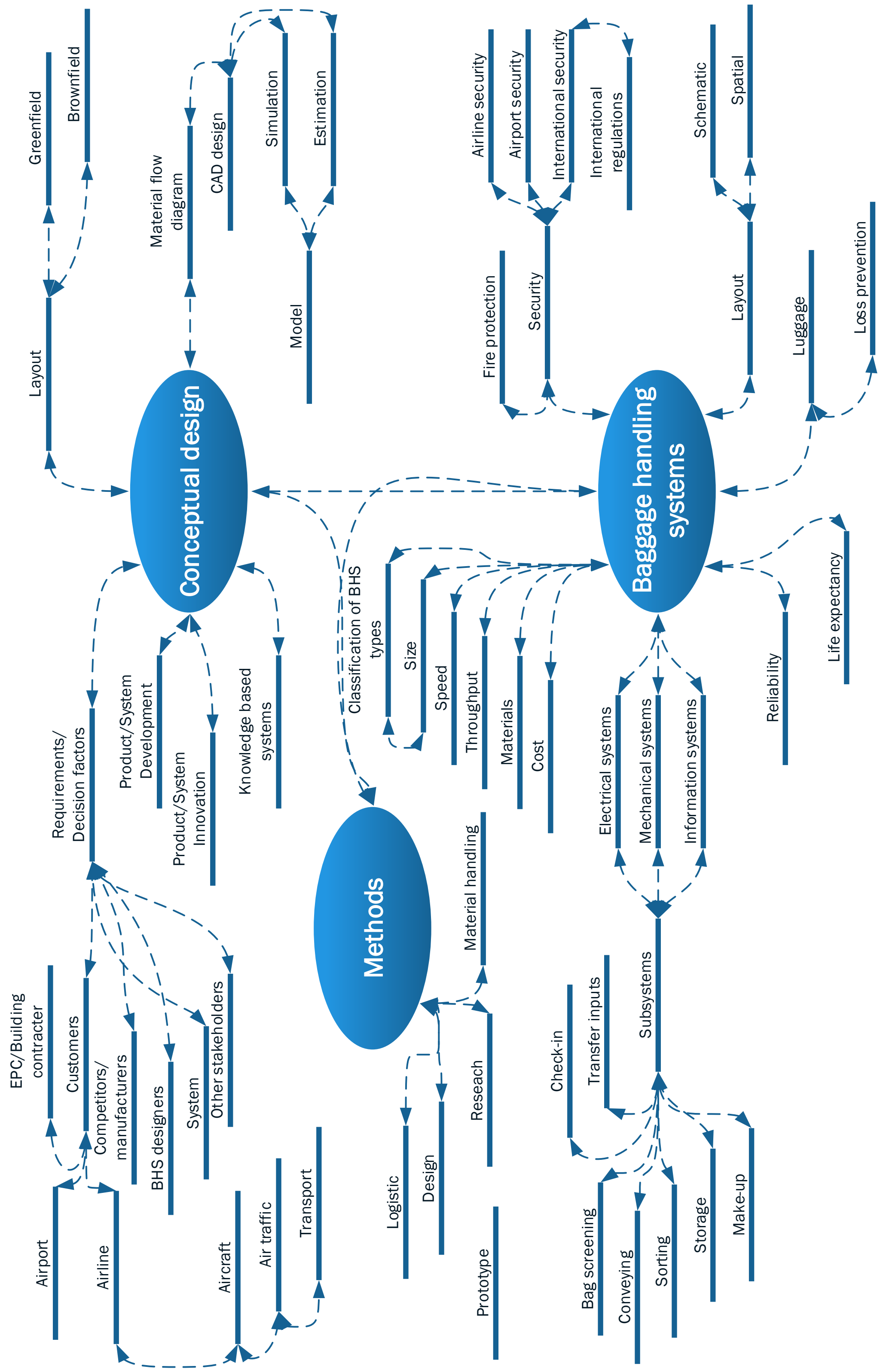


Figure C.1: A brainstorm diagram of words that are associated with baggage handling, conceptual design and methods. These may be used in the literature survey for relevant works.

D

STAKEHOLDERS

Assessing stakeholders may yield valuable information on requirements that define the design process. An extensive stakeholder analysis for baggage handling system design has already been performed at Vanderlande Industries by Lemain [43]. The analysis is concerned with investigating as many stakeholders as possible in the design of baggage handling system and their requirements. They may range from belt designers to ground handlers and airlines, all who affect and are affected by the baggage handling system in any way. From this analysis and personal correspondence with ten Berge [5], stakeholders are selected that are considered to have an influence on conceptual designs. Several stakeholders and their influences are:

Governments National governmental bodies are involved in every airport project. They may either be a controlling organization or be related to the requesting group of decision makers. As controlling organization governments will determine work and safety regulation for the building and system. They will also determine landing rights for airlines and amount of annual landings, as well as the amount of aircraft movements for the airport. Several factors that they have influence on:

- Work regulations
- Safety regulations
- Landing rights
- Airspace access
- Aircraft movements
- Opening times

Municipalities Representing the desires of local people and businesses in the vicinity of an airport, municipalities will have influence on the expansion of airports and the opening times of an airport. However, since airports are projects of national concern, local authorities may be overruled by their government. As far as information is available, this stakeholder is therefore not taken into account.

Customs & State Police As part of the national government, the customs and state police organizations have to check baggage for unwanted substances, items, animals, or people entering the country. So in the terminating baggage flow, they have a big influence to prevent things and may also want to check departing passengers. They will follow governmental regulation of incoming baggage scanning, whether to scan it all or just selectively, but these rules are interpreted differently by each airport. Generally, baggage and people are sampled by these organizations but it may be decided to check every item. It may therefore be coupled to the 100% HBS. Several factors they have influence on:

- Screening equipment
- Screening equipment regulations

Airport group The airport group is a combination of airport owners that can consist of airlines, private investors or a governmental body. They are the main decision makers for baggage handling systems and consider factors like:

- Capital expenditure

- Operational expenditure
- Capacity
- Throughput
- Flight schedule
- In-system processing time
- Required space
- Reliability

A design project is initiated by the airport group which may consist of governmental organizations, airlines and private investors. This group awards a project or directly contacts baggage handling system manufacturers. Together with the other stakeholders, they may select representatives to form a decision making unit. A DMU is formed to convey the requirements of each stakeholder to the manufacturers unanimously. As decision making units are typically not trained in forming requirements, this may be done in cooperation with a consultancy firm. Some DMUs are authorized to make design decisions, whilst others purely serve an informative function to their management. These mentioned factors must also be proofed with a sensitivity analysis. Group must be informed well of the system requirements before making a final decision.

Airlines Are one of the main users of baggage handling systems. As main users (being dependent upon the service) they have influence on the performance of the system. Some airlines may even be owners of baggage handling systems and thus have a larger stake in factors like:

- Capacity
- Throughput
- Flight schedule
- In-system processing time
- Operational expenditure
- Reliability

Ground handlers A group of companies specialized in handling baggage. They operate the baggage handling system and cargo transport towards the aircraft. Although they do not have a significant influence on the design itself, they do have to give permission based on factors like:

- Ergonomics
- Work regulations

They represent themselves in predetermined heights of loading equipment, the movement that is required to load an item and the amount of tons per person. This will be determined in the detailed design. Thus are very important, but not on the foreground in the design process. Workers are represented by a person in the DMU and will also have to agree with the final design. Do not necessarily have influence on the design itself but go/no-go decision capabilities. These requirements are elaborated with the detailed design and do not play a part in the conceptual design.

Passengers Indirectly have influence on which airport they choose but do not directly determine how the baggage handling system will be designed. An important aspect of a baggage handling system design is the user interface. The user is however not a part of the DMU so unable to convey its requirements in the conceptual design phase.

Contractors & Suppliers In both the public tender process and the private tender process, contractors and suppliers of materials have influence on the design by having developed certain systems and solutions. These solutions are adapted to the requirements of a tender. It may happen that several solutions of competitors are also adopted when this is regarded as better. So therefore, competitors also have an influence. Suppliers may at some point bring in their own ideas into the tender phase. This may lead to changing requirements at the customers side.

Consultants Guiding an airports group and DMU into the tender awarding process. Does have a direct influence on the final decision because of proximity to the customer and their dependency on a consultants knowledge. Consultants are often used in awarding and completing a tender document, which ultimately defines the outline of the baggage handling system. They may also provide a brand neutral design as a reference for suppliers.

Architects Design the building and surroundings for the DMU and thus have a say over the amount of space that is available for the baggage handling system. The system may not be too large but required size expansions may be discussed with the architect. This is mentioned as the system envelope, which may change during a new terminal design. They have less influence during system redesigns, but may still play a part.

- Required space

Investors Determine the amount of available money for the project, and have a say in how it is spent. They often push a project towards a public tender, as this will bring them more security. Although they do not have a direct influence on the design itself, they do have to give a go/no-go on the final design plan, to approve their investors. These investors will particularly be interested in:

- Capital expenditure
- Operational expenditure

It is seen that different parties in the design of baggage handling systems present different design requirements. Some of these requirements are not regarded during the conceptual design. Others are represented by the decision making unit. Conflicting requirements are resolved by the DMU itself and presented as a tender contract. It may be seen that multiple parties are involved with both the operating expenditure as well as the capital expenditure of a baggage handling system.

E

BAGGAGE HANDLING EQUIPMENT

A short introduction to baggage handling is given in chapter 1 to provide a context for this thesis. This introduction was elaborated with three examples of system solutions. There are however more solutions available for designing baggage handling systems, which are presented in this chapter. Before mentioning various solutions, a full definition and purpose for baggage handling system is given:

> *"A baggage handling system is a type of transport system installed in airports and which transports checked baggage from ticket counters to areas where the bags can be loaded onto aircraft. A BHS also transports checked baggage coming from aircraft to baggage claims or to an area where the bag can be loaded onto another aircraft."* - Lodewijks [46]

Operations occurring between check-in counters and the make-up area are considered as baggage handling. The part between the make-up area and the loading of aircraft is taken as cargo handling since at this stage loose baggage, ULDs and cargo may be combined. Activities that occur in a baggage handling system and the solutions thereto can be categorized in various ways. One way to perform the categorization is to assess solutions on functional level. Various solutions to functions are presented in section E.1 with multiple parts of equipment for each function. The IATA has also made a recommendation for baggage handling system equipment conforming to the system's peak hour flow rate. This recommendation is matched and compared to the functional grouping in section E.2.

E.1. BAGGAGE HANDLING EQUIPMENT

In the previous section, different functions of a baggage handling system are explained. This section will order feasible solutions according to these functions. The obtained knowledge is retrieved from several publications as well as the portfolio of SPPAL, Beumer Group and Vanderlande Industries. Subsections E.1.1-E.1.4 discuss the basic functions from subsection 4.3.2, whilst subsections E.1.5-E.1.8 discuss the additional functions. As the gathered information shows overlap between products from different sources, it has been generalized on principle of operation. It is also important to mention that an ICS can be used for both transport and sorting.

E.1.1. IN-FEED SOLUTIONS

A baggage handling system may be designed to receive items from different sources. These sources can be subdivided into on-site check-in, off-site check-in and arriving baggage inputs. On-site check-in and arriving baggage inputs are located in airport terminals, whereas off-site check-in inputs may be located at an airport's car park or at a city's railway station. Several on-site check-in systems are:

Check-in counter A conventional baggage handling system input is the well known check-in counter depicted in Figure E.1. Every counter is manned by personnel from a specific airline and passengers hand in their luggage at these counters. Several options for the layout of these counters exist, they may for example be placed in a line or an island formation and in a single or a mirrored alignment. A typical counter has two to three belt conveyors for the transport of bags, with one conveyor containing a weight sensor. After completing the check-in process, luggage is transferred to a collection conveyor behind

Figure E.1: A conventional check-in counter for baggage with a limited size [15].

Figure E.2: The uDrop system at Paris Charles de Gaulle Airport [16].

Figure E.3: A Lufthansa counter for handing in special baggage [17].

the check-in counter. This collection conveyor supplies the transport of items to a sorting device or security screening system.

Baggage drop-off point As a response to the high labour costs, several companies have developed systems which allow passengers to self-drop baggage as is depicted in Figure E.2. These self drop-off systems range from simple open systems to more complex systems containing detection equipment. After putting an item on the belt conveyor, a typical self drop-off counter weighs the item and prints a label that must be attached to the baggage. The bag is then transported to a collections conveyor which runs behind a line of drop-off systems and enters the baggage handling system.

Special baggage counter Conventional baggage is often limited in size and baggage handling systems are adapted to this size. It may however happen that passengers wish to take larger items with them. These items are referred to as special baggage, oversized baggage, odd size baggage or out-of-gauge baggage. To still allow passengers in taking these special items on aircraft, a select amount of special baggage counters are available on airports (see Figure E.3) for items such as golf clubs, kayaks, weapons and ammunition, bicycles and musical instruments. Depending on the airport's requirements, special baggage may be transported directly to the aircraft or via the baggage handling system. It may also happen that passengers are referred to cargo handlers for items that do not satisfy to the oversized baggage limitations.

Besides the on-site check-in baggage, passengers may also find that it is possible to hand in their luggage at other locations. The discussed check-in options for example may also be available at specific locations in the airport's car park. These baggage in-feeds are also connected with the baggage handling system. In rare cases it is even possible for passengers to check in luggage at large railway stations that directly connect to the airport. After check-in at such a railway station, baggage is loaded into ULDs and transferred into a custom wagon. When arriving at the airport railway station, the ULDs are unloaded from the train and emptied into the baggage handling system.

Opposing departure baggage, arriving baggage consists of terminating baggage and transfer baggage. Terminating baggage will be reclaimed by the passenger, but still needs to be transported from an aircraft to the reclaim area. This may be performed by loading the luggage from carts and ULDs onto a lateral or using an automated unloading device. These unloading methods are also used for transfer baggage, which is separated from terminating baggage in aircraft's hold. Transfer baggage has to be screened again, as it is considered to be hold baggage for the new flight. Additionally, transfer baggage is sometimes transported directly from arriving aircraft to departing aircraft, in case the transfer window is narrow and it is expected that the baggage handling system will not process the transfer bags timely.

E.1.2. SECURITY MEASURES

As was mentioned in the introduction of this thesis, security has become an important part of the baggage handling operation. National and international regulations govern that nowadays 100% of the hold baggage has to be screened before it enters the aircraft [95]. This includes transfer baggage as well. Screening may be performed with several devices and clearance is assessed by either a computer algorithm or security officer with an on-screen resolution (OSR) system [39]. The various methods are manual checking, x-ray scanning, tomography scanning and diffraction scanning.

Different ideas exist throughout the world about which place is most suited for hold baggage screening. One idea is to perform the HBS near the check-in counters such that rejected bags can be identified and retrieved immediately, as well as the bag owner. It is however difficult to match the capacity of expensive screening machines with the capacity of check-in counters. That is why more efficient hold baggage screening systems are located after merging and diverting several check-in collection conveyors. These systems consist of a tiered approach to baggage screening, as is depicted in Figure 2.1. All baggage is screened by a multiview x-ray and an algorithm or security officer decides its clearance. Uncleared bags are rescanned by a more detailed tomography scan, after which it is reviewed. If the bag is still not cleared, it continues to the manual inspection or trace detection area. Here, a decision is made to dispose of the bag or to clear it. All cleared bags are diverted from the dispose line to a line which continues the baggage handling.

Whilst hold baggage screening occurs for all baggage that enters the cargo hold of the aircraft, customs screening occurs when baggage is terminating at the airport. The strictness on customs screening is largely dependent on country regulations. Customs screening is performed either by random picking passengers and manually opening bags, or by a screening device positioned before the baggage reclaim. Similar methods as hold baggage screening may be used, independent of the airports size. Since hold baggage screening is mandatory, the only remaining options are the location of screening and whether to do it in-line or on a standalone device.

E.1.3. TRANSPORTATION

Unlike the in-feed and the security systems, the transport function has widely varying solutions. Three of these solutions have already been presented in section 1.1, but still require elaborating. Multiple transport solutions can also be adopted in a single baggage handling system and transport may recur between other processes.

Manual transport Already mentioned in chapter 1 and depicted in Figure 1.2 is the manual transportation of baggage, with or without aiding devices. On very small airports which require limited capacity or on airports with check-in counters close to the make-up area, manual transport may be a viable option. This manual transport may be assisted by tractor pulled carts and ULDs, as depicted in Figure E.4. It should be noted that transport between a make-up area and an aircraft loading area also uses carts and ULDs, but this part of the baggage handling system is not considered in this thesis.

Belt conveyor A commonly used type of equipment for transporting baggage are belt conveyors. Belt conveyors consist of a minimum of two pulleys that carry a belt and are available in multiple widths and lengths to satisfy luggage size requirements. Belts may consist of straight, curved and inclined sections, each with a one or more powered pulleys. The velocity of a belt can be regulated by the motor controller and the motor power is dependent on maximum allowable weight on the belt. Figure 1.3 depicts several different sections of belts respectively. There exists an important trade-off between long and short belt sections. Short sections can be activated shortly for a single piece of baggage, whereas long belts need to run continuously until the item is passed. This results in succeeding short sections requiring less energy than an equally long single belt. The trade-off in this case is that the succeeding short belt are more vulnerable to breakdowns than a single long belt, due to probability.

Chute In order to have baggage descend without the use of power, chutes may be used. Chutes are available in straight sections and curved sections, as depicted in Figure E.5. Curved sections are specifically useful when luggage needs to be lowered in a limited amount of space. Using chutes after individualization in a baggage handling system is not recommended since the individualization of baggage is lost upon entering the chute. A typical location for chutes in baggage handling systems is before manual make-up solutions, where it is not necessary to identify individual items.

Elevator Similarly to chutes requiring limited space for baggage descent, item elevators can be used for both ascending and descending items but do require a source of power. The main advantage of item elevators is that the individual character of baggage is retained. Multiple solutions are available for the lifting of items, for example vertical conveyors, spiral conveyors, lifts and forklifts.

Individual carrier system A system which is offered by three major baggage handling equipment providers is the individual carrier system (ICS). The fundamental principle of such systems is that every single baggage item is loaded in a carrier and the carrier transported to its destination via rail or belt tracks. In these systems, either the track or the carrier is powered and may achieve velocities of up to $15\frac{m}{s}$ on

Figure E.4: A mixed train consisting of a tractor, three ULD carriers and two carts respectively for transport between make-up and aircraft [18].

Figure E.5: Spiral chute that allows items to drop multiple levels [19].

straight sections. Loading of items on these carriers is performed with a belt conveyor, whereas unloading may be performed by tilting the carrier or activating a cross-belt on suitable carriers. The tracks of an ICS system may have switches, which allow these systems to sort carriers to specific destinations.

E.1.4. SYSTEM OUTLETS

Whether a baggage handling system is design for loading carts and ULDs returning baggage to the passengers or a composition of both, all systems require an outlet. These outlets can be subdivided into solutions for make-up areas or solutions for baggage reclaims. Several devices are available that assist in the loading of bags into carts and ULDs. These devices increase the productivity of manual loading. Considering the make-up area, the following solutions are available:

Lateral Laterals are long, straight belt conveyors that are positioned at such a height that items can easily be transferred between lateral and cart or ULD. An example of a lateral is depicted in Figure E.6.

Carousel Carousels can be used to hold baggage that is about to be loaded into a cart or ULD. They may also serve as a short term buffer for baggage that has arrived earlier than aircraft loading has started. When loading needs to be performed, baggage can easily be picked up from the carousel and loaded. In Figure E.7, an inclined make-up carousel is depicted. These carousels do not necessarily need to be inclined and may also be flat. Holding capacities are largely determined by the length and width of a carousel.

Chute Instead of using chutes as a medium for vertical transport of baggage items, they can also be used as baggage handling system outlets. Items are dropped into the chute from which a worker loads them. This is depicted in Figure E.8.

Automatic loader A new development in airport technology is the robotic loading of baggage into carts and ULDs. The robot receives items piece by piece from a queuing system and an algorithm determines the most suitable location for each item. Several of these systems have already been implemented at Schiphol Amsterdam Airport, which can be seen in Figure E.9.

Independent of the type of outlet bin, lateral or carousel, baggage can be loaded manually into carts and ULDs or with the help of loading aids. These aids are designed to relieve the work load of employees following lifting load restrictions in several countries. Loading aids may come in the form of movable platforms or cranes with suction cups. When using an automated loader, the choice between manual loading and aided loading becomes irrelevant as the machine is able to operate without supervision. It is however common to have supervision on these machines, as the technology is still relatively new.

Figure E.6: Manually loading carts with baggage from laterals [20].

Figure E.7: Loading baggage carts from an inclined conveyor [20].

Figure E.8: Chutes collect baggage for a single destination [20].

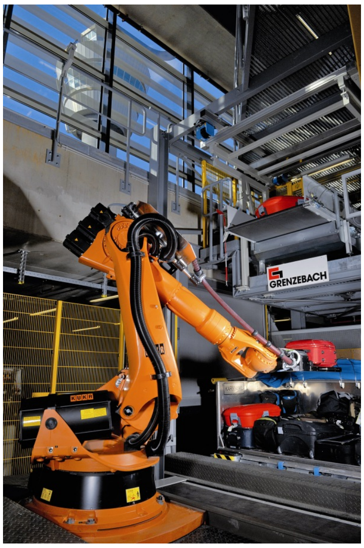


Figure E.9: Automated loading robot for loading carts and ULDs [21].

Figure E.10: Wall outlet for special baggage at Salt Lake City Airport. Due to being a popular skiing destination, this airport has a ski-equipment reclaim area [22].

Besides loading aircraft, a baggage handling system also serves as solution to deliver baggage to passengers in the reclaim area. These outputs may also consist of laterals and carousels, sometimes called racetracks. A special type of output that is used for reclaiming odd sized baggage is the wall outlet depicted in Figure E.10, and may be considered as a type of bin. This is used since odd sized baggage is only a small part of all baggage and typically does not fit on a reclaim carousel.

E.1.5. BAGGAGE SORTING

An important additional function to baggage handling is the sorting of baggage for several destinations. As was mentioned earlier, baggage from several check-in desks is carried on a collection conveyor. Since different check-in counters may have various destinations, it becomes necessary to sort the baggage for each destination. More efficient use of security screening machines is also a reason for merging and sorting baggage items. Due to the solutions for transport, sorting may take place in four different ways:

Manually In manual sorting, baggage is transported to the make-up area, from which workers take and sort the baggage to the correct destination. As one can imagine, this is a time consuming and labour intensive process. It may however be viable to implement manual sorting on several airports which for example require a low amount of throughput or need a flexible capacity.

In-line conveyor systems When requiring more throughput, belt systems with sorting capabilities become of interest. Basic solutions to diverting are the pusher, single swing-arm diverter and double swing-arm diverter, which can be used in high-speed diverting. These three solutions can be applied to belt conveyors as well as flat carousels. A fourth solution is the vertical sorter, which allows one belt conveyor to sort on two belt conveyors stacked vertically.

Tilting and cross-belt sorters Besides sorting baggage on belts, dedicated sorting equipment may be used.

In a tilting tray sorter, baggage is loaded on trays which tilt when the correct outlet is reached. The system consists of a chain of trays which loops around and connects to the first tray. Trays may be loaded and unloaded from both sides of the tray. Similar to this tilting tray system is the crossbelt sorter, but instead of tilting the tray to unload, the sorter has a small belt which activates at the correct destination and unloads the item. The final dedicated sorter is the sliding shoe sorter, also called sliding bar sorter. Again quite similar to the tilting tray sorter, a sliding bar sorter consists of multiple trays with items loaded on it. When reaching the correct destination, a sliding bar is activated and pushes the item off the tray. The shoe sorter can compared with the sliding bar sorter, with major differences being that the bar is replaced by a smaller bars called shoes and the tray is replaced by a continuous medium. Due to this, items with varying lengths can be sorted.

In-line ICS The individual carrier system as described in subsection E.1.3 has the ability to transport baggage as well as sort it. This sorting takes place by routing carriers to different locations with track switches and merges. At these destinations, carriers are unloaded. These unloads take place by tilting the tray when it is either at rest or moving. Tilting a tray at rest is referred to as static tilting whilst tilting a moving tray is referred to as dynamic tilting. It may also happen that multiple unloading point on a single track are used for sorting baggage items to their destinations.

E.1.6. Storage of baggage

In some cases, it is desired to store baggage for a certain amount of time. Items may be stored for as short as several minutes to hours or overnight storage. The demand therefor is caused by departing and transfer passenger arriving early at an airport, before make-up of their flight initiated. To compensate for this, baggage can be stored with:

Manual storage On smaller airports, it is often not necessary to store baggage for a long time. However when passengers arrive early and make-up of a flight has not started yet, suitcases are put aside to be loaded later on. This can be seen as manual storage of baggage.

Conveyor buffer Storing baggage on belt conveyors is typically used for staging early bags before flight, in order to have the make-up continually loading carts or ULDs. The belt conveyor may be equipped with a detector which forwards the belt only when a new item arrivers. In more advanced systems, multiple metering conveyors (see subsection E.1.7 may be used successively to have evenly spaced baggage.

Virtual storage A principle for storing baggage for short amounts of times is with virtual storage in sorting loops. Larger baggage handling systems may contain several loops which allow for storing a bag by letting it pass a loop. This is advantageous as it does not require dedicated storing capacity, it does however reduce the throughput capacity of the sorting system.

Carrier lanes More advanced systems can be used in case make-up has not started for an aircraft. Baggage is stored on carriers statically and dynamically. The static carrier storage, in which trays are stored in one lane after another, is depicted in Figure E.11. Such a system may also contain a loop in which carriers are kept moving, the dynamic carrier storage. The control system uses dynamically stored carriers when it is expected that the trays are requested within a short amount of time.

Storage and retrieval systems Another option for more advanced storage is a storage and retrieval system (SRS). These systems are fully automatic and a controller decides when to release each stored item into the baggage handling system again. Storing may happen with a moving elevator that can store trays in racks, single sided or double sided. Another option for automatic storage is the lift & run system, in which the system lifts trays and runs them on track to the storage bay. A lift & run system thus has a higher throughput.

E.1.7. Individualization

Processes such as in-line screening, sorting and storing require items to be individualized before they can be handled further. Additionally, the loading of ICS trays also requires single baggage items. To achieve this, metering conveyors are used. These conveyors can be quickly regulated based on the occupation of the conveyors ahead. Metering conveyors are placed in groups of four before systems such as screening machines, sorter inductions or belt conveyor storage lanes.

Figure E.11: Lane storage of baggage on an ICS at Bergen Airport [23].

Figure E.12: Storage of baggage in an automated storage and retrieval system [24].

In small baggage handling systems, items may be identified and individualized manually. This ensures that items are clearly separated when they enter the security screening. In larger systems, items are separated with the use of a metering conveyor. This type of conveyor is slightly larger than than the maximum allowable baggage size and is able to detect when an item passes. With the detection, it now becomes possible to separate items from each other and either load them in ICS trays, or continue on with screening, sorting and storing.

E.1.8. IDENTIFICATION

Individualized items which are discussed in the previous subsection, require identification when in larger baggage handling systems. This identification is used by the system controllers to divert items in order to complete the routing of items towards their destination. Several ways to identify individual baggage pieces are with:

Barcode A traditional way of identifying baggage is with the use of barcodes printed on the baggage label. These barcodes are assigned to an item by the airline according to IATA standards and may be shared with alliance partners. Reading of barcodes may occur with a standard barcode reader, a camera system and algorithm or a 3D-imaging system. Current developments have led to passengers being able to print their baggage label and barcode at home [41] and a globally unique 10 digit license plate for each baggage item [40]. A disadvantage of barcodes is that they have to be visible in order to scan them and are sometimes folded. A solution to this is the optical character recognition (OCR) of labels, allowing an algorithm to recognize folds and improve positive read percentages.

Radio-frequency identification Further development of identification principles has yielded detection with radio-frequency identification (RFID). These devices may have passive or active implementation. An active device requires a power source, whereas a passive tag is powered by incoming signals. A major advantage of RFID is that tags do not need to be visible. However, they are more expensive than using barcodes and are not an international standard. Still, RFID tags are used in baggage handling systems mainly to identify carriers in an ICS, but other applications are possible.

Pattern recognition A method which does not need a tag is pattern recognition. Due to every bag having a specific colour, size, shape and scratch marks, it is possible to identify a bag by its pattern. A drawback to this method is that the vision system needs to be able to cope with deformation of the item.

These three types of identification may be performed with a computer algorithm. Such algorithms however sometimes do not recognize the tag and require the bag to be identified manually. This is done at a manual encoding station, where an employee enters the barcode or RFID tag in the system.

E.2. CATEGORIZATION OF SOLUTIONS

The previous sections have discussed solutions for baggage systems following a categorization to functions. This is however not a historical approach to partitioning solutions. To compare categorization on throughput and functions, Table 2.2 is devised. The throughput is divided into categories according to IATA standards and elaborated upon in Bradley [25]. The standard does not mention options for the functions: security screening, individualization and identification. The following description is given for each category:

Category A Baggage handling systems in category A are expected to handle less than 1000 bags per peak hour. Recommended equipment for this category is elaborated in Table 2.2. In order to create redundancy for a manual system, a covered hall or apron area should be available of at least twice the size of the sorting area. In the case of automatic sorting, the system is recommended to be capable of processing 50% of the peak baggage flow rate.

Category B A category B system is expected to handle between 1000 and 5000 bags per peak hour. With such a system, IATA recommends the use of automatic sorting systems as described in Table 2.2. The designed system is required to have a redundancy 75% of the peak flow rate.

Category C The largest baggage handling systems are part of category C, with a throughput of at least 5000 bags per peak hour. Such systems contain alternative routes to destinations, making it complex. In case of the primary system failing, the redundant system should be capable of processing at least 75% of peak flow rate

As clear as the boundaries are defined by Bradley [25], applicable situations for real world baggage handling systems are not nearly as well defined [39]. Nevertheless such a categorization is useful to get an initial idea of the situation and provides a rough estimation of the situation. Ultimately the baggage handling stakeholders decide which concept will be implemented. It is notable to mention that not all solutions from section E.1 are mentioned in the recommendation, and that security screening, individualization and identification options are lacking.

F

Morphological overviews

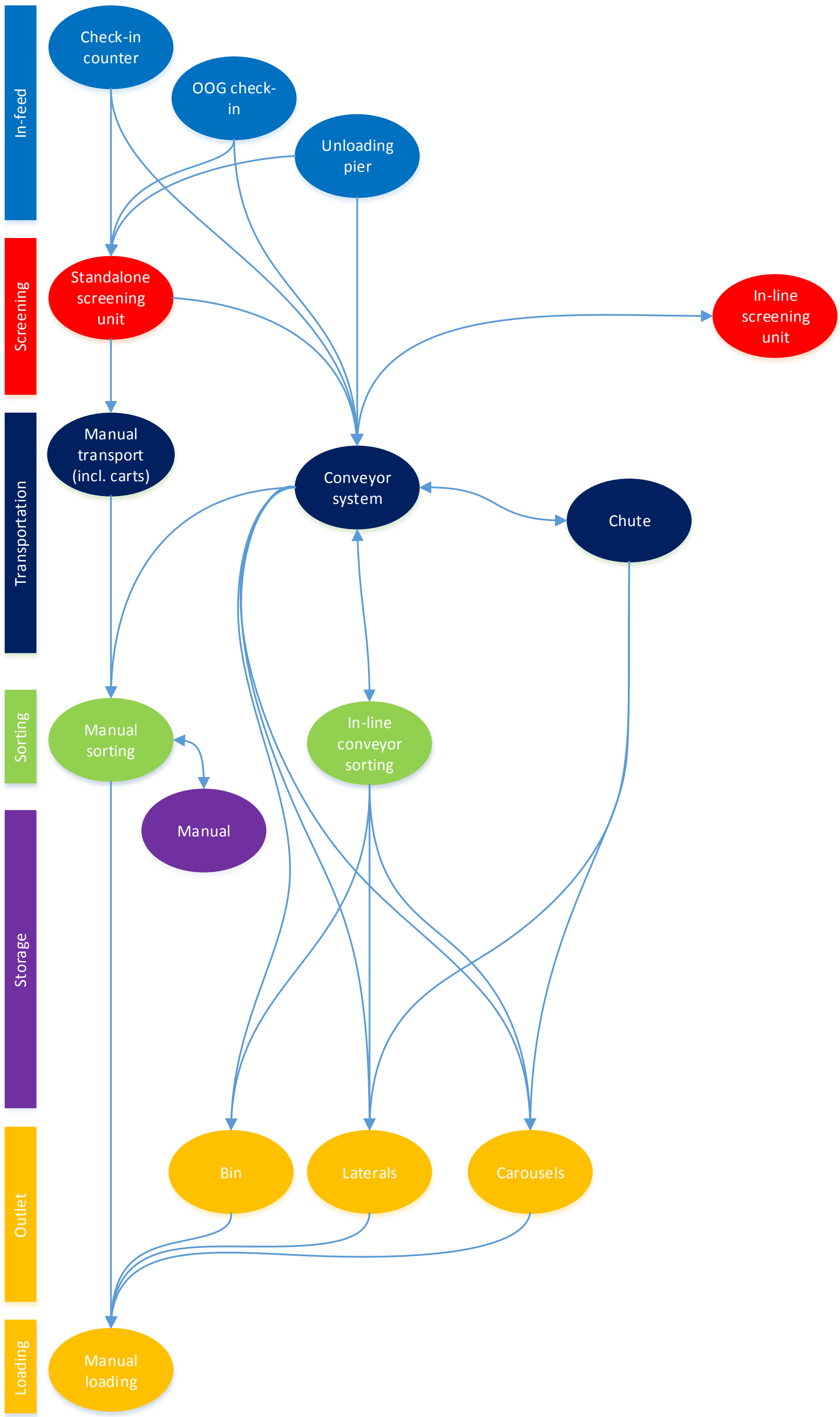


Figure F.1: The morphology for baggage handling systems classified as small by IATA. Solutions that can be created with this diagram may apply to baggage handling systems with a capacity of 0 to 1000 bags per hour, during peak moments at an airport.

