Evaluation of conceptual design choices using dependency structure matrix methods

Meeusen, K.A.

Award date:
2019

Department of Mechanical Engineering

Evaluation of Conceptual Design Choices using Dependency Structure Matrix Methods

Graduation Project

CST 2019.026

Student:	K.A. Meeusen (Koen)
ID nr.:	0657087
Supervisors:	
	Dr.ir. L.F.P. Etman
	Dr.ir. T. Wilschut
Company Supervisors:	
	Ir. A.T.A. Peijnenburg
	H. Segers

Eindhoven University of Technology
Department of Mechanical Engineering
Control Systems Technology Group

March 20, 2019

Technische Universiteit
Eindhoven
University of Technology

Graduation project

Evaluation of conceptual design choices using dependency structure matrix methods

Introduction

In engineering design, engineers face many conceptual design choices that have a profound impact on a system's key-performance-indicators (KPI's), such as reliability, availability, and life-cycle-cost. Conceptual design choices occur at various stages of the design process, may be inter-dependent, and are often based on qualitative and incomplete data. Additionally, exploring each design concept in great detail is cost and time ineffective, and therefore, usually not feasible. Properly evaluating and motivating each conceptual choice is a non-trivial but import task in design processes, in which insight into the system architecture is required.

The goal of this project is two-fold. First of all, it is aimed to valid the previously developed Elephant Specification Language (ESL), which is a novel language for writing multi-level system specifications and the automated derivation of component, function, and variable dependencies. The network of dependencies provides a model of the system architecture. Secondly, it is aimed to show the effect of design choices on system architecture and critical points w.r.t. KPI's within that system architecture. A critical point 'short list' enables engineers to focus their efforts in evaluating multiple design concepts. As subject of study, the vacuum wafer-handler (WH) developed by VDL ETG is used.

Assignment

The assignment consist of the following tasks:

1. Specify and analyze the 'as is' WH system architecture at multiple decomposition levels.
2. Identify critical points in the 'as is' system architecture w.r.t. selected KPI's.
3. Specify and analyze an alternative WH system architecture at multiple decomposition levels. Document the alternative conceptual design choices and what effect those choices have on the system architecture.
4. Identify critical points in the alternative system architecture w.r.t. selected KPI's.
5. Compare the 'as is' and alternative system architecture. What are the commonalities and differences?
6. Compare the critical points w.r.t. the selected KPI's in both system architectures, which are easiest to manage or have the least impact? Elaborate on the trade-offs in the comparison.

Size

45 ECTS

Reference

Browning, T. R., 2016. "Design structure matrix extensions and innovations: a survey and new opportunities". IEEE Transactions on Engineering Management, 63(1), pp. 27- 52.

Start June 2018
Finish February 2019
Student K.A. Meeusen
Supervisors ir. A.T.A. Peijnenburg, Dr. ir. L.F.P. Etman, Ir. T. Wilschut

Technische Universiteit
Eindhoven
University of Technology

Declaration concerning the TU/e Code of Scientific Conduct for the Master's thesis

I have read the TU/e Code of Scientific Conduct[i].

I hereby declare that my Master's thesis has been carried out in accordance with the rules of the TU/e Code of Scientific Conduct

Date

19-03-2019

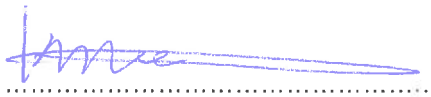
Name

K.A. Meeusen

ID-number

0657087

Signature



Submit the signed declaration to the student administration of your department.

[i] See: https://www.tue.nl/en/our-university/about-the-university/organization/integrity/scientific-integrity/
The Netherlands Code of Conduct for Scientific Integrity, endorsed by 6 umbrella organizations, including the VSNU, can be found here also. More information about scientific integrity is published on the websites of TU/e and VSNU

Preface

This report contains the result of my master graduation project at the department of Mechanical Engineering at Eindhoven University of Technology and the VDL Enabling Technologies Group. Investigation if ESL and DSM methods can support the evaluation of conceptual design choices in the development of high-tech engineering systems is performed in this project. As a case problem the waferhandler, made and designed by VDL ETG is used. The research in this report is performed from June 2018 until March 2019.

This report is the result of my graduation project at the VDL Enabling Technologies Group. The months that I worked on this project, I really enjoyed working on this project and time flew by. When I got stuck on part of the protect, I worked on another part to the keep momentum up. This way there was always some progress which I could discuss with my supervisors.

I would like to thank my supervisors, Pascal Etman, Tim Wilschut, Ton Peijnenburg and Huib Segers for their time, feedback and support during this project. The meetings were always inspiring and were the basis for new ideas.

During my project I talked to a lot of people at VDL ETG. In particular, I would like to thank Joep Wijn, Roel Meulenbroeks and William v.d. Put for the time they took to explain the wafer handler in more detail.

I would like to thank my fellow students, Toby Erens and Jelte de Jong, who also performed their graduation project at the exact same time in the System Engineering Laboratory.

Last but not least, I want to thank my parents, John and Mirjam Meeusen, for their support over the years.

Koen Meeusen
Eindhoven, March 20, 2019

Summary

As engineering systems are becoming increasingly complex, ever growing requirement sets have to be satisfied. During early design phases, engineering systems are decomposed. The way the modules, sub-modules, and components are related to each other is called the system architecture.

Since there is no unique solution to a design problem, conceptual design choices have to be made. With increasing size and complexity of systems, conceptual design choices have more profound effects on the system's architecture. Before making a motivated decision, information is needed. This information is not always available. A tool to deal with the information available to understand, design and improve systems is the Dependency Structure Matrix or Design Structure Matrix (DSM).

The Elephant Specification Language (ESL) is a language for writing multi-level design-specifications for engineering systems. From an ESL specification dependencies between system components are derived and visualized in a DSM. The goal of this graduation project is to investigate if ESL and DSM methods can support the evaluation of conceptual design choices in the development of high-tech engineering systems. As a case problem, the Waferhandler (WH) from VDL ETG is used to evaluate and validate the proposed approach.