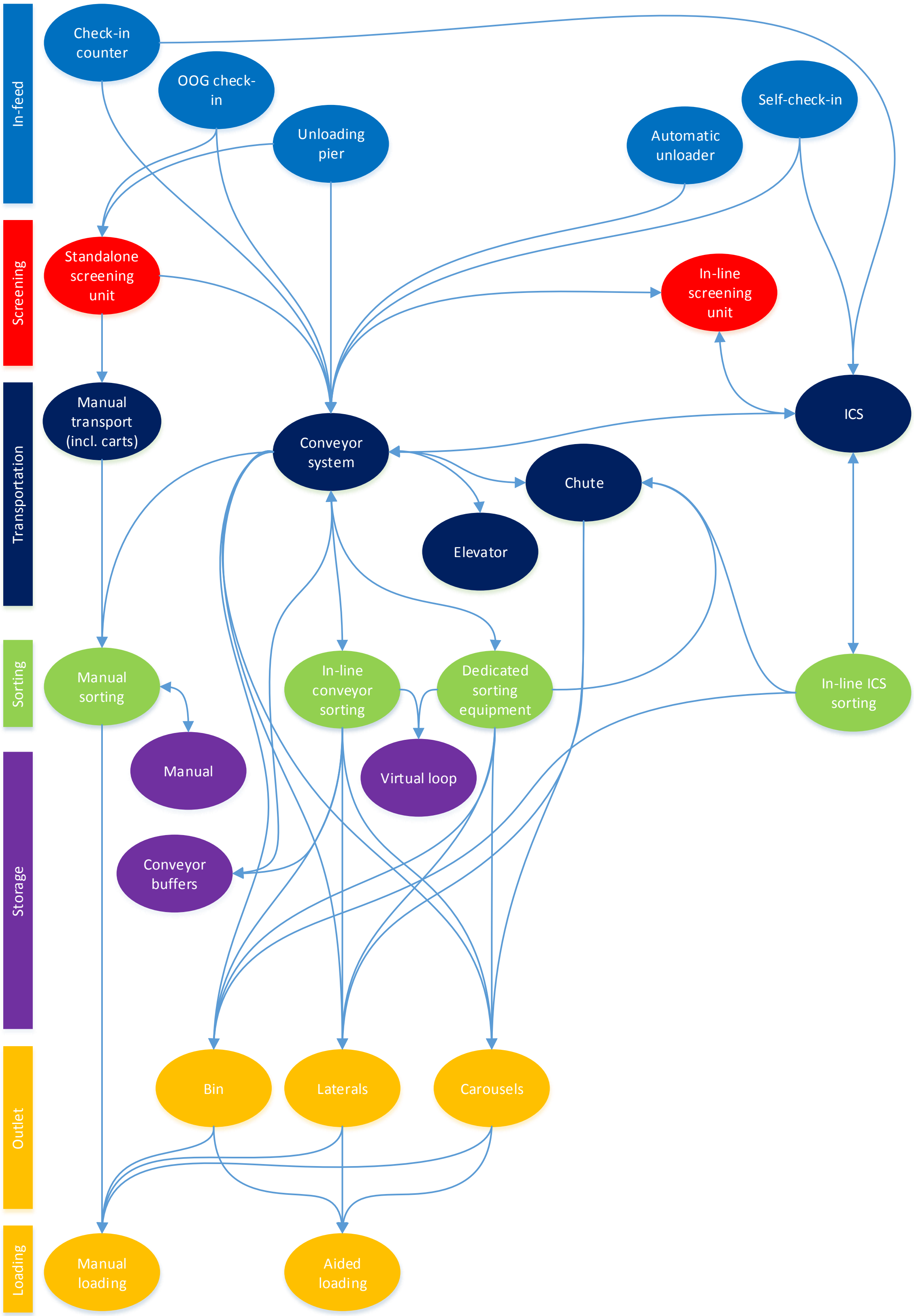


Figure F.2: The morphology for baggage handling systems classified as medium by IATA. Solutions that can be created with this diagram may apply to baggage handling systems with a capacity of 1000 to 5000 bags per hour, during peak moments at an airport.

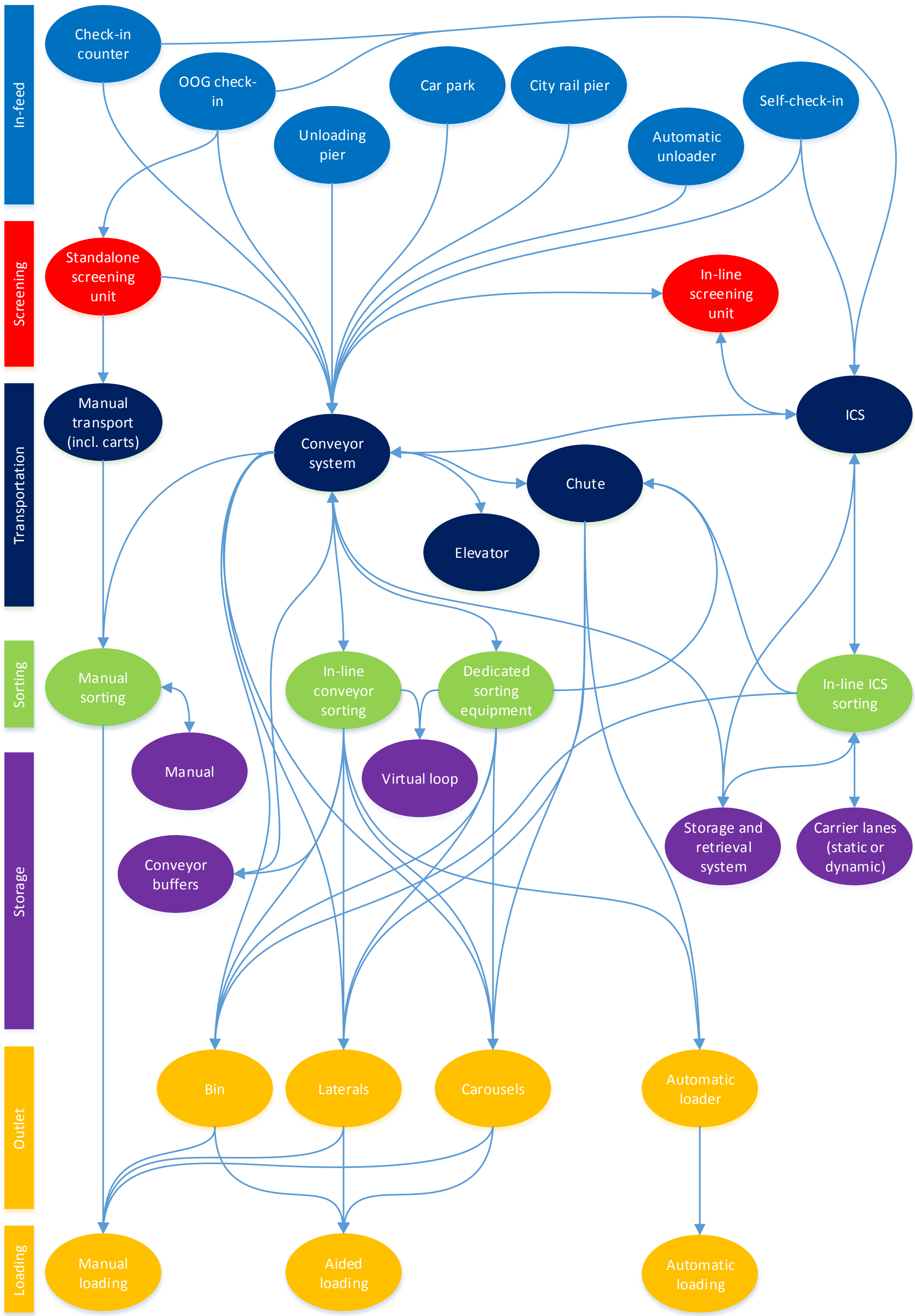


Figure F.3: The morphology for baggage handling systems classified as large by IATA. Solutions that can be created with this diagram may apply to baggage handling systems with a capacity of more than 5000 bags per hour, during peak moments at an airport.

G

DESIGN CASES

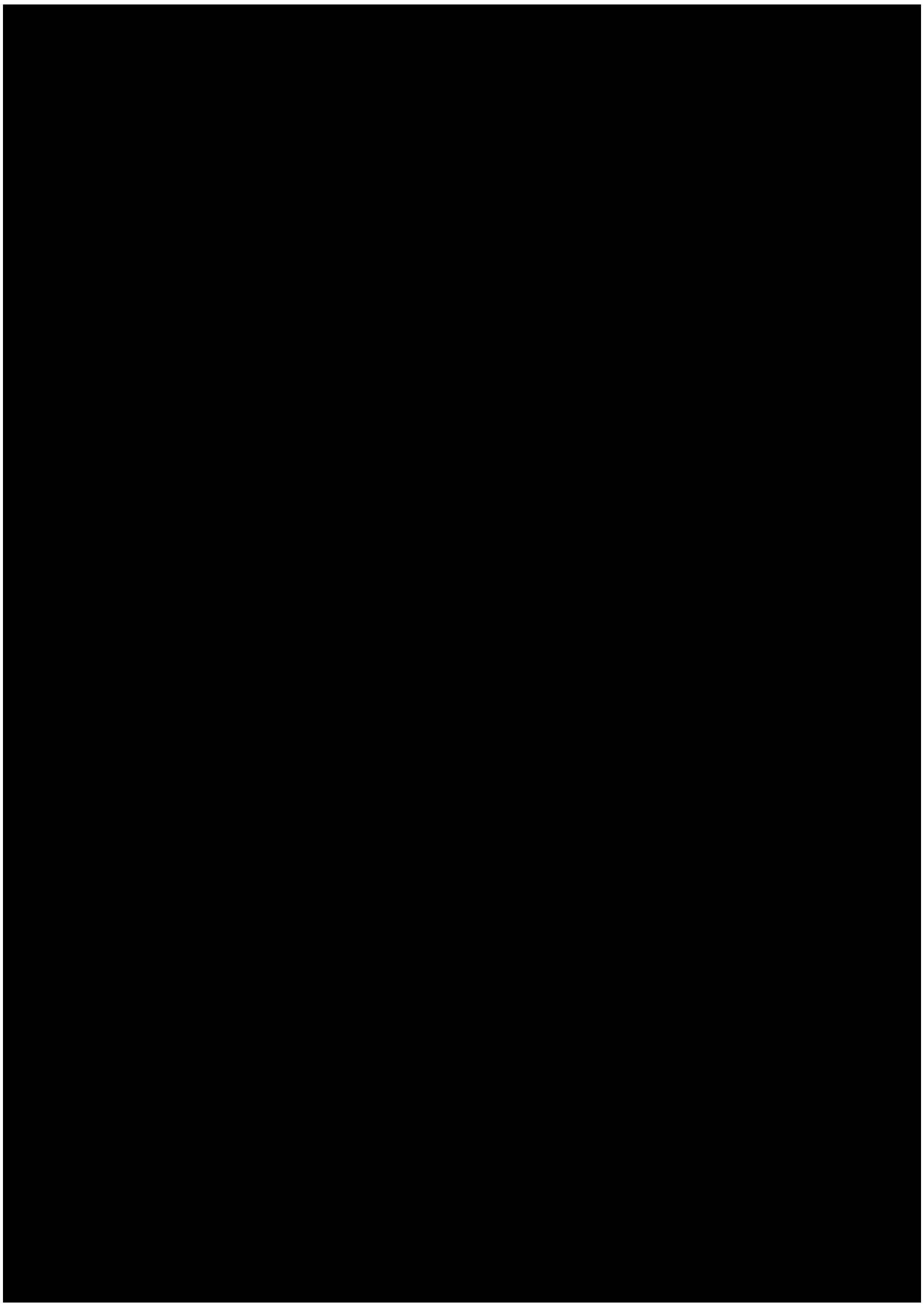


Figure G.1: Design case C1.

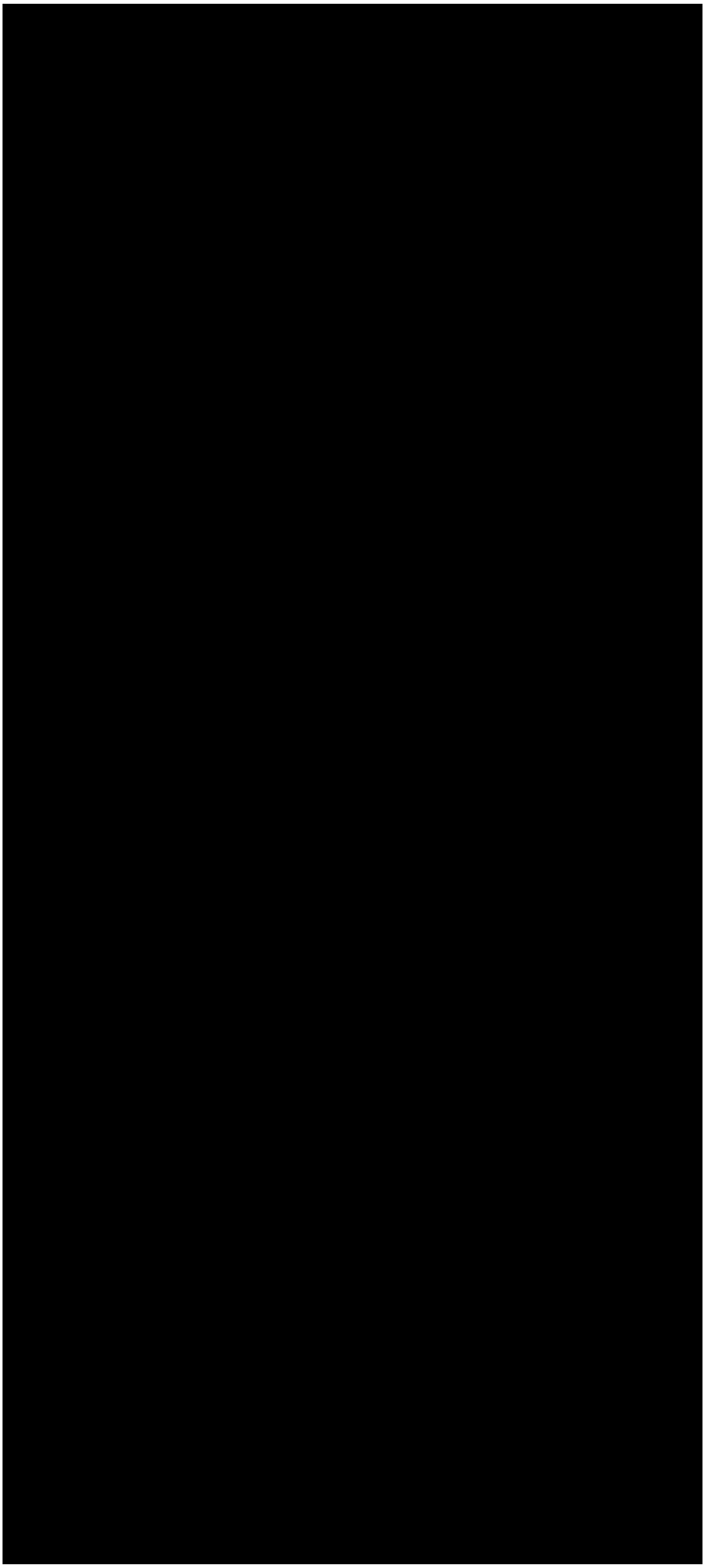


Figure G.2: Design case C2.

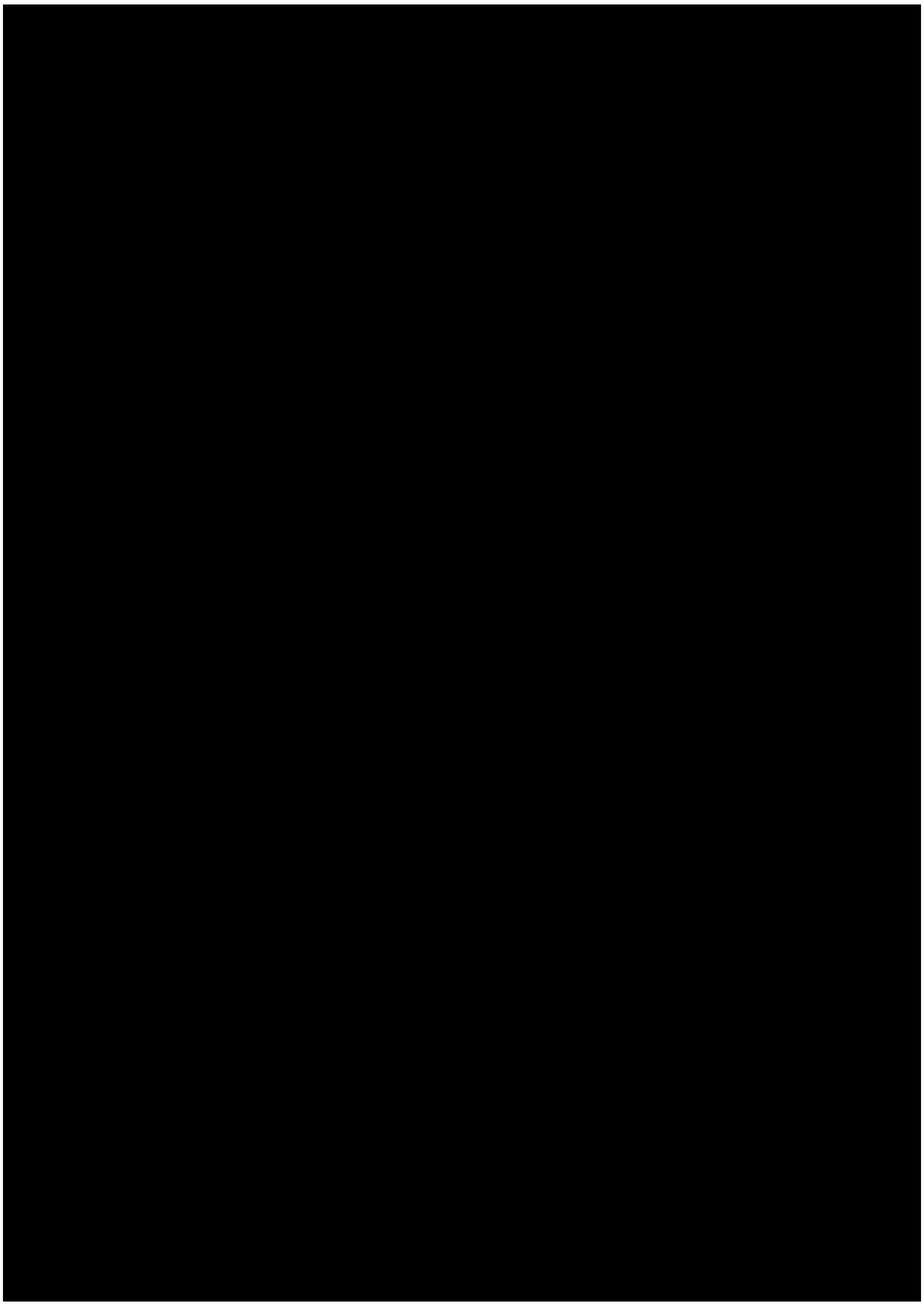


Figure G.3: Design case C3.

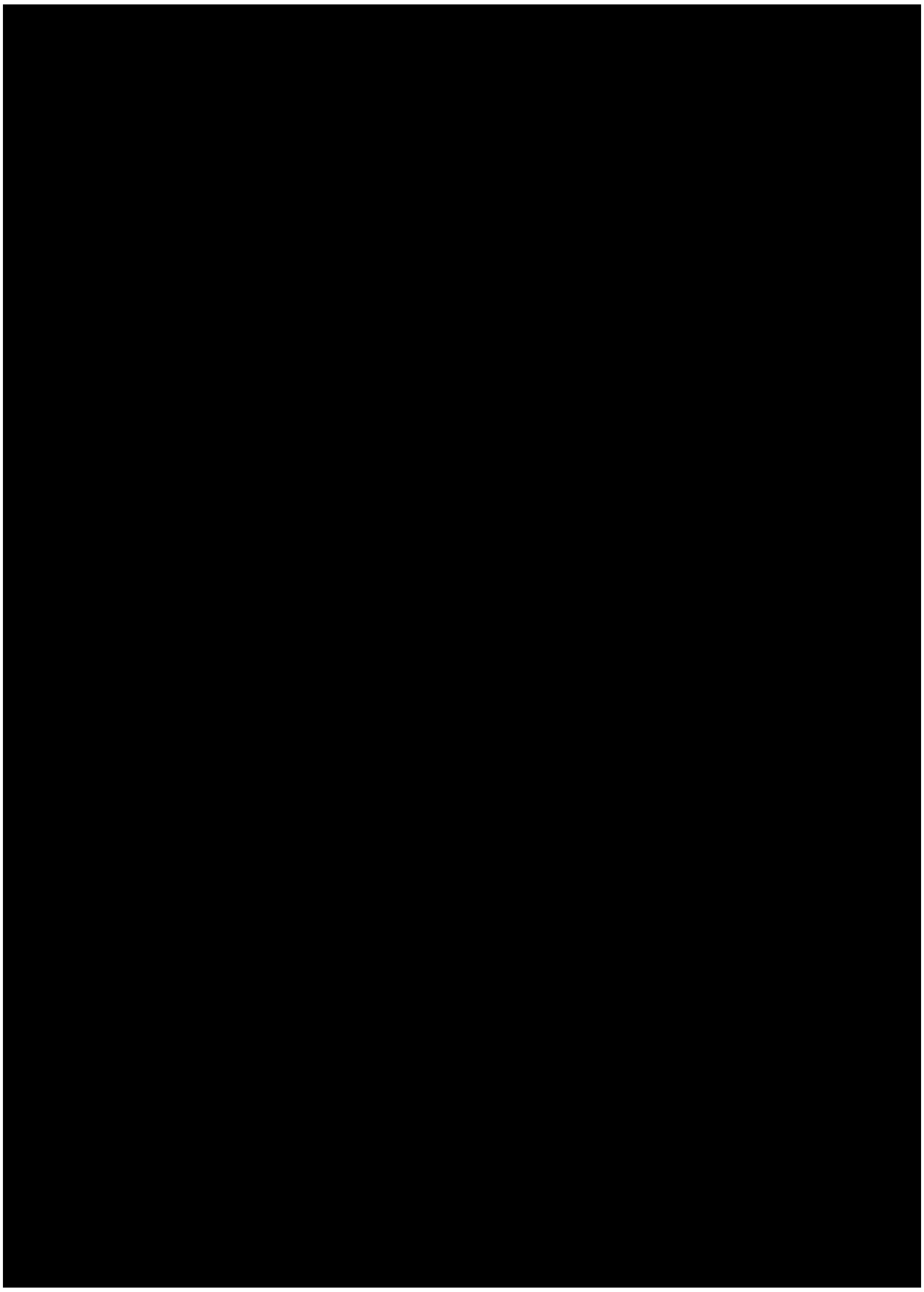


Figure G.4: Design case C4, part 1.

[redacted]

Figure G.5: Design case C4, part 2.

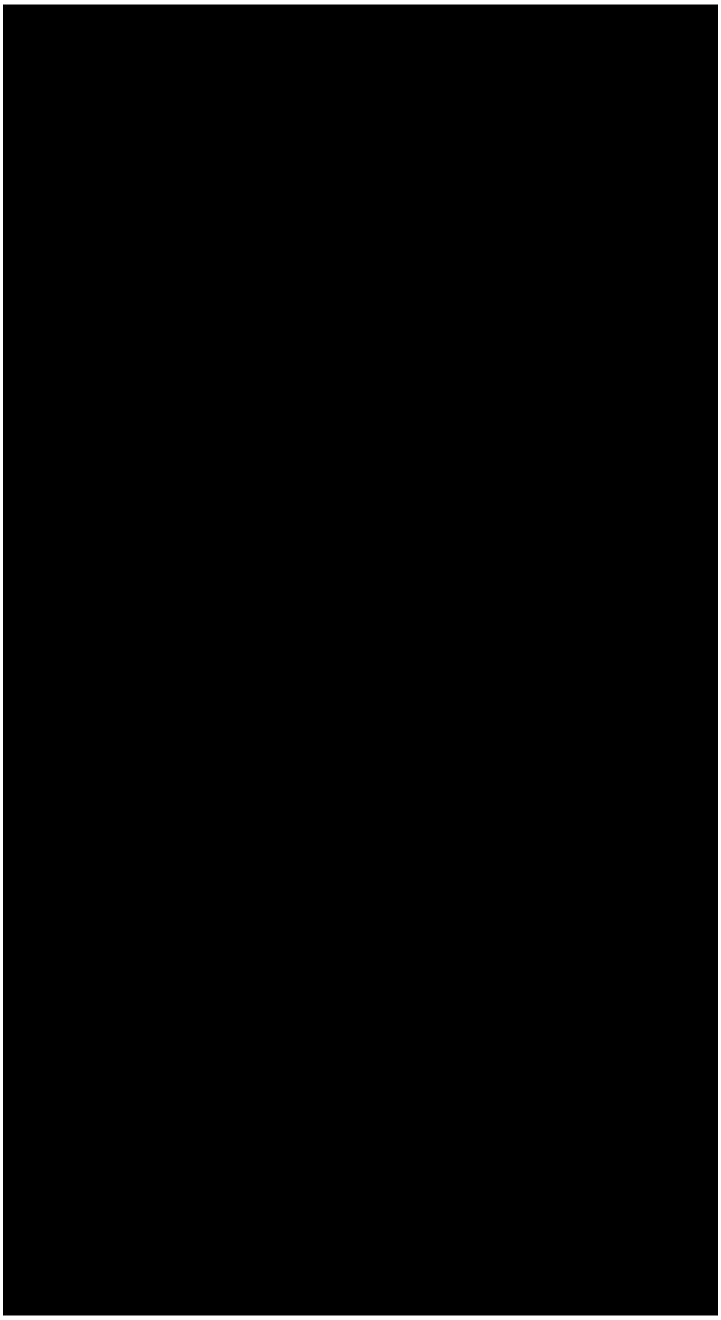


Figure G.6: Design case C5.

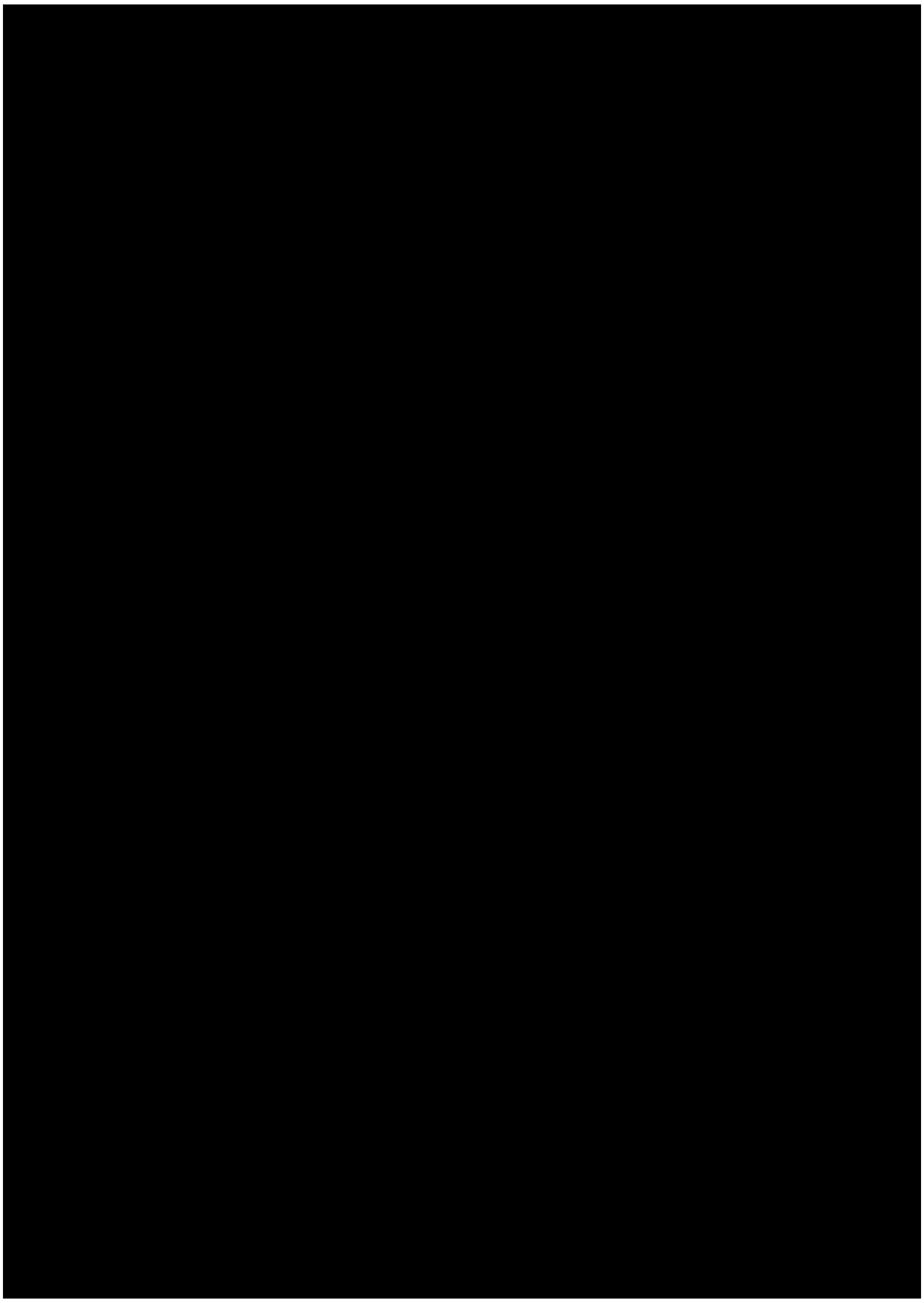


Figure G.7: Design case C6.

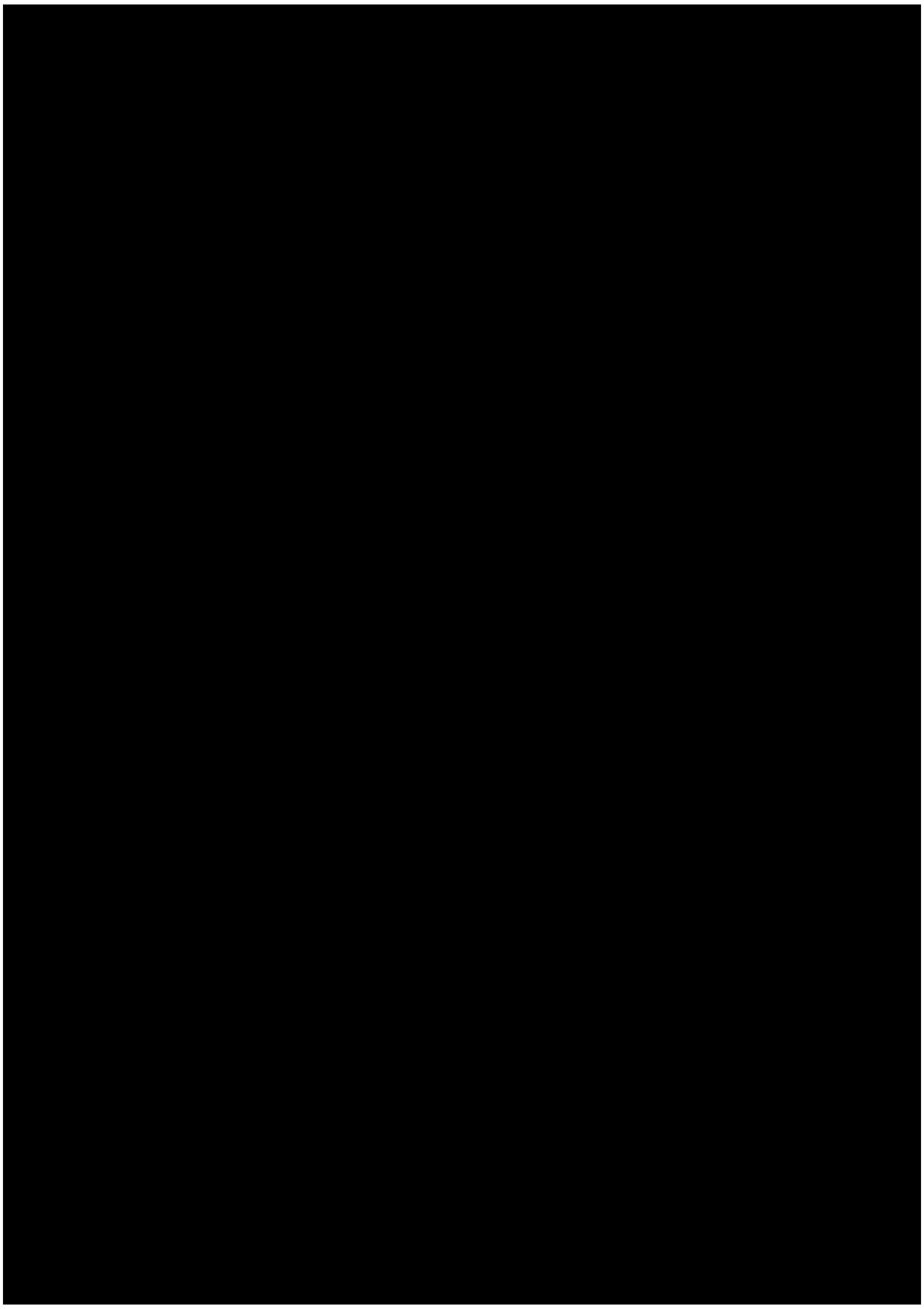


Figure G.8: Design case C7.

H

INSTRUCTIONS

BaggageFlow macro instructions

The following pages will instruct you on how to use the BaggageFlow macro in Microsoft Offfice Excel. These instruction are part of a thesis on baggage handling system (BHS) design and it is advised to fully read the into the macro before applying it in real world cases. This document describes the purpose of the macro, required input data and presentation of output data. Several remarks are made afterwards.

It should be noted that the provided figures for this document are screenshots from a Dutch version of Microsoft Office Excel 2013 and may be different in other versions of the program. Due to the nature of Excel, the macro is backwards compatible with older versions of Excel and all language plugins.

Purpose

The aim of this macro is to assist system architects in the conceptual design of baggage handling systems, as these systems may become complex for larger airports. Architects are assisted by showing several design options to different functions and accompanying performances in the system. From these key performance indicators (KPIs), decisions may be made on what to include in the concept design. The analytic hierarchy process is proposed for making these decisions. The basis for using this method and functional division are described in the thesis and is therefore an important read before using Pielage's design method and this macro in determining concepts.

Starting the macro

Before opening the file, it is advised to store the original document in a separate folder or location on a hard drive and create a renamed copy of this file to work in. This ensures that if the macro is accidentally changed by a user, a working back-up is still available.

The file consists of several tabs with specific inputs and outputs and a macro. It is required to enable this macro in Excel before it can be used. Excel blocks macros to protect users from malicious content. This safeguard should only be enabled when the file is obtained from a trusted source. Opening the file for the first time shows a prompt in which the macro may be enabled (Figure 1). This prompt will then disappear. When the message of Figure 2 is shown after clicking the "Perform calculation" button in Figure 3, the macro is not enabled. A new prompt can be obtained by saving your progress, closing the file and opening it again.

Macro input data

Figure 1 and Figure 3 show two of the three input tabs for the macro. The first figure depicts how the flight schedule may be entered. Each line represents a new flight in the flight schedule. For each flight, several properties must be added for the program to function properly. A close up of the required data is given in Figure 4. Aircraft data is represented by: airline, airline type, flight number, flight character and capacity. The macro requires the flight character and capacity to be filled in. The other entries for aircraft data are optional. The airline name is used to determine how many baggage drop-offs occur for that airline and how many counters are needed. The airline type and flight number are not used in the calculation but serve as a reference when adding extra flights. The flight character governs how baggage is handled, represented by data in Figure 3. Capacity represents the available seats in the aircraft.

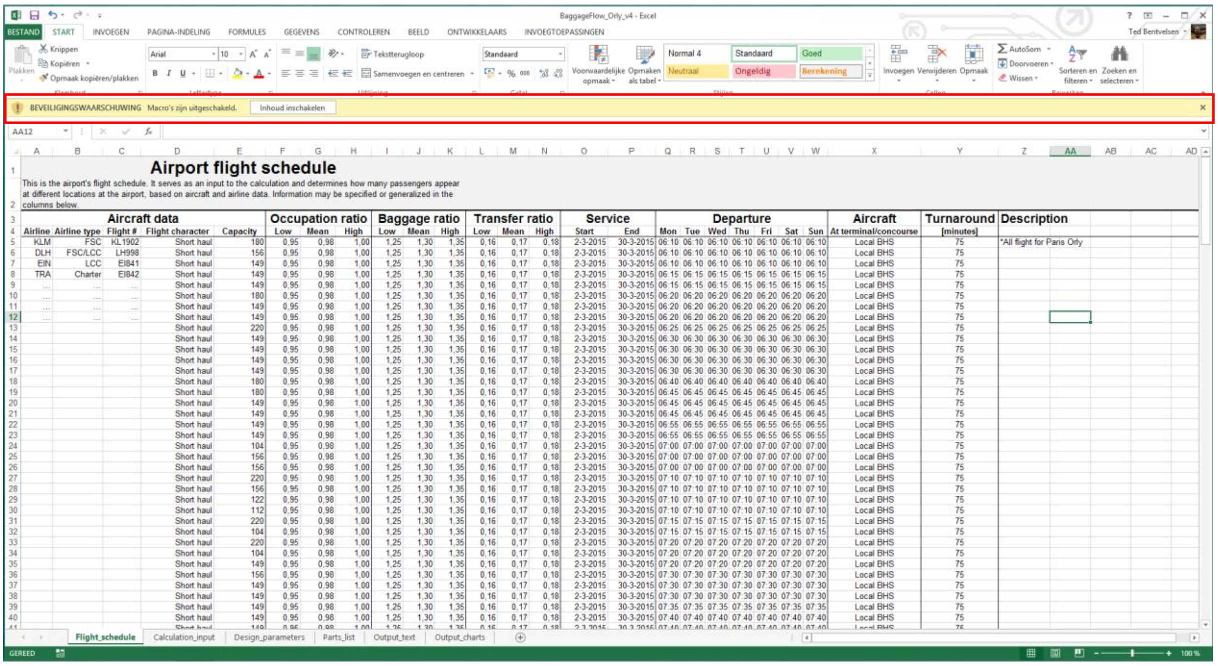


Figure 1 - First opening of the BaggageFlow Excel file. Enabling macros is marked in red.

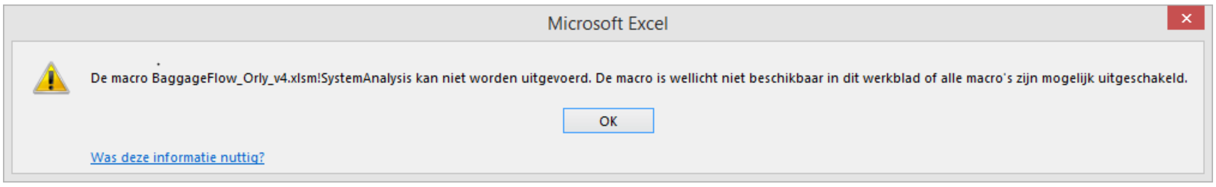


Figure 2 - Message occurring when the "perform calculation" button is clicked but macros aren't enabled in the file.

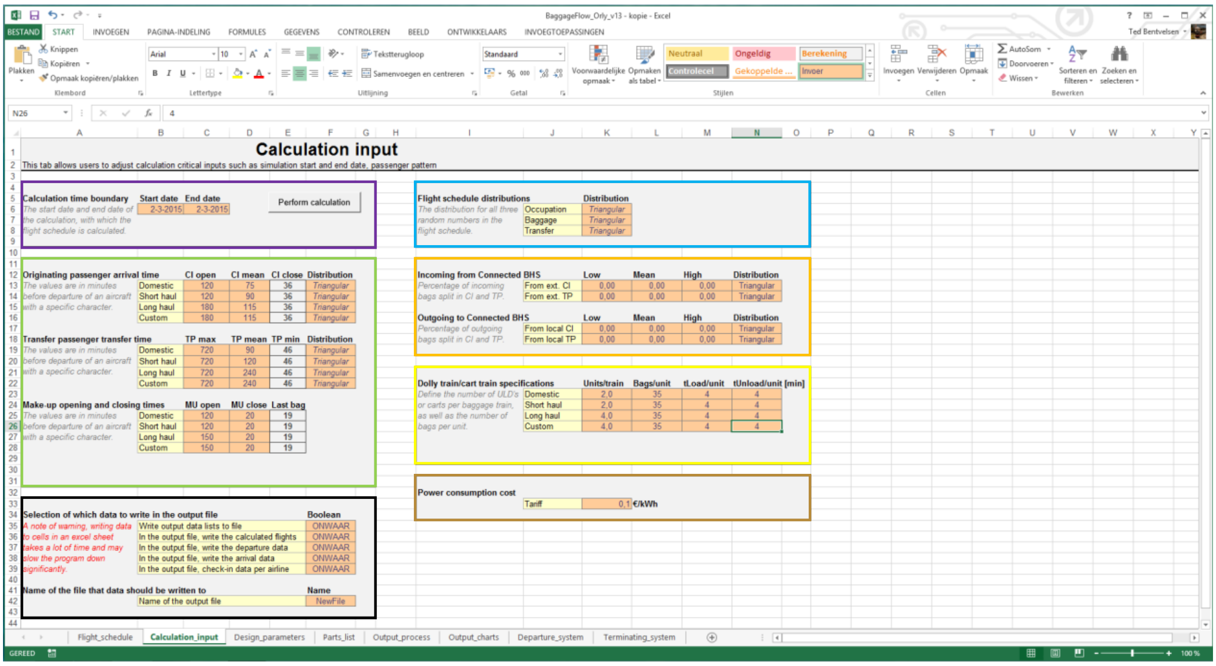


Figure 3 - Additional inputs for the model that allow for specifying the handling of baggage. From this tab, the "perform calculation" button (marked purple) may be clicked to start the simulation and calculation.

Airport flight schedule

This is the airport's flight schedule. It serves as an input to the calculation and determines how many passengers appear at different locations at the airport, based on aircraft and airline data. Information may be specified or generalized in the columns below.

	Occupation ratio			Baggage ratio			Transfer ratio		
	Low	Mean	High	Low	Mean	High	Low	Mean	High
	0,80	0,90	1,00	1,00	1,20	1,45	0,15	0,20	0,22
	Apply to all			Apply to all			Apply to all		

Aircraft data					Occupation ratio			Baggage ratio			Transfer ratio			Service		Departure							Aircraft	Turnaround	Description
Airline	Airline type	Flight #	Flight character	Capacity	Low	Mean	High	Low	Mean	High	Low	Mean	High	Start	End	Mon	Tue	Wed	Thu	Fri	Sat	Sun	At terminal/concourse	[minutes]	
AA	LCC	AA0000	Domestic	95	0,80	0,90	1,00	0,80	0,95	1,00	0,00	0,00	0,00	1-1-2016	1-8-2016	09:15		09:15		09:15	09:15		Local BHS	60	*Example flight
AA	LCC	AA0001	Domestic	95	0,80	0,90	1,00	0,80	0,95	1,00	0,00	0,00	0,00	1-1-2016	1-8-2016	18:25		18:25		18:25	18:25		Local BHS	60	*Example flight
AB	FSC	AB0000	Short haul	95	0,75	0,80	1,00	1,00	1,20	1,45	0,15	0,20	0,22	1-1-2016	1-8-2016	10:30	10:30	10:30	10:30	10:30	10:30	10:30	Local BHS	60	*Example flight
AB	FSC	AB0001	Short haul	95	0,75	0,80	1,00	1,00	1,20	1,45	0,15	0,20	0,22	1-1-2016	1-8-2016	10:35	10:35	10:35	10:35	10:35	10:35	10:35	Local BHS	60	*Example flight
AB	FSC	AB0002	Short haul	180	0,80	0,95	1,00	1,00	1,20	1,45	0,15	0,20	0,22	1-1-2016	1-8-2016	21:55	21:55			21:55	21:55	21:55	Local BHS	60	*Example flight
AB	FSC	AB0003	Long haul	180	0,80	0,90	1,00	1,00	1,20	1,45	0,15	0,20	0,22	1-1-2016	1-8-2016	08:00		08:00		08:00		08:00	Local BHS	75	*Example flight
AB	FSC	AB0004	Long haul	250	0,80	0,90	1,00	1,00	1,20	1,45	0,15	0,20	0,22	1-1-2016	1-8-2016	13:25	13:25	13:25	13:25	13:25	13:25	13:25	Connected BHS	75	*Example flight
AC	FSC	AC0000	Custom	250	0,70	0,90	1,00	1,00	1,20	1,45	0,15	0,20	0,22	1-1-2016	1-8-2016	15:00	15:00		15:00	15:00	15:00		Local BHS	75	*Example flight

Figure 4 - A close up of the required flight schedule data.

In the flight schedule, several ratios are required. These are occupation ratio, baggage ratio and transfer ratio. The occupation ratio represents how many of the available seats are filled and should be between 0 and 1. How many bags each passenger has is governed by the baggage ratio, which should be at higher than 0. The transfer ratio determines how many seats in a departing aircraft are occupied by transferring passengers and should be between 0 and 1. All ratios are represented by a low, mean and high value. These may represent different values for different distributions:

Distribution	Low	Mean	High
Constant	-	Constant value	-
Uniform	Lower bound value	-	Upper bound value
Triangular	Lower bound value	Most likely value	Upper bound value
Normal	-	Mean	Variance
Custom	-	-	-

Additional distribution types may be added later on via custom entries. Changing these distributions for all three ratios can be done in the blue marked area in Figure 3 and holds for the entire flight schedule. By filling in the service start date and end date, the days in which an aircraft is operational may be selected. This is supplemented by adding or leaving out the departure times for each weekday. Taking the first line of Figure 4 as an example, it may be seen that aircraft AA0000 starts service at 1-1-2016 and ends its service at 1-8-2016. Between these two dates, it departs at 09:15 on Mondays, Wednesdays, Fridays and Saturdays. To allow for determining baggage flows on incoming and outgoing connections of the BHS, it may be selected whether the aircraft is serviced by the local BHS or connected BHS. The turnaround time of 60 minutes represents the arrival time of the aircraft at the gate in will create an influx of baggage into the system for terminating or transferring passengers. Recall that the transfer passengers are determined for departing aircraft. In order to correctly simulate their arrival, their arrival times are matched with arriving aircraft. Transfer passengers that cannot find be matched with arriving aircraft are disregarded by the simulation.

Further specification of passenger behaviour and baggage handling properties may be done in the "Calculation_input" tab. The simulation start date and end date must be provided. Aircraft within this range are taken into account by the simulation. Simulating 230 aircraft per day for a year has resulted in a calculation time of 18 minutes on an Intel i5-5200U processor with 8GB RAM available when all background processes were minimised. It should be noted that this calculation time scales linearly with the amount of simulation days and that different computers have different processing speeds. A large amount of random access memory (RAM) is required when simulating many aircraft.

For each of the four flight characters entered in the previous tab, check-in windows, transfer windows and make-up windows may be entered at the green marked area in Figure 3. The check-in and transfer opening and mean times may be entered here and should be rounded to units. The opening time should also be larger than the mean time, and mean time should be larger than the closing time. Passengers will arrive within this window according to the selected distribution. The minimum connecting time (MCT) for transfers and check-in close times are determined based on the transport times that may be changed in the next tab: “Design_parameters”, depicted in Figure 5. The make-up open and closing time may also be entered and should be larger than the last bag time. This last bag time is also determined with information from the next tab.

Information on the previously described incoming and outgoing connections may be given in the orange marked area. These values are distributed according to the previously described distributions. For both the incoming and the outgoing connections, baggage is divided into originating and transfer baggage. Examples are incoming baggage from external system check-ins or transfer piers and outgoing baggage from local check-ins or transfer piers. A low, mean and high value must be entered and aircraft need to be directed to the connected system in the “Flight_schedule” tab. When this is not done, no incoming and outgoing bags will be observed.

The yellow marked area in Figure 3 represents the baggage train specifications. For each flight character, the amount of units per train, items per unit, aircraft loading and unloading time may be entered. The energy costs per kilowatt hour may be entered in the brown marked area. As per request, it is possible to have the macro output a list of all simulated flights with data, departure data, arrival data and data per airline. This may be selected in the black marked area.

The final input tab “Design_parameters” is depicted in Figure 5. In this tab, transport time and distance may be entered per flow in the red marked area. A depiction of the model is given as a reference for each flow. The distance and time are used to calculate the average transport velocity for that flow. This should be taken into account when selecting a transport medium. The orange marked area may be used to change the formatting of results and charts. As different designers prefer various denotations, bags per minute as opposed to bags per hour, they may be changed here. The amount of days represented in charts may also be changed, at the cost of axis resolution. Design factors may be adjusted in the yellow marked area. The blue marked area denotes function process times in minutes. Splits in flows may be changed in the green marked area, based on percentages from 0% to 100%. A Boolean is used to change the setting of storing outgoing baggage in the local storage facilities in the purple marked area. Make-up loading speeds may be adjusted in the black marked area.

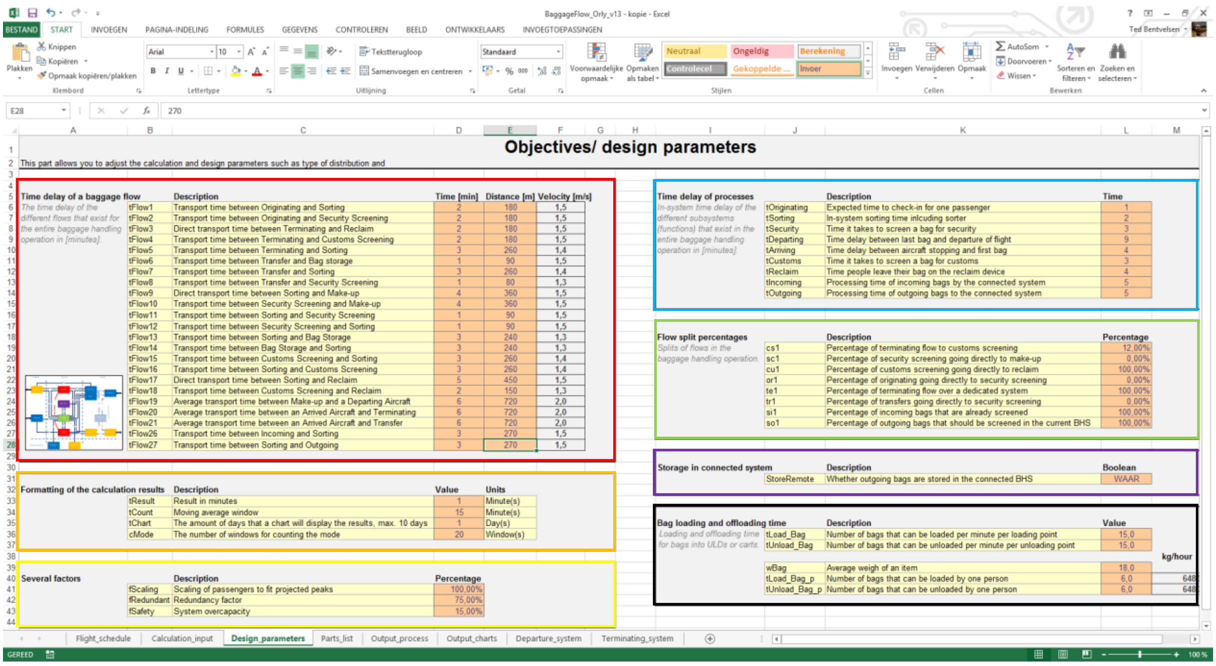


Figure 5 - Design parameters tab of the excel macro.

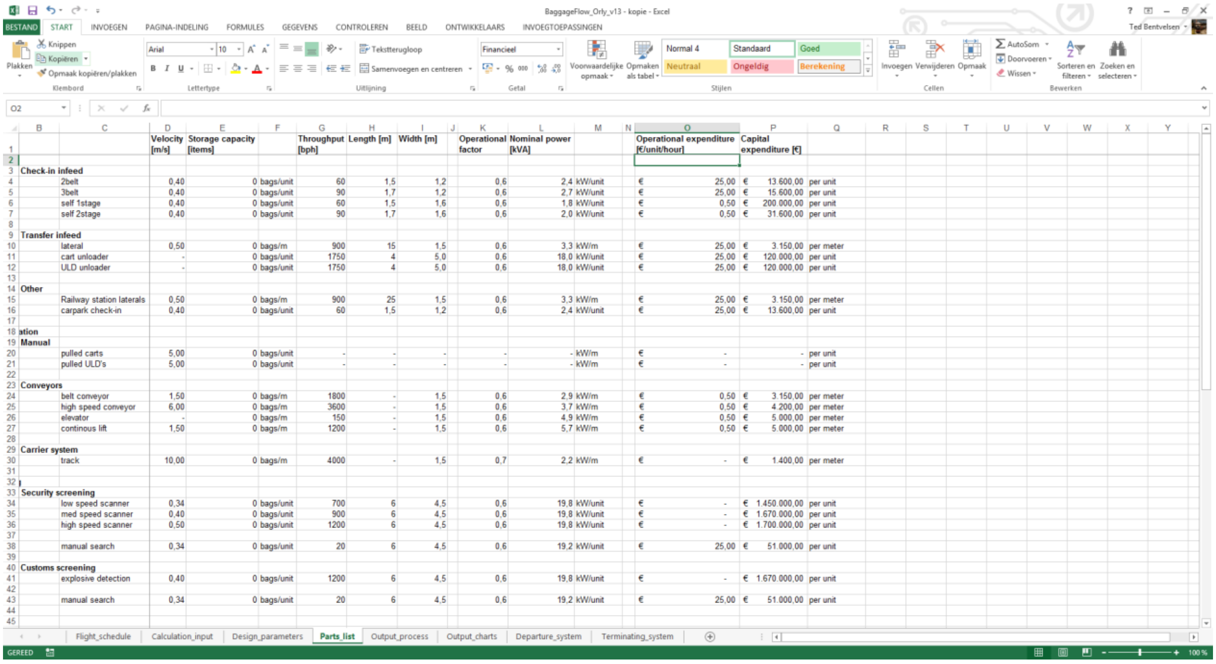


Figure 6 - Parts list tab for the Excel macro. This does not need to be changed unless properties of systems have changed.

Standardised parts and their properties are given in the “Parts_list” tab. This tab is depicted in Figure 6. Parts are subdivided into categories. For each part, velocity, storage capacity, throughput, length, width, operational factor, nominal power, operational expenditure and capital expenditure are given. For length based systems, these values are given per meter. Unit based systems are given per unit. In these values, drive, power and control systems are included.

Macro output data

After changing all properties into desired values, the macro may be activated by pressing the “Perform calculation” button as depicted in Figure 3. As was mentioned it may take some time to complete the simulation and calculation, as numerous aircraft and baggage items have to be taken into account. During the simulation and calculation, Excel may tell you it is “Not responding”. This indicates that the simulation and calculation are still running. The macro is protected against infinite looping, so it will eventually show results or encounter an error. Errors are always shown in prompts and will end the calculation.

After successfully simulating, the output data in the output tabs have changed. The first available tab is “Output_process” in which the process model is depicted. This may be seen in Figure 7. For each process and flow, an average and maximum flow is given. For accumulating functions, storage positions are also depicted. Charts of these functions are depicted in the “Output_charts” tab of the file. This is depicted in Figure 8 and Figure 9. Among these charts are also the amount of stored carts and ULD’s on the apron as well as a chart with check-ins per airline.

The presented data is used to determine the KPIs of subsystems for each function and flow. These may be seen in Figure 10 and Figure 11, which show the departure and terminating system respectively. Transfers are included in the departure system. Tabs “Departure_system” and “Terminating_system” may be used and are separated to maintain overview. The KPIs are presented for average, 95% and maximum throughput. Sampling check-ins shows that 95% is reached by installing 9 two-belt check-ins, 6 three-belt check-ins, 9 one-stage drop-off or 6 two-stage drop-off. Trade-offs can now be made.

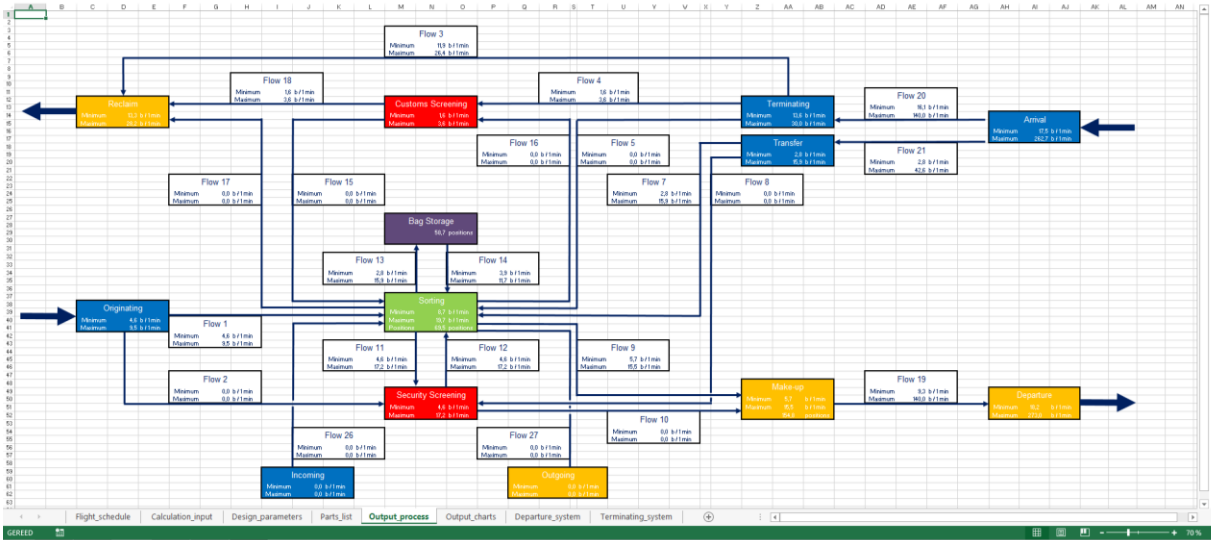


Figure 7 - Output process tab of the Excel macro containing average and maximum throughput per function and flow.

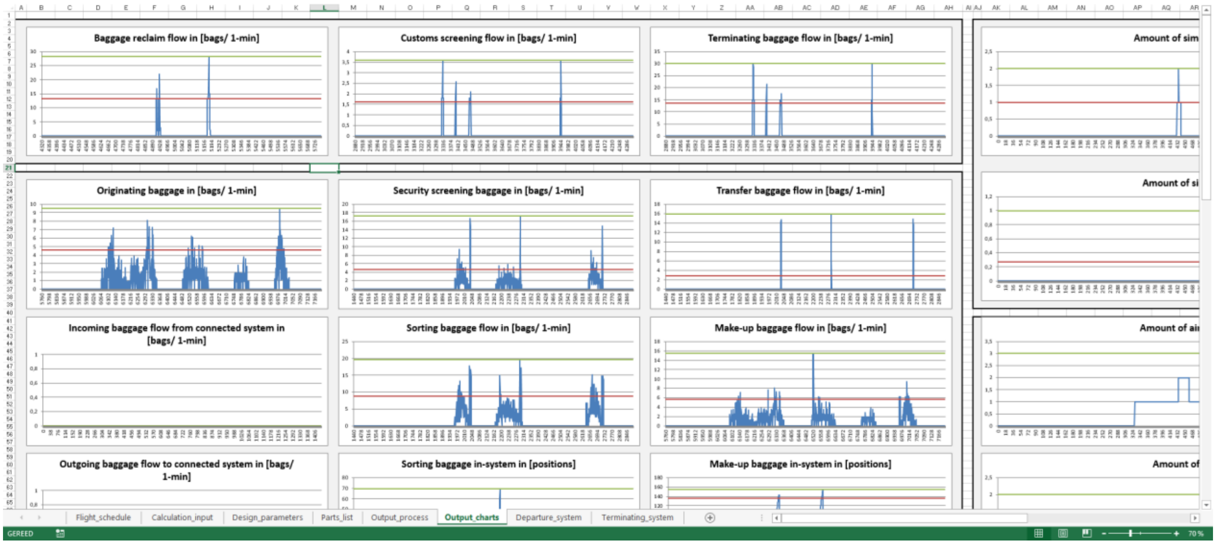


Figure 8 - Several charts depicting the day with highest peak throughput for each function. Different days may be depicted.

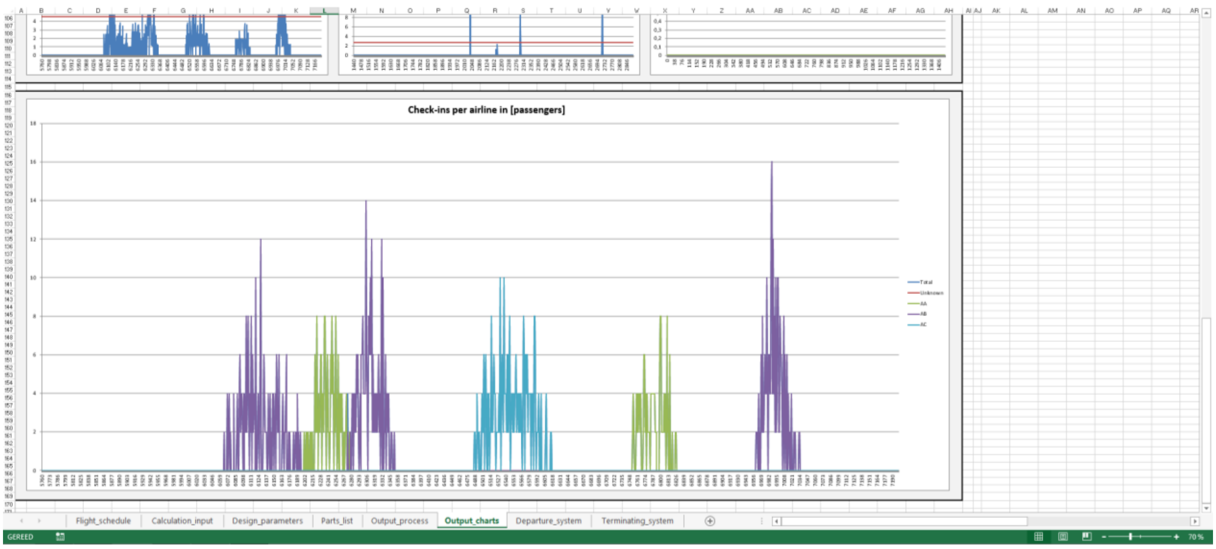


Figure 9 – Additional chart shown in the output charts tab.

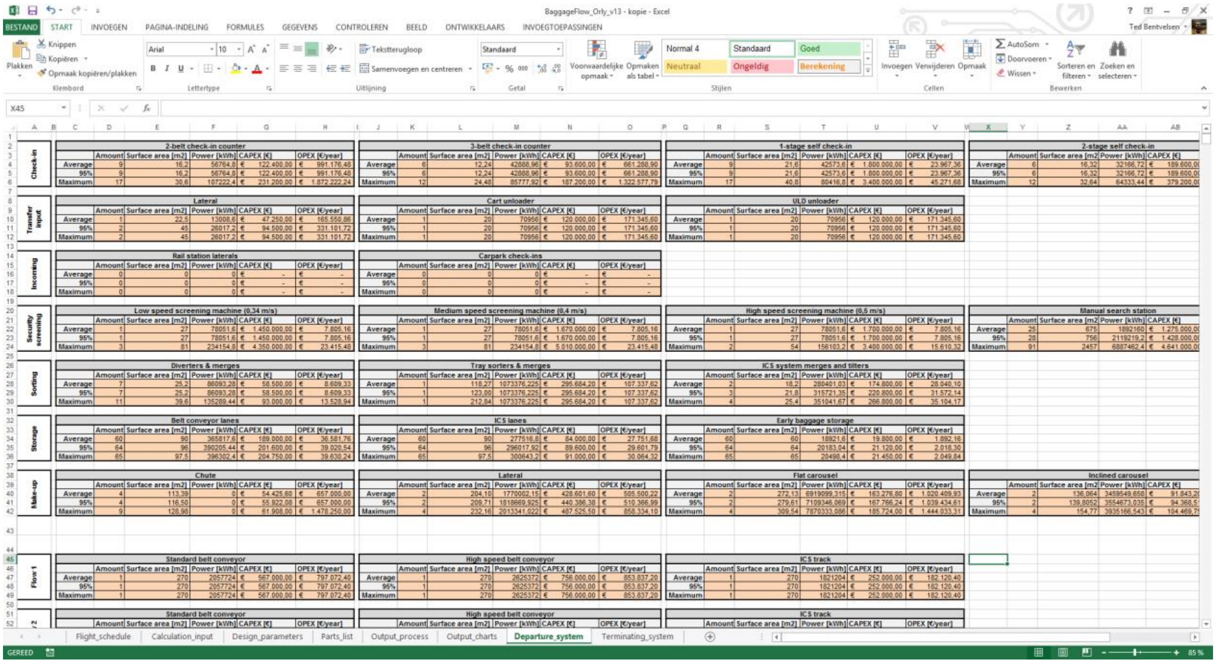


Figure 10 - Departure system calculation outputs.

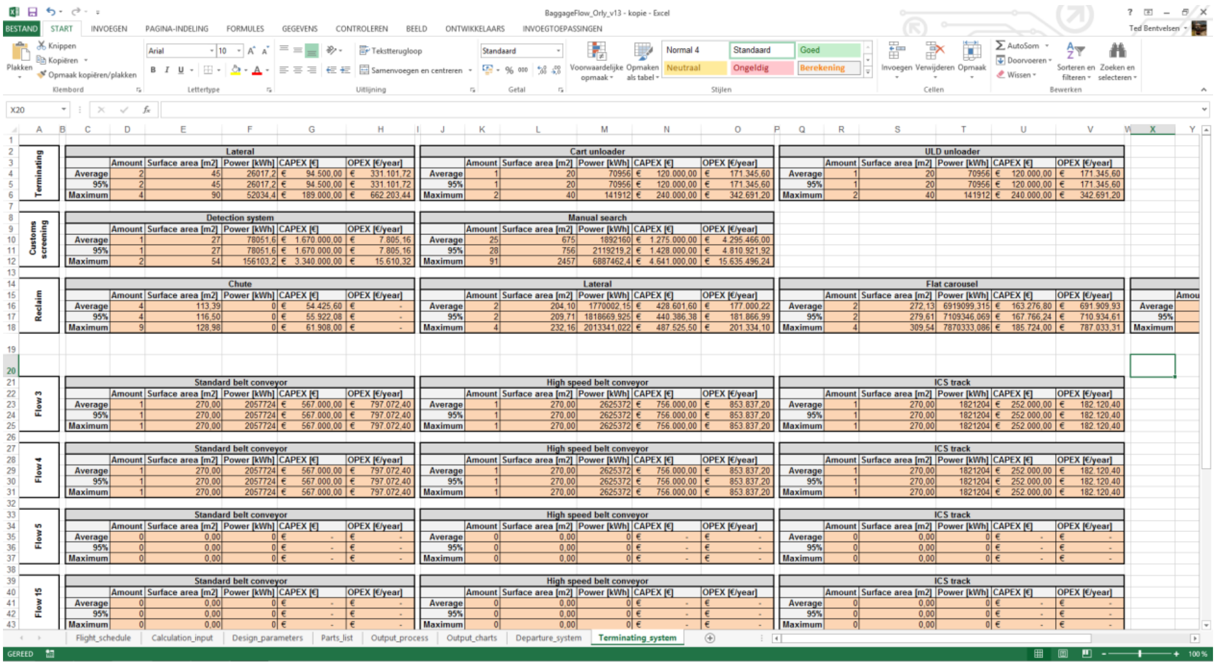


Figure 11 - Terminating system calculation outputs.

I

EXPERT CRITERIA RANKINGS

Table I.1: A table containing the results to the pairwise comparison of criteria by XXXX. The consistency ratio of this comparison is 0.26.

	AL	CO	FL	IN	IT	LC	PD	PI	RA	SI	DS
AL	1,00	1,00	1,00	1,00	0,14	0,33	0,20	0,33	0,20	0,14	0,14
CO	1,00	1,00	1,00	1,00	0,20	1,00	0,20	0,14	1,00	1,00	1,00
FL	1,00	1,00	1,00	1,00	1,00	1,00	0,20	0,20	3,00	3,00	0,20
IN	1,00	1,00	1,00	1,00	1,00	5,00	1,00	0,33	1,00	3,00	1,00
IT	7,00	5,00	1,00	1,00	1,00	5,00	5,00	0,20	0,33	0,20	0,20
LC	3,00	1,00	1,00	0,20	0,20	1,00	1,00	0,20	3,00	0,20	0,20
PD	5,00	5,00	5,00	1,00	0,20	1,00	1,00	0,20	1,00	0,20	1,00
PI	3,00	7,00	5,00	3,00	5,00	5,00	5,00	1,00	5,00	5,00	1,00
RA	5,00	1,00	0,33	1,00	3,00	0,33	1,00	0,20	1,00	1,00	0,20
SI	7,00	1,00	0,33	0,33	5,00	5,00	5,00	0,20	1,00	1,00	1,00
DS	7,00	1,00	5,00	1,00	5,00	5,00	1,00	1,00	5,00	1,00	1,00

Table I.2: A table containing the results to the pairwise comparison of criteria by XXXX. The consistency ratio of this comparison is 0.20.

	AL	CO	FL	IN	IT	LC	PD	PI	RA	SI	DS
AL	1,00	0,14	1,00	3,00	0,14	0,20	1,00	0,14	1,00	3,00	0,20
CO	7,00	1,00	7,00	3,00	1,00	3,00	1,00	1,00	1,00	5,00	3,00
FL	1,00	0,14	1,00	0,33	0,33	1,00	1,00	0,14	1,00	1,00	4,00
IN	0,33	0,33	3,00	1,00	0,33	3,00	1,00	0,14	1,00	5,00	5,00
IT	7,00	1,00	3,00	3,00	1,00	3,00	1,00	1,00	1,00	3,00	3,00
LC	5,00	0,33	1,00	0,33	0,33	1,00	1,00	0,33	1,00	1,00	3,00
PD	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
PI	7,00	1,00	7,00	7,00	1,00	3,00	1,00	1,00	1,00	5,00	5,00
RA	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
SI	0,33	0,20	1,00	0,20	0,33	1,00	1,00	0,20	1,00	1,00	5,00
DS	5,00	0,33	0,25	0,20	0,33	0,33	1,00	0,20	1,00	0,20	1,00

Table I.3: A table containing the results to the pairwise comparison of criteria by XXXX. The consistency ratio of this comparison is 0.13.

	AL	CO	FL	IN	IT	LC	PD	PI	RA	SI	DS
AL	1,00	2,00	0,20	3,00	2,00	1,00	0,33	0,20	0,33	1,00	0,50
CO	0,50	1,00	1,00	5,00	0,33	2,00	1,00	0,20	1,00	1,00	0,25
FL	5,00	1,00	1,00	5,00	1,00	1,00	1,00	0,20	0,33	3,00	1,00
IN	0,33	0,20	0,20	1,00	0,33	3,00	0,20	0,17	0,20	0,33	0,25
IT	0,50	3,00	1,00	3,00	1,00	4,00	1,00	1,00	0,20	1,00	1,00
LC	1,00	0,50	1,00	0,33	0,25	1,00	0,25	0,17	0,33	1,00	0,25
PD	3,00	1,00	1,00	5,00	1,00	4,00	1,00	0,17	1,00	3,00	1,00
PI	5,00	5,00	5,00	6,00	1,00	6,00	6,00	1,00	1,00	1,00	3,00
RA	3,00	1,00	3,00	5,00	5,00	3,00	1,00	1,00	1,00	3,00	1,00
SI	1,00	1,00	0,33	3,00	1,00	1,00	0,33	1,00	0,33	1,00	0,33
DS	2,00	4,00	1,00	4,00	1,00	4,00	1,00	0,33	1,00	3,00	1,00

Table I.4: A table containing the results to the pairwise comparison of criteria by XXXX. The consistency ratio of this comparison is 0.14.