For evaluation of conceptual design choices using ESL and DSM a four step method is presented:

1. An ESL specification is created of the current system architecture, from which a DSM is generated. The DSM is the basis for projections of Key Performance Indicators (KPIs) on the DSM. Critical areas are defined based on the KPI projections.
2. The theory of inventive problem solving (TRIZ) is selected as method for generating an alternative system architecture. Critical areas of the original system architecture are input to TRIZ. By translating the KPIs into the TRIZ technical parameters inventive principles are suggested by the contradictions matrix.
3. The inventive principles, are applied to the WH system architecture. A new DSM is generated and KPI values for the alternative design are projected to the DSM.
4. Pairwise comparison, part of the Analytic Hierarchic Process (AHP), is used to compare the system architectures. To enhance the pairwise comparison the Changeover-Delta DSM (CD-DSM) is used. These allow a pairwise comparison, where changes in dependencies and KPI values can be compared. This allows a design team to pairwise compare system architectures and support conceptual design choices.

For the WH case study, the TRIZ inventive principles are; segmentation, parameter change and cushioning in advance. By segmentation of the LoadLock and Robot Vacuum Chamber the wafer input path and output path of the vacuum WH are split. Pairwise comparison shows that the generated alternative system architecture is preferred over the original system architecture. Due to segmentation of the WH, smaller volumes of the vacuum have to be restored after repairs and modules are easier accessible. This results in an improvement in repair times compared to the original system architecture.

The proposed method and case problem show that ESL and DSM can be used in the early conceptual design stage to support the evaluation of conceptual design choices. The usage of ESL, DSM, TRIZ and AHP combined proves to be a successful combination. It is expected that the proposed method will be sufficient for most systems.

Contents

Chapter 1

Introduction

Nowadays, engineering systems are becoming increasingly complex as they have to satisfy ever growing requirement sets (Pahl and Beitz [1]). In the early design phases, engineering systems (or products) to be developed are gradually decomposed into modules, sub-modules, and components each with its own set of requirements. Together, the (sub)-modules and components have to fulfil the top-level requirements. The way the modules, sub-modules, and components are related to each other, is called the system architecture.

Since there is (generally) no unique solution to a design problem, many conceptual design choices have to be made. The conceptual design phase, is an early design phase were systems are abstracted to find essential problems, to establish function structures, to determine suitable working principles and to combine them into working structures [1]. This includes the design of interfaces, processes and control strategies within a system. Conceptual design choices, are the choices made related to the outcome of the conceptual design phase. Each (conceptual) design choice affects the system's architecture.

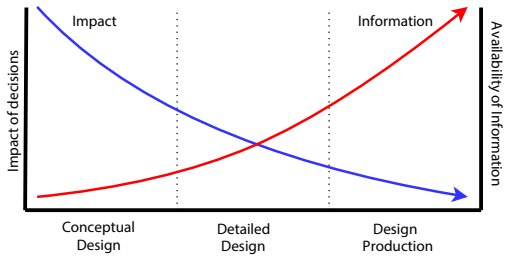


Figure 1.1: Impact of decisions and availability of information in the design phases, modified from [2]

With increasing size and complexity of systems, conceptual design choices have more profound effects on the system's architecture. A particular conceptual design decision can limit or exclude certain solutions due to interdependency of (sub)-modules. Thus, decisions can have a significant impact, as shown in Figure 1.1. Before making a motivated decision, information is needed. This information is not always available, incomplete, subject to uncertainty and imprecision, or simply too much to understand the consequences [3]. As the development of the design progresses and decisions are made, more information regarding the system's architecture becomes available.

A tool to deal with the information available to understand, design and improve systems, is the Dependency Structure Matrix or Design Structure Matrix (DSM) (Steward [4]). A DSM (of which an example is shown in Figure 1.2) is a system representation by means of an $N \times N$ matrix (with identical row and column labels), which can be used for analysis, design and management of systems. In a DSM, the system is decomposed into N elements which represent sub-systems or components. Marks (binary) or numbers in the matrix represent relationships between these elements. The usage of simple marks only shows that a relationship is present, whereas a numerical DSM also depicts the strength or importance of the various relationships. Both DSM types provide a visual model, for analysis by the user. The DSM has been used for analysis in a large number of applications, of which an overview is given in both Browning [5] and Eppinger and Browning [6].

In de conceptual design phase there is also the difficulty that there is often multiple ways to realize a particular function, which may affect the product decomposition and its dependency structure. Hence, tooling to support this activity is desired.

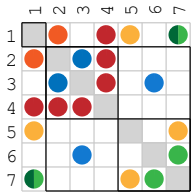


Figure 1.2: Example DSM, from [7]

In previous research, Wilschut [7] has developed the Elephant Specification Language (ESL) which is a computer readable language for writing multi-level design-specifications for engineering systems. From an ESL specification dependencies between system components, functions, and (design) variables can automatically be derived and visualized in a DSM. Such a function specification language with accompanying automated DSM generation may present the opportunity to effectively model design alternatives in the early design phase. Hence, this research investigates the suitability of ESL for this purpose and develops dedicated DSM modelling methods to support this. A case problem from VDL ETG is used to evaluate and validate the proposed approach.

1.1 VDL ETG

Founded in the year 1900 as Philips Machinefabrieken, VDL Enabling Technologies Group [8] (Eindhoven, the Netherlands), is a company which designs and produces solutions for, among others, the semi-conductor, medical, solar and defence industries. One of the products developed for a semiconductor customer is the Waferhandler (WH). The WH is a module of a AMSL Waferstepper (a lithography system) and feeds wafers from the track (transport) onto the Waferstage where the lithography process will take place. After lithography, the wafer is removed by the WH and placed back on a (different) track. Within the WH several processes (like: measurements, robotic movements, pneumatics, thermal and hydraulic heating and cooling) are executed simultaneously. The various engineering disciplines result in a complex system architecture. In this research, the Waferhandler for the newest generation of Wafersteppers is used as a case study to validate the proposed method for evaluating conceptual design choices on system architectures.

Each version of the WH is an evolution of previous versions. For each version, the entire design process must be followed. Hence, determining whether a modification of the design has a positive effect on the WH performance is desired.

A new WH design must fulfil a set of specifications and requirements. Typically, multiple design solutions meet the requirements set. Poorly defined requirements can lead to a non-optimal solution, or even failure to reach the system requirements due to the influence and dependencies of other components. To deal with dependencies of the (sub-)modules and components, insight is needed on how these (sub-)modules and components influence each other.

By evaluating possible changes on the subsystems, one can look at the effects within the larger system. This can help to motivate conceptual design choices when designing a complex system such as the WH. The considered design change is to be evaluated against a short list of Key Performance Indicators (KPIs) in the critical areas of the WH. These KPIs represent selected key performance objectives and are measurable values that demonstrate how effectively a system can achieve those objectives, such as reliability, availability, throughput and contamination.

For evaluation of design alternatives, the WH system architectures resulting from those design choices need to be compared and the effects on the KPIs quantified. By making different conceptual design choices, alternative system architectures may be formulated. Hence, the need for a method through which system architectures can easily be compared. Moreover, the effect of the alternative system architectures on the KPIs need to be displayed. A KPI comparison of candidate system architectures can give insight in the consequences of conceptual design choices. These insights can then be used to motivate conceptual design choices.

DSMs are used to visualize dependencies between components. If two system architectures are visualized by DSM models, a comparison between the DSM can provide insight into the differences and commonalities between

the systems. Furthermore, as presented in Brady [9], KPI values can be projected to the DSM. This visualizes the effect of the KPI onto dependencies between components. The KPIs projections, can help to identify the critical areas and to evaluate the effect of changes to the system architecture.

1.2 Research objective

The impact of configuration changes on the system and its architecture can be visualized and analysed by means of the DSM. The ESL language facilitates DSM model building through the concise specification of component functions, parameter flows, and their respective dependencies at various levels in the decomposition tree (product breakdown). DSM relations are derived from the functional specifications and visualized into a component DSM, function DSM and parameter DSM, respectively.

In the existing DSM literature methods have been proposed to visualize the effect of design changes or technology changes. However, a method to compare to design alternatives regarding their pros and cons from the viewpoint of the system performance seems to be missing. What is more, a method is desired that provides an integrated approach towards functional requirements specification, design alternatives generation, configuration comparison and selection. DSM modelling and the ESL language are considered to be two key elements to this end. Therefore, the research goal of this thesis is:
Investigate if ESL and DSM methods can support the evaluation of conceptual design choices in the development of high-tech engineering systems.
To achieve the main objective, the research goals are:

1. Investigate how ESL and DSM modelling can be used in cooperation to identify critical areas with respect KPIs within a system architecture.
2. To investigate how alternative system configuration designs can be generated to improve KPI critical areas in the machine.
3. To investigate how the commonalities and differences between KPI critical areas of two system configurations can be compared and evaluated.
4. To develop a method that provides the desired integrated approach towards system modelling, design concept generation, and design concepts comparison.
5. To carry out the research and develop the method using the NXE:3400C wafer handler developed by VDL ETG as the main case problem.

The outcome of the research is a method for supporting conceptual design that embodies ESL for creating structure design specifications; DSM for visualizing and analysing the system architecture; Theory of Inventive Problem Solving (TRIZ) [10][11] for finding design alternatives; and Analytical Hierarchical Process (AHP) [12] for comparing the proposed alternative system architectures.

To support the method, a toolbox adding functionality to the ESL compiler is presented. The toolbox adds the necessary functionality needed to perform the presented method for evaluation of conceptual design choices. The toolbox supports to derive dependencies between elements defined in ESL and use these dependencies to visualize differences, additions, eliminations and commonalties between system architectures.

1.3 Report outline

The general outline is as follows; Chapter 2 presents the theoretical background of the methods used and other research related to these methods. Subsequently, Chapter 3 introduces a method for the evaluation of design alternatives. Chapter 4 evaluates the usage of ESL and presents additions made to the ESL compiler developed during this project. These additions are needed to compute and generate some of the DSMs presented in the subsequent chapters.

A case study with the Waferhandler will be preformed to validate the method presented in Chapter 3. Chapter 5 analyses the current system architecture of the wafer handler and identifies critical areas with respect to a set of Key Performance Indicators. These critical points are used as input to generate an alternative system architecture which is presented in Chapter 6. Both system architectures are compared in Chapter 7. Finally, Chapter 8 concludes this report and presents recommendations for further research.

Chapter 2

Related work

The goal of the project is to evaluation of conceptual design choices using dependency structure matrix methods. Methods are needed for the purpose of modelling, alternative generation and evaluation of system architectures. In this chapter, the background of the methods used in this project are introduced. For each method a short introduction and explanation is given regarding the working principles of the method.

First, some definitions are given regarding system architectures and conceptual design. The second section introduces Design structure matrices and ESL, methods used for modelling and analysis of system architectures. Section three, explains TRIZ, used for finding alternative designs. Finally, the Analytical Hierarchic Process (AHP) method is presented as a method for evaluation of design changes.

2.1 System Architectures

Systems are built according to a system architecture. A definition of a system architecture, concerning engineering systems (or products) is given by Ulrich [13]:

> *"(1) the arrangement of functional elements; (2) the mapping of functional elements on physical components; (3) the specification of the interfaces among interacting physical components."*

A system architecture is the result of design choices in order to fulfil the requirements of the system to be created. The design choices must address the fundamental and practical difficulties which occur due to conceptual design choices. The system architecture resulting from those conceptual design choices, determines which working principles are used and the order of the processes.

Ulrich [13] shows that different approaches can result in different designs. If opted for a modular design, modules can be exchanged throughout the system. A negative effect can be the risk of creating organizational barriers to architectural innovation [13]. Hence, a design choice made in the beginning of the design process and can have negative influence further in the project. Crawley [14] provides more information regarding the management of complex systems by using system architectures.

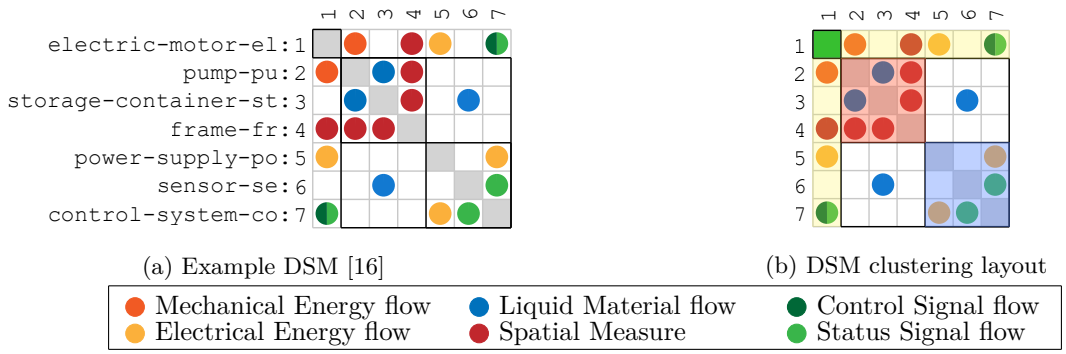
Jaakkola [15]) argues that a system architecture can be used to:

- *explain* - the architecture explains the structure of the system.
- *guide* - the architecture guides the engineer to design within specified constraints.
- *enable* - the architecture provides a high level mechanism to implement the requirements of the product. Whereby the mechanisms to be implemented are determined or influenced by the conceptual design choices.