	AL	CO	FL	IN	IT	LC	PD	PI	RA	SI	DS
AL	1,00	0,14	0,33	1,00	1,00	0,33	3,00	0,33	1,00	1,00	3,00
CO	7,00	1,00	3,00	3,00	3,00	3,00	5,00	0,33	3,00	3,00	3,00
FL	3,00	0,33	1,00	3,00	1,00	0,33	3,00	0,20	1,00	1,00	1,00
IN	1,00	0,33	0,33	1,00	0,33	0,33	1,00	0,33	3,00	3,00	0,33
IT	1,00	0,33	1,00	3,00	1,00	0,33	3,00	0,33	3,00	3,00	0,33
LC	3,00	0,33	3,00	3,00	3,00	1,00	3,00	0,33	3,00	0,33	1,00
PD	0,33	0,20	0,33	1,00	0,33	0,33	1,00	0,33	1,00	1,00	0,33
PI	3,00	3,00	5,00	3,00	3,00	3,00	3,00	1,00	3,00	3,00	3,00
RA	1,00	0,33	1,00	0,33	0,33	0,33	1,00	0,33	1,00	1,00	0,33
SI	1,00	0,33	1,00	0,33	0,33	3,00	1,00	0,33	1,00	1,00	0,33
DS	0,33	0,33	1,00	3,00	3,00	1,00	3,00	0,33	3,00	3,00	1,00

Table I.5: A table containing the results to the pairwise comparison of criteria by XXXX. The consistency ratio of this comparison is 0.08.

	AL	CO	FL	IN	IT	LC	PD	PI	RA	SI	DS
AL	1,00	3,00	1,00	3,00	1,00	5,00	0,33	0,20	1,00	3,00	0,50
CO	0,33	1,00	0,20	0,33	0,33	3,00	0,25	0,20	0,33	3,00	0,33
FL	1,00	5,00	1,00	5,00	1,00	5,00	1,00	0,33	3,00	1,00	1,00
IN	0,33	3,00	0,20	1,00	0,33	5,00	0,20	0,14	0,20	1,00	0,20
IT	1,00	3,00	1,00	3,00	1,00	5,00	0,33	0,33	3,00	3,00	0,50
LC	0,20	0,33	0,20	0,20	0,20	1,00	0,20	0,14	0,20	1,00	0,20
PD	3,00	4,00	1,00	5,00	3,00	5,00	1,00	1,00	3,00	3,00	1,00
PI	5,00	5,00	3,00	7,00	3,00	7,00	1,00	1,00	5,00	5,00	3,00
RA	1,00	3,00	0,33	5,00	0,33	5,00	0,33	0,20	1,00	3,00	0,33
SI	0,33	0,33	1,00	1,00	0,33	1,00	0,33	0,20	0,33	1,00	0,33
DS	2,00	3,00	1,00	5,00	2,00	5,00	1,00	0,33	3,00	3,00	1,00

J

DESIGN CASE CONCEPT RANKINGS

J.1. DESIGN CASE C7

J.2. DESIGN CASE C8

Table J.1: Multi-criteria analysis for the originating process.

	Weight	2-belt check-in	3-belt check-in	1-st. self-check-in	2-st. self-check-in
System time	0,059	2	3	5	4
System throughput	0,187	2	5	4	3
Storage capacity	0,035	0	0	0	0
Required space	0,058	4	5	4	2
Energy consumption	0,167	2	4	4	5
Operational expenditure	0,147	2	2	4	5
Capital expenditure	0,348	4	5	2	3

Table J.2: Multi-criteria analysis for the screening process.

	Weight	$0{,}34^m/_s$ security screening	$0{,}40^m/_s$ security screening	$0{,}50^m/_s$ security screening
System time	0,059	3	4	5
System throughput	0,187	5	4	3
Storage capacity	0,035	0	0	0
Required space	0,058	3	4	5
Energy consumption	0,167	3	4	5
Operational expenditure	0,147	3	4	5
Capital expenditure	0,348	3	4	5

Table J.3: Multi-criteria analysis for the sorting process.

	Weight	Diverts	Tilting tray	ICS sorting
System time	0,059	3	4	5
System throughput	0,187	4	3	5
Storage capacity	0,035	0	5	5
Required space	0,058	4	3	5
Energy consumption	0,167	5	3	4
Operational expenditure	0,147	5	3	4
Capital expenditure	0,348	5	3	4

Table J.4: Multi-criteria analysis for the transport processes.

	Weight	$1{,}5^m/_s$ belt conveyor	$6{,}0^m/_s$ belt conveyors	$10{,}0^m/_s$ ICS track
System time	0,059	3	4	5
System throughput	0,187	3	4	5
Storage capacity	0,035	0	0	0
Required space	0,058	5	5	5
Energy consumption	0,167	4	3	5
Operational expenditure	0,147	4	3	5
Capital expenditure	0,348	4	3	5

Table J.5: Multi-criteria analysis for the make-up process.

	Weight	Chutes	Laterals	Flat carousel	Inclined carousel
System time	0,059	0	0	0	0
System throughput	0,187	2	3	4	5
Storage capacity	0,035	2	3	4	5
Required space	0,058	5	3	2	4
Energy consumption	0,167	5	4	2	3
Operational expenditure	0,147	4	5	2	3
Capital expenditure	0,348	5	2	3	4

Table J.6: Multi-criteria decision on terminating systems for design case Dresden.

	Weight	Lateral	Cart unloader	ULD unloader
In-system time	0,076	3	3	3
System throughput	0,164	4	5	5
Storage capacity	0,164	5	4	4
Required space	0,049	4	5	5
Energy consumption	0,150	5	4	4
Operational expenditure	0,287	4	5	5
Capital expenditure	0,109	5	4	4

Table J.7: Multi-criteria analysis for the reclaim process.

	Weight	Chutes	Laterals	Flat carousel	Inclined carousel
System time	0,059	0	0	0	0
System throughput	0,187	2	3	4	5
Storage capacity	0,035	2	3	4	5
Required space	0,058	5	3	2	4
Energy consumption	0,167	5	4	2	3
Operational expenditure	0,147	4	5	2	3
Capital expenditure	0,348	5	2	3	4

Table J.8: Multi-criteria decision on check-ins systems for design case Paris Orly.

	Weight	2-belt check-in	3-belt check-in	1-st. self-check-in	2-st. self-check-in
In-system time	0,076	3	3	3	3
System throughput	0,164	3	5	3	5
Storage capacity	0,164	4	5	3	3
Required space	0,049	4	5	2	4
Energy consumption	0,150	2	4	4	5
Operational expenditure	0,287	2	3	4	5
Capital expenditure	0,109	4	5	4	4

Table J.9: Multi-criteria decision on screening systems for design case Paris Orly.

	Weight	$0,34^m/_s$ security screening	$0,40^m/_s$ security screening	$0,50^m/_s$ security screening
In-system time	0,076	3	4	5
System throughput	0,164	3	4	5
Storage capacity	0,164	0	0	0
Required space	0,049	3	4	5
Energy consumption	0,150	3	4	5
Operational expenditure	0,287	3	4	5
Capital expenditure	0,109	3	4	5

Table J.10: Multi-criteria decision on sorting systems for design case Paris Orly.

	weight	Diverts	Tilting tray sorter	ICS sorting
In-system time	0,076	4	4	4
System throughput	0,164	4	4	4
Storage capacity	0,164	0	0	0
Required space	0,049	4	3	5
Energy consumption	0,150	5	3	4
Operational expenditure	0,287	5	3	4
Capital expenditure	0,109	5	3	4

Table J.11: Multi-criteria decision on transport systems for design case Paris Orly.

	Weight	$1{,}5\,^m/_s$ belt conveyor	$6{,}0\,^m/_s$ belt conveyors	$10{,}0\,^m/_s$ ICS track
In-system time	0,076	4	5	5
System throughput	0,164	4	5	5
Storage capacity	0,164	0	0	0
Required space	0,049	3	4	5
Energy consumption	0,150	3	4	5
Operational expenditure	0,287	4	4	5
Capital expenditure	0,109	3	4	5

Table J.12: Multi-criteria decision on make-up systems for design case Paris Orly.

	Weight	Chutes	Laterals	Flat carousel	Inclined carousel
In-system time	0,076	3	3	3	3
System throughput	0,164	4	4	4	5
Storage capacity	0,164	4	4	4	5
Required space	0,049	5	3	3	4
Energy consumption	0,150	5	4	2	3
Operational expenditure	0,287	4	5	2	3
Capital expenditure	0,109	4	3	3	5

Table J.13: Multi-criteria decision on transfer systems for design case Paris Orly.

	Weight	Lateral	Cart unloader	ULD unloader
In-system time	0,076	3	3	3
System throughput	0,164	4	5	5
Storage capacity	0,164	5	4	4
Required space	0,049	4	5	5
Energy consumption	0,150	5	4	4
Operational expenditure	0,287	4	5	5
Capital expenditure	0,109	5	4	4

Table J.14: Multi-criteria decision on storage systems for design case Paris Orly.

	Weight	Belt conveyor lanes	ICS storage lanes	SRS
In-system time	0,076	4	5	3
System throughput	0,164	3	4	5
Storage capacity	0,164	3	3	3
Required space	0,049	3	3	5
Energy consumption	0,150	3	4	5
Operational expenditure	0,287	3	4	5
Capital expenditure	0,109	3	4	5

Table J.15: Multi-criteria decision on terminating systems for design case Paris Orly.

	Weight	Lateral	Cart unloader	ULD unloader
In-system time	0,076	3	3	3
System throughput	0,164	4	5	5
Storage capacity	0,164	5	4	4
Required space	0,049	4	5	5
Energy consumption	0,150	5	4	4
Operational expenditure	0,287	4	5	5
Capital expenditure	0,109	5	4	4

Table J.16: Multi-criteria decision on reclaim systems for design case Paris Orly.

	Weight	Chutes	Laterals	Flat carousel	Inclined carousel
In-system time	0,076	3	3	3	3
System throughput	0,164	4	4	4	5
Storage capacity	0,164	4	4	4	5
Required space	0,049	5	3	3	4
Energy consumption	0,150	5	4	2	3
Operational expenditure	0,287	4	5	2	3
Capital expenditure	0,109	4	3	3	5

K

PROGRAM CODE

K.1. DECLARATION OF GLOBAL VARIABLES

```
Option Explicit

'''''''''''''''''''''''''''''''''''''''''''''''
'''
'''    Declaration of global variables
'''
'''''''''''''''''''''''''''''''''''''''''''''''
Dim Hour As Long
Dim Day As Long
Dim Week As Long
Dim Rand As Integer

Dim SimStart As Date
Dim SimEnd As Date
Dim SimLength As Long
Dim SimLengthDays As Long
Dim ScheduleBegin As Long
Dim ScheduleEnd As Long
Dim WeekdayStart As Long
Dim ColumnMonday As Long

Dim tOriginating As Integer
Dim tSorting As Integer
Dim tSecurity As Integer
Dim tDeparting As Integer
Dim tArriving As Integer
Dim tCustoms As Integer
Dim tReclaim As Integer
Dim tIncoming As Integer
Dim tOutgoing As Integer

Dim tFlow1 As Integer
Dim tFlow2 As Integer
Dim tFlow3 As Integer
Dim tFlow4 As Integer
Dim tFlow5 As Integer
Dim tFlow6 As Integer
Dim tFlow7 As Integer
Dim tFlow8 As Integer
Dim tFlow9 As Integer
Dim tFlow10 As Integer
Dim tFlow11 As Integer
Dim tFlow12 As Integer
Dim tFlow13 As Integer
Dim tFlow14 As Integer
Dim tFlow15 As Integer
Dim tFlow16 As Integer
Dim tFlow17 As Integer
Dim tFlow18 As Integer
Dim tFlow19 As Integer
Dim tFlow20 As Integer
Dim tFlow21 As Integer
Dim tFlow26 As Integer
Dim tFlow27 As Integer

```

```
Dim tRoute1 As Integer
Dim tRoute2 As Integer
Dim tRoute7 As Integer
Dim tRoute8 As Integer
Dim tRoute26 As Integer
Dim tMaxOriginating As Integer
Dim tMaxTransfer As Integer
Dim tMaxIncoming As Integer
Dim tMaxStorage As Integer

Dim tSpread As Integer
Dim tSafety As Integer
Dim fSafety As Double
Dim fScaling As Double
Dim fRedundant As Double
Dim tResult As Integer
Dim tCount As Integer
Dim tChart As Integer
Dim cMode As Integer
Dim Distribution As String
Dim NewEntry As String
Dim Airline As String
Dim StoreRemote As Boolean

Dim hbs1 As Double
Dim cs1 As Double
Dim sc1 As Double
Dim cu1 As Double
Dim or1 As Double
Dim te1 As Double
Dim tr1 As Double
Dim si1 As Double
Dim so1 As Double

Dim WriteToFile As Boolean
Dim WriteFlightPlan As Boolean
Dim WriteDeparture As Boolean
Dim WriteArrival As Boolean
Dim WritePerAirline As Boolean
Dim FileName As String

Dim tLoad_Bag As Double
Dim tUnload_Bag As Double

Dim Schedule As Worksheet
Dim Calculation As Worksheet
Dim Parameters As Worksheet
Dim Options As Worksheet
Dim Parts As Worksheet
Dim Results As Worksheet
Dim Charts As Worksheet
Dim Process As Worksheet

Dim NewBook As Workbook
Dim NewSchedule As Worksheet
Dim NewDeparture As Worksheet
Dim NewArrival As Worksheet
Dim NewOriginating As Worksheet
Dim Sheet As Worksheet
Dim Series As SeriesCollection

Dim ArrivalFlightTime() As Variant
Dim DepartureFlightTime() As Variant
Dim FlightIndex() As Variant
Dim ArrivalFlightList() As Variant
Dim DepartureFlightList() As Variant
Dim AllFlights() As String
Dim MakeUpList() As Double

Dim FlowData(1 To 50, 0 To 5) As Double
Dim ProcessData(1 To 19, 0 To 5) As Double
Dim SlotData(1 To 4, 0 To 5) As Long
Dim BoxPlotData() As Double

Dim SlotRecord() As Long
Dim ProcessRecord() As Double
Dim FlowRecord() As Double

Dim OriginatingAirline() As Double
Dim AirlineList() As String
Dim ProcessNames() As Variant
Dim FlowNames() As Variant
Dim SlotNames() As Variant
```

K.2. MAIN PROGRAM

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''
'''   General program to initiate every sub in order
'''
''''''''''''''''''''''''''''''''''''''''''''''''''''''
Sub SystemAnalysis()
    Dim i As Long
    Dim SimulationTimer(0 To 11) As Double

    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    'Do some things that make excel vba code go a little faster and measure time
    Application.ScreenUpdating = True 'False
    SimulationTimer(0) = Timer

    'Perform the initialization statements
    Call Initialize
    SimulationTimer(1) = Timer

    'Lock the random number system to a specific seed/stream
    Call RandomNumber
    SimulationTimer(2) = Timer

    'Perform the flight list calculations, giving: departure time, arrival time, taken seats, ...
        transferring passengers, terminating passengers
    Call FlightList
    SimulationTimer(3) = Timer

    'Generate baggage at the times that passengers may arrive according to the flightschedule ...
        or random number generator
    Call GenerateBags
    SimulationTimer(4) = Timer

    'Perform the calculation for the departure baggage train
    Call DepartureProcess
    SimulationTimer(5) = Timer

    'Perform the calculation for the arrival baggage train
    Call ArrivalProcess
    SimulationTimer(6) = Timer

    'Step throught each minute of the preferred calculation time and adapt the system to it
    Call DiscreteTime
    SimulationTimer(7) = Timer

    'Calculate the minimum, maximum, etc. of every
    Call DataProcessing
    SimulationTimer(8) = Timer

    'Cut the charts down to 1 or 2 days and put the data in it
    Call PopulateCharts
    SimulationTimer(9) = Timer

    'Write maximum, minimum and positions to process diagram
    Call ProcessOutput
    SimulationTimer(10) = Timer

    'If needed, write the data to a file
    Call WriteFile
    SimulationTimer(11) = Timer

    'Write time data to text output
    For i = 1 To 11
        Results.Cells(i, "A") = SimulationTimer(i) - SimulationTimer(i - 1)
    Next i
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    'Do some things that make excel behave normally again and display runtime
    Application.ScreenUpdating = True

    'Add End to make sure that global variables are cleared and will change upon new calculation
    End
End Sub
```

K.3. INITIALIZE

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

```
'''
'''    Read calculation data from sheets, create some constants and redimension arrays to ...
    appropriate size
'''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Sub Initialize()
    Dim ValMax As Long

    'Define which sheet is which for later use. Change this when you change the name of a sheet
    Set Schedule = wsh_Schedule
    Set Calculation = wsh_Calculation
    Set Parameters = wsh_Parameters
    Set Options = wsh_Options
    Set Parts = wsh_Parts
    Set Results = wsh_Output_text
    Set Charts = wsh_Output_charts
    Set Process = wsh_Output_process
    Set Check = wsh_Check
    Set Systems = wsh_Output_systems
    Set Terminat = wsh_Output_terminat

    'Declare some constants, determine the time interval and declare where the flight schedule ...
        starts
    Hour = 60
    Day = 24
    Week = 7

    SimStart = Format(Calculation.Cells(6, "B"), "Short Date")
    SimEnd = Format(Calculation.Cells(6, "C"), "Short Date")
    SimLengthDays = Round((SimEnd - SimStart), 0)
    SimLength = (SimLengthDays + 1) * Day * Hour
    ScheduleBegin = 8                        'Defines at which line the flight schedule starts
    ScheduleEnd = WorksheetFunction.Max(Schedule.Cells(Schedule.Rows.Count, ...
        "A").End(xlUp).Row, Schedule.Cells(Schedule.Rows.Count, "E").End(xlUp).Row, ...
        Schedule.Cells(Schedule.Rows.Count, "G").End(xlUp).Row, ...
        Schedule.Cells(Schedule.Rows.Count, "O").End(xlUp).Row, ...
        Schedule.Cells(Schedule.Rows.Count, "Q").End(xlUp).Row, ...
        Schedule.Cells(Schedule.Rows.Count, "W").End(xlUp).Row, ...
        Schedule.Cells(Schedule.Rows.Count, "X").End(xlUp).Row, ...
        Schedule.Cells(Schedule.Rows.Count, "Y").End(xlUp).Row)
    ColumnMonday = 17                        'The column which defines monday for the departures

    tSpread = 4
    tSafety = 0                              'Safety time for calculating BagStore vs. No BagStore
    fSafety = 1                              'Safety factor for linear scaling of suitcases
    fRedundant = Parameters.Cells(42, "D")   'Required redundancy of the system
    fScaling = Parameters.Cells(41, "D")
    tResult = Parameters.Cells(33, "D")
    tCount = Parameters.Cells(34, "D")
    tChart = Parameters.Cells(35, "D")
    cMode = Parameters.Cells(36, "D")
    cEnergy = Calculation.Cells(33, "K")

    'Describe process and flow times as well as fractions
    'Time it takes to complete a process
    tOriginating = Parameters.Cells(6, "L")
    tSorting = Parameters.Cells(7, "L")
    tSecurity = Parameters.Cells(8, "L")
    tDeparting = Parameters.Cells(9, "L")
    tArriving = Parameters.Cells(10, "L")
    tCustoms = Parameters.Cells(11, "L")
    tReclaim = Parameters.Cells(12, "L")
    tIncoming = Parameters.Cells(13, "L")
    tOutgoing = Parameters.Cells(14, "L")

    'Time it takes for bags to transport on this line (the number behind tFlow.. matches the ...
        visio diagram)
    tFlow1 = Parameters.Cells(6, "D")
    tFlow2 = Parameters.Cells(7, "D")
    tFlow3 = Parameters.Cells(8, "D")
    tFlow4 = Parameters.Cells(9, "D")
    tFlow5 = Parameters.Cells(10, "D")
    tFlow6 = Parameters.Cells(11, "D")
    tFlow7 = Parameters.Cells(12, "D")
    tFlow8 = Parameters.Cells(13, "D")
    tFlow9 = Parameters.Cells(14, "D")
    tFlow10 = Parameters.Cells(15, "D")
    tFlow11 = Parameters.Cells(16, "D")
    tFlow12 = Parameters.Cells(17, "D")
    tFlow13 = Parameters.Cells(18, "D")
    tFlow14 = Parameters.Cells(19, "D")
    tFlow15 = Parameters.Cells(20, "D")
    tFlow16 = Parameters.Cells(21, "D")
    tFlow17 = Parameters.Cells(22, "D")
```

```
    tFlow18 = Parameters.Cells(23, "D")
    tFlow19 = Parameters.Cells(24, "D")
    tFlow20 = Parameters.Cells(25, "D")
    tFlow21 = Parameters.Cells(26, "D")
    tFlow26 = Parameters.Cells(27, "D")
    tFlow27 = Parameters.Cells(28, "D")

    tRoute1 = tOriginating + tFlow1 + tSorting + tFlow11 + tSecurity + tFlow12 + tSorting + ...
        tFlow13
    tRoute2 = tOriginating + tFlow2 + tSecurity + tFlow12 + tSorting + tFlow13
    tRoute7 = tFlow7 + tSorting + tFlow11 + tSecurity + tFlow12 + tSorting + tFlow13
    tRoute8 = tFlow8 + tSecurity + tFlow12 + tSorting + tFlow13
    tRoute26 = tIncoming + tFlow26 + tSorting + tFlow11 + tSecurity + tFlow12 + tSorting + tFlow13
    tMaxOriginating = tOriginating + WorksheetFunction.Max(tFlow1 + tSorting + tFlow11, ...
        tFlow2) + tSecurity + WorksheetFunction.Max(tFlow12 + tSorting + tFlow9, tFlow10)
    tMaxTransfer = WorksheetFunction.Max(tFlow7 + tSorting + tFlow11, tFlow8) + tSecurity + ...
        WorksheetFunction.Max(tFlow12 + tSorting + tFlow9, tFlow10)
    tMaxIncoming = tIncoming + tFlow26 + WorksheetFunction.Max(tSorting + tFlow9, tSorting + ...
        tFlow11 + tSecurity + tFlow10, tSorting + tFlow11 + tSecurity + tFlow12 + tSorting + ...
        tFlow9)
    tMaxStorage = tFlow14 + tSorting + tFlow9

    hbs1 = 1    'What fraction of the originating and tansfer flow will be screened for ...
        security, KEEP AS 1 ALWAYS!
    cs1 = Parameters.Cells(19, "L")     'What fraction of the terminating flow  will be ...
        screened by customs
    sc1 = Parameters.Cells(20, "L")     'What fraction of screened bags by security, that do ...
        not have to be stored, is sent directly to make-up
    cu1 = Parameters.Cells(21, "L")     'What fraction of screened bags by customs is sent ...
        directly to reclaim
    or1 = Parameters.Cells(22, "L")     'In case (hbs1 > 0), what fraction of bags that need ...
        to be scanned is sent directly to security screening
    te1 = Parameters.Cells(23, "L")     'What fraction of bags uses dedicated equipment for ...
        transport to customs screening and reclaim
    tr1 = Parameters.Cells(24, "L")     'What fraction of the transfer flow that should be ...
        screened is sent directly to screening, bypassing sorting
    si1 = Parameters.Cells(25, "L")     'What fraction of incoming bags is already screened
    so1 = Parameters.Cells(26, "L")     'What fraction of outgoing bags should already be screened

    StoreRemote = Parameters.Cells(31, "L") 'The remote system handles storage to this part of ...
        the system
    tLoad_Bag = Parameters.Cells(36, "L")
    tUnload_Bag = Parameters.Cells(37, "L")

    WriteToFile = Calculation.Cells(35, "F")
    WriteFlightPlan = Calculation.Cells(36, "F")
    WriteDeparture = Calculation.Cells(37, "F")
    WriteArrival = Calculation.Cells(38, "F")
    WritePerAirline = Calculation.Cells(39, "F")
    FileName = Calculation.Cells(42, "F")

    ProcessNames = Array("Originating baggage", "Sorting baggage in-system", "Sorting baggage ...
        flow", "Security screening baggage", "SecurityA", "SecurityB", "SecurityC", "Make-up ...
        baggage in-system", "Make-up baggage flow", "Baggage storage positions", "Customs ...
        screening flow", "Baggage reclaim flow", "Terminating baggage flow", "Transfer baggage ...
        flow", "Arriving baggage flow", "Departing baggage flow", "Incoming baggage flow from ...
        connected system", "Outgoing baggage flow to connected system", "Filled ULDs and carts ...
        waiting for loading into an aircraft")
    FlowNames = Array("Flow1", "Flow1a", "Flow1b", "Flow1c", "Flow1d", "Flow2", "Flow2a", ...
        "Flow2b", "Flow2c", "Flow3", "Flow4", "Flow5", "Flow6", "Flow7", "Flow7a", "Flow7b", ...
        "Flow7c", "Flow7d", "Flow8", "Flow8a", "Flow8b", "Flow8c", "Flow9", "Flow10", ...
        "Flow11", "Flow11a", "Flow11b", "Flow11c", "Flow12", "Flow12a", "Flow12b", "Flow12c", ...
        "Flow12d", "Flow13", "Flow14", "Flow14a", "Flow14b", "Flow15", "Flow16", "Flow17", ...
        "Flow18", "Flow19", "Flow20", "Flow21", "Flow26", "Flow26a", "Flow26b", "Flow26c", ...
        "Flow26d", "Flow27")
    SlotNames = Array("Amount of aircraft serviced by make-up simultaneously", "Amount of ...
        simultaneous transfer unloads taking place", "Amount of simultaneous terminating ...
        unloads taking place", "Amount of baggage trains loading simultaneously")

    'Dimension arrays based on next flow, so that the first few calculations most will be zero
    ReDim ArrivalFlightTime(0 To ScheduleEnd) As Variant
    ReDim DepartureFlightTime(0 To ScheduleEnd) As Variant
    ReDim FlightIndex(0 To ScheduleEnd) As Variant
    ReDim ArrivalFlightList(0 To WorksheetFunction.RoundUp((SimLengthDays + 1) / 7, 0) * ...
        WorksheetFunction.CountA(Schedule.Range(Schedule.Cells(ScheduleBegin, "Q"), ...
        Schedule.Cells(ScheduleEnd, "W"))), 1 To 6) As Variant
    ReDim DepartureFlightList(0 To WorksheetFunction.RoundUp((SimLengthDays + 1) / 7, 0) * ...
        WorksheetFunction.CountA(Schedule.Range(Schedule.Cells(ScheduleBegin, "Q"), ...
        Schedule.Cells(ScheduleEnd, "W"))), 1 To 6) As Variant
    ReDim AllFlights(ScheduleBegin To ScheduleEnd) As String

    ReDim BoxPlotData(1 To 19, 1 To cMode) As Double
    ReDim FlowBoxPlot(1 To 50, 1 To cMode) As Double
    ReDim SlotBoxPlot(1 To 4, 1 To cMode) As Double
```

```

    ReDim AirlineList(1 To 2) As String
    ReDim SlotRecord(1 To 4, 0 To SimLength) As Long

    ValMax = WorksheetFunction.Max(tOriginating, tSecurity, tCustoms) + 1
    ReDim ProcessRecord(1 To 19, -ValMax To SimLength) As Double

    ValMax = WorksheetFunction.Max(tSorting + tFlow1, tFlow2, tFlow3, tFlow4, tFlow5 + ...
        tSorting, tSorting + tFlow7, tFlow8, tFlow9, tFlow10, tFlow11, tSorting + tFlow12, ...
        tSorting + tFlow14, tFlow5 + tSorting, tFlow16, tFlow17, tFlow18, tFlow20, tFlow21, ...
        tFlow26 + tSorting, tFlow27) + 1
    ReDim FlowRecord(1 To 50, -ValMax To SimLength) As Double
End Sub
```

K.4. RANDOMIZER

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''
'''  Several lines of code to create a stream and seed
'''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Sub RandomNumber()
    'Initialize random number generator and seed/stream. From here on the "Rnd" function ...
        generates a repeatable random variable (stream).
    Rand = Rnd(-1)
    Randomize 1
End Sub
```

K.5. FLIGHT LIST

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''
'''  Prepare the flight list that contains which flights are calculated
'''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Sub FlightList()
    Dim CheckCell As Long
    Dim wLine As Long
    Dim ArrLine As Long
    Dim DepLine As Long
    Dim Count As Long
    Dim ArrayIndex As Long

    Dim i As Long
    Dim j As Long
    Dim k As Long
    Dim l As Long
    Dim m As Long

    Dim FoundItem As Boolean

    'Read all airline names in an array
    AirlineList(1) = "Total"
    For i = LBound(AllFlights) To UBound(AllFlights)
        AllFlights(i) = Schedule.Cells(i, "A")

        NewEntry = AllFlights(i)
        FoundItem = False

        'Loop through AirlineList names to find whether entry is already there
        For j = 1 To UBound(AirlineList)
            If AirlineList(j) = NewEntry Then
                FoundItem = True
            End If
        Next j

        If FoundItem = False Then
            'Add new entry to airline list
            ReDim Preserve AirlineList(1 To UBound(AirlineList) + 1)
            AirlineList(UBound(AirlineList)) = NewEntry
        End If
    Next i

    ReDim OriginatingAirline(-(tOriginating + 1) To SimLength, LBound(AirlineList) To ...
        UBound(AirlineList)) As Double
```

```

    'Load the arriving and departing flights in the flight schedule
    ArrLine = 0
    DepLine = 0
    WeekdayStart = WorksheetFunction.Weekday(SimStart, vbMonday)
    For i = 0 To SimLengthDays

        wLine = 0
        For j = ScheduleBegin To Schedule.Cells(Schedule.Rows.Count, "O").End(xlUp).Row
            If Schedule.Cells(j, "O") ≤ Schedule.Cells(j, "P") Then
                If Schedule.Cells(j, "O") ≤ SimStart + i And Schedule.Cells(j, "P") ≥ SimStart ...
                    + i Then
                    CheckCell = ColumnMonday + (((WeekdayStart - 1) + i) - Week * ...
                        Fix(((WeekdayStart - 1) + i) / Week))

                    If Not IsEmpty(Schedule.Cells(j, CheckCell)) Then
                        ArrivalFlightTime(wLine) = Round((i + Schedule.Cells(j, CheckCell)) * ...
                            Day * Hour - Schedule.Cells(j, "Y"), 0)
                        DepartureFlightTime(wLine) = Round((i + Schedule.Cells(j, CheckCell)) ...
                            * Day * Hour, 0)
                        FlightIndex(wLine) = j
                        wLine = wLine + 1
                    End If
                End If
            Else
                MsgBox "The scheduled flight in line " & j & " has an end date before its ...
                    starting date. This flight will be skipped in this calculation.", ...
                    vbOKOnly, "Notification"
            End If
        Next j


        'Write sorted arrival list with: | (1) ArrivalFlightTime | (2) FlightIndex | (3) ...
            TerminatingSeats | (4) RemainingSeats | (5) TerminatingBags | (6) TransferBags |
        For j = 0 To WorksheetFunction.Count(ArrivalFlightTime) - 1
            ArrayIndex = Application.Match(WorksheetFunction.Min(ArrivalFlightTime), ...
                ArrivalFlightTime, False) - 1

            If ArrivalFlightTime(ArrayIndex) > 0 Then
                ArrivalFlightList(ArrLine, 1) = ArrivalFlightTime(ArrayIndex)
                ArrivalFlightList(ArrLine, 2) = FlightIndex(ArrayIndex)
                ArrivalFlightList(ArrLine, 3) = Round(Schedule.Cells(FlightIndex(ArrayIndex), ...
                    "E") * (1 - Probability(Calculation.Cells(8, "K"), ...
                    Schedule.Cells(FlightIndex(ArrayIndex), "L"), ...
                    Schedule.Cells(FlightIndex(ArrayIndex), "N"), ...
                    Schedule.Cells(FlightIndex(ArrayIndex), "M"), Rnd)) * ...
                    Probability(Calculation.Cells(6, "K"), ...
                    Schedule.Cells(FlightIndex(ArrayIndex), "F"), ...
                    Schedule.Cells(FlightIndex(ArrayIndex), "H"), ...
                    Schedule.Cells(FlightIndex(ArrayIndex), "G"), Rnd), 0)
                ArrivalFlightList(ArrLine, 4) = Schedule.Cells(FlightIndex(ArrayIndex), "E") - ...
                    ArrivalFlightList(ArrLine, 3)
                ArrivalFlightList(ArrLine, 5) = 0
                ArrivalFlightList(ArrLine, 6) = 0

                'Apply scaling to aircraft capacity
                ArrivalFlightList(ArrLine, 3) = Round(ArrivalFlightList(ArrLine, 3) * ...
                    fScaling, 0)
                ArrivalFlightList(ArrLine, 4) = Round(ArrivalFlightList(ArrLine, 4) * ...
                    fScaling, 0)
            End If

            ArrivalFlightTime(ArrayIndex) = Empty
            ArrLine = ArrLine + 1
        Next j


        'Write sorted departure list with: | (1) DepartureFlightTime | (2) FlightIndex | (3) ...
            TotalTakenSeats | (4) TransferPassengers | (5) OriginatingBags | (6) TransferBags |
        For j = 0 To WorksheetFunction.Count(DepartureFlightTime) - 1
            ArrayIndex = Application.Match(WorksheetFunction.Min(DepartureFlightTime), ...
                DepartureFlightTime, False) - 1

            If DepartureFlightTime(ArrayIndex) > 0 Then
                DepartureFlightList(DepLine, 1) = DepartureFlightTime(ArrayIndex)
                DepartureFlightList(DepLine, 2) = FlightIndex(ArrayIndex)
                DepartureFlightList(DepLine, 3) = ...
                    Round(Schedule.Cells(FlightIndex(ArrayIndex), "E") * ...
                    Probability(Calculation.Cells(6, "K"), ...
                    Schedule.Cells(FlightIndex(ArrayIndex), "F"), ...
                    Schedule.Cells(FlightIndex(ArrayIndex), "H"), ...
                    Schedule.Cells(FlightIndex(ArrayIndex), "G"), Rnd), 0)
                DepartureFlightList(DepLine, 4) = Round(DepartureFlightList(DepLine, 3) * ...
                    Probability(Calculation.Cells(6, "K"), ...
```

```
                        Schedule.Cells(FlightIndex(ArrayIndex), "L"), ...
                        Schedule.Cells(FlightIndex(ArrayIndex), "N"), ...
                        Schedule.Cells(FlightIndex(ArrayIndex), "M"), Rnd), 0)
                    DepartureFlightList(DepLine, 5) = 0
                    DepartureFlightList(DepLine, 6) = 0

                    'Apply scaling to aircraft capacity
                    DepartureFlightList(DepLine, 3) = Round(DepartureFlightList(DepLine, 3) * ...
                        fScaling, 0)
                    DepartureFlightList(DepLine, 4) = Round(DepartureFlightList(DepLine, 4) * ...
                        fScaling, 0)
                End If