To understand the effect of design choices on the system architecture, modelling and visualization of system architectures can provide insight into the consequences of design choices. The advantage of a graphical overview, is that a system engineer can see how a system architecture is constructed in one clear overview. One of the methods to create a visual representation of the system is the Design Structure Matrix.

2.2 Modelling system architectures

For the purpose of modelling system architectures, the Design Structure Matrix (DSM)(Steward [4]) has proven to be an effective and analytically compact method to analyses system architectures (Eppinger and Browning [6]). A DSM is a graphical system representation, wherein the system is decomposed into elements, modules or components. The number of elements (n) determines the size of the DSM, namely ($n \times n$). On both axes of the DSM, the elements are listed in the same order. An example DSM is shown in Figure 2.1a.



(a) Example DSM [16] (b) DSM clustering layout

Figure 2.1: Example DSM and the DSM structure

The purpose of a DSM is to give a visual representation of the system architecture. Dependencies between defined elements of the systems architecture are highlighted in the off diagonal matrix entries. By using markings, numerical values or colour codes (as in Figure 2.1a), different dependency types can be indicated. The usage of simple marks (binary) only shows that a relationship is present, whereas a numbers and colours (both numerical) in the DSM can show different dependency types or quantify them.

The rows and columns of the DSM shown in Figure 2.1a are sorted with a clustering algorithm to highlight the system architecture. Clustering is a permutation of the rows and columns of the DSM, such that clusters are created. Clustering groups elements, which have a strong dependency on each other, into groups around the diagonal of the matrix. Clustering is one way to look at the system, but other views may exist. Hence, a DSM (and clustering) is a method to facilitate the analysis of the system.

Figure 2.1 shows two identical DSMs, but Figure 2.1b highlights the clustering layout of the DSM. In the solid green cluster (which in this example consists of just a single element) depicts the bus cluster. Elements in the bus cluster often fulfil collective functions (e.g. controllers) or depend on many elements (e.g. a chassis). The other clusters (transparent red and blue) contain elements with a strong dependency on other elements in that same cluster. The yellow entries contain dependencies between the bus elements and the non-bus elements. The unmarked squares in Figure 2.1b show dependencies between (elements within) the red and blue cluster

Clustering can be done manually, or by algorithm. Examples of clustering algorithms include the multi-level Markov clusterings algorithm developed for DSMs by Wilschut [17] and the cost (minimization) based clustering algorithm by Thebeau [18]. It is important to note, that clustering provides a specific view on the system and does not have a unique result. It is up to the maker of the DSM to determine which clustering result meets the engineering insights.

By decomposing a system, smaller sub-systems can be defined of lower complexity, to obtain a better overview and get access to more information regarding the various interfaces between the subsystems. Furthermore, dependencies between subsystems can be noted, giving insight into integration problems and the behaviour of the system. By evaluating design changes on the subsystems, one can look at the effects within the larger system.

A DSM can be hand-made or computer generated. Wilschut [19], compares a hand-made DSM to a (from ESL specifications) computer generated DSM. It is concluded that a computer generated DSM is less time consuming and makes multiple dependency types easier to apply. The DSM is generated by means of the Elephant Specification Language (ESL). From an ESL specification, dependencies between system components, functions, and (design) variables are automatically derived and visualized in a DSM.

2.2.1 Elephant Specification Language

ESL is a functional requirement-based specification language. The syntax of ESL is presented in Wilschut [20]. The functional basis of Hirtz [21] (extended by Tilstra [22]) is used for the functional requirements. By describing the needs, requirements and constraints (between components) in a fixed format, the ESL compiler is able to derive dependencies between the defined components. This is shown in Wilschut [19]. From these derived dependencies, a Multi Domain Matrix (MDM) is generated. This MDM consists of a component DSM, a function DSM and a parameter DSM, as well as mapping matrices describing the mapping relations between the three DSMs.

By decomposing a component into subcomponents, the lower level (system) architecture can be visualized by a DSM [22]. In [20] it is shown how ESL can be used to define multi level system architectures using a single ESL specification. The ESL compiler then generates a clustered DSM for each level of decomposition. The multi-level Markov clusterings algorithm implemented into the ESL compiler is presented in Wilschut [17].

An application of ESL is given in Wilschut [23], where a lock renovation pilot study is presented. Using ESL the effects of changes to the system architecture / structure are highlighted. Since this project is a continuation of the research performed by Wilschut, ESL will be used to specify the WH system.

2.2.2 System Quantification Methods

To assess the impact of Key Performance Indicators (KPIs), DSM projections can be used. A projection DSM, is the mapping of values to the dependencies and/ or components of a DSM. The (already) clustered DSM is used as a basis and the order of the elements remain unchanged. Hence, only the projection of the values to the DSMTwo types of projections are defined:

Quantification of dependencies can give insight in which dependencies are most important and which system modules contain most of the high impact dependencies. To quantify dependencies, a general qualification scheme such a proposed by Pimmler [24] may be used. They use a 5-point scale (ranged between -2 and +2) to assign weights to dependencies. These weights indicated in the DSM, creating a weighted DSM.

In Brady [9], each DSM element is given a score regarding the technology readiness, which expresses the matureness of the technology used for a particular element. The goal of Brady's research was to find the modules of the NASA Mars rover that contained many of such high risk elements. Each dependency score is constructed out of the values of the corresponding elements and the type of dependency. The used dependency types are based on the functional basis reconciled flow set of Hirtz [21]. The scores are projected on the (already clustered) DSM. Based on the projection, areas of interest are defined. These areas of interest provide insight where the technology readiness needs to be improved.

In Wilschut [16], reliability and availability used as attributes to the elements of the DSM. The scores on the nonlinear scale 1,3,9 are used to represent the associated risk. Such a scoring is commonly employed in Quality function deployment to analyse the system properties.

2.3 Generating Design Alternatives

Alternative system designs can be created by small changes or entirely new designs. All designs need to go through the entire development cycle to determine the effect of the changes. One way to come up with design changes is by seeking to turn negative effects into neutral or positive effects. To come up with solutions, creative thinking is required.

In Altshuller [10] a systematic approach to support creative thinking is presented. This approach later evolved into TRIZ. TRIZ is a Russian abbreviation of Teoriya Resheniya Izobreatatelskikh Zadatch (Theory of the Solution of Inventive Problems). Like the name suggests, TRIZ is a systematic method to come up with inventive solutions for design problems one might encounter.

The TRIZ method to solve problems, is to approach the contradictions. A problem typically concerns the improvement of one parameter performance without deteriorating another parameter. TRIZ seeks to solve such a contradiction by departing from 40 inventive design principles (Altshuller [11]). Design principles are

suggested by the contradiction matrix, where the parameters to improve are set against parameters not to deteriorate. The 40 principles are based on design principles of (granted) patents. Reapplying these principles to similar problems may provide a solution in which a similar contradiction is eliminated.

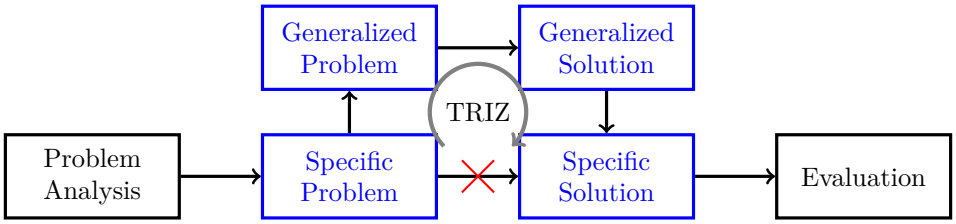


Figure 2.2: The TRIZ approach, modified from [25]

The key principle of TRIZ is shown in Figure 2.2. From analysis, a specific problem can be defined. Instead of creating a specific solution straight away, a different approach is used. The specific problem is translated into a generalized problem. This is then solved by a generalized solution. This way, existing solutions can be used to solve the generalized problem and double work (or reinventing the wheel) is avoided. The generalized solution is then adapted to create the specific solution to the original specific problem. This method (and other TRIZ information) is extensively explained in both Haines-Gadd [26] and in Gadd [27].

An example of TRIZ problem solving, using the 40 inventive principles, can be a heavy object (e.g. a stone) that needs to be moved from A to point B. The specific problem is that the object needs to be moved with the lowest effort possible. Translating this to a general problem, one can say the amount of energy to move the object needs to be optimized without changing the weight or shape. TRIZ shows that the principle "Equal potential" might provide a general solution. Equal potential can be interpreted as, not raising or lowering the object. In case of the heavy object (the specific solution), rollers or a cart can be placed under the object, avoiding the need to raise and lower the object while moving.

TRIZ is also used in system architecting. In Bonnema [25], TRIZ is used to find alternatives within an architecting approach (presented in Bonnema [28]) which connects key drivers to system functions using a coupling matrix. Using the relations and key functions, architectures are created and alternatives can be found using TRIZ. The effects of the changes are only marked positive or negative. No quantitative method is coupled to the method presented in Bonnema [25]. Chechurin [29] shows that if QFD and TRIZ are combined, QFD defines the contradictory requirements and TRIZ is used to eliminate them. Tseng [30], extends the usage of QFD and TRIZ with a task sequencing (asymmetrical) DSM. So it is concluded that TRIZ can be used to find alternative system architectures, whereby a quantitative ranking can provide the contradictory requirements to be solved.

2.4 Evaluating Design

To determine whether an alternative design is an improvement, the alternative system architectures have to be compared. Furthermore, a decision has to be made which system architecture is most optimal.

The Analytic Hierarchy Process (AHP) (Saaty [12] [31]) is one of the most popular methods to the purpose of decision making in engineering applications (Triantaphyllou [32]). AHP consists of two parts, namely:

1. Pairwise comparison between the criteria
2. Pairwise comparison between the candidates

Pairwise comparisons are used to calculate weights of importance. The advantage of pairwise comparison is that criteria which have no direct relation to each other can be compared. Only a decision has to be made regarding which criteria is more, or equally, important over another. One or multiple persons (representing stakeholders or domain experts) can decide which criteria is more or equally important over another.

All criteria are put through a pairwise comparison, this eventually results in a normalized vector containing scaled weights of importance.

The full AHP procedure including calculations is explained and presented in [32]. For convenience the AHP calculations are also presented in Section A.4.

After the criteria are compared, candidates (the available options to be chosen from) can be pairwise compared along each criteria. The result is a matrix with decision results. When this matrix is multiplied with the criteria vector, a second (normalized) vector is the result. The highest value(s) is/ are the preferred design based on the decisions made.

Darwish [33] presents a framework, to make decisions regarding system architectures. To this purpose they use AHP, based on quantitative attributes. The quantifications of system architectures are used to compare the design and make a decision regarding the optimal system architecture.

Chapter 3

Method

This chapter presents a method for generating, evaluating and comparing design alternatives. The method combines Dependency Structure Matrix (DSM)[4] modelling, Theory of Inventive Problem Solving (TRIZ)[10] and Analytical Hierarchical Process (AHP)[12] techniques. DSM modelling is used to create system architecture models, visualize the effect of design changes, and to highlight critical areas with respect to various Key Performance Indicators (KPIs). TRIZ is used as a structured method to create design alternatives. AHP is used to compare those alternatives based on KPI and domain expert opinions. All methods are combined to create a method which can generate alternatives and evaluate the changes. A schematic overview of the entire method, presented in this chapter, can be seen in Figure 3.1. This method consists of four main steps:

1. Design specification and analysis
2. Alternative design generation
3. Alternative design specification and analysis
4. Design comparison

In step 1 (red block in Figure 3.1), a system requirement specification is created, stating what the original system must do. The Elephant Specification Language (ESL) (Wilschut [20]), a language designed for writing structured multi-level system specifications, is used as a tool within the method to automatically derive (functional) dependencies between components. By visualizing these dependencies in a DSM, a model of the system architecture can be created. Critical points within the system are defined using the ranking of the various components within the system with respect to the selected Key Performance Indicators (KPIs). To this purpose, a method similar to Brady [9] is used.

The results of step 1 are used as input in step 2 (the green block) to generate alternative designs. The alternative design consists of changes and/or improvements to the original system architecture.

Of generated alternative, a ESL specification is specified in step 3 (the yellow block). In the ESL specification, the changes to the system architecture are implemented. As in the original system design, a DSM is derived and components within the system are ranked using the same KPIs as the original system.