                DepartureFlightTime(ArrayIndex) = Empty
                FlightIndex(ArrayIndex) = Empty
                DepLine = DepLine + 1
            Next j
        Next i
End Sub
```

K.6. GENERATE BAGS

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''
'''  Code that generates bags and places them in the correct locations
'''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Sub GenerateBags()
    Dim Count As Long
    Dim OriginatingRow As Integer
    Dim TransferRow As Integer
    Dim MakeUpRow As Integer
    Dim BagTrainRow As Integer

    Dim IncomingCI As Long
    Dim IncomingTP As Long
    Dim OutgoingCI As Long
    Dim OutgoingTP As Long

    Dim OriginatingPassengers As Long
    Dim TransferPassengers As Long
    Dim TerminatingPassengers As Long

    Dim INopen As Long
    Dim INclose As Long
    Dim INmean As Long
    Dim MUopen As Long
    Dim MUclose As Long
    Dim LoopPassengers As Long
    Dim PassengerTime As Long

    Dim NewBags As Double
    Dim tNewBags As Long
    Dim tOldBags As Long
    Dim TrainCapacity As Double
    Dim tTransferTrain As Long

    Dim Character As String
    Dim AtTerminal As String

    Dim i As Long
    Dim j As Long
    Dim k As Long
    Dim l As Long
    Dim m As Long

    Dim FoundItem As Boolean
    Dim ValMax As Long

    ValMax = WorksheetFunction.Max(Calculation.Cells(25, "C") - Calculation.Cells(25, "D"), ...
        Calculation.Cells(26, "C") - Calculation.Cells(26, "D"), Calculation.Cells(27, "C") - ...
        Calculation.Cells(27, "D"), Calculation.Cells(28, "C") - Calculation.Cells(28, "D"))
    ReDim MakeUpList(LBound(DepartureFlightList, 1) To UBound(DepartureFlightList, 1), -2 To ...
        ValMax) As Double

    'Perform calculations for originating/transferring/terminating passengers
    For i = LBound(DepartureFlightList, 1) To UBound(DepartureFlightList, 1)
        If Not IsEmpty(DepartureFlightList(i, 1)) Then
```

```
        'Define which row needs to be selected for check-in, transfer and make-up time ...
            (for the departing aircraft)
        Character = Schedule.Cells(DepartureFlightList(i, 2), "D")
        Select Case Character
            Case Options.Cells(5, "D")
                OriginatingRow = 13
                TransferRow = 19
                MakeUpRow = 25
            Case Options.Cells(6, "D")
                OriginatingRow = 14
                TransferRow = 20
                MakeUpRow = 26
            Case Options.Cells(7, "D")
                OriginatingRow = 15
                TransferRow = 21
                MakeUpRow = 27
            Case Options.Cells(8, "D")
                OriginatingRow = 16
                TransferRow = 22
                MakeUpRow = 28
        End Select

        'Look at which terminal the aircraft will arrive and depart
        AtTerminal = Schedule.Cells(DepartureFlightList(i, 2), "X")
        If AtTerminal = Options.Cells(6, "F") Then
            IncomingCI = 0
            IncomingTP = 0
            OutgoingCI = Round((DepartureFlightList(i, 3) - DepartureFlightList(i, 4)) * ...
                Probability(Calculation.Cells(17, "N"), Calculation.Cells(17, "K"), ...
                Calculation.Cells(17, "M"), Calculation.Cells(17, "L"), Rnd), 0)
            OutgoingTP = Round(DepartureFlightList(i, 4) * ...
                Probability(Calculation.Cells(18, "N"), Calculation.Cells(18, "K"), ...
                Calculation.Cells(18, "M"), Calculation.Cells(18, "L"), Rnd), 0)

            OriginatingPassengers = OutgoingCI
            TransferPassengers = OutgoingTP
        ElseIf AtTerminal = Options.Cells(5, "F") Then
            IncomingCI = Round((DepartureFlightList(i, 3) - DepartureFlightList(i, 4)) * ...
                Probability(Calculation.Cells(13, "N"), Calculation.Cells(13, "K"), ...
                Calculation.Cells(13, "M"), Calculation.Cells(13, "L"), Rnd), 0)
            IncomingTP = Round(DepartureFlightList(i, 4) * ...
                Probability(Calculation.Cells(14, "N"), Calculation.Cells(14, "K"), ...
                Calculation.Cells(14, "M"), Calculation.Cells(14, "L"), Rnd), 0)
            OutgoingCI = 0
            OutgoingTP = 0

            OriginatingPassengers = (DepartureFlightList(i, 3) - DepartureFlightList(i, ...
                4)) - IncomingCI
            TransferPassengers = DepartureFlightList(i, 4) - IncomingTP
        End If

        'Determine make up opening time
        MUopen = DepartureFlightList(i, 1) - Calculation.Cells(MakeUpRow, "C")
        MUclose = DepartureFlightList(i, 1) - Calculation.Cells(MakeUpRow, "D")
        MakeUpList(i, -1) = MUopen
        MakeUpList(i, -2) = MUclose

        'Count from "i" to where the departure time of aircraft "i" equals that of ...
            arriving aircraft "Count"
        If DepartureFlightList(i, 1) ≥ ArrivalFlightList(i, 1) Then
            Count = i
            Do Until DepartureFlightList(i, 1) < ArrivalFlightList(Count, 1) Or Count ≥ ...
                UBound(DepartureFlightList, 1)
                Count = Count + 1
            Loop
        ElseIf DepartureFlightList(i, 1) < ArrivalFlightList(i, 1) Then
            Count = i
            Do Until DepartureFlightList(i, 1) > ArrivalFlightList(Count, 1) Or Count ≤ ...
                LBound(DepartureFlightList, 1)
                Count = Count - 1
            Loop
        End If

        'Loop through the 6 cases that exist to calculate the times departing passengers ...
            check in.
        For j = 1 To 6
            Select Case j
                Case 1, 3, 5
                    INopen = DepartureFlightList(i, 1) - Calculation.Cells(OriginatingRow, ...
                        "C")
                    INclose = DepartureFlightList(i, 1) - ...
                        Calculation.Cells(OriginatingRow, "E")
                    INmean = DepartureFlightList(i, 1) - Calculation.Cells(OriginatingRow, ...
                        "D")
```

```
                    Distribution = Calculation.Cells(OriginatingRow, "F")
                Case 2, 4, 6
                    INopen = DepartureFlightList(i, 1) - Calculation.Cells(TransferRow, "C")
                    INclose = DepartureFlightList(i, 1) - Calculation.Cells(TransferRow, "E")
                    INmean = DepartureFlightList(i, 1) - Calculation.Cells(TransferRow, "D")
                    Distribution = Calculation.Cells(TransferRow, "F")
            End Select

            Select Case j
                Case 1
                    LoopPassengers = OriginatingPassengers
                Case 2
                    LoopPassengers = TransferPassengers
                Case 3
                    LoopPassengers = IncomingCI
                Case 4
                    LoopPassengers = IncomingTP
                Case 5
                    LoopPassengers = OutgoingCI
                Case 6
                    LoopPassengers = OutgoingTP
            End Select

            If LoopPassengers > 0 Then
                For k = 1 To LoopPassengers
                    PassengerTime = Round(Probability(Distribution, INopen, INclose, ...
                        INmean, Rnd), 0)
                    NewBags = Round(Probability(Calculation.Cells(7, "K"), ...
                        Schedule.Cells(DepartureFlightList(i, 2), "I"), ...
                        Schedule.Cells(DepartureFlightList(i, 2), "K"), ...
                        Schedule.Cells(DepartureFlightList(i, 2), "J"), Rnd), 2)

                    Select Case j
                        Case 1
                        'This case handles the originating passengers that enter local ...
                            aircraft
                            FoundItem = False
                            If PassengerTime < 0 Then
                                FoundItem = False
                            Else
                                FoundItem = True
                            End If

                            If FoundItem Then
                                ProcessRecord(1, PassengerTime) = ProcessRecord(1, ...
                                    PassengerTime) + NewBags
                                DepartureFlightList(i, 5) = DepartureFlightList(i, 5) + ...
                                    NewBags

                                'Create and fill the list of check-ins per airline only ...
                                    for originating passengers
                                Airline = Schedule.Cells(DepartureFlightList(i, 2), "A")
                                For l = LBound(AirlineList) + 1 To UBound(AirlineList)
                                    If AirlineList(l) = Airline Then
                                        For m = 0 To tOriginating
                                            OriginatingAirline(PassengerTime, l) = ...
                                                OriginatingAirline(PassengerTime, l) + 1
                                            OriginatingAirline(PassengerTime, 1) = ...
                                                OriginatingAirline(PassengerTime, 1) + 1
                                        Next m
                                    End If
                                Next l

                                'Create an entry in Flow14 for to be stored bags
                                If PassengerTime < MUopen - ...
                                    (WorksheetFunction.Max(tRoute1, tRoute2) + tSafety) Then
                                    FlowRecord(3, PassengerTime + tOriginating) = ...
                                        FlowRecord(3, PassengerTime + tOriginating) + (1 - ...
                                        (or1 * hbs1)) * NewBags
                                    FlowRecord(8, PassengerTime + tOriginating) = ...
                                        FlowRecord(8, PassengerTime + tOriginating) + (or1 ...
                                        * hbs1) * NewBags

                                    'Add bags to storage outflow (spread), and make-up ...
                                        baggage list
                                    For m = 1 To tSpread + 1
                                        FlowRecord(36, MUopen + (m - 1)) = FlowRecord(36, ...
                                            MUopen + (m - 1)) + (NewBags / (tSpread + 1))
                                        MakeUpList(i, tMaxStorage + (m - 1)) = ...
                                            MakeUpList(i, tMaxStorage + (m - 1)) + ...
                                            (NewBags / (tSpread + 1))
                                    Next m
                                Else
```

```
                        FlowRecord(2, PassengerTime + tOriginating) = ...
                            FlowRecord(2, PassengerTime + tOriginating) + (1 - ...
                            (or1 * hbs1)) * NewBags
                        FlowRecord(7, PassengerTime + tOriginating) = ...
                            FlowRecord(7, PassengerTime + tOriginating) + (or1 ...
                            * hbs1) * NewBags
                        MakeUpList(i, PassengerTime - MUopen + ...
                            tMaxOriginating) = MakeUpList(i, PassengerTime - ...
                            MUopen + tMaxOriginating) + NewBags
                    End If
                End If

            Case 2
            'This case handles the local transfers that enter local aircraft
                FoundItem = False
                l = Count + 1

                If PassengerTime ≤ 0 Then
                    PassengerTime = 0
                    FoundItem = False
                Else
                    Do
                        l = l - 1
                        If Not IsEmpty(ArrivalFlightList(l, 1)) Then
                            If ArrivalFlightList(l, 1) < PassengerTime Then
                                If ArrivalFlightList(l, 4) > 0 And ...
                                    Schedule.Cells(ArrivalFlightList(l, 2), ...
                                    "X") = Options.Cells(5, "F") Then
                                    FoundItem = True
                                    PassengerTime = ArrivalFlightList(l, 1)
                                    ArrivalFlightList(l, 4) = ...
                                        ArrivalFlightList(l, 4) - 1
                                End If
                            End If
                        End If
                    Loop Until FoundItem = True Or l ≤ 1 Or ...
                        (DepartureFlightList(i, 1) - ArrivalFlightList(l - 1, ...
                        1)) > Calculation.Cells(TransferRow, "C")
                End If

                If FoundItem Then
                    'Check which character the arriving flight has. Note: this ...
                        may be different from the departing flight
                    Character = Schedule.Cells(ArrivalFlightList(l, 2), "D")
                    Select Case Character
                        Case Options.Cells(5, "D")
                            BagTrainRow = 23
                        Case Options.Cells(6, "D")
                            BagTrainRow = 24
                        Case Options.Cells(7, "D")
                            BagTrainRow = 25
                        Case Options.Cells(8, "D")
                            BagTrainRow = 26
                    End Select

                    'Add the bags to arriving aircraft and both flight lists ...
                        calculate train capacity
                    tOldBags = WorksheetFunction.RoundUp(ArrivalFlightList(l, ...
                        6) / tUnload_Bag, 0)

                    ProcessRecord(15, PassengerTime) = ProcessRecord(15, ...
                        PassengerTime) + NewBags
                    ArrivalFlightList(l, 6) = ArrivalFlightList(l, 6) + NewBags
                    DepartureFlightList(i, 6) = DepartureFlightList(i, 6) + ...
                        NewBags

                    tNewBags = WorksheetFunction.RoundUp(ArrivalFlightList(l, ...
                        6) / tUnload_Bag, 0)
                    TrainCapacity = Calculation.Cells(BagTrainRow, "K") * ...
                        Calculation.Cells(BagTrainRow, "L")
                    tTransferTrain = ArrivalFlightList(l, 1) + tArriving + ...
                        WorksheetFunction.RoundUp(ArrivalFlightList(l, 6) / ...
                        TrainCapacity, 0) * ...
                        Round(Calculation.Cells(BagTrainRow, "K") * ...
                        Calculation.Cells(BagTrainRow, "N"), 0)

                    'Add the newbags to the flows
                    FlowRecord(44, tTransferTrain) = FlowRecord(44, ...
                        tTransferTrain) + NewBags
                    ProcessRecord(14, tTransferTrain + tFlow21 + tNewBags) = ...
                        ProcessRecord(14, tTransferTrain + tFlow21 + tNewBags) ...
                        + NewBags

```

```
                If tTransferTrain + tFlow21 + tNewBags < MUopen - ...
                     (WorksheetFunction.Max(tRoute7, tRoute8) + tSafety) Then
                    FlowRecord(16, tTransferTrain + tFlow21 + tNewBags) = ...
                         FlowRecord(16, tTransferTrain + tFlow21 + ...
                         tNewBags) + (1 - tr1) * NewBags
                    FlowRecord(21, tTransferTrain + tFlow21 + tNewBags) = ...
                         FlowRecord(21, tTransferTrain + tFlow21 + ...
                         tNewBags) + tr1 * NewBags

                    'Add bags to storage outflow (spread)
                    For m = 1 To tSpread + 1
                        FlowRecord(36, MUopen + (m - 1)) = FlowRecord(36, ...
                             MUopen + (m - 1)) + (NewBags / (tSpread + 1))
                        MakeUpList(i, (m - 1) + tMaxStorage) = ...
                             MakeUpList(i, (m - 1) + tMaxStorage) + ...
                             (NewBags / (tSpread + 1))
                    Next m
                Else
                    FlowRecord(15, tTransferTrain + tFlow21 + tNewBags) = ...
                         FlowRecord(15, tTransferTrain + tFlow21 + ...
                         tNewBags) + (1 - tr1) * NewBags
                    FlowRecord(20, tTransferTrain + tFlow21 + tNewBags) = ...
                         FlowRecord(20, tTransferTrain + tFlow21 + ...
                         tNewBags) + tr1 * NewBags
                    MakeUpList(i, tTransferTrain + tFlow21 + tNewBags - ...
                         MUopen + tMaxTransfer) = MakeUpList(i, ...
                         tTransferTrain + tFlow21 + tNewBags - MUopen + ...
                         tMaxTransfer) + NewBags
                End If

                'Test if the newly added bag also adds an extra minute to ...
                     unloading time, if so, block off an extra minute for ...
                     the unload slot
                If tNewBags > tOldBags Then
                    SlotRecord(2, tTransferTrain + tFlow21 + tNewBags) = ...
                         SlotRecord(2, tTransferTrain + tFlow21 + tNewBags) ...
                         + 1
                End If
            End If

        Case 3
        'This case handles the originating passengers from connected ...
             BHS(s) that enter local aircraft
            FoundItem = False

            If PassengerTime < 0 Then
                FoundItem = False
            Else
                FoundItem = True
            End If

            If FoundItem Then
                ProcessRecord(17, PassengerTime) = ProcessRecord(17, ...
                     PassengerTime) + NewBags
                DepartureFlightList(i, 5) = DepartureFlightList(i, 5) + ...
                     NewBags

                If PassengerTime < MUopen - (tRoute26 + tSafety) Then
                    'Store bags that need to be screened after being ...
                         screened and store bags that don't need to be ...
                         screened immediately
                    'Flow (26c) goes directly to baggage storage and flow ...
                         (26b) goes to storage via security screening
                    FlowRecord(47, PassengerTime + tIncoming) = ...
                         FlowRecord(47, PassengerTime + tIncoming) + (1 - ...
                         si1) * NewBags
                    FlowRecord(48, PassengerTime + tIncoming) = ...
                         FlowRecord(48, PassengerTime + tIncoming) + si1 * ...
                         NewBags

                    'Add bags to storage outflow (spread)
                    For m = 1 To tSpread + 1
                        FlowRecord(36, MUopen + (m - 1)) = FlowRecord(36, ...
                             MUopen + (m - 1)) + (NewBags / (tSpread + 1))
                        MakeUpList(i, (m - 1) + tMaxStorage) = ...
                             MakeUpList(i, (m - 1) + tMaxStorage) + ...
                             (NewBags / (tSpread + 1))
                    Next m
                Else
                    'Send bags that don't need to be screened straight to ...
                         make-up
                    'Flow (26d) will go directly to make-up, whilst flow ...
                         (26a) will go to make-up via screening
```

```
            FlowRecord(46, PassengerTime + tIncoming) = ...
                FlowRecord(46, PassengerTime + tIncoming) + (1 - ...
                si1) * NewBags
            FlowRecord(49, PassengerTime + tIncoming) = ...
                FlowRecord(49, PassengerTime + tIncoming) + si1 * ...
                NewBags
            MakeUpList(i, PassengerTime - MUopen + tMaxIncoming) = ...
                MakeUpList(i, PassengerTime - MUopen + ...
                tMaxIncoming) + NewBags
        End If
    End If

Case 4
'This case handles the transfer passengers from the connected BHS ...
    to the local terminal
    FoundItem = False
    l = Count + 1

    If PassengerTime ≤ 0 Then
        PassengerTime = 0
        FoundItem = True
    Else
        Do
            l = l - 1
            If Not IsEmpty(ArrivalFlightList(l, 1)) Then
                If ArrivalFlightList(l, 1) < PassengerTime Then
                    If ArrivalFlightList(l, 4) > 0 And ...
                        Schedule.Cells(ArrivalFlightList(l, 2), ...
                        "X") = Options.Cells(6, "F") Then
                        FoundItem = True
                        PassengerTime = ArrivalFlightList(l, 1)
                        ArrivalFlightList(l, 4) = ...
                            ArrivalFlightList(l, 4) - 1
                    End If
                End If
            End If
        Loop Until FoundItem = True Or l ≤ 1 Or ...
            (DepartureFlightList(i, 1) - ArrivalFlightList(l - 1, ...
            1)) > Calculation.Cells(TransferRow, "C")
    End If

    If FoundItem Then
        'Check which character the arriving flight has. Note: this ...
            may be different from the departing flight
        Character = Schedule.Cells(ArrivalFlightList(l, 2), "D")
        Select Case Character
            Case Options.Cells(5, "D")
                BagTrainRow = 23
            Case Options.Cells(6, "D")
                BagTrainRow = 24
            Case Options.Cells(7, "D")
                BagTrainRow = 25
            Case Options.Cells(8, "D")
                BagTrainRow = 26
        End Select

        'Add the bags to arriving aircraft and both flight lists ...
            calculate train capacity
        ArrivalFlightList(l, 6) = ArrivalFlightList(l, 6) + NewBags
        DepartureFlightList(i, 6) = DepartureFlightList(i, 6) + ...
            NewBags

        TrainCapacity = Calculation.Cells(BagTrainRow, "K") * ...
            Calculation.Cells(BagTrainRow, "L")
        tTransferTrain = ArrivalFlightList(l, 1) + tArriving + ...
            WorksheetFunction.RoundUp(ArrivalFlightList(l, 6) / ...
            TrainCapacity, 0) * ...
            Round(Calculation.Cells(BagTrainRow, "K") * ...
            Calculation.Cells(BagTrainRow, "N"), 0)
        tNewBags = WorksheetFunction.RoundUp(ArrivalFlightList(l, ...
            6) / tUnload_Bag, 0)

        ProcessRecord(17, tTransferTrain + tFlow21 + tNewBags) = ...
            ProcessRecord(17, tTransferTrain + tFlow21 + tNewBags) ...
            + NewBags
        If tTransferTrain + tFlow21 + tNewBags < MUopen - ...
            (tRoute26 + tSafety) Then
            'Store bags that need to be screened after being ...
                screened and store bags that don't need to be ...
                screened immediately
            'Flow (26c) goes directly to baggage storage and flow ...
                (26b) goes to storage via security screening
            FlowRecord(47, tTransferTrain + tFlow21 + tNewBags + ...
                tIncoming) = FlowRecord(47, tTransferTrain + ...
```

```
                tFlow21 + tNewBags + tIncoming) + (1 - si1) * NewBags
            FlowRecord(48, tTransferTrain + tFlow21 + tNewBags + ...
                tIncoming) = FlowRecord(48, tTransferTrain + ...
                tFlow21 + tNewBags + tIncoming) + si1 * NewBags

            'Add bags to storage outflow (spread)
            For m = 1 To tSpread + 1
                FlowRecord(36, MUopen + (m - 1)) = FlowRecord(36, ...
                    MUopen + (m - 1)) + (NewBags / (tSpread + 1))
                MakeUpList(i, (m - 1) + tMaxStorage) = ...
                    MakeUpList(i, (m - 1) + tMaxStorage) + ...
                    (NewBags / (tSpread + 1))
            Next m
        Else
            'Send bags that don't need to be screened straight to ...
                make-up
            'Flow (26d) will go directly to make-up, whilst flow ...
                (26a) will go to make-up via screening
            FlowRecord(46, tTransferTrain + tFlow21 + tNewBags + ...
                tIncoming) = FlowRecord(46, tTransferTrain + ...
                tFlow21 + tNewBags + tIncoming) + (1 - si1) * NewBags
            FlowRecord(49, tTransferTrain + tFlow21 + tNewBags + ...
                tIncoming) = FlowRecord(49, tTransferTrain + ...
                tFlow21 + tNewBags + tIncoming) + si1 * NewBags
            MakeUpList(i, tTransferTrain + tFlow21 + tNewBags - ...
                MUopen + tMaxIncoming) = MakeUpList(i, ...
                tTransferTrain + tFlow21 + tNewBags - MUopen + ...
                tMaxIncoming) + NewBags
        End If
    End If

Case 5
'This case handles local originating passengers that enter ...
    aircraft on a connected BHS
    FoundItem = False
    If PassengerTime < 0 Then
        FoundItem = False
    Else
        FoundItem = True
    End If

    If FoundItem Then
        ProcessRecord(1, PassengerTime) = ProcessRecord(1, ...
            PassengerTime) + NewBags
        DepartureFlightList(i, 5) = DepartureFlightList(i, 5) + ...
            NewBags

        'Create and fill the list of check-ins per airline only ...
            for originating passengers
        Airline = Schedule.Cells(DepartureFlightList(i, 2), "A")
        For l = LBound(AirlineList) To UBound(AirlineList)
            If AirlineList(l) = Airline Then
                For m = 0 To tOriginating
                    OriginatingAirline(PassengerTime, l) = ...
                        OriginatingAirline(PassengerTime, l) + 1
                    OriginatingAirline(PassengerTime, 1) = ...
                        OriginatingAirline(PassengerTime, 1) + 1
                Next m
            End If
        Next l

        If PassengerTime < MUopen - ...
            (WorksheetFunction.Max(tRoute1, tRoute2) + tSafety) ...
            And StoreRemote = False Then
            'Bag must be screened and then stored locally until ...
                flight will depart or make-up is open
            FlowRecord(3, PassengerTime + tOriginating) = ...
                FlowRecord(3, PassengerTime + tOriginating) + (1 - ...
                (or1 * hbs1)) * NewBags
            FlowRecord(8, PassengerTime + tOriginating) = ...
                FlowRecord(8, PassengerTime + tOriginating) + (or1 ...
                * hbs1) * NewBags

            'Add bags to storage outflow
            For m = 1 To tSpread + 1
                FlowRecord(37, MUopen + (m - 1)) = FlowRecord(37, ...
                    MUopen + (m - 1)) + (NewBags / (tSpread + 1))
            Next m
        Else
            'Store bags in remote system with the possibility of ...
                screening locally or on the remote system
            'Flow (1d) is the part that does not have to be ...
                screened and should go directly to OUT
```

```
            'Flow (1c) is the part that needs to be screened and ...
                will go via sorting to screening to OUT
            'Flow (2c)is the part that needs to go directly to ...
                screening before it is sent to OUT via sorting
            FlowRecord(4, PassengerTime + tOriginating) = ...
                FlowRecord(4, PassengerTime + tOriginating) + so1 ...
                * (1 - (or1 * hbs1)) * NewBags
            FlowRecord(5, PassengerTime + tOriginating) = ...
                FlowRecord(5, PassengerTime + tOriginating) + (1 - ...
                so1) * NewBags
            FlowRecord(9, PassengerTime + tOriginating) = ...
                FlowRecord(9, PassengerTime + tOriginating) + so1 ...
                * or1 * hbs1 * NewBags
        End If
    End If

Case 6
'This case handles local transfer passengers that enter aircraft ...
    on a remote terminal
    FoundItem = False
    l = Count + 1

    If PassengerTime ≤ 0 Then
        PassengerTime = 0
        FoundItem = True
    Else
        Do
            l = l - 1
            If Not IsEmpty(ArrivalFlightList(l, 1)) Then
                If ArrivalFlightList(l, 1) < PassengerTime Then
                    If ArrivalFlightList(l, 4) > 0 And ...
                        Schedule.Cells(ArrivalFlightList(l, 2), ...
                        "X") = Options.Cells(5, "F") Then
                        FoundItem = True
                        PassengerTime = ArrivalFlightList(l, 1)
                        ArrivalFlightList(l, 4) = ...
                            ArrivalFlightList(l, 4) - 1
                    End If
                End If
            End If
        Loop Until FoundItem = True Or l ≤ 1 Or ...
            (DepartureFlightList(i, 1) - ArrivalFlightList(l - 1, ...
            1)) > Calculation.Cells(TransferRow, "C")
    End If

    If FoundItem Then
        'Check which character the arriving flight has. Note: this ...
            may be different from the departing flight
        Character = Schedule.Cells(ArrivalFlightList(l, 2), "D")
        Select Case Character
            Case Options.Cells(5, "D")
                BagTrainRow = 23
            Case Options.Cells(6, "D")
                BagTrainRow = 24
            Case Options.Cells(7, "D")
                BagTrainRow = 25
            Case Options.Cells(8, "D")
                BagTrainRow = 26
        End Select

        'Add the bags to arriving aircraft and both flight lists ...
            calculate train capacity
        tOldBags = WorksheetFunction.RoundUp(ArrivalFlightList(l, ...
            6) / tUnload_Bag, 0)

        ProcessRecord(15, PassengerTime) = ProcessRecord(15, ...
            PassengerTime) + NewBags
        ArrivalFlightList(l, 6) = ArrivalFlightList(l, 6) + NewBags
        DepartureFlightList(i, 6) = DepartureFlightList(i, 6) + ...
            NewBags

        tNewBags = WorksheetFunction.RoundUp(ArrivalFlightList(l, ...
            6) / tUnload_Bag, 0)
        TrainCapacity = Calculation.Cells(BagTrainRow, "K") * ...
            Calculation.Cells(BagTrainRow, "L")
        tTransferTrain = ArrivalFlightList(l, 1) + tArriving + ...
            WorksheetFunction.RoundUp(ArrivalFlightList(l, 6) / ...
            TrainCapacity, 0) * ...
            Round(Calculation.Cells(BagTrainRow, "K") * ...
            Calculation.Cells(BagTrainRow, "N"), 0)

        'Add the newbags to the flows
        FlowRecord(44, tTransferTrain) = FlowRecord(44, ...
            tTransferTrain) + NewBags
```

```
                                        ProcessRecord(14, tTransferTrain + tFlow21 + tNewBags) = ...
                                            ProcessRecord(14, tTransferTrain + tFlow21 + tNewBags) ...
                                            + NewBags

                                        If tTransferTrain + tFlow21 + tNewBags < MUopen - ...
                                            (WorksheetFunction.Max(tRoute7, tRoute8) + tSafety) ...
                                            And StoreRemote = False Then
                                            'Bag must be screened and then stored locally until ...
                                                flight will make-up is open
                                            FlowRecord(16, tTransferTrain + tFlow21 + tNewBags) = ...
                                                FlowRecord(16, tTransferTrain + tFlow21 + ...
                                                tNewBags) + (1 - (or1 * hbs1)) * NewBags
                                            FlowRecord(21, tTransferTrain + tFlow21 + tNewBags) = ...
                                                FlowRecord(21, tTransferTrain + tFlow21 + ...
                                                tNewBags) + (or1 * hbs1) * NewBags

                                            'Add bags to storage outflow (spread)
                                            For m = 1 To tSpread + 1
                                                FlowRecord(37, MUopen + (m - 1)) = FlowRecord(37, ...
                                                    MUopen + (m - 1)) + (NewBags / (tSpread + 1))
                                            Next m
                                        Else
                                            'Store bags in remote system with the possibility of ...
                                                screening locally or on the remote system
                                            'Flow (7d) is the part that does not have to be ...
                                                screened and should go directly to OUT
                                            'Flow (7c) is the part that needs to be screened and ...
                                                will go via sorting to screening to OUT
                                            'Flow (8c) is the part that needs to go directly to ...
                                                screening before it is sent to OUT via sorting
                                            FlowRecord(17, tTransferTrain + tFlow21 + tNewBags) = ...
                                                FlowRecord(17, tTransferTrain + tFlow21 + ...
                                                tNewBags) + so1 * (1 - (or1 * hbs1)) * NewBags
                                            FlowRecord(18, tTransferTrain + tFlow21 + tNewBags) = ...
                                                FlowRecord(18, tTransferTrain + tFlow21 + ...
                                                tNewBags) + (1 - so1) * NewBags
                                            FlowRecord(22, tTransferTrain + tFlow21 + tNewBags) = ...
                                                FlowRecord(22, tTransferTrain + tFlow21 + ...
                                                tNewBags) + so1 * or1 * hbs1 * NewBags
                                        End If

                                        'Test if the newly added bag also adds an extra minute to ...
                                            unloading time, if so, block off an extra minute for ...
                                            the unload slot
                                        If tNewBags > tOldBags Then
                                            SlotRecord(2, tTransferTrain + tFlow21 + tNewBags) = ...
                                                SlotRecord(2, tTransferTrain + tFlow21 + tNewBags) ...
                                                + 1
                                        End If
                                    End If

                            End Select
                        Next k
                    End If
                Next j
            End If
        Next i
    End Sub
```

K.7. DEPARTURE PROCESS

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''
'''  Code that determines how baggage trains on the departure side are handled
'''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Sub DepartureProcess()
    Dim BagTrainRow As Integer
    Dim TrainLength As Double
    Dim MaxTrainLength As Double
    Dim TotalBags As Double
    Dim UnitCapacity As Double
    Dim TotalUnits As Double
    Dim NewTrainBags As Double
    Dim ThisTrainBags As Double
    Dim tTransferTrain As Long

    Dim Character As String

```

```
    Dim i As Long
    Dim j As Long
    Dim k As Long

    'Calculate how many bags transport is needed between make-up and departing flights
    For i = LBound(DepartureFlightList, 1) To UBound(DepartureFlightList, 1)
        If Not IsEmpty(DepartureFlightList(i, 1)) Then
            'Add departing flight to departing flight flow
            ProcessRecord(16, DepartureFlightList(i, 1)) = ProcessRecord(16, ...
                DepartureFlightList(i, 1)) + DepartureFlightList(i, 5) + ...
                DepartureFlightList(i, 6)

            Character = Schedule.Cells(DepartureFlightList(i, 2), "D")
            Select Case Character
                Case Options.Cells(5, "D")
                    BagTrainRow = 23
                Case Options.Cells(6, "D")
                    BagTrainRow = 24
                Case Options.Cells(7, "D")
                    BagTrainRow = 25
                Case Options.Cells(8, "D")
                    BagTrainRow = 26
            End Select

            MaxTrainLength = Calculation.Cells(BagTrainRow, "K")
            UnitCapacity = Calculation.Cells(BagTrainRow, "L")
            TotalBags = DepartureFlightList(i, 5) + DepartureFlightList(i, 6)

            'Mark loading time of aircraft in the SlotRecord list
            For j = MakeUpList(i, -1) To MakeUpList(i, -2)
                SlotRecord(1, j) = SlotRecord(1, j) + 1
            Next j

            NewTrainBags = 0
            TotalUnits = 0
            For j = 0 To (MakeUpList(i, -2) - MakeUpList(i, -1))
                NewTrainBags = NewTrainBags + MakeUpList(i, j)

                If NewTrainBags > (MaxTrainLength * UnitCapacity) Or j ≥ (MakeUpList(i, -2) - ...
                    MakeUpList(i, -1)) Then
                    If j ≥ (MakeUpList(i, -2) - MakeUpList(i, -1)) Then
                        ThisTrainBags = NewTrainBags
                    Else
                        ThisTrainBags = WorksheetFunction.Min(NewTrainBags, MaxTrainLength * ...
                            UnitCapacity)
                    End If
                    NewTrainBags = WorksheetFunction.Max(0, NewTrainBags - ThisTrainBags)
                    TrainLength = WorksheetFunction.RoundUp(ThisTrainBags / UnitCapacity, 0)
                    TotalUnits = TotalUnits + TrainLength

                    tTransferTrain = MakeUpList(i, -1) + j
                    FlowRecord(42, tTransferTrain) = FlowRecord(42, tTransferTrain) + ...
                        ThisTrainBags

                    For k = tTransferTrain - WorksheetFunction.RoundUp(ThisTrainBags / ...
                        tLoad_Bag, 0) To tTransferTrain
                        SlotRecord(4, k) = SlotRecord(4, k) + 1
                    Next k

                    For k = tTransferTrain To DepartureFlightList(i, 1) - tDeparting - ...
                        WorksheetFunction.RoundUp(TotalUnits * Calculation.Cells(BagTrainRow, ...
                        "M"), 0)
                        ProcessRecord(19, k) = ProcessRecord(19, k) + TrainLength
                    Next k
                End If
            Next j
        End If
    Next i
End Sub
```

K.8. ARRIVAL PROCESS

```
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''
'''  Code that determines how baggage trains on the terminating (arrival) side are handled
'''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Sub ArrivalProcess()
    Dim BagTrainRow As Integer
```

```
    Dim TerminatingPassengers As Long
    Dim NewBags As Double
    Dim tNewBags As Long

    Dim TrainCapacity As Double
    Dim TrainAmount As Long
    Dim TrainBags As Double
    Dim RemainingBags As Double
    Dim UnloadedBags As Double
    Dim tTransferTrain As Long
    Dim tTerminatingTrain As Long