Finally, in step 4 (the blue block), a comparison between both original and alternative system architectures is preformed. Both designs are compared using a Changeover-Delta DSM (explained in Section 3.4). For each KPI, the system designs are compared

In the remainder of this chapter each of these steps is explained in further detail.

3.1 Modelling System Architectures

The system requirement specifications are specified using ESL. Relations within the system are described in functional requirements and elements are defined on multiple decomposition levels. This enables to specify the system at multiple levels of granularity. The greater the decomposition level, the smaller the system granules, the greater the level of detail of the specification (Maier [34]). Depending on the system decomposition, only the functional requirements between the elements specified on each specific decomposition level have to be defined. For lower levels of decomposition, the higher layer functional requirements are automatically taken into account.

Using the multi-level system specifications, component, function and variable dependencies can automatically derived using the ESL compiler [20]. Programming design changes into the specification is achieved by adding or

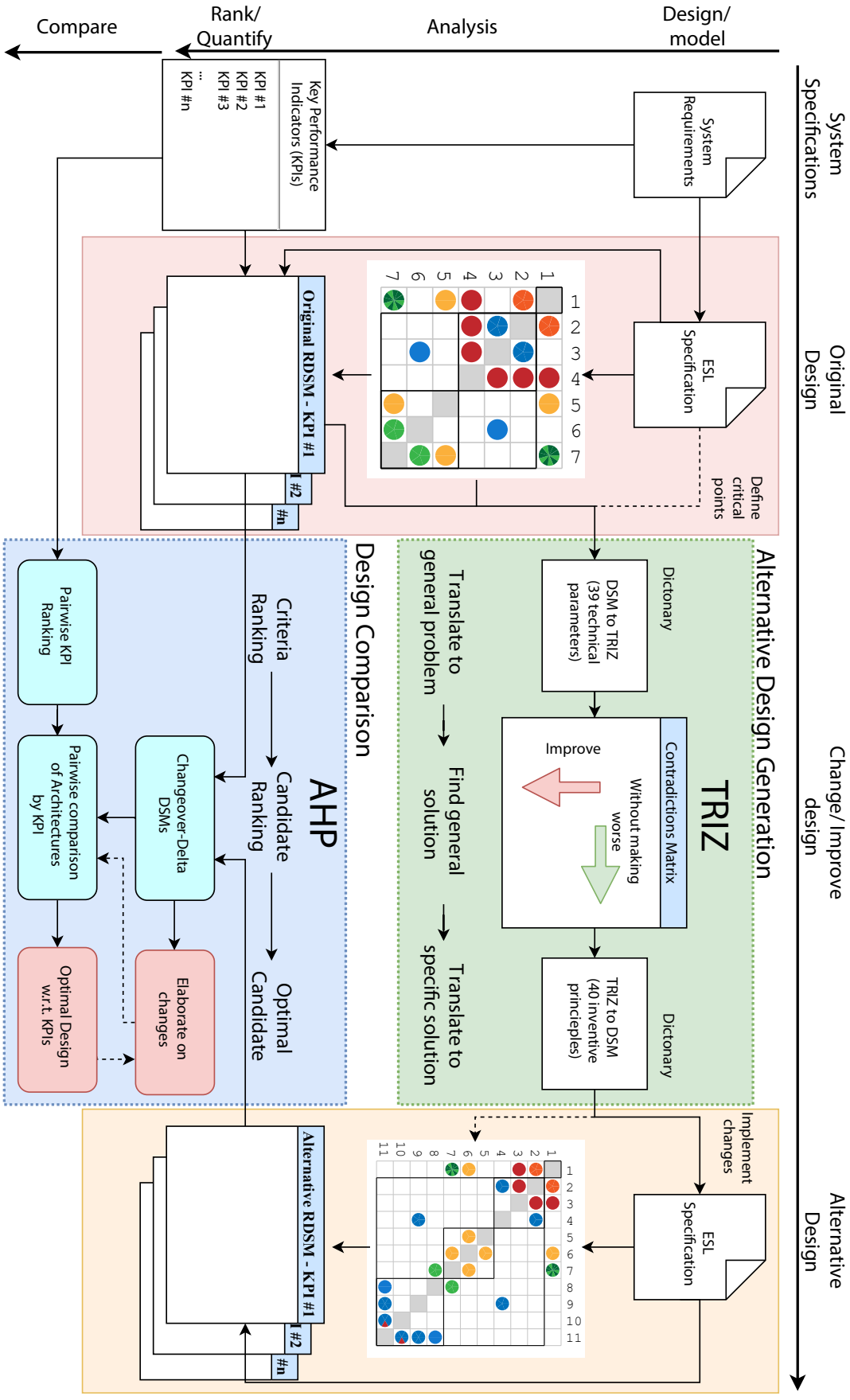


Figure 3.1: Method to generate design alternatives and system evaluation.

removing components in the decomposition tree and/ or changing the goal-requirements between components. This is especially useful when modelling the alternative system architectures.

A number of modelling decisions are made regarding the modelling of the system architecture. Due to different approaches, a system can be modelled in several ways. The most important decisions are:

- *Internal modelling is applied* - The internal processes of elements are known, internal processes can be modelled instead of simple input/output transformations.
- *Intermediate elements are taken into account* - A dependency between elements A and B passing through C, is modelled as two dependencies: $A \rightarrow C$ and $C \rightarrow B$.
- *Physically connected elements are modelled as mechanical (positional reaction) flows* - Two parts or assemblies bolted together, are modelled as mechanical positional flows, like in a free-body diagram.

The derived dependencies,from the ESL specification, are visualized in a DSM. This DSM is clustered using the flow-based Markov clustering algorithm of Wilschut [17]. As explained in Section 2.2, clustering combines elements which have a strong dependency on each other into groups. The clustering algorithm will first determine the bus elements. Hence, during clustering the bus size is determined first. Next, the non-bus clusters and granularity of clustering are determined. The clustered DSM is the basis for the projection of the system performance.

Part of modelling the system architecture, is the projection of the Key Performance Indicator (KPI) values creating ranking DSMs (RDSMs). The KPIs used in this project are shown in Table 3.1 and are divided into a set of fixed and non-fixed KPIs. The fixed KPIs are elementary system performance requirements, of which deteriorated performance results in an unsellable system. Hence, the fixed KPIs are not allowed to deteriorate.

Non-fixed KPIs related to the machine uptime. The overall balance of non-fixed KPIs must improve for the alternative system architecture to be an improvement. A detailed explanation of the KPIs is presented in Appendix A.1.

Key Performance Indicators (KPIs)	
Fixed:	Non-Fixed:
Positional Accuracy (wafer) Wafer Contamination Wafer Temperature Wafer Throughput	(Machine) Availability (Machine) Reliability

Table 3.1: Key Performance Indicators use within this project

Table 3.1 shows that fixed KPIs are product (wafer) related, non-fixed KPIs are related to machine performance (uptime). This difference allows for a different type of KPI projection. Not all elements in the system architecture have a direct (or even indirect) relation to one or more fixed KPIs. For example, a frame provides structural integrity, but does not influence any of the fixed KPIs. For non-fixed KPIs, a ranking can be given for each element within the DSM and all elements can be taken into account. For the fixed KPIs, only the relevant DSM elements are taken into account (even a frame can break). Hence, the type of ranking depends on the KPI and are presented in Table 3.2.

Key Performance Indicators (KPIs)	Projection (Ranking)
(Machine) Availability (Machine) Reliability	Availability Impact
Wafer Contamination	Allowable contribution
Wafer Temperature Wafer Throughput Positional Accuracy (wafer)	Allowable deviation w.r.t. (local) nominal value

Table 3.2: Ranking per KPI

Both non-fixed KPIs are combined in a single ranking. The availability impact is the true effect of both non-fixed KPIs. If a component has a long repair time but never fails, the effect of the component on the machines

availability impact is minimal or even negligible.

The availability impact calculation can be seen in Equation (3.1) and is a combination of the Mean Time To Repair (MTTR) and the fail-rate (λ) of a component or assembly. The result is the average amount of hours of downtime per year a component or assembly will cause. A high fail-rate or long repair times, will affect the availability impact in a negative way. The availability impact values include preventive maintenance times. Availability impact is therefore to total average downtime per year per machine.

A modification of Brady [9] is used to rank the non-fixed KPIs. Like [9], each element within the DSM gets a value assigned with respect to the availability impact of the element. For each component the availability impact is calculated as in Equation (3.1). If one or more dependencies exist between two components, both values are added together (compared to multiplication in [9]) as in Equation (3.2).

$$\text{Av-impact} = MTTR \times \lambda[hours/year] \tag{3.1}$$

$$\text{Entry value}(i,j) = \frac{sign(\#\text{ dependencies}(i,j)) \times (\text{Av-impact}(i) + \text{Av-impact}(j))}{\text{Total Av-impact}} \times 100\% \tag{3.2}$$

The entry values of (3.2) can be interpreted as the percentage of total downtime to be expected if a dependency between the elements fails (and both elements are affected). The entry values, give an indication of the reliability and fail-rate of the components, and the criticality of the dependencies between these components. A critical dependency can influence the fail-rate of the elements. Changes to these dependencies (finding alternatives), can change the components such that the reliability and failure rate improves.

Fixed KPIs give an indication regarding the system's performance with respect to the KPI. The indication of the system performance can support conclusions with respect to the non-fixed KPIs. If a dependency of a non-fixed KPI is identified as critical, the fixed KPIs can provide insight if the criticality is by said KPI.

A logarithmic scale is used to rank the fixed KPIs. The impact of the KPI related to the dependencies is highlighted. If the KPI is a summation (like contamination), there is a maximum value not to be exceeded. The used scale is the allowable (order of magnitude) of the dependencies contribution. Dependencies, allowed to have a small contribution (small order of magnitude), are assigned a low value (1). Larger contributions (higher order of magnitude), are assigned high values (9). Intermediate orders of magnitude are assigned to intermediate values (3).

The allowable deviation (expressed in a order of magnitude) to a local nominal value is used if the KPI has a target value which must be reached. For example; a high order allowable temperature deviation is assigned a value of 9. Low order allowable temperature deviation, is assigned a value of 1. Intermediate orders are assigned intermediate values (3).

In both cases, the KPI ranking is independent of the target value. Furthermore, a small order of magnitude means a small margin and is therefore harder to achieve. Dependencies that do not contribute to the KPI are left blank, but are marked as non-contributing.

The result of the KPI ranking, is a set of RDSMs. The set of RDSMs, rank the system in terms of the KPIs. These rankings can be used to find areas of interest. Critical areas are defined as groups of elements whereby both the non-fixed KPI rankings show critical values (high availability impact). The areas of interest can then be used to create design alternatives to improve the non-fixed KPIs. The fixed KPI rankings only establish a baseline, which is not allowed to deteriorate.

3.2 Generating Design Alternatives

For the purpose of generating alternative designs, TRIZ is selected. The main condition for the design changes, is that a set of KPIs is fixed in terms of performance and another set of non-fixed KPIs are free to change. The goal of the alternative design is therefore to improve the non-fixed KPIs without making the fixed KPIs worse.

The 40 inventive principles, part of TRIZ, aim to improve a selected (TRIZ) parameter without making other (TRIZ parameters) worse. The resulting inventive principles can then be applied to the existing system architecture to create an alternative system design.

As seen in Figure 3.1, the alternative generation consists of three steps:

1. Problem translation to TRIZ domain
2. Problem solving, by contradiction matrix
3. Solution translation to WH-domain

In order to use the 40 inventive principles, the problems must be specified in terms the 39 TRIZ parameters. The specified critical areas must be translated to the TRIZ domain. The KPIs, relevant to these areas, are defined in terms of (out of a possible 39) TRIZ parameters for which a first dictionary is used. In [26] and [27] definitions of the technical parameters are given.

By describing the KPIs in terms of the technical parameters, the TRIZ contradictions matrix can be used to find suitable inventive solutions. An example of TRIZ can be found in Section 2.3, the contradictions matrix is shown in Appendix A.2.

Inventive principles, provided by the contradiction matrix, need to be translated to specific solutions to the critical areas. A second dictionary (provided by [26] and [27]) describes the meaning and essence of the inventive principles. The set of specific solutions is checked for suitable options. These options are then selected to be implemented. The selection and implementation is the translation of the inventive principles to the WH-domain. The inventive principles are applied to the ESL specification. As a results, the dependencies of the alternative system architecture are generated. These can be visualized in a DSM of the alternative system architecture.

A second method for the generation of alternatives is presented in Appendix A. This method relates to the business cases alternatives instead of architectural changes. This report only focusses on architectural changes.