    Dim Character As String

    Dim i As Long
    Dim j As Long
    Dim k As Long

    'This case only handles the amount of terminating bags for locally arriving aircraft, ...
        flow20 is calculated later
    'because this flow can only be determined after all transfer bags have been unloaded.
    For i = LBound(ArrivalFlightList, 1) To UBound(ArrivalFlightList, 1)
        If Not IsEmpty(ArrivalFlightList(i, 1)) Then
            'Calculate how many passengers are on board and how many bags they have in total.
            TerminatingPassengers = ArrivalFlightList(i, 3)
            NewBags = 0
            If TerminatingPassengers > 0 Then
                For j = 1 To TerminatingPassengers
                    NewBags = NewBags + Round(Probability(Calculation.Cells(7, "K"), ...
                        Schedule.Cells(ArrivalFlightList(i, 2), "I"), ...
                        Schedule.Cells(ArrivalFlightList(i, 2), "K"), ...
                        Schedule.Cells(ArrivalFlightList(i, 2), "J"), Rnd), 2)
                Next j
            End If

            'Add the total amount of bags to the ArrivalFlightList containing flight data and ...
                if the aircraft is handled by the local BHS, add it to ProcessRecord(15, ..)
            ArrivalFlightList(i, 5) = NewBags
            If Schedule.Cells(ArrivalFlightList(i, 2), "X") = Options.Cells(5, "F") Then
                ProcessRecord(15, ArrivalFlightList(i, 1)) = ProcessRecord(15, ...
                    ArrivalFlightList(i, 1)) + NewBags

                Character = Schedule.Cells(ArrivalFlightList(i, 2), "D")
                Select Case Character
                    Case Options.Cells(5, "D")
                        BagTrainRow = 23
                    Case Options.Cells(6, "D")
                        BagTrainRow = 24
                    Case Options.Cells(7, "D")
                        BagTrainRow = 25
                    Case Options.Cells(8, "D")
                        BagTrainRow = 26
                End Select

                'Determine the time that the transfer baggage will be finished and terminating ...
                    baggage can be started
                TrainCapacity = Calculation.Cells(BagTrainRow, "K") * ...
                    Calculation.Cells(BagTrainRow, "L")
                tTransferTrain = ArrivalFlightList(i, 1) + tArriving + ...
                    WorksheetFunction.RoundUp(ArrivalFlightList(i, 6) / TrainCapacity, 0) * ...
                    Round(Calculation.Cells(BagTrainRow, "K") * Calculation.Cells(BagTrainRow, ...
                    "N"), 0)

                TrainAmount = WorksheetFunction.RoundUp(NewBags / TrainCapacity, 0)
                RemainingBags = NewBags
                If TrainAmount > 0 Then
                    For j = 1 To TrainAmount
                        TrainBags = WorksheetFunction.Min(TrainCapacity, RemainingBags)
                        RemainingBags = RemainingBags - TrainBags

                        If j = 1 Then
                            tTerminatingTrain = tTransferTrain + ...
                                WorksheetFunction.RoundUp(TrainBags / ...
                                Calculation.Cells(BagTrainRow, "L"), 0) * ...
                                Calculation.Cells(BagTrainRow, "N")
                        Else
                            tTerminatingTrain = tTerminatingTrain + ...
                                WorksheetFunction.RoundUp(TrainBags / ...
                                Calculation.Cells(BagTrainRow, "L"), 0) * ...
                                Calculation.Cells(BagTrainRow, "N")
                        End If

                        FlowRecord(43, tTerminatingTrain) = FlowRecord(43, tTerminatingTrain) ...
                            + TrainBags
```

```
                        tNewBags = WorksheetFunction.RoundUp(TrainBags / tUnload_Bag, 0)
                        For k = 0 To tNewBags - 1
                            UnloadedBags = WorksheetFunction.Min(tUnload_Bag, TrainBags)
                            TrainBags = TrainBags - UnloadedBags
                            ProcessRecord(13, tTerminatingTrain + tFlow20 + k) = ...
                                ProcessRecord(13, tTerminatingTrain + tFlow20 + k) + UnloadedBags
                            SlotRecord(3, tTerminatingTrain + tFlow20 + k) = SlotRecord(3, ...
                                tTerminatingTrain + tFlow20 + k) + 1
                        Next k
                    Next j
                End If
            End If
        End If
    Next i
End Sub
```

K.9. DISCRETE TIME

```
''''''''''''''''''''''''''''''''''
'''
'''  The time step calculation
'''
''''''''''''''''''''''''''''''''''
Sub DiscreteTime()
    Dim i As Long

    'Run the simulation to calculate all intermediate steps (recursive equations)
    For i = 0 To SimLength
        'Calculate terminating flow to reclaim
        FlowRecord(10, i) = (1 - cs1) * te1 * ProcessRecord(13, i)
        FlowRecord(11, i) = cs1 * te1 * ProcessRecord(13, i)
        FlowRecord(12, i) = (1 - te1) * ProcessRecord(13, i)

        FlowRecord(39, i) = cs1 * FlowRecord(12, i - tFlow5 - tSorting)

        ProcessRecord(11, i) = FlowRecord(11, i - tFlow4) + FlowRecord(39, i - tFlow16)

        FlowRecord(38, i) = (1 - cu1) * ProcessRecord(11, i - tCustoms)
        FlowRecord(40, i) = (1 - cs1) * FlowRecord(12, i - tFlow5 - tSorting) + FlowRecord(38, ...
            i - tFlow15 - tSorting)
        FlowRecord(41, i) = cu1 * ProcessRecord(11, i - tCustoms)

        ProcessRecord(12, i) = FlowRecord(10, i - tFlow3) + FlowRecord(40, i - tFlow17) + ...
            FlowRecord(41, i - tFlow18)

        'Calculate the rest of the system
        FlowRecord(1, i) = FlowRecord(2, i) + FlowRecord(3, i) + FlowRecord(4, i) + ...
            FlowRecord(5, i)
        FlowRecord(6, i) = FlowRecord(7, i) + FlowRecord(8, i) + FlowRecord(9, i)

        FlowRecord(14, i) = FlowRecord(15, i) + FlowRecord(16, i) + FlowRecord(17, i) + ...
            FlowRecord(18, i)
        FlowRecord(19, i) = FlowRecord(20, i) + FlowRecord(21, i) + FlowRecord(22, i)

        FlowRecord(35, i) = FlowRecord(36, i) + FlowRecord(37, i)
        FlowRecord(45, i) = FlowRecord(46, i) + FlowRecord(47, i) + FlowRecord(48, i) + ...
            FlowRecord(49, i)

        FlowRecord(26, i) = FlowRecord(2, i - tSorting - tFlow1) + FlowRecord(15, i - tSorting ...
            - tFlow7) + FlowRecord(46, i - tSorting - tFlow26)
        FlowRecord(27, i) = FlowRecord(3, i - tSorting - tFlow1) + FlowRecord(16, i - tSorting ...
            - tFlow7) + FlowRecord(47, i - tSorting - tFlow26)
        FlowRecord(28, i) = FlowRecord(4, i - tSorting - tFlow1) + FlowRecord(17, i - tSorting ...
            - tFlow7)
        FlowRecord(25, i) = FlowRecord(26, i) + FlowRecord(27, i) + FlowRecord(28, i)

        ProcessRecord(5, i) = FlowRecord(7, i - tFlow1) + FlowRecord(20, i - tFlow8) + ...
            FlowRecord(26, i - tFlow11)
        ProcessRecord(6, i) = FlowRecord(8, i - tFlow1) + FlowRecord(21, i - tFlow8) + ...
            FlowRecord(27, i - tFlow11)
        ProcessRecord(7, i) = FlowRecord(9, i - tFlow2) + FlowRecord(22, i - tFlow8) + ...
            FlowRecord(28, i - tFlow11)
        ProcessRecord(4, i) = ProcessRecord(5, i) + ProcessRecord(6, i) + ProcessRecord(7, i)

        FlowRecord(24, i) = sc1 * ProcessRecord(5, i - tSecurity)
        FlowRecord(30, i) = (1 - sc1) * ProcessRecord(5, i - tSecurity)
        FlowRecord(31, i) = ProcessRecord(6, i - tSecurity)
        FlowRecord(32, i) = ProcessRecord(7, i - tSecurity)
        FlowRecord(29, i) = FlowRecord(30, i) + FlowRecord(31, i) + FlowRecord(32, i)
```

```
        FlowRecord(23, i) = FlowRecord(30, i - tSorting - tFlow12) + FlowRecord(36, i - ...
            tSorting - tFlow14) + FlowRecord(49, i - tSorting - tFlow26)
        FlowRecord(34, i) = FlowRecord(31, i - tSorting - tFlow12) + FlowRecord(48, i - ...
            tSorting - tFlow26)

        FlowRecord(50, i) = FlowRecord(5, i - tSorting - tFlow1) + FlowRecord(18, i - tSorting ...
            - tFlow1) + FlowRecord(32, i - tSorting - tFlow12) + FlowRecord(37, i - tSorting - ...
            tFlow14)

        ProcessRecord(8, i) = ProcessRecord(8, i - 1) + FlowRecord(23, i) + FlowRecord(24, i) ...
            - FlowRecord(42, i)
        ProcessRecord(9, i) = FlowRecord(23, i - tFlow9) + FlowRecord(24, i - tFlow10)
        ProcessRecord(18, i) = FlowRecord(50, i - tFlow27)

        ProcessRecord(10, i) = ProcessRecord(10, i - 1) + FlowRecord(34, i) - FlowRecord(35, i)
        ProcessRecord(2, i) = ProcessRecord(2, i - 1) + FlowRecord(1, i) + FlowRecord(12, i) + ...
            FlowRecord(14, i) + FlowRecord(29, i) + FlowRecord(35, i) + FlowRecord(38, i) + ...
            FlowRecord(45, i) - FlowRecord(23, i) - FlowRecord(25, i) - FlowRecord(34, i) - ...
            FlowRecord(39, i) - FlowRecord(40, i) - FlowRecord(50, i)
        ProcessRecord(3, i) = FlowRecord(1, i - tFlow1) + FlowRecord(12, i - tFlow5) + ...
            FlowRecord(14, i - tFlow7) + FlowRecord(29, i - tFlow12) + FlowRecord(35, i - ...
            tFlow14) + FlowRecord(38, i - tFlow15) + FlowRecord(45, i - tFlow26)
    Next i
End Sub
```

K.10. DATA PROCESSING

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''
'''  Processing the calculation data into minimum, maximum, average, etc.
'''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Sub DataProcessing()
    Dim Temp As Double
    Dim WindowStart As Long
    Dim WindowEnd As Long
    Dim FuncAdd As Double
    Dim FuncSub As Double
    Dim FlowAdd As Double
    Dim FlowSub As Double
    Dim SlotAdd As Double
    Dim SlotSub As Double
    Dim MultiplyResult As Double

    ReDim ProcessPrint(LBound(BoxPlotData, 1) To UBound(BoxPlotData, 1), 1 To 3) As Double
    ReDim FlowPrint(LBound(FlowBoxPlot, 1) To UBound(FlowBoxPlot, 1), 1 To 3) As Double
    ReDim SlotPrint(LBound(SlotBoxPlot, 1) To UBound(SlotBoxPlot, 1), 1 To 3) As Double

    Dim i As Long
    Dim j As Long
    Dim k As Long

    'Determine the maximum [minute/5-minute/hourly] baggage flow, first calculate minute max ...
        and then multiply by hour! as REAL MAXIMUM
    'But also show maximum hour average! as REAL MINIMUM. Do this all in an array, so there ...
        are not so many variables and code needed!
    'ProcessData array: | (0) Temporary value | (1) SimulationTotalFlow | (2) MinuteMax | (3) ...
        tCount-Average | (4) MinuteMinimum | (5) LocationOfMax |
    'FlowData array: | (0) Temporary value | (1) SimulationTotalFlow | (2) MinuteMax | (3) ...
        tCount-Average | (4) MinuteMinimum | (5) LocationOfMax |
    'SlotsData array: | (0) Temporary value | (1) SimulationTotalFlow | (2) MinuteMax | (3) ...
        tCount-Average | (4) MinuteMinimum | (5) LocationOfMax |
    For i = 0 To SimLength
        ''''''''''''''''''''''''''''''''''''''''''''''
        '''
        '''  Perform data analysis on BHS functions
        '''
        ''''''''''''''''''''''''''''''''''''''''''''''
        For j = 1 To 19
            'Add for total sum
            ProcessData(j, 1) = ProcessData(j, 1) + ProcessRecord(j, i)

            'Compare for maximum per minute and denote the time
            If ProcessRecord(j, i) > ProcessData(j, 2) Then
                ProcessData(j, 2) = ProcessRecord(j, i)
                ProcessData(j, 5) = i
            End If

```

```
            'Compare for minimum per minute
            If ProcessRecord(j, i) < ProcessData(j, 4) Then
                ProcessData(j, 4) = ProcessRecord(j, i)
            End If

            'Determine average over time = tCount
            WindowStart = WorksheetFunction.Max(0, i - tCount)
            WindowEnd = i

            FuncAdd = ProcessRecord(j, WindowEnd)
            If WindowStart = 0 Then
                FuncSub = 0
            Else
                FuncSub = ProcessRecord(j, WindowStart)
            End If

            ProcessData(j, 0) = ProcessData(j, 0) + FuncAdd - FuncSub
            If ProcessData(j, 0) > ProcessData(j, 3) Then
                ProcessData(j, 3) = ProcessData(j, 0)
            End If
        Next j

        '''''''''''''''''''''''''''''''''''''''''''''''
        '''
        '''   Perform data analysis on BHS flows
        '''
        '''''''''''''''''''''''''''''''''''''''''''''''
        For j = 1 To 50
            'Add for total sum
            FlowData(j, 1) = FlowData(j, 1) + FlowRecord(j, i)

            'Compare for maximum per minute and denote the time
            If FlowRecord(j, i) > FlowData(j, 2) Then
                FlowData(j, 2) = FlowRecord(j, i)
                FlowData(j, 5) = i
            End If

            'Compare for minimum per minute
            If FlowRecord(j, i) < FlowData(j, 4) Then
                FlowData(j, 4) = FlowRecord(j, i)
            End If

            'Determine average over time = tCount
            WindowStart = WorksheetFunction.Max(0, i - tCount)
            WindowEnd = i

            FlowAdd = FlowRecord(j, WindowEnd)
            If WindowStart = 0 Then
                FlowSub = 0
            Else
                FlowSub = FlowRecord(j, WindowStart)
            End If

            FlowData(j, 0) = FlowData(j, 0) + FlowAdd - FlowSub
            If FlowData(j, 0) > FlowData(j, 3) Then
                FlowData(j, 3) = FlowData(j, 0)
            End If
        Next j

        '''''''''''''''''''''''''''''''''''''''''''''''
        '''
        '''   Perform data analysis on BHS slots
        '''
        '''''''''''''''''''''''''''''''''''''''''''''''
        For j = 1 To 4
            'Add for total sum
            SlotData(j, 1) = SlotData(j, 1) + SlotRecord(j, i)

            'Compare for maximum per minute and denote the time
            If SlotRecord(j, i) > SlotData(j, 2) Then
                SlotData(j, 2) = SlotRecord(j, i)
                SlotData(j, 5) = i
            End If

            'Compare for minimum per minute
            If SlotRecord(j, i) < SlotData(j, 4) Then
                SlotData(j, 4) = SlotRecord(j, i)
            End If

            'Determine average over time = tCount
            WindowStart = WorksheetFunction.Max(0, i - tCount)
            WindowEnd = i

            SlotAdd = SlotRecord(j, WindowEnd)
```

```
                If WindowStart = 0 Then
                    SlotSub = 0
                Else
                    SlotSub = SlotRecord(j, WindowStart)
                End If

                SlotData(j, 0) = SlotData(j, 0) + SlotAdd - SlotSub
                If SlotData(j, 0) > SlotData(j, 3) Then
                    SlotData(j, 3) = SlotData(j, 0)
                End If
            Next j
        Next i

        'Categorize occurences between min/max in a number of cMode categories
        For i = 0 To SimLength
            'Process data
            For j = 1 To 19
                Temp = (ProcessData(j, 2) - (ProcessData(j, 3) / tCount)) / cMode
                If ProcessRecord(j, i) > ProcessData(j, 3) / tCount Then
                    For k = 1 To cMode
                        If ProcessRecord(j, i) < (ProcessData(j, 3) / tCount) + k * Temp And ...
                             ProcessRecord(j, i) > (ProcessData(j, 3) / tCount) + (k - 1) * Temp Then
                            BoxPlotData(j, k) = BoxPlotData(j, k) + 1
                        End If
                    Next k
                End If
            Next j

            'Flow data
            For j = 1 To 50
                Temp = (FlowData(j, 2) - (FlowData(j, 3) / tCount)) / cMode
                If FlowRecord(j, i) > FlowData(j, 3) / tCount Then
                    For k = 1 To cMode
                        If FlowRecord(j, i) < (FlowData(j, 3) / tCount) + k * Temp And ...
                             FlowRecord(j, i) > (FlowData(j, 3) / tCount) + (k - 1) * Temp Then
                            FlowBoxPlot(j, k) = FlowBoxPlot(j, k) + 1
                        End If
                    Next k
                End If
            Next j

            'Slot data
            For j = 1 To 4
                Temp = (SlotData(j, 2) - (SlotData(j, 3) / tCount)) / cMode
                If SlotRecord(j, i) > SlotData(j, 3) / tCount Then
                    For k = 1 To cMode
                        If SlotRecord(j, i) < (SlotData(j, 3) / tCount) + k * Temp And ...
                             SlotRecord(j, i) > (SlotData(j, 3) / tCount) + (k - 1) * Temp Then
                            SlotBoxPlot(j, k) = SlotBoxPlot(j, k) + 1
                        End If
                    Next k
                End If
            Next j
        Next i

        'Determine which cMode is most likely to occur and its amount of bags
        'Process data
        For i = 1 To 19
            Temp = 1
            MultiplyResult = Hour
            Select Case i
                Case 2, 8, 10, 19
                    MultiplyResult = 1
            End Select

            ProcessPrint(i, 1) = (ProcessData(i, 3) / tCount) * MultiplyResult
            ProcessPrint(i, 3) = ProcessData(i, 2) * MultiplyResult
            ProcessPrint(i, 2) = (ProcessPrint(i, 1) + ProcessPrint(i, 3)) / 2
            For j = LBound(BoxPlotData, 2) To UBound(BoxPlotData, 2)
                If BoxPlotData(i, j) ≥ Temp Then
                    Temp = BoxPlotData(i, j)
                    ProcessPrint(i, 2) = ((j * ((ProcessData(i, 2) - (ProcessData(i, 3) / tCount)) ...
                         / cMode)) + (ProcessData(i, 3) / tCount)) * MultiplyResult
                End If
            Next j
        Next i

        For i = 1 To 50
            Temp = 1
            FlowPrint(i, 1) = (FlowData(i, 3) / tCount) * Hour
            FlowPrint(i, 3) = FlowData(i, 2) * Hour
            FlowPrint(i, 2) = (FlowPrint(i, 1) + FlowPrint(i, 3)) / 2
            For j = LBound(FlowBoxPlot, 2) To UBound(FlowBoxPlot, 2)
                If FlowBoxPlot(i, j) ≥ Temp Then
```

```
                Temp = FlowBoxPlot(i, j)
                FlowPrint(i, 2) = ((j * ((FlowData(i, 2) - (FlowData(i, 3) / tCount)) / ...
                    cMode)) + (FlowData(i, 3) / tCount)) * Hour
            End If
        Next j
    Next i

    For i = 1 To 4
        Temp = 1
        SlotPrint(i, 1) = (SlotData(i, 3) / tCount)
        SlotPrint(i, 3) = SlotData(i, 2)
        SlotPrint(i, 2) = (SlotPrint(i, 1) + SlotPrint(i, 3)) / 2
        For j = LBound(SlotBoxPlot, 2) To UBound(SlotBoxPlot, 2)
            If SlotBoxPlot(i, j) ≥ Temp Then
                Temp = SlotBoxPlot(i, j)
                SlotPrint(i, 2) = ((j * ((SlotData(i, 2) - (SlotData(i, 3) / tCount)) / ...
                    cMode)) + (SlotData(i, 3) / tCount))
            End If
        Next j
    Next i

    Results.Range("B2:B2").Resize(UBound(ProcessPrint, 1) - LBound(ProcessPrint, 1) + 1, ...
        UBound(ProcessPrint, 2) - LBound(ProcessPrint, 2) + 1) = ProcessPrint
    Results.Range("B24:B24").Resize(UBound(FlowPrint, 1) - LBound(FlowPrint, 1) + 1, ...
        UBound(FlowPrint, 2) - LBound(FlowPrint, 2) + 1) = FlowPrint
    Results.Range("B77:B77").Resize(UBound(SlotPrint, 1) - LBound(SlotPrint, 1) + 1, ...
        UBound(SlotPrint, 2) - LBound(SlotPrint, 2) + 1) = SlotPrint
End Sub
```

K.11. CALCULATION 1

```
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''
'''  Processing the calculation data into minimum, maximum, average, etc.
'''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Sub CalcUnit()
    Dim nCounters As Double
    Dim nBelt As Double
    Dim nPier As Double
    Dim nTray As Double
    Dim nSort As Double
    Dim nEBS As Double

    Dim i As Long
    Dim j As Long
    Dim k As Long
    Dim l As Long
    Dim flow As Long

    Dim OperationalYear As Double

    Dim nMerges As Double
    Dim nDiverters As Double

    nCounters = 0.5
    nBelt = 0.75
    nPier = 0.4
    nTray = 0.75
    nSort = 0.8
    nEBS = 0.4
    OperationalYear = 365 * 24


    'Check-ins
    i = 4
    j = 4
    k = 4
    Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(1, 1) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(1, 2) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(1, 3) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)

    Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
```

```
    Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))

    Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nCounters * OperationalYear)
    Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nCounters * OperationalYear)
    Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nCounters * OperationalYear)

    Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Parts.Cells(k, "P")

    Systems.Cells(i + 0, j + 4) = Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nCounters * ...
        OperationalYear + (Systems.Cells(i + 0, j + 2) * cEnergy)
    Systems.Cells(i + 1, j + 4) = Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nCounters * ...
        OperationalYear + (Systems.Cells(i + 1, j + 2) * cEnergy)
    Systems.Cells(i + 2, j + 4) = Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nCounters * ...
        OperationalYear + (Systems.Cells(i + 2, j + 2) * cEnergy)

    j = 11
    k = 5
    Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(1, 1) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(1, 2) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(1, 3) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)

    Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))

    Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nCounters * OperationalYear)
    Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nCounters * OperationalYear)
    Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nCounters * OperationalYear)

    Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Parts.Cells(k, "P")

    Systems.Cells(i + 0, j + 4) = Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nCounters * ...
        OperationalYear + (Systems.Cells(i + 0, j + 2) * cEnergy)
    Systems.Cells(i + 1, j + 4) = Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nCounters * ...
        OperationalYear + (Systems.Cells(i + 1, j + 2) * cEnergy)
    Systems.Cells(i + 2, j + 4) = Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nCounters * ...
        OperationalYear + (Systems.Cells(i + 2, j + 2) * cEnergy)

    j = 18
    k = 6
    Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(1, 1) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(1, 2) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(1, 3) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)

    Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))

    Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nCounters * OperationalYear)
    Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nCounters * OperationalYear)
    Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nCounters * OperationalYear)

    Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Parts.Cells(k, "P")

    Systems.Cells(i + 0, j + 4) = Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nCounters * ...
        OperationalYear + (Systems.Cells(i + 0, j + 2) * cEnergy)
```

```
    Systems.Cells(i + 1, j + 4) = Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nCounters * ...
        OperationalYear + (Systems.Cells(i + 1, j + 2) * cEnergy)
    Systems.Cells(i + 2, j + 4) = Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nCounters * ...
        OperationalYear + (Systems.Cells(i + 2, j + 2) * cEnergy)

    j = 25
    k = 7
    Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(1, 1) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(1, 2) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(1, 3) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)

    Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))

    Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nCounters * OperationalYear)
    Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nCounters * OperationalYear)
    Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nCounters * OperationalYear)

    Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Parts.Cells(k, "P")

    Systems.Cells(i + 0, j + 4) = Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nCounters * ...
        OperationalYear + (Systems.Cells(i + 0, j + 2) * cEnergy)
    Systems.Cells(i + 1, j + 4) = Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nCounters * ...
        OperationalYear + (Systems.Cells(i + 1, j + 2) * cEnergy)
    Systems.Cells(i + 2, j + 4) = Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nCounters * ...
        OperationalYear + (Systems.Cells(i + 2, j + 2) * cEnergy)



    'Screening
    i = 22
    j = 4
    k = 34
    Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 1) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 2) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 3) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)

    Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))

    Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)

    Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Parts.Cells(k, "P")

    Systems.Cells(i + 0, j + 4) = Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Systems.Cells(i + 0, j + 2) * cEnergy)
    Systems.Cells(i + 1, j + 4) = Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Systems.Cells(i + 1, j + 2) * cEnergy)
    Systems.Cells(i + 2, j + 4) = Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Systems.Cells(i + 2, j + 2) * cEnergy)

    j = 11
    k = 35
    Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 1) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 2) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
```

```
    Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 3) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)

    Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))

    Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)

    Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Parts.Cells(k, "P")

    Systems.Cells(i + 0, j + 4) = Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Systems.Cells(i + 0, j + 2) * cEnergy)
    Systems.Cells(i + 1, j + 4) = Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Systems.Cells(i + 1, j + 2) * cEnergy)
    Systems.Cells(i + 2, j + 4) = Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Systems.Cells(i + 2, j + 2) * cEnergy)

    j = 18
    k = 36
    Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 1) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 2) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 3) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)

    Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))

    Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)

    Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Parts.Cells(k, "P")

    Systems.Cells(i + 0, j + 4) = Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Systems.Cells(i + 0, j + 2) * cEnergy)
    Systems.Cells(i + 1, j + 4) = Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Systems.Cells(i + 1, j + 2) * cEnergy)
    Systems.Cells(i + 2, j + 4) = Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Systems.Cells(i + 2, j + 2) * cEnergy)

    j = 25
    k = 38
    Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 1) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 2) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 3) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)

    Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))

    Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
```

```

Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Parts.Cells(k, "P")
Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Parts.Cells(k, "P")
Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Parts.Cells(k, "P")

Systems.Cells(i + 0, j + 4) = Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
    OperationalYear + (Systems.Cells(i + 0, j + 2) * cEnergy)
Systems.Cells(i + 1, j + 4) = Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
    OperationalYear + (Systems.Cells(i + 1, j + 2) * cEnergy)
Systems.Cells(i + 2, j + 4) = Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
    OperationalYear + (Systems.Cells(i + 2, j + 2) * cEnergy)


'Transfer
i = 10
j = 4
k = 10
Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(14, 1) * (1 + ...
    fRedundant)) / Parts.Cells(k, "G"), 0)
Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(14, 2) * (1 + ...
    fRedundant)) / Parts.Cells(k, "G"), 0)
Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(14, 3) * (1 + ...
    fRedundant)) / Parts.Cells(k, "G"), 0)

Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
    Parts.Cells(k, "I"))
Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
    Parts.Cells(k, "I"))
Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
    Parts.Cells(k, "I"))

Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
    Parts.Cells(k, "L") * nBelt * OperationalYear)
Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
    Parts.Cells(k, "L") * nBelt * OperationalYear)
Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
    Parts.Cells(k, "L") * nBelt * OperationalYear)

Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Parts.Cells(k, "P") * ...
    Parts.Cells(k, "H")
Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Parts.Cells(k, "P") * ...
    Parts.Cells(k, "H")
Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Parts.Cells(k, "P") * ...
    Parts.Cells(k, "H")

Systems.Cells(i + 0, j + 4) = Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
    OperationalYear + (Systems.Cells(i + 0, j + 2) * cEnergy)
Systems.Cells(i + 1, j + 4) = Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
    OperationalYear + (Systems.Cells(i + 1, j + 2) * cEnergy)
Systems.Cells(i + 2, j + 4) = Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
    OperationalYear + (Systems.Cells(i + 2, j + 2) * cEnergy)

j = 11
k = 11
Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(14, 1) * (1 + ...
    fRedundant)) / Parts.Cells(k, "G"), 0)
Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(14, 2) * (1 + ...
    fRedundant)) / Parts.Cells(k, "G"), 0)
Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(14, 3) * (1 + ...
    fRedundant)) / Parts.Cells(k, "G"), 0)

Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
    Parts.Cells(k, "I"))
Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
    Parts.Cells(k, "I"))
Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
    Parts.Cells(k, "I"))

Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
    Parts.Cells(k, "L") * nBelt * OperationalYear)
Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
    Parts.Cells(k, "L") * nBelt * OperationalYear)
Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
    Parts.Cells(k, "L") * nBelt * OperationalYear)

Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Parts.Cells(k, "P")
Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Parts.Cells(k, "P")
Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Parts.Cells(k, "P")

Systems.Cells(i + 0, j + 4) = Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
    OperationalYear + (Systems.Cells(i + 0, j + 2) * cEnergy)
Systems.Cells(i + 1, j + 4) = Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
    OperationalYear + (Systems.Cells(i + 1, j + 2) * cEnergy)
```

```
    Systems.Cells(i + 2, j + 4) = Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Systems.Cells(i + 2, j + 2) * cEnergy)

    j = 18
    k = 12
    Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(14, 1) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(14, 2) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(14, 3) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)

    Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))

    Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)

    Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Parts.Cells(k, "P")

    Systems.Cells(i + 0, j + 4) = Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Systems.Cells(i + 0, j + 2) * cEnergy)
    Systems.Cells(i + 1, j + 4) = Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Systems.Cells(i + 1, j + 2) * cEnergy)
    Systems.Cells(i + 2, j + 4) = Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Systems.Cells(i + 2, j + 2) * cEnergy)


    'Incoming
    i = 16
    j = 4
    k = 15
    Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(17, 1) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(17, 2) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(17, 3) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)

    Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))

    Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nPier * OperationalYear)
    Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nPier * OperationalYear)
    Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nPier * OperationalYear)

    Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Parts.Cells(k, "P") * ...
        Parts.Cells(k, "H")
    Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Parts.Cells(k, "P") * ...
        Parts.Cells(k, "H")
    Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Parts.Cells(k, "P") * ...
        Parts.Cells(k, "H")

    Systems.Cells(i + 0, j + 4) = Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nPier * ...
        OperationalYear + (Systems.Cells(i + 0, j + 2) * cEnergy)
    Systems.Cells(i + 1, j + 4) = Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nPier * ...
        OperationalYear + (Systems.Cells(i + 1, j + 2) * cEnergy)
    Systems.Cells(i + 2, j + 4) = Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nPier * ...
        OperationalYear + (Systems.Cells(i + 2, j + 2) * cEnergy)

    j = 11
    k = 16
    Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(17, 1) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(17, 2) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
```

```
    Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(17, 3) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)

    Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))

    Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nPier * OperationalYear)
    Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nPier * OperationalYear)
    Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nPier * OperationalYear)

    Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Parts.Cells(k, "P")
    Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Parts.Cells(k, "P")

    Systems.Cells(i + 0, j + 4) = Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nPier * ...
        OperationalYear + (Systems.Cells(i + 0, j + 2) * cEnergy)
    Systems.Cells(i + 1, j + 4) = Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nPier * ...
        OperationalYear + (Systems.Cells(i + 1, j + 2) * cEnergy)
    Systems.Cells(i + 2, j + 4) = Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nPier * ...
        OperationalYear + (Systems.Cells(i + 2, j + 2) * cEnergy)


    'Storage
    i = 34
    j = 4
    k = 82
    Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(10, 1) * (1 + 0.1)), 0)
    Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(10, 2) * (1 + 0.1)), 0)
    Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(10, 3) * (1 + 0.1)), 0)

    Systems.Cells(i + 0, j + 1) = (Systems.Cells(i + 0, j) / Parts.Cells(k, "E")) * ...
        Parts.Cells(k, "I")
    Systems.Cells(i + 1, j + 1) = (Systems.Cells(i + 1, j) / Parts.Cells(k, "E")) * ...
        Parts.Cells(k, "I")
    Systems.Cells(i + 2, j + 1) = (Systems.Cells(i + 2, j) / Parts.Cells(k, "E")) * ...
        Parts.Cells(k, "I")

    Systems.Cells(i + 0, j + 2) = (Systems.Cells(i + 0, j) / Parts.Cells(k, "E")) * ...
        (Parts.Cells(k, "K") * Parts.Cells(k, "L") * nEBS * OperationalYear)
    Systems.Cells(i + 1, j + 2) = (Systems.Cells(i + 1, j) / Parts.Cells(k, "E")) * ...
        (Parts.Cells(k, "K") * Parts.Cells(k, "L") * nEBS * OperationalYear)
    Systems.Cells(i + 2, j + 2) = (Systems.Cells(i + 2, j) / Parts.Cells(k, "E")) * ...
        (Parts.Cells(k, "K") * Parts.Cells(k, "L") * nEBS * OperationalYear)

    Systems.Cells(i + 0, j + 3) = (Systems.Cells(i + 0, j) / Parts.Cells(k, "E")) * ...
        Parts.Cells(k, "P")
    Systems.Cells(i + 1, j + 3) = (Systems.Cells(i + 1, j) / Parts.Cells(k, "E")) * ...
        Parts.Cells(k, "P")
    Systems.Cells(i + 2, j + 3) = (Systems.Cells(i + 2, j) / Parts.Cells(k, "E")) * ...
        Parts.Cells(k, "P")

    Systems.Cells(i + 0, j + 4) = (Systems.Cells(i + 0, j) / Parts.Cells(k, "E")) * ...
        Parts.Cells(k, "O") * nEBS * OperationalYear + (Systems.Cells(i + 0, j + 2) * cEnergy)
    Systems.Cells(i + 1, j + 4) = (Systems.Cells(i + 1, j) / Parts.Cells(k, "E")) * ...
        Parts.Cells(k, "O") * nEBS * OperationalYear + (Systems.Cells(i + 1, j + 2) * cEnergy)
    Systems.Cells(i + 2, j + 4) = (Systems.Cells(i + 2, j) / Parts.Cells(k, "E")) * ...
        Parts.Cells(k, "O") * nEBS * OperationalYear + (Systems.Cells(i + 2, j + 2) * cEnergy)

    j = 11
    k = 83
    Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(10, 1) * (1 + 0.1)), 0)
    Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(10, 2) * (1 + 0.1)), 0)
    Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(10, 3) * (1 + 0.1)), 0)

    Systems.Cells(i + 0, j + 1) = (Systems.Cells(i + 0, j) / Parts.Cells(k, "E")) * ...
        Parts.Cells(k, "I")
    Systems.Cells(i + 1, j + 1) = (Systems.Cells(i + 1, j) / Parts.Cells(k, "E")) * ...
        Parts.Cells(k, "I")
    Systems.Cells(i + 2, j + 1) = (Systems.Cells(i + 2, j) / Parts.Cells(k, "E")) * ...
        Parts.Cells(k, "I")

    Systems.Cells(i + 0, j + 2) = (Systems.Cells(i + 0, j) / Parts.Cells(k, "E")) * ...
        (Parts.Cells(k, "K") * Parts.Cells(k, "L") * nEBS * OperationalYear)
    Systems.Cells(i + 1, j + 2) = (Systems.Cells(i + 1, j) / Parts.Cells(k, "E")) * ...
        (Parts.Cells(k, "K") * Parts.Cells(k, "L") * nEBS * OperationalYear)
    Systems.Cells(i + 2, j + 2) = (Systems.Cells(i + 2, j) / Parts.Cells(k, "E")) * ...
        (Parts.Cells(k, "K") * Parts.Cells(k, "L") * nEBS * OperationalYear)
```

```

Systems.Cells(i + 0, j + 3) = (Systems.Cells(i + 0, j) / Parts.Cells(k, "E")) * ...
    Parts.Cells(k, "P")
Systems.Cells(i + 1, j + 3) = (Systems.Cells(i + 1, j) / Parts.Cells(k, "E")) * ...
    Parts.Cells(k, "P")
Systems.Cells(i + 2, j + 3) = (Systems.Cells(i + 2, j) / Parts.Cells(k, "E")) * ...
    Parts.Cells(k, "P")

Systems.Cells(i + 0, j + 4) = (Systems.Cells(i + 0, j) / Parts.Cells(k, "E")) * ...
    Parts.Cells(k, "O") * nEBS * OperationalYear + (Systems.Cells(i + 0, j + 2) * cEnergy)
Systems.Cells(i + 1, j + 4) = (Systems.Cells(i + 1, j) / Parts.Cells(k, "E")) * ...
    Parts.Cells(k, "O") * nEBS * OperationalYear + (Systems.Cells(i + 1, j + 2) * cEnergy)
Systems.Cells(i + 2, j + 4) = (Systems.Cells(i + 2, j) / Parts.Cells(k, "E")) * ...
    Parts.Cells(k, "O") * nEBS * OperationalYear + (Systems.Cells(i + 2, j + 2) * cEnergy)

j = 18
k = 84
Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(10, 1) * (1 + 0.1)), 0)
Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(10, 2) * (1 + 0.1)), 0)
Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(10, 3) * (1 + 0.1)), 0)

Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) / Parts.Cells(k, "E")
Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) / Parts.Cells(k, "E")
Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) / Parts.Cells(k, "E")

Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * Parts.Cells(k, "K") * ...
    Parts.Cells(k, "L") * nEBS * OperationalYear
Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * Parts.Cells(k, "K") * ...
    Parts.Cells(k, "L") * nEBS * OperationalYear
Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * Parts.Cells(k, "K") * ...
    Parts.Cells(k, "L") * nEBS * OperationalYear

Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Parts.Cells(k, "P")
Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Parts.Cells(k, "P")
Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Parts.Cells(k, "P")

Systems.Cells(i + 0, j + 4) = Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nEBS * ...
    OperationalYear + (Systems.Cells(i + 0, j + 2) * cEnergy)
Systems.Cells(i + 1, j + 4) = Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nEBS * ...
    OperationalYear + (Systems.Cells(i + 1, j + 2) * cEnergy)
Systems.Cells(i + 2, j + 4) = Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nEBS * ...
    OperationalYear + (Systems.Cells(i + 2, j + 2) * cEnergy)


'Make-up
i = 40
j = 4
k = 69
Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, 1) ...
    * (1 + fRedundant)), (ProcessPrint(9, 1) * (1 + fRedundant)) / Parts.Cells(k, "G")), 0)
Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, 2) ...
    * (1 + fRedundant)), (ProcessPrint(9, 2) * (1 + fRedundant)) / Parts.Cells(k, "G")), 0)
Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, 3) ...
    * (1 + fRedundant)), (ProcessPrint(9, 3) * (1 + fRedundant)) / Parts.Cells(k, "G")), 0)

Systems.Cells(i + 0, j + 1) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, "I")
Systems.Cells(i + 1, j + 1) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, "I")
Systems.Cells(i + 2, j + 1) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, "I")

Systems.Cells(i + 0, j + 2) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
    "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
Systems.Cells(i + 1, j + 2) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
    "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
Systems.Cells(i + 2, j + 2) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
    "K") * Parts.Cells(k, "L") * nBelt * OperationalYear

Systems.Cells(i + 0, j + 3) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, "P")
Systems.Cells(i + 1, j + 3) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, "P")
Systems.Cells(i + 2, j + 3) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, "P")

Systems.Cells(i + 0, j + 4) = (Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
    OperationalYear) + Systems.Cells(i + 0, j + 2) * cEnergy
Systems.Cells(i + 1, j + 4) = (Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
    OperationalYear) + Systems.Cells(i + 1, j + 2) * cEnergy
Systems.Cells(i + 2, j + 4) = (Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
    OperationalYear) + Systems.Cells(i + 2, j + 2) * cEnergy

j = 11
k = 70
Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, 1) ...
    * (1 + fRedundant)), (ProcessPrint(9, 1) * (1 + fRedundant)) / Parts.Cells(k, "G")), 0)
Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, 2) ...
    * (1 + fRedundant)), (ProcessPrint(9, 2) * (1 + fRedundant)) / Parts.Cells(k, "G")), 0)
```

```
        Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, 3) ...
             * (1 + fRedundant)), (ProcessPrint(9, 3) * (1 + fRedundant)) / Parts.Cells(k, "G")), 0)

        Systems.Cells(i + 0, j + 1) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, "I")
        Systems.Cells(i + 1, j + 1) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, "I")
        Systems.Cells(i + 2, j + 1) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, "I")

        Systems.Cells(i + 0, j + 2) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
             "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Systems.Cells(i + 1, j + 2) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
             "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Systems.Cells(i + 2, j + 2) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
             "K") * Parts.Cells(k, "L") * nBelt * OperationalYear

        Systems.Cells(i + 0, j + 3) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, "P")
        Systems.Cells(i + 1, j + 3) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, "P")
        Systems.Cells(i + 2, j + 3) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, "P")

        Systems.Cells(i + 0, j + 4) = (Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
             OperationalYear) + Systems.Cells(i + 0, j + 2) * cEnergy
        Systems.Cells(i + 1, j + 4) = (Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
             OperationalYear) + Systems.Cells(i + 1, j + 2) * cEnergy
        Systems.Cells(i + 2, j + 4) = (Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
             OperationalYear) + Systems.Cells(i + 2, j + 2) * cEnergy

        j = 18
        k = 71
        Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, 1) ...
             * (1 + fRedundant)), (ProcessPrint(9, 1) * (1 + fRedundant)) / Parts.Cells(k, "G")), 0)
        Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, 2) ...
             * (1 + fRedundant)), (ProcessPrint(9, 2) * (1 + fRedundant)) / Parts.Cells(k, "G")), 0)
        Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, 3) ...
             * (1 + fRedundant)), (ProcessPrint(9, 3) * (1 + fRedundant)) / Parts.Cells(k, "G")), 0)

        Systems.Cells(i + 0, j + 1) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, "I")
        Systems.Cells(i + 1, j + 1) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, "I")
        Systems.Cells(i + 2, j + 1) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, "I")

        Systems.Cells(i + 0, j + 2) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
             "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Systems.Cells(i + 1, j + 2) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
             "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Systems.Cells(i + 2, j + 2) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
             "K") * Parts.Cells(k, "L") * nBelt * OperationalYear

        Systems.Cells(i + 0, j + 3) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, "P")
        Systems.Cells(i + 1, j + 3) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, "P")
        Systems.Cells(i + 2, j + 3) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, "P")

        Systems.Cells(i + 0, j + 4) = (Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
             OperationalYear) + Systems.Cells(i + 0, j + 2) * cEnergy
        Systems.Cells(i + 1, j + 4) = (Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
             OperationalYear) + Systems.Cells(i + 1, j + 2) * cEnergy
        Systems.Cells(i + 2, j + 4) = (Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
             OperationalYear) + Systems.Cells(i + 2, j + 2) * cEnergy

        j = 25
        k = 72
        Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, 1) ...
             * (1 + fRedundant)), (ProcessPrint(9, 1) * (1 + fRedundant)) / Parts.Cells(k, "G")), 0)
        Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, 2) ...
             * (1 + fRedundant)), (ProcessPrint(9, 2) * (1 + fRedundant)) / Parts.Cells(k, "G")), 0)
        Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, 3) ...
             * (1 + fRedundant)), (ProcessPrint(9, 3) * (1 + fRedundant)) / Parts.Cells(k, "G")), 0)

        Systems.Cells(i + 0, j + 1) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, "I")
        Systems.Cells(i + 1, j + 1) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, "I")
        Systems.Cells(i + 2, j + 1) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, "I")

        Systems.Cells(i + 0, j + 2) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
             "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Systems.Cells(i + 1, j + 2) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
             "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Systems.Cells(i + 2, j + 2) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
             "K") * Parts.Cells(k, "L") * nBelt * OperationalYear

        Systems.Cells(i + 0, j + 3) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, "P")
        Systems.Cells(i + 1, j + 3) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, "P")
        Systems.Cells(i + 2, j + 3) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, "P")

        Systems.Cells(i + 0, j + 4) = (Systems.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
             OperationalYear) + Systems.Cells(i + 0, j + 2) * cEnergy
        Systems.Cells(i + 1, j + 4) = (Systems.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
             OperationalYear) + Systems.Cells(i + 1, j + 2) * cEnergy
```

```
    Systems.Cells(i + 2, j + 4) = (Systems.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear) + Systems.Cells(i + 2, j + 2) * cEnergy

    '''''''''''''''''''''''''''''''''''
    ''  Flows
    '''''''''''''''''''''''''''''''''''
    For flow = 1 To 12
        Select Case flow
            Case 1
                l = 1
            Case 2
                l = 6
            Case 3
                l = 14
            Case 4
                l = 19
            Case 5
                l = 23
            Case 6
                l = 24
            Case 7
                l = 25
            Case 8
                l = 29
            Case 9
                l = 34
            Case 10
                l = 35
            Case 11
                l = 45
            Case 12
                l = 50
        End Select

        i = 47 + ((flow - 1) * 6)
        j = 4
        k = 24
        Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp(FlowPrint(l, 1) / Parts.Cells(k, ...
            "G"), 0)
        Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp(FlowPrint(l, 2) / Parts.Cells(k, ...
            "G"), 0)
        Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp(FlowPrint(l, 3) / Parts.Cells(k, ...
            "G"), 0)

        Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "I")
        Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "I")
        Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "I")

        Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear

        Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "P")
        Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "P")
        Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "P")

        Systems.Cells(i + 0, j + 4) = (Systems.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "O") * nBelt * OperationalYear) + Systems.Cells(i + 0, j + 2) * cEnergy
        Systems.Cells(i + 1, j + 4) = (Systems.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "O") * nBelt * OperationalYear) + Systems.Cells(i + 1, j + 2) * cEnergy
        Systems.Cells(i + 2, j + 4) = (Systems.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "O") * nBelt * OperationalYear) + Systems.Cells(i + 2, j + 2) * cEnergy

        j = 11
        k = 25
        Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp(FlowPrint(l, 1) / Parts.Cells(k, ...
            "G"), 0)
        Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp(FlowPrint(l, 2) / Parts.Cells(k, ...
            "G"), 0)
        Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp(FlowPrint(l, 3) / Parts.Cells(k, ...
            "G"), 0)

        Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "I")
```

```
            Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "I")
            Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "I")

            Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
            Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
            Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear

            Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "P")
            Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "P")
            Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "P")

            Systems.Cells(i + 0, j + 4) = (Systems.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "O") * nBelt * OperationalYear) + Systems.Cells(i + 0, j + 2) * cEnergy
            Systems.Cells(i + 1, j + 4) = (Systems.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "O") * nBelt * OperationalYear) + Systems.Cells(i + 1, j + 2) * cEnergy
            Systems.Cells(i + 2, j + 4) = (Systems.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "O") * nBelt * OperationalYear) + Systems.Cells(i + 2, j + 2) * cEnergy

            j = 18
            k = 30
            Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp(FlowPrint(l, 1) / Parts.Cells(k, ...
                "G"), 0)
            Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp(FlowPrint(l, 2) / Parts.Cells(k, ...
                "G"), 0)
            Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp(FlowPrint(l, 3) / Parts.Cells(k, ...
                "G"), 0)

            Systems.Cells(i + 0, j + 1) = Systems.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "I")
            Systems.Cells(i + 1, j + 1) = Systems.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "I")
            Systems.Cells(i + 2, j + 1) = Systems.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "I")

            Systems.Cells(i + 0, j + 2) = Systems.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
            Systems.Cells(i + 1, j + 2) = Systems.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
            Systems.Cells(i + 2, j + 2) = Systems.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear

            Systems.Cells(i + 0, j + 3) = Systems.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "P")
            Systems.Cells(i + 1, j + 3) = Systems.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "P")
            Systems.Cells(i + 2, j + 3) = Systems.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "P")

            Systems.Cells(i + 0, j + 4) = (Systems.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "O") * nBelt * OperationalYear) + Systems.Cells(i + 0, j + 2) * cEnergy
            Systems.Cells(i + 1, j + 4) = (Systems.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "O") * nBelt * OperationalYear) + Systems.Cells(i + 1, j + 2) * cEnergy
            Systems.Cells(i + 2, j + 4) = (Systems.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
                Parts.Cells(k, "O") * nBelt * OperationalYear) + Systems.Cells(i + 2, j + 2) * cEnergy
        Next flow
    End Sub
```

K.12. CALCULATION 2

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''
'''  Fill the charts with the necessary information
'''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Sub CalcUnit2()
    Dim nCounters As Double
    Dim nBelt As Double
    Dim nPier As Double
    Dim nTray As Double
    Dim nSort As Double
    Dim nEBS As Double
```

```
    Dim i As Long
    Dim j As Long
    Dim k As Long
    Dim l As Long
    Dim flow As Long

    Dim OperationalYear As Double

    Dim nMerges As Double
    Dim nDiverters As Double
    Dim nLoading As Double
    Dim nOffloading As Double

    nCounters = 0.5
    nBelt = 0.75
    nPier = 0.4
    nTray = 0.75
    nSort = 0.8
    nEBS = 0.4
    OperationalYear = 365 * 24


    'Sorting diverters and merges
    For i = 1 To 3
        nMerges = 0
        nDiverters = 0

        If (Systems.Cells(i + 3, "D") / 20) > Systems.Cells(i + 46, "D") Then
            nMerges = nMerges + WorksheetFunction.RoundUp((Systems.Cells(i + 3, "D") / 20) - ...
                Systems.Cells(i + 46, "D"), 0)
        ElseIf (Systems.Cells(i + 3, "D") / 20) < Systems.Cells(i + 46, "D") Then
            nDiverters = nDiverters + WorksheetFunction.RoundUp(Systems.Cells(i + 46, "D") - ...
                (Systems.Cells(i + 3, "D") / 20), 0)
        End If

        If Systems.Cells(i + 82, "D") < Systems.Cells(i + 46, "D") + Systems.Cells(i + 58, ...
            "D") + Systems.Cells(i + 106, "D") Then
            nMerges = nMerges + Systems.Cells(i + 46, "D") + Systems.Cells(i + 58, "D") + ...
                Systems.Cells(i + 106, "D") - Systems.Cells(i + 82, "D")
        ElseIf Systems.Cells(i + 82, "D") > Systems.Cells(i + 46, "D") + Systems.Cells(i + 58, ...
            "D") + Systems.Cells(i + 106, "D") Then
            nDiverters = nDiverters + Systems.Cells(i + 82, "D") - (Systems.Cells(i + 46, "D") ...
                + Systems.Cells(i + 58, "D") + Systems.Cells(i + 106, "D"))
        End If

        If Systems.Cells(i + 46, "D") < Systems.Cells(i + 21, "R") Then
            nDiverters = nDiverters + Systems.Cells(i + 21, "R") - Systems.Cells(i + 46, "D")
        ElseIf Systems.Cells(i + 46, "D") > Systems.Cells(i + 21, "R") Then
            nMerges = nMerges + Systems.Cells(i + 46, "D") - Systems.Cells(i + 21, "R")
        End If

        If Systems.Cells(i + 88, "D") < Systems.Cells(i + 21, "R") Then
            nMerges = nMerges + Systems.Cells(i + 21, "R") - Systems.Cells(i + 88, "D")
        ElseIf Systems.Cells(i + 88, "D") > Systems.Cells(i + 21, "R") Then
            nDiverters = nDiverters + Systems.Cells(i + 88, "D") - Systems.Cells(i + 21, "R")
        End If

        nMerges = nMerges + (Systems.Cells(i + 88, "D") * Systems.Cells(i + 94, "D") + ...
            Systems.Cells(i + 100, "D") * Systems.Cells(i + 70, "D"))
        nDiverters = nDiverters + (Systems.Cells(i + 88, "D") * Systems.Cells(i + 94, "D") + ...
            Systems.Cells(i + 100, "D") * Systems.Cells(i + 70, "D"))
        nDiverters = nDiverters + (Systems.Cells(i + 39, "Y") - 1)

        Systems.Cells(i + 27, "D") = nMerges + nDiverters
        Systems.Cells(i + 27, "E") = nMerges * Parts.Cells(47, "H") * Parts.Cells(47, "I") + ...
            nDiverters * Parts.Cells(50, "H") * Parts.Cells(50, "I")
        Systems.Cells(i + 27, "F") = nMerges * Parts.Cells(47, "K") * Parts.Cells(47, "L") * ...
            nBelt * OperationalYear + nDiverters * Parts.Cells(50, "K") * Parts.Cells(50, "L") ...
            * nBelt * OperationalYear
        Systems.Cells(i + 27, "G") = nMerges * Parts.Cells(47, "P") + nDiverters * ...
            Parts.Cells(50, "P")
        Systems.Cells(i + 27, "H") = nMerges * Parts.Cells(47, "O") * nBelt * OperationalYear ...
            + nDiverters * Parts.Cells(50, "O") * nSort * OperationalYear + Systems.Cells(i + ...
            27, "F") * cEnergy
    Next i

    'Tray sorter systems
    i = 28
    j = 11
    Systems.Cells(i + 0, j) = WorksheetFunction.RoundUp(((ProcessPrint(3, 1) * (1 + ...
        fRedundant) * 0.6) / Parts.Cells(58, "G")), 0)
    Systems.Cells(i + 1, j) = WorksheetFunction.RoundUp(((ProcessPrint(3, 2) * (1 + ...
        fRedundant) * 0.6) / Parts.Cells(58, "G")), 0)
```

```
    Systems.Cells(i + 2, j) = WorksheetFunction.RoundUp(((ProcessPrint(3, 3) * (1 + ...
        fRedundant) * 0.6) / Parts.Cells(58, "G")), 0)

    nMerges = Systems.Cells(48, "D") + Systems.Cells(60, "D") + Systems.Cells(90, "D") + ...
        Systems.Cells(102, "D") + Systems.Cells(108, "D")
    nDiverters = Systems.Cells(72, "D") + Systems.Cells(78, "D") + Systems.Cells(84, "D") + ...
        Systems.Cells(96, "D") + Systems.Cells(114, "D")

    Systems.Cells(i + 0, j + 1) = ((ProcessPrint(2, 1) * (1 + fRedundant) * 0.6) * 1.2 + ...
        ((nMerges + nDiverters) * Parts.Cells(56, "H"))) * Parts.Cells(58, "I")
    Systems.Cells(i + 1, j + 1) = ((ProcessPrint(2, 2) * (1 + fRedundant) * 0.6) * 1.2 + ...
        ((nMerges + nDiverters) * Parts.Cells(56, "H"))) * Parts.Cells(58, "I")
    Systems.Cells(i + 2, j + 1) = ((ProcessPrint(2, 3) * (1 + fRedundant) * 0.6) * 1.2 + ...
        ((nMerges + nDiverters) * Parts.Cells(56, "H"))) * Parts.Cells(58, "I")

    Systems.Cells(i + 0, j + 2) = (Systems.Cells(i + 0, j + 1) / Parts.Cells(58, "I")) * ...
        Parts.Cells(58, "K") * Parts.Cells(58, "L") * nSort * OperationalYear
    Systems.Cells(i + 1, j + 2) = (Systems.Cells(i + 0, j + 1) / Parts.Cells(58, "I")) * ...
        Parts.Cells(58, "K") * Parts.Cells(58, "L") * nSort * OperationalYear
    Systems.Cells(i + 2, j + 2) = (Systems.Cells(i + 0, j + 1) / Parts.Cells(58, "I")) * ...
        Parts.Cells(58, "K") * Parts.Cells(58, "L") * nSort * OperationalYear

    Systems.Cells(i + 0, j + 3) = (Systems.Cells(i + 0, j + 1) / Parts.Cells(58, "I")) * ...
        Parts.Cells(58, "P")
    Systems.Cells(i + 1, j + 3) = (Systems.Cells(i + 0, j + 1) / Parts.Cells(58, "I")) * ...
        Parts.Cells(58, "P")
    Systems.Cells(i + 2, j + 3) = (Systems.Cells(i + 0, j + 1) / Parts.Cells(58, "I")) * ...
        Parts.Cells(58, "P")

    Systems.Cells(i + 0, j + 4) = Systems.Cells(i + 0, j + 2) * cEnergy
    Systems.Cells(i + 1, j + 4) = Systems.Cells(i + 0, j + 2) * cEnergy
    Systems.Cells(i + 2, j + 4) = Systems.Cells(i + 0, j + 2) * cEnergy

    'ICS systems
    For i = 1 To 3
        nLoading = WorksheetFunction.RoundUp(ProcessPrint(1, i) / Parts.Cells(62, "G") + ...
            ProcessPrint(4, i) / Parts.Cells(62, "G") + ProcessPrint(14, i) / Parts.Cells(62, ...
            "G") + ProcessPrint(17, i) / Parts.Cells(62, "G"), 0)
        nOffloading = WorksheetFunction.RoundUp(ProcessPrint(4, i) / Parts.Cells(65, "G") + ...
            SlotPrint(4, i) / Parts.Cells(65, "G") + ProcessPrint(18, i) / Parts.Cells(65, ...
            "G"), 0)

        Systems.Cells(i + 27, "R") = nLoading + nOffloading
        Systems.Cells(i + 27, "S") = nLoading * Parts.Cells(62, "H") * Parts.Cells(62, "I") + ...
            nOffloading * Parts.Cells(65, "H") * Parts.Cells(65, "I")
        Systems.Cells(i + 27, "T") = nLoading * Parts.Cells(62, "K") * Parts.Cells(62, "L") * ...
            nTray * OperationalYear + nOffloading * Parts.Cells(65, "K") * Parts.Cells(65, ...
            "L") * nTray * OperationalYear
        Systems.Cells(i + 27, "U") = nLoading * Parts.Cells(62, "P") + nOffloading * ...
            Parts.Cells(65, "P")
        Systems.Cells(i + 27, "V") = Systems.Cells(i + 27, "T") * cEnergy
    Next i


    'Terminating system
    i = 4
    j = 4
    k = 10
    Terminat.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(13, 1) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Terminat.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(13, 2) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Terminat.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(13, 3) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)

    Terminat.Cells(i + 0, j + 1) = Terminat.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Terminat.Cells(i + 1, j + 1) = Terminat.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Terminat.Cells(i + 2, j + 1) = Terminat.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))

    Terminat.Cells(i + 0, j + 2) = Terminat.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Terminat.Cells(i + 1, j + 2) = Terminat.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Terminat.Cells(i + 2, j + 2) = Terminat.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)

    Terminat.Cells(i + 0, j + 3) = Terminat.Cells(i + 0, j) * Parts.Cells(k, "P") * ...
        Parts.Cells(k, "H")
    Terminat.Cells(i + 1, j + 3) = Terminat.Cells(i + 1, j) * Parts.Cells(k, "P") * ...
        Parts.Cells(k, "H")
```

```
    Terminat.Cells(i + 2, j + 3) = Terminat.Cells(i + 2, j) * Parts.Cells(k, "P") * ...
        Parts.Cells(k, "H")

    Terminat.Cells(i + 0, j + 4) = Terminat.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Terminat.Cells(i + 0, j + 2) * cEnergy)
    Terminat.Cells(i + 1, j + 4) = Terminat.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Terminat.Cells(i + 1, j + 2) * cEnergy)
    Terminat.Cells(i + 2, j + 4) = Terminat.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Terminat.Cells(i + 2, j + 2) * cEnergy)

    j = 11
    k = 11
    Terminat.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(13, 1) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Terminat.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(13, 2) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Terminat.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(13, 3) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)

    Terminat.Cells(i + 0, j + 1) = Terminat.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Terminat.Cells(i + 1, j + 1) = Terminat.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Terminat.Cells(i + 2, j + 1) = Terminat.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))

    Terminat.Cells(i + 0, j + 2) = Terminat.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Terminat.Cells(i + 1, j + 2) = Terminat.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Terminat.Cells(i + 2, j + 2) = Terminat.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)

    Terminat.Cells(i + 0, j + 3) = Terminat.Cells(i + 0, j) * Parts.Cells(k, "P")
    Terminat.Cells(i + 1, j + 3) = Terminat.Cells(i + 1, j) * Parts.Cells(k, "P")
    Terminat.Cells(i + 2, j + 3) = Terminat.Cells(i + 2, j) * Parts.Cells(k, "P")

    Terminat.Cells(i + 0, j + 4) = Terminat.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Terminat.Cells(i + 0, j + 2) * cEnergy)
    Terminat.Cells(i + 1, j + 4) = Terminat.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Terminat.Cells(i + 1, j + 2) * cEnergy)
    Terminat.Cells(i + 2, j + 4) = Terminat.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Terminat.Cells(i + 2, j + 2) * cEnergy)

    j = 18
    k = 12
    Terminat.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(13, 1) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Terminat.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(13, 2) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Terminat.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(13, 3) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)

    Terminat.Cells(i + 0, j + 1) = Terminat.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Terminat.Cells(i + 1, j + 1) = Terminat.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Terminat.Cells(i + 2, j + 1) = Terminat.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))

    Terminat.Cells(i + 0, j + 2) = Terminat.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Terminat.Cells(i + 1, j + 2) = Terminat.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Terminat.Cells(i + 2, j + 2) = Terminat.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)

    Terminat.Cells(i + 0, j + 3) = Terminat.Cells(i + 0, j) * Parts.Cells(k, "P")
    Terminat.Cells(i + 1, j + 3) = Terminat.Cells(i + 1, j) * Parts.Cells(k, "P")
    Terminat.Cells(i + 2, j + 3) = Terminat.Cells(i + 2, j) * Parts.Cells(k, "P")

    Terminat.Cells(i + 0, j + 4) = Terminat.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Terminat.Cells(i + 0, j + 2) * cEnergy)
    Terminat.Cells(i + 1, j + 4) = Terminat.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Terminat.Cells(i + 1, j + 2) * cEnergy)
    Terminat.Cells(i + 2, j + 4) = Terminat.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Terminat.Cells(i + 2, j + 2) * cEnergy)


    'Screening
    i = 10
    j = 4
    k = 41
```

```
    Terminat.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 1) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Terminat.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 2) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Terminat.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 3) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)

    Terminat.Cells(i + 0, j + 1) = Terminat.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Terminat.Cells(i + 1, j + 1) = Terminat.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Terminat.Cells(i + 2, j + 1) = Terminat.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))

    Terminat.Cells(i + 0, j + 2) = Terminat.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Terminat.Cells(i + 1, j + 2) = Terminat.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Terminat.Cells(i + 2, j + 2) = Terminat.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)

    Terminat.Cells(i + 0, j + 3) = Terminat.Cells(i + 0, j) * Parts.Cells(k, "P")
    Terminat.Cells(i + 1, j + 3) = Terminat.Cells(i + 1, j) * Parts.Cells(k, "P")
    Terminat.Cells(i + 2, j + 3) = Terminat.Cells(i + 2, j) * Parts.Cells(k, "P")

    Terminat.Cells(i + 0, j + 4) = Terminat.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Terminat.Cells(i + 0, j + 2) * cEnergy)
    Terminat.Cells(i + 1, j + 4) = Terminat.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Terminat.Cells(i + 1, j + 2) * cEnergy)
    Terminat.Cells(i + 2, j + 4) = Terminat.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Terminat.Cells(i + 2, j + 2) * cEnergy)

    j = 11
    k = 43
    Terminat.Cells(i + 0, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 1) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Terminat.Cells(i + 1, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 2) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)
    Terminat.Cells(i + 2, j) = WorksheetFunction.RoundUp((ProcessPrint(4, 3) * (1 + ...
        fRedundant)) / Parts.Cells(k, "G"), 0)

    Terminat.Cells(i + 0, j + 1) = Terminat.Cells(i + 0, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Terminat.Cells(i + 1, j + 1) = Terminat.Cells(i + 1, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))
    Terminat.Cells(i + 2, j + 1) = Terminat.Cells(i + 2, j) * (Parts.Cells(k, "H") * ...
        Parts.Cells(k, "I"))

    Terminat.Cells(i + 0, j + 2) = Terminat.Cells(i + 0, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Terminat.Cells(i + 1, j + 2) = Terminat.Cells(i + 1, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)
    Terminat.Cells(i + 2, j + 2) = Terminat.Cells(i + 2, j) * (Parts.Cells(k, "K") * ...
        Parts.Cells(k, "L") * nBelt * OperationalYear)

    Terminat.Cells(i + 0, j + 3) = Terminat.Cells(i + 0, j) * Parts.Cells(k, "P")
    Terminat.Cells(i + 1, j + 3) = Terminat.Cells(i + 1, j) * Parts.Cells(k, "P")
    Terminat.Cells(i + 2, j + 3) = Terminat.Cells(i + 2, j) * Parts.Cells(k, "P")

    Terminat.Cells(i + 0, j + 4) = Terminat.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Terminat.Cells(i + 0, j + 2) * cEnergy)
    Terminat.Cells(i + 1, j + 4) = Terminat.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Terminat.Cells(i + 1, j + 2) * cEnergy)
    Terminat.Cells(i + 2, j + 4) = Terminat.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear + (Terminat.Cells(i + 2, j + 2) * cEnergy)

    'Make-up
    i = 16
    j = 4
    k = 75
    Terminat.Cells(i + 0, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, ...
        1) * (1 + fRedundant)), (ProcessPrint(9, 1) * (1 + fRedundant)) / Parts.Cells(k, ...
        "G")), 0)
    Terminat.Cells(i + 1, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, ...
        2) * (1 + fRedundant)), (ProcessPrint(9, 2) * (1 + fRedundant)) / Parts.Cells(k, ...
        "G")), 0)
    Terminat.Cells(i + 2, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, ...
        3) * (1 + fRedundant)), (ProcessPrint(9, 3) * (1 + fRedundant)) / Parts.Cells(k, ...
        "G")), 0)

    Terminat.Cells(i + 0, j + 1) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "I")
    Terminat.Cells(i + 1, j + 1) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "I")
```

```
    Terminat.Cells(i + 2, j + 1) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "I")

    Terminat.Cells(i + 0, j + 2) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
    Terminat.Cells(i + 1, j + 2) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
    Terminat.Cells(i + 2, j + 2) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "K") * Parts.Cells(k, "L") * nBelt * OperationalYear

    Terminat.Cells(i + 0, j + 3) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "P")
    Terminat.Cells(i + 1, j + 3) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "P")
    Terminat.Cells(i + 2, j + 3) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "P")

    Terminat.Cells(i + 0, j + 4) = (Terminat.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear) + Terminat.Cells(i + 0, j + 2) * cEnergy
    Terminat.Cells(i + 1, j + 4) = (Terminat.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear) + Terminat.Cells(i + 1, j + 2) * cEnergy
    Terminat.Cells(i + 2, j + 4) = (Terminat.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear) + Terminat.Cells(i + 2, j + 2) * cEnergy

    j = 11
    k = 76
    Terminat.Cells(i + 0, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, ...
        1) * (1 + fRedundant)), (ProcessPrint(9, 1) * (1 + fRedundant)) / Parts.Cells(k, ...
        "G")), 0)
    Terminat.Cells(i + 1, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, ...
        2) * (1 + fRedundant)), (ProcessPrint(9, 2) * (1 + fRedundant)) / Parts.Cells(k, ...
        "G")), 0)
    Terminat.Cells(i + 2, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, ...
        3) * (1 + fRedundant)), (ProcessPrint(9, 3) * (1 + fRedundant)) / Parts.Cells(k, ...
        "G")), 0)

    Terminat.Cells(i + 0, j + 1) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "I")
    Terminat.Cells(i + 1, j + 1) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "I")
    Terminat.Cells(i + 2, j + 1) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "I")

    Terminat.Cells(i + 0, j + 2) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
    Terminat.Cells(i + 1, j + 2) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
    Terminat.Cells(i + 2, j + 2) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "K") * Parts.Cells(k, "L") * nBelt * OperationalYear

    Terminat.Cells(i + 0, j + 3) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "P")
    Terminat.Cells(i + 1, j + 3) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "P")
    Terminat.Cells(i + 2, j + 3) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "P")

    Terminat.Cells(i + 0, j + 4) = (Terminat.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear) + Terminat.Cells(i + 0, j + 2) * cEnergy
    Terminat.Cells(i + 1, j + 4) = (Terminat.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear) + Terminat.Cells(i + 1, j + 2) * cEnergy
    Terminat.Cells(i + 2, j + 4) = (Terminat.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
        OperationalYear) + Terminat.Cells(i + 2, j + 2) * cEnergy

    j = 18
    k = 77
    Terminat.Cells(i + 0, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, ...
        1) * (1 + fRedundant)), (ProcessPrint(9, 1) * (1 + fRedundant)) / Parts.Cells(k, ...
        "G")), 0)
    Terminat.Cells(i + 1, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, ...
        2) * (1 + fRedundant)), (ProcessPrint(9, 2) * (1 + fRedundant)) / Parts.Cells(k, ...
        "G")), 0)
    Terminat.Cells(i + 2, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, ...
        3) * (1 + fRedundant)), (ProcessPrint(9, 3) * (1 + fRedundant)) / Parts.Cells(k, ...
        "G")), 0)

    Terminat.Cells(i + 0, j + 1) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "I")
    Terminat.Cells(i + 1, j + 1) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "I")
    Terminat.Cells(i + 2, j + 1) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
        "I")

```

```
        Terminat.Cells(i + 0, j + 2) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
            "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Terminat.Cells(i + 1, j + 2) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
            "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Terminat.Cells(i + 2, j + 2) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
            "K") * Parts.Cells(k, "L") * nBelt * OperationalYear

        Terminat.Cells(i + 0, j + 3) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
            "P")
        Terminat.Cells(i + 1, j + 3) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
            "P")
        Terminat.Cells(i + 2, j + 3) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
            "P")

        Terminat.Cells(i + 0, j + 4) = (Terminat.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
            OperationalYear) + Terminat.Cells(i + 0, j + 2) * cEnergy
        Terminat.Cells(i + 1, j + 4) = (Terminat.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
            OperationalYear) + Terminat.Cells(i + 1, j + 2) * cEnergy
        Terminat.Cells(i + 2, j + 4) = (Terminat.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
            OperationalYear) + Terminat.Cells(i + 2, j + 2) * cEnergy

        j = 25
        k = 78
        Terminat.Cells(i + 0, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, ...
            1) * (1 + fRedundant)), (ProcessPrint(9, 1) * (1 + fRedundant)) / Parts.Cells(k, ...
            "G")), 0)
        Terminat.Cells(i + 1, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, ...
            2) * (1 + fRedundant)), (ProcessPrint(9, 2) * (1 + fRedundant)) / Parts.Cells(k, ...
            "G")), 0)
        Terminat.Cells(i + 2, j) = WorksheetFunction.RoundUp(WorksheetFunction.Max((SlotPrint(4, ...
            3) * (1 + fRedundant)), (ProcessPrint(9, 3) * (1 + fRedundant)) / Parts.Cells(k, ...
            "G")), 0)

        Terminat.Cells(i + 0, j + 1) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
            "I")
        Terminat.Cells(i + 1, j + 1) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
            "I")
        Terminat.Cells(i + 2, j + 1) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
            "I")

        Terminat.Cells(i + 0, j + 2) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
            "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Terminat.Cells(i + 1, j + 2) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
            "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Terminat.Cells(i + 2, j + 2) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
            "K") * Parts.Cells(k, "L") * nBelt * OperationalYear

        Terminat.Cells(i + 0, j + 3) = (ProcessPrint(8, 1) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
            "P")
        Terminat.Cells(i + 1, j + 3) = (ProcessPrint(8, 2) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
            "P")
        Terminat.Cells(i + 2, j + 3) = (ProcessPrint(8, 3) / Parts.Cells(k, "E")) * Parts.Cells(k, ...
            "P")

        Terminat.Cells(i + 0, j + 4) = (Terminat.Cells(i + 0, j) * Parts.Cells(k, "O") * nBelt * ...
            OperationalYear) + Terminat.Cells(i + 0, j + 2) * cEnergy
        Terminat.Cells(i + 1, j + 4) = (Terminat.Cells(i + 1, j) * Parts.Cells(k, "O") * nBelt * ...
            OperationalYear) + Terminat.Cells(i + 1, j + 2) * cEnergy
        Terminat.Cells(i + 2, j + 4) = (Terminat.Cells(i + 2, j) * Parts.Cells(k, "O") * nBelt * ...
            OperationalYear) + Terminat.Cells(i + 2, j + 2) * cEnergy

        ''''''''''''''''''''''''''''''''''''
        ''  Flows
        ''''''''''''''''''''''''''''''''''''
        For flow = 1 To 7
            Select Case flow
                Case 1
                    l = 10
                Case 2
                    l = 11
                Case 3
                    l = 12
                Case 4
                    l = 38
                Case 5
                    l = 39
                Case 6
                    l = 40
                Case 7
                    l = 41
            End Select

            i = 23 + ((flow - 1) * 6)
            j = 4
```

```
        k = 24
        Terminat.Cells(i + 0, j) = WorksheetFunction.RoundUp(FlowPrint(l, 1) / Parts.Cells(k, ...
            "G"), 0)
        Terminat.Cells(i + 1, j) = WorksheetFunction.RoundUp(FlowPrint(l, 2) / Parts.Cells(k, ...
            "G"), 0)
        Terminat.Cells(i + 2, j) = WorksheetFunction.RoundUp(FlowPrint(l, 3) / Parts.Cells(k, ...
            "G"), 0)

        Terminat.Cells(i + 0, j + 1) = Terminat.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "I")
        Terminat.Cells(i + 1, j + 1) = Terminat.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "I")
        Terminat.Cells(i + 2, j + 1) = Terminat.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "I")

        Terminat.Cells(i + 0, j + 2) = Terminat.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Terminat.Cells(i + 1, j + 2) = Terminat.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Terminat.Cells(i + 2, j + 2) = Terminat.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear

        Terminat.Cells(i + 0, j + 3) = Terminat.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "P")
        Terminat.Cells(i + 1, j + 3) = Terminat.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "P")
        Terminat.Cells(i + 2, j + 3) = Terminat.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "P")

        Terminat.Cells(i + 0, j + 4) = (Terminat.Cells(i + 0, j) * Results.Cells(l + 23, "G") ...
            * Parts.Cells(k, "O") * nBelt * OperationalYear) + Terminat.Cells(i + 0, j + 2) * ...
            cEnergy
        Terminat.Cells(i + 1, j + 4) = (Terminat.Cells(i + 1, j) * Results.Cells(l + 23, "G") ...
            * Parts.Cells(k, "O") * nBelt * OperationalYear) + Terminat.Cells(i + 1, j + 2) * ...
            cEnergy
        Terminat.Cells(i + 2, j + 4) = (Terminat.Cells(i + 2, j) * Results.Cells(l + 23, "G") ...
            * Parts.Cells(k, "O") * nBelt * OperationalYear) + Terminat.Cells(i + 2, j + 2) * ...
            cEnergy

        j = 11
        k = 25
        Terminat.Cells(i + 0, j) = WorksheetFunction.RoundUp(FlowPrint(l, 1) / Parts.Cells(k, ...
            "G"), 0)
        Terminat.Cells(i + 1, j) = WorksheetFunction.RoundUp(FlowPrint(l, 2) / Parts.Cells(k, ...
            "G"), 0)
        Terminat.Cells(i + 2, j) = WorksheetFunction.RoundUp(FlowPrint(l, 3) / Parts.Cells(k, ...
            "G"), 0)

        Terminat.Cells(i + 0, j + 1) = Terminat.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "I")
        Terminat.Cells(i + 1, j + 1) = Terminat.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "I")
        Terminat.Cells(i + 2, j + 1) = Terminat.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "I")

        Terminat.Cells(i + 0, j + 2) = Terminat.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Terminat.Cells(i + 1, j + 2) = Terminat.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Terminat.Cells(i + 2, j + 2) = Terminat.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear

        Terminat.Cells(i + 0, j + 3) = Terminat.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "P")
        Terminat.Cells(i + 1, j + 3) = Terminat.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "P")
        Terminat.Cells(i + 2, j + 3) = Terminat.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "P")

        Terminat.Cells(i + 0, j + 4) = (Terminat.Cells(i + 0, j) * Results.Cells(l + 23, "G") ...
            * Parts.Cells(k, "O") * nBelt * OperationalYear) + Terminat.Cells(i + 0, j + 2) * ...
            cEnergy
        Terminat.Cells(i + 1, j + 4) = (Terminat.Cells(i + 1, j) * Results.Cells(l + 23, "G") ...
            * Parts.Cells(k, "O") * nBelt * OperationalYear) + Terminat.Cells(i + 1, j + 2) * ...
            cEnergy
        Terminat.Cells(i + 2, j + 4) = (Terminat.Cells(i + 2, j) * Results.Cells(l + 23, "G") ...
            * Parts.Cells(k, "O") * nBelt * OperationalYear) + Terminat.Cells(i + 2, j + 2) * ...
            cEnergy

        j = 18
        k = 30
        Terminat.Cells(i + 0, j) = WorksheetFunction.RoundUp(FlowPrint(l, 1) / Parts.Cells(k, ...
            "G"), 0)
```

```
        Terminat.Cells(i + 1, j) = WorksheetFunction.RoundUp(FlowPrint(l, 2) / Parts.Cells(k, ...
            "G"), 0)
        Terminat.Cells(i + 2, j) = WorksheetFunction.RoundUp(FlowPrint(l, 3) / Parts.Cells(k, ...
            "G"), 0)

        Terminat.Cells(i + 0, j + 1) = Terminat.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "I")
        Terminat.Cells(i + 1, j + 1) = Terminat.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "I")
        Terminat.Cells(i + 2, j + 1) = Terminat.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "I")

        Terminat.Cells(i + 0, j + 2) = Terminat.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Terminat.Cells(i + 1, j + 2) = Terminat.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear
        Terminat.Cells(i + 2, j + 2) = Terminat.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "K") * Parts.Cells(k, "L") * nBelt * OperationalYear

        Terminat.Cells(i + 0, j + 3) = Terminat.Cells(i + 0, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "P")
        Terminat.Cells(i + 1, j + 3) = Terminat.Cells(i + 1, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "P")
        Terminat.Cells(i + 2, j + 3) = Terminat.Cells(i + 2, j) * Results.Cells(l + 23, "G") * ...
            Parts.Cells(k, "P")

        Terminat.Cells(i + 0, j + 4) = (Terminat.Cells(i + 0, j) * Results.Cells(l + 23, "G") ...
            * Parts.Cells(k, "O") * nBelt * OperationalYear) + Terminat.Cells(i + 0, j + 2) * ...
            cEnergy
        Terminat.Cells(i + 1, j + 4) = (Terminat.Cells(i + 1, j) * Results.Cells(l + 23, "G") ...
            * Parts.Cells(k, "O") * nBelt * OperationalYear) + Terminat.Cells(i + 1, j + 2) * ...
            cEnergy
        Terminat.Cells(i + 2, j + 4) = (Terminat.Cells(i + 2, j) * Results.Cells(l + 23, "G") ...
            * Parts.Cells(k, "O") * nBelt * OperationalYear) + Terminat.Cells(i + 2, j + 2) * ...
            cEnergy
    Next flow
End Sub
```

K.13. POPULATE CHARTS

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''
'''  Fill the charts with the necessary information
'''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Sub PopulateCharts()
    Dim j As Long
    Dim k As Long
    Dim ChartLength As Long
    Dim ChartStart As Long
    Dim DayOfMaximum As Long
    Dim MultiplyResult As Boolean
    Dim MultiplyBy As Integer

    ChartLength = WorksheetFunction.Min(Abs(tChart), Abs(SimLengthDays) + 1, 10) * Day * Hour
    ReDim ChartValues(0 To ChartLength) As Double
    ReDim ChartXValues(0 To ChartLength) As Long

    For j = 1 To 19
        MultiplyResult = True
        Select Case j
            Case 2, 8, 10, 19
                MultiplyResult = False
        End Select

        'Define where the chart should start
        DayOfMaximum = Fix(ProcessData(j, 5) / (Day * Hour)) * Day * Hour
        ChartStart = DayOfMaximum - Fix(tChart / (Day * Hour))

        If ChartStart < 0 Then
            ChartStart = 0
        ElseIf ChartStart + ChartLength > SimLength Then
            ChartStart = SimLength - ChartLength
        End If

        'Create and update charts for all functions
        With Charts.ChartObjects(j).Chart
            .HasTitle = True
            .ChartType = xlLine
```

```
                .HasLegend = False

                If MultiplyResult = True Then
                    .ChartTitle.Text = ProcessNames(j - 1) & " in [bags/ " & tResult & "-min]"
                    MultiplyBy = tResult
                ElseIf MultiplyResult = False Then
                    .ChartTitle.Text = ProcessNames(j - 1) & " in [positions]"
                    MultiplyBy = 1
                End If

                'Delete the old chart series containing data, and build 3 new ones
                For k = 1 To .SeriesCollection.Count
                    .SeriesCollection(1).Delete
                Next k
                .SeriesCollection.NewSeries
                .SeriesCollection.NewSeries
                .SeriesCollection.NewSeries

                'Fill the first series with the minute data and x-values
                For k = 0 To ChartLength
                    ChartValues(k) = ProcessRecord(j, k + ChartStart) * MultiplyBy
                    ChartXValues(k) = k + ChartStart
                Next k
                .SeriesCollection(1).XValues = ChartXValues
                .SeriesCollection(1).Values = ChartValues

                'Mark the average per tCount with a line
                For k = 0 To ChartLength
                    ChartValues(k) = (ProcessData(j, 3) / tCount) * MultiplyBy
                Next k
                .SeriesCollection(2).Values = ChartValues

                'Mark the maximum of the function with a line
                For k = 0 To ChartLength
                    ChartValues(k) = ProcessData(j, 2) * MultiplyBy
                Next k
                .SeriesCollection(3).Values = ChartValues
            End With
        Next j

        For j = 1 To 4
            'Define where the chart should start
            DayOfMaximum = Fix(SlotData(j, 5) / (Day * Hour)) * Day * Hour
            ChartStart = DayOfMaximum - Fix(tChart / (Day * Hour))

            If ChartStart < 0 Then
                ChartStart = 0
            ElseIf ChartStart + ChartLength > SimLength Then
                ChartStart = SimLength - ChartLength
            End If

            'Create and update charts for all functions
            With Charts.ChartObjects(j + 19).Chart
                .HasTitle = True
                .ChartTitle.Text = SlotNames(j - 1) & " in [positions]"
                .ChartType = xlLine
                .HasLegend = False

                'Delete the old chart series containing data, and build 3 new ones
                For k = 1 To .SeriesCollection.Count
                    .SeriesCollection(1).Delete
                Next k
                .SeriesCollection.NewSeries
                .SeriesCollection.NewSeries
                .SeriesCollection.NewSeries

                'Fill the first series with the minute data and x-values
                For k = 0 To ChartLength
                    ChartValues(k) = SlotRecord(j, k + ChartStart)
                    ChartXValues(k) = k + ChartStart
                Next k
                .SeriesCollection(1).XValues = ChartXValues
                .SeriesCollection(1).Values = ChartValues

                'Mark the average per tCount with a line
                For k = 0 To ChartLength
                    ChartValues(k) = (SlotData(j, 3) / tCount)
                Next k
                .SeriesCollection(2).Values = ChartValues

                'Mark the maximum of the function with a line
                For k = 0 To ChartLength
                    ChartValues(k) = SlotData(j, 2)
                Next k
```

```
            .SeriesCollection(3).Values = ChartValues
        End With
    Next j

    'Find out where the largest amount of originating passengers is
    DayOfMaximum = Fix(ProcessData(1, 5) / (Day * Hour)) * Day * Hour
    ChartStart = DayOfMaximum - Fix(tChart / (Day * Hour))

    If ChartStart < 0 Then
        ChartStart = 0
    ElseIf ChartStart + ChartLength > SimLength Then
        ChartStart = SimLength - ChartLength
    End If

    AirlineList(2) = "Unknown"
    With Charts.ChartObjects(24).Chart
        .HasTitle = True
        .ChartTitle.Text = "Check-ins per airline in [passengers]"
        .ChartType = xlLine
        .HasLegend = True

        For k = 1 To .SeriesCollection.Count
            .SeriesCollection(1).Delete
        Next k

        For j = LBound(OriginatingAirline, 2) To UBound(OriginatingAirline, 2)
            For k = 0 To ChartLength
                ChartValues(k) = OriginatingAirline(k + ChartStart, j)
                ChartXValues(k) = k + ChartStart
            Next k
            .SeriesCollection.NewSeries
            .SeriesCollection(j).XValues = ChartXValues
            .SeriesCollection(j).Values = ChartValues
            .SeriesCollection(j).Name = AirlineList(j)
        Next j
    End With
End Sub
```

K.14. PROCESS OUTPUT

```
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''
'''  Present the processed data in the process flow chart
'''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Sub ProcessOutput()
    Dim i As Integer
    Dim j As Integer

    'Arrival results
    Process.Cells(14, "D") = (ProcessData(12, 3) / tCount) * tResult
    Process.Cells(15, "D") = ProcessData(12, 2) * tResult
    Process.Cells(14, "E") = "b / " & tResult & " min"
    Process.Cells(15, "E") = "b / " & tResult & " min"

    Process.Cells(11, "I") = (FlowData(41, 3) / tCount) * tResult
    Process.Cells(12, "I") = FlowData(41, 2) * tResult
    Process.Cells(11, "J") = "b / " & tResult & " min"
    Process.Cells(12, "J") = "b / " & tResult & " min"

    Process.Cells(5, "N") = (FlowData(10, 3) / tCount) * tResult
    Process.Cells(6, "N") = FlowData(10, 2) * tResult
    Process.Cells(5, "O") = "b / " & tResult & " min"
    Process.Cells(6, "O") = "b / " & tResult & " min"

    Process.Cells(24, "G") = (FlowData(40, 3) / tCount) * tResult
    Process.Cells(25, "G") = FlowData(40, 2) * tResult
    Process.Cells(24, "H") = "b / " & tResult & " min"
    Process.Cells(25, "H") = "b / " & tResult & " min"

    Process.Cells(24, "K") = (FlowData(38, 3) / tCount) * tResult
    Process.Cells(25, "K") = FlowData(38, 2) * tResult
    Process.Cells(24, "L") = "b / " & tResult & " min"
    Process.Cells(25, "L") = "b / " & tResult & " min"

    Process.Cells(19, "Q") = (FlowData(39, 3) / tCount) * tResult
    Process.Cells(20, "Q") = FlowData(39, 2) * tResult
    Process.Cells(19, "R") = "b / " & tResult & " min"
    Process.Cells(20, "R") = "b / " & tResult & " min"
```

```

    Process.Cells(19, "U") = (FlowData(12, 3) / tCount) * tResult
    Process.Cells(20, "U") = FlowData(12, 2) * tResult
    Process.Cells(19, "V") = "b / " & tResult & " min"
    Process.Cells(20, "V") = "b / " & tResult & " min"

    Process.Cells(11, "T") = (FlowData(11, 3) / tCount) * tResult
    Process.Cells(12, "T") = FlowData(11, 2) * tResult
    Process.Cells(11, "U") = "b / " & tResult & " min"
    Process.Cells(12, "U") = "b / " & tResult & " min"

    Process.Cells(14, "AA") = (ProcessData(13, 3) / tCount) * tResult
    Process.Cells(15, "AA") = ProcessData(13, 2) * tResult
    Process.Cells(14, "AB") = "b / " & tResult & " min"
    Process.Cells(15, "AB") = "b / " & tResult & " min"

    Process.Cells(14, "N") = (ProcessData(11, 3) / tCount) * tResult
    Process.Cells(15, "N") = ProcessData(11, 2) * tResult
    Process.Cells(14, "O") = "b / " & tResult & " min"
    Process.Cells(15, "O") = "b / " & tResult & " min"

    Process.Cells(13, "AE") = (FlowData(43, 3) / tCount) * tResult
    Process.Cells(14, "AE") = FlowData(43, 2) * tResult
    Process.Cells(13, "AF") = "b / " & tResult & " min"
    Process.Cells(14, "AF") = "b / " & tResult & " min"

    Process.Cells(16, "AI") = (ProcessData(15, 3) / tCount) * tResult
    Process.Cells(17, "AI") = ProcessData(15, 2) * tResult
    Process.Cells(16, "AJ") = "b / " & tResult & " min"
    Process.Cells(17, "AJ") = "b / " & tResult & " min"

    Process.Cells(20, "AE") = (FlowData(44, 3) / tCount) * tResult
    Process.Cells(21, "AE") = FlowData(44, 2) * tResult
    Process.Cells(20, "AF") = "b / " & tResult & " min"
    Process.Cells(21, "AF") = "b / " & tResult & " min"

    Process.Cells(19, "AA") = (ProcessData(14, 3) / tCount) * tResult
    Process.Cells(20, "AA") = ProcessData(14, 2) * tResult
    Process.Cells(19, "AB") = "b / " & tResult & " min"
    Process.Cells(20, "AB") = "b / " & tResult & " min"

    Process.Cells(24, "V") = (FlowData(14, 3) / tCount) * tResult
    Process.Cells(25, "V") = FlowData(14, 2) * tResult
    Process.Cells(24, "W") = "b / " & tResult & " min"
    Process.Cells(25, "W") = "b / " & tResult & " min"

    Process.Cells(24, "Z") = (FlowData(19, 3) / tCount) * tResult
    Process.Cells(25, "Z") = FlowData(19, 2) * tResult
    Process.Cells(24, "AA") = "b / " & tResult & " min"
    Process.Cells(25, "AA") = "b / " & tResult & " min"

    'Departure results
    Process.Cells(40, "D") = (ProcessData(1, 3) / tCount) * tResult
    Process.Cells(41, "D") = ProcessData(1, 2) * tResult
    Process.Cells(40, "E") = "b / " & tResult & " min"
    Process.Cells(41, "E") = "b / " & tResult & " min"

    Process.Cells(42, "G") = (FlowData(1, 3) / tCount) * tResult
    Process.Cells(43, "G") = FlowData(1, 2) * tResult
    Process.Cells(42, "H") = "b / " & tResult & " min"
    Process.Cells(43, "H") = "b / " & tResult & " min"

    Process.Cells(49, "G") = (FlowData(6, 3) / tCount) * tResult
    Process.Cells(50, "G") = FlowData(6, 2) * tResult
    Process.Cells(49, "H") = "b / " & tResult & " min"
    Process.Cells(50, "H") = "b / " & tResult & " min"

    Process.Cells(39, "N") = (ProcessData(3, 3) / tCount) * tResult
    Process.Cells(40, "N") = ProcessData(3, 2) * tResult
    Process.Cells(41, "N") = ProcessData(2, 2)
    Process.Cells(39, "O") = "b / " & tResult & " min"
    Process.Cells(40, "O") = "b / " & tResult & " min"

    Process.Cells(51, "N") = (ProcessData(4, 3) / tCount) * tResult
    Process.Cells(52, "N") = ProcessData(4, 2) * tResult
    Process.Cells(51, "O") = "b / " & tResult & " min"
    Process.Cells(52, "O") = "b / " & tResult & " min"

    Process.Cells(29, "N") = ProcessData(10, 2)

    Process.Cells(45, "L") = (FlowData(25, 3) / tCount) * tResult
    Process.Cells(46, "L") = FlowData(25, 2) * tResult
    Process.Cells(45, "M") = "b / " & tResult & " min"
    Process.Cells(46, "M") = "b / " & tResult & " min"
```

```

    Process.Cells(45, "P") = (FlowData(29, 3) / tCount) * tResult
    Process.Cells(46, "P") = FlowData(29, 2) * tResult
    Process.Cells(45, "Q") = "b / " & tResult & " min"
    Process.Cells(46, "Q") = "b / " & tResult & " min"

    Process.Cells(34, "L") = (FlowData(34, 3) / tCount) * tResult
    Process.Cells(35, "L") = FlowData(34, 2) * tResult
    Process.Cells(34, "M") = "b / " & tResult & " min"
    Process.Cells(35, "M") = "b / " & tResult & " min"

    Process.Cells(34, "P") = (FlowData(35, 3) / tCount) * tResult
    Process.Cells(35, "P") = FlowData(35, 2) * tResult
    Process.Cells(34, "Q") = "b / " & tResult & " min"
    Process.Cells(35, "Q") = "b / " & tResult & " min"

    Process.Cells(45, "U") = (FlowData(23, 3) / tCount) * tResult
    Process.Cells(46, "U") = FlowData(23, 2) * tResult
    Process.Cells(45, "V") = "b / " & tResult & " min"
    Process.Cells(46, "V") = "b / " & tResult & " min"

    Process.Cells(54, "V") = (FlowData(24, 3) / tCount) * tResult
    Process.Cells(55, "V") = FlowData(24, 2) * tResult
    Process.Cells(54, "W") = "b / " & tResult & " min"
    Process.Cells(55, "W") = "b / " & tResult & " min"

    Process.Cells(50, "AA") = (ProcessData(9, 3) / tCount) * tResult
    Process.Cells(51, "AA") = ProcessData(9, 2) * tResult
    Process.Cells(52, "AA") = ProcessData(8, 2)
    Process.Cells(50, "AB") = "b / " & tResult & " min"
    Process.Cells(51, "AB") = "b / " & tResult & " min"

    Process.Cells(49, "AE") = (FlowData(42, 3) / tCount) * tResult
    Process.Cells(50, "AE") = FlowData(42, 2) * tResult
    Process.Cells(49, "AF") = "b / " & tResult & " min"
    Process.Cells(50, "AF") = "b / " & tResult & " min"

    Process.Cells(51, "AI") = (ProcessData(16, 3) / tCount) * tResult
    Process.Cells(52, "AI") = ProcessData(16, 2) * tResult
    Process.Cells(51, "AJ") = "b / " & tResult & " min"
    Process.Cells(52, "AJ") = "b / " & tResult & " min"

    Process.Cells(56, "K") = (FlowData(45, 3) / tCount) * tResult
    Process.Cells(57, "K") = FlowData(45, 2) * tResult
    Process.Cells(56, "L") = "b / " & tResult & " min"
    Process.Cells(57, "L") = "b / " & tResult & " min"

    Process.Cells(56, "Q") = (FlowData(50, 3) / tCount) * tResult
    Process.Cells(57, "Q") = FlowData(50, 2) * tResult
    Process.Cells(56, "R") = "b / " & tResult & " min"
    Process.Cells(57, "R") = "b / " & tResult & " min"

    Process.Cells(61, "J") = (ProcessData(17, 3) / tCount) * tResult
    Process.Cells(62, "J") = ProcessData(17, 2) * tResult
    Process.Cells(61, "K") = "b / " & tResult & " min"
    Process.Cells(62, "K") = "b / " & tResult & " min"

    Process.Cells(61, "R") = (ProcessData(18, 3) / tCount) * tResult
    Process.Cells(62, "R") = ProcessData(18, 2) * tResult
    Process.Cells(61, "T") = "b / " & tResult & " min"
    Process.Cells(62, "T") = "b / " & tResult & " min"

    For i = LBound(ProcessData, 1) To UBound(ProcessData, 1)
        For j = LBound(ProcessData, 2) To UBound(ProcessData, 2)
            Check.Cells(i + 20, j + 1) = ProcessData(i, j)
        Next j
    Next i
End Sub
```

K.15. WRITE FILE

```
''''''''''''''''''''''''''''''''''''''''''''''''''''
'''
'''  If desired, writ the information to a file
'''
''''''''''''''''''''''''''''''''''''''''''''''''''''
Sub WriteFile()
    Dim i As Long
    Dim j As Long
```

```

    If WriteToFile = True Then
        'Printing statement for checking the calculation

        Set NewBook = Workbooks.Add
        With NewBook
            .Title = "All Sales"
            .Subject = "Sales"
            .SaveAs (ThisWorkbook.Path & "\" & FileName & ".xlsx")
        End With

        Set NewDeparture = NewBook.Sheets.Add
        Set NewArrival = NewBook.Sheets.Add
        Set NewSchedule = NewBook.Sheets.Add
        Set NewOriginating = NewBook.Sheets.Add

        NewSchedule.Name = "Flight_plan"
        NewDeparture.Name = "Departure_data"
        NewArrival.Name = "Arrival_data"
        NewOriginating.Name = "Check-in_per_airline"

        For Each Sheet In NewBook.Worksheets
            If Sheet.Name <> "Flight_plan" And Sheet.Name <> "Departure_data" And Sheet.Name ...
                <> "Arrival_data" And Sheet.Name <> "Check-in_per_airline" Then
                Application.DisplayAlerts = False
                Sheet.Delete
                Application.DisplayAlerts = True
            End If
        Next Sheet


        If WriteFlightPlan = True Then
            NewSchedule.Cells(1, "A") = "Arrival flight schedule printout data"
            NewSchedule.Cells(1, "H") = "Arrival flight schedule printout data"

            NewSchedule.Cells(2, "A") = "Arrival time [minutes]"
            NewSchedule.Cells(2, "B") = "Flight plan line"
            NewSchedule.Cells(2, "C") = "Terminating passenger seats"
            NewSchedule.Cells(2, "D") = "Empty seats"
            NewSchedule.Cells(2, "E") = "Number of terminating bags"
            NewSchedule.Cells(2, "F") = "Number of transfer bags"

            NewSchedule.Cells(2, "H") = "Departure time [minutes]"
            NewSchedule.Cells(2, "I") = "Flight plan line"
            NewSchedule.Cells(2, "J") = "Total seats taken"
            NewSchedule.Cells(2, "K") = "Transfer passengers"
            NewSchedule.Cells(2, "L") = "Number of originating bags"
            NewSchedule.Cells(2, "M") = "Number of transfer bags"

            'Writing all flights to the output file
            For i = LBound(ArrivalFlightList, 1) To UBound(ArrivalFlightList, 1)
                NewSchedule.Cells(i + 3, 1) = ArrivalFlightList(i, 1)
                NewSchedule.Cells(i + 3, 2) = ArrivalFlightList(i, 2)
                NewSchedule.Cells(i + 3, 3) = ArrivalFlightList(i, 3)
                NewSchedule.Cells(i + 3, 4) = ArrivalFlightList(i, 4)
                NewSchedule.Cells(i + 3, 5) = ArrivalFlightList(i, 5)
                NewSchedule.Cells(i + 3, 6) = ArrivalFlightList(i, 6)
            Next i

            For i = LBound(DepartureFlightList, 1) To UBound(DepartureFlightList, 1)
                NewSchedule.Cells(i + 3, 8) = DepartureFlightList(i, 1)
                NewSchedule.Cells(i + 3, 9) = DepartureFlightList(i, 2)
                NewSchedule.Cells(i + 3, 10) = DepartureFlightList(i, 3)
                NewSchedule.Cells(i + 3, 11) = DepartureFlightList(i, 4)
                NewSchedule.Cells(i + 3, 12) = DepartureFlightList(i, 5)
                NewSchedule.Cells(i + 3, 13) = DepartureFlightList(i, 6)
            Next i
        End If

        If WriteDeparture = True Then
            NewDeparture.Cells(1, "B") = "System infeed"
            NewDeparture.Cells(1, "M") = "In-system"
            NewDeparture.Cells(1, "X") = "System outlet"

            NewDeparture.Cells(2, "A") = "Simulation minute"
            NewDeparture.Cells(2, "B") = "Originating"
            NewDeparture.Cells(2, "C") = "Flow 1"
            NewDeparture.Cells(2, "D") = "Flow 2"

            NewDeparture.Cells(2, "F") = "Transfer"
            NewDeparture.Cells(2, "G") = "Flow 7"
            NewDeparture.Cells(2, "H") = "Flow 8"

            NewDeparture.Cells(2, "J") = "Incoming"
```

```
            NewDeparture.Cells(2, "K") = "Flow 26"

            NewDeparture.Cells(2, "M") = "Flow 11"
            NewDeparture.Cells(2, "N") = "Security"
            NewDeparture.Cells(2, "O") = "Flow 12"

            NewDeparture.Cells(2, "Q") = "Flow 13"
            NewDeparture.Cells(2, "R") = "Baggage store"
            NewDeparture.Cells(2, "S") = "Flow 14"

            NewDeparture.Cells(2, "U") = "Sorting"
            NewDeparture.Cells(2, "V") = "Sorting positions"

            NewDeparture.Cells(2, "X") = "Flow 9"
            NewDeparture.Cells(2, "Y") = "Flow 10"
            NewDeparture.Cells(2, "Z") = "Make-up"
            NewDeparture.Cells(2, "AA") = "Make-up storage"


            NewDeparture.Cells(2, "AC") = "Flow 26"
            NewDeparture.Cells(2, "AD") = "Outgoing"

            For i = 0 To SimLength
                NewDeparture.Cells(i + 3, "A") = i
                NewDeparture.Cells(i + 3, "B") = ProcessRecord(1, i)
                NewDeparture.Cells(i + 3, "C") = FlowRecord(1, i)
                NewDeparture.Cells(i + 3, "D") = FlowRecord(6, i)

                NewDeparture.Cells(i + 3, "F") = ProcessRecord(14, i)
                NewDeparture.Cells(i + 3, "G") = FlowRecord(14, i)
                NewDeparture.Cells(i + 3, "H") = FlowRecord(19, i)

                NewDeparture.Cells(i + 3, "J") = ProcessRecord(17, i)
                NewDeparture.Cells(i + 3, "K") = FlowRecord(45, i)

                NewDeparture.Cells(i + 3, "M") = FlowRecord(25, i)
                NewDeparture.Cells(i + 3, "N") = ProcessRecord(4, i)
                NewDeparture.Cells(i + 3, "O") = FlowRecord(29, i)

                NewDeparture.Cells(i + 3, "Q") = FlowRecord(34, i)
                NewDeparture.Cells(i + 3, "R") = ProcessRecord(10, i)
                NewDeparture.Cells(i + 3, "S") = FlowRecord(35, i)

                NewDeparture.Cells(i + 3, "U") = ProcessRecord(3, i)
                NewDeparture.Cells(i + 3, "V") = ProcessRecord(2, i)

                NewDeparture.Cells(i + 3, "X") = FlowRecord(23, i)
                NewDeparture.Cells(i + 3, "Y") = FlowRecord(24, i)
                NewDeparture.Cells(i + 3, "Z") = ProcessRecord(9, i)
                NewDeparture.Cells(i + 3, "AA") = ProcessRecord(8, i)

                NewDeparture.Cells(i + 3, "AC") = FlowRecord(45, i)
                NewDeparture.Cells(i + 3, "AD") = ProcessRecord(18, i)
            Next i
        End If

        If WriteArrival = True Then
            NewArrival.Cells(1, "B") = "System infeed"
            NewArrival.Cells(1, "G") = "In-system"
            NewArrival.Cells(1, "L") = "System outlet"

            NewArrival.Cells(2, "A") = "Calculation minute"
            NewArrival.Cells(2, "B") = "Terminating"
            NewArrival.Cells(2, "C") = "Flow 3"
            NewArrival.Cells(2, "D") = "Flow 4"
            NewArrival.Cells(2, "E") = "Flow 5"

            NewArrival.Cells(2, "G") = "Flow 16"
            NewArrival.Cells(2, "H") = "Customs screening"
            NewArrival.Cells(2, "I") = "Flow 15"
            NewArrival.Cells(2, "J") = "Flow 18"

            NewArrival.Cells(2, "L") = "Flow 17"
            NewArrival.Cells(2, "M") = "Reclaim"

            For i = 0 To SimLength
                NewArrival.Cells(i + 3, "A") = i
                NewArrival.Cells(i + 3, "B") = ProcessRecord(13, i)
                NewArrival.Cells(i + 3, "C") = FlowRecord(10, i)
                NewArrival.Cells(i + 3, "D") = FlowRecord(11, i)
                NewArrival.Cells(i + 3, "E") = FlowRecord(12, i)

                NewArrival.Cells(i + 3, "G") = FlowRecord(39, i)
                NewArrival.Cells(i + 3, "H") = ProcessRecord(11, i)
```

```
                NewArrival.Cells(i + 3, "I") = FlowRecord(38, i)
                NewArrival.Cells(i + 3, "J") = FlowRecord(41, i)

                NewArrival.Cells(i + 3, "L") = FlowRecord(40, i)
                NewArrival.Cells(i + 3, "M") = ProcessRecord(12, i)
            Next i
        End If

        If WritePerAirline = True Then
            NewOriginating.Cells(1, "B") = "Originating flow per airline"
            NewOriginating.Cells(2, "A") = "Calculation minute"

            For i = LBound(AirlineList) To UBound(AirlineList)
                NewOriginating.Cells(2, i + 1) = AirlineList(i)
            Next i


            For i = 0 To SimLength
                NewOriginating.Cells(i + 3, "A") = i

                For j = LBound(OriginatingAirline, 2) To UBound(OriginatingAirline, 2)
                    NewOriginating.Cells(i + 3, j + 1) = OriginatingAirline(i, j)
                Next j
            Next i
        End If
        NewBook.Close (True)
    End If
End Sub
```

K.16. WRITE FILE

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''
'''  Private function to calculate the random numbers needed throughout the calculation
'''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Private Function Probability(Distr As String, a As Variant, b As Variant, c As Variant, x As ...
     Double) As Double
    'A function to generate a probability of different distributions with the inverse cdf. ...
        Distr assigns the_
    'case, a assigns the minimum, b assigns the maximum and c the mean or mode. X is the ...
        random variable.
    Dim var As Double
    Dim norm As Double
    Dim triangular As Double
    Dim threshold As Double

    Dim u1 As Double
    Dim u2 As Double
    Dim v1 As Double
    Dim v2 As Double
    Dim w As Double

    Select Case Distr
        Case "Normal"
            Do
                u1 = Rnd    'not tested yet, could be a problem with Rnd function twice, or ...
                    that it cannot be called within a function?
                u2 = Rnd
                v1 = 2 * u1 - 1
                v2 = 2 * u2 - 1
                w = v1 ^ 2 + v2 ^ 2
            Loop Until w ≤ 1

            var = (b - c) / 3
            norm = v1 * VBA.Sqr(-2 * VBA.Log(w) / w)
            Probability = c + VBA.Sqr(var) * norm

        Case "Exponential"
            Probability = -VBA.Log(1 - x) * c
        Case "Erlang"
            'Add some here
        Case "Triangular"
            If (b - a) < 0.001 Then
                Probability = 0
            Else
                threshold = ((c - a) / (b - a))
                If x ≤ threshold Then
                    triangular = VBA.Sqr(threshold * x)
```

```
                Else
                    triangular = 1 - VBA.Sqr((1 - threshold) * (1 - x))
                End If
                Probability = a + (b - a) * triangular
            End If
        Case "Uniform"
            Probability = a + (b - a) * x
        Case "Constant"
            Probability = c
        Case "Custom1"
            'Custom distribution if required
    End Select
End Function
```