3.3 Evaluating Design Alternatives

To determine whether a design alternative is a design improvement, all generated designs have to be compared and evaluated. The number of generated alternative designs and used KPIs may vary. There are a number of methods capable of handling a variable number of criteria (KPIs) and candidate designs (alternatives), AHP is selected in this project. Other decision methods are available [35], but the ability to handle a variable number of designs and KPIs make that AHP is used. Within AHP, KPIs are treated as criteria and the system architectures as candidates.

In Figure 3.1, the steps of AHP are shown. The method has two sets of input. KPIs, needed for criteria ranking and the candidate designs (including the KPI mappings) for the design comparison. Criteria ranking is used to determine the importance of each KPI relatively to other KPIs. The importance of each KPI is needed during the final candidate selection, but also gives insight in the importance of the KPI themselves.

All AHP calculations are presented in Saalty [31] and also in Triantaphyllou [32], for the convenience the AHP calculations are also explained in Section A.4.

Figure 3.2 shows the pairwise comparison between system architectures as preformed in this project. Two DSMs, are pairwise compared by means of a Changeover-Delta DSMs. The original DSM (red) serves as a basis, by which the second DSM (green) is compared. Hence, the clustering of the red DSM can be seen in the overlapping part of the Changeover-Delta DSM.

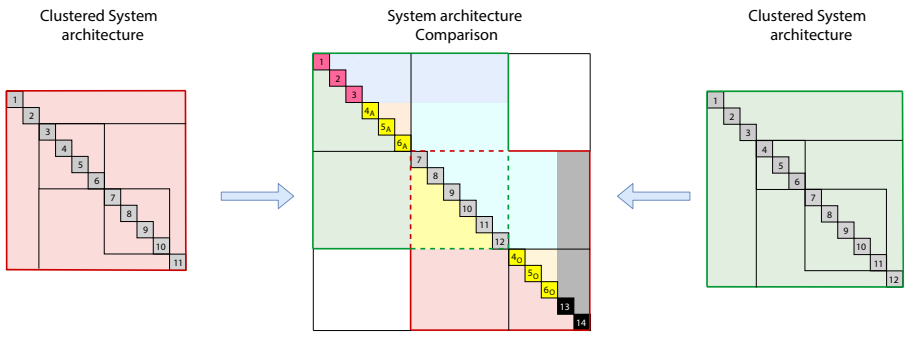


Figure 3.2: Pairwise system architecture comparison

To the purpose of pairwise candidate comparison, the Changeover-Delta DSM (CD-DSM) is created in this project. The CD-DSM (explained in Section 3.4) provides in a single overview, the differences, commodities and changes of dependencies and elements. The goal of the CD-DSM is to support the pairwise comparison between designs and motivate decisions.

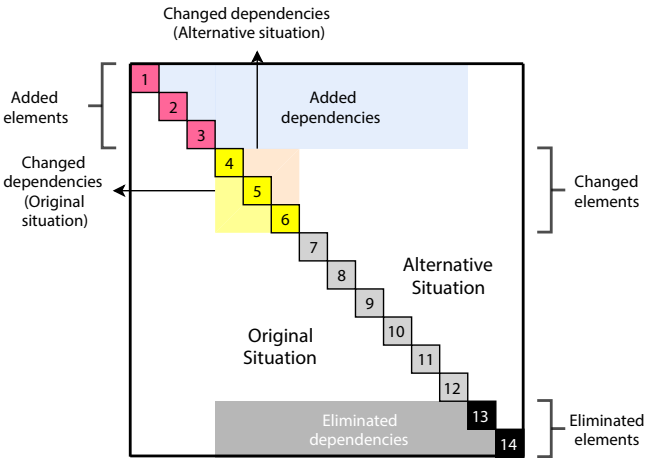
To determine if the KPIs (fixed and non-fixed) are improved or deteriorated, a CD-DSM an be created of the RDSMs, containing the ranking values of the KPIs. To this purpose the KPI mappings generated can be used.

For two candidates (the original design and one alternative), AHP reduces to direct comparison between the designs. This this case the AHP consistency-checks are not possible. For more then two candidates, the consistency has to be calculated in order to check if the pairwise comparisons of the candidates are done properly, the consistency calculations are presented in Section A.4.

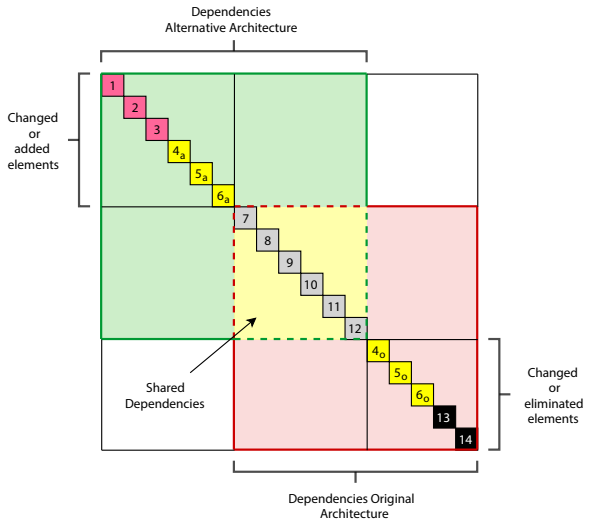
3.4 The Changeover-Delta DSM

The Changeover-Delta DSM (CD-DSM) is a combination of existing methods for the visualization of changes between two DSMs (or system architectures):

- *ΔDSM [36]* - shows the differences between two DSMs (see Figure 3.3a). Some notes regarding the Δ-DSM can be found in Section A.5.
- *Changeover DSM [S.D. Eppinger, personal conversation 27-11-2018]* - shows the transformation between elements and dependencies between two DSMs (see Figure 3.3b).



(a) Schematic layout of the ΔDSM



(b) Schematic layout of the Changeover DSM

Figure 3.3: System Architecture DSM Comparison Methods

The CD-DSM is created during this project and schematically shown in Figure 3.4. The CD-DSM is a combination between the Changeover DSM and a ΔDSM, whereby the symmetry of the DSM is used to combine both methods. The result is a single figure, where the transformation and differences between designs can be seen. The lower triangle shows the changeover between the designs. The upper triangle of the DSM (above the diagonal) shows the changed dependencies, based on a Δ-DSMs.

The elements displayed on the diagonal, highlight whether the element is added, removed or has changed dependencies compared to the original system architecture. A specific colour is given to the element for identification. These colour codes are (with corresponding elements):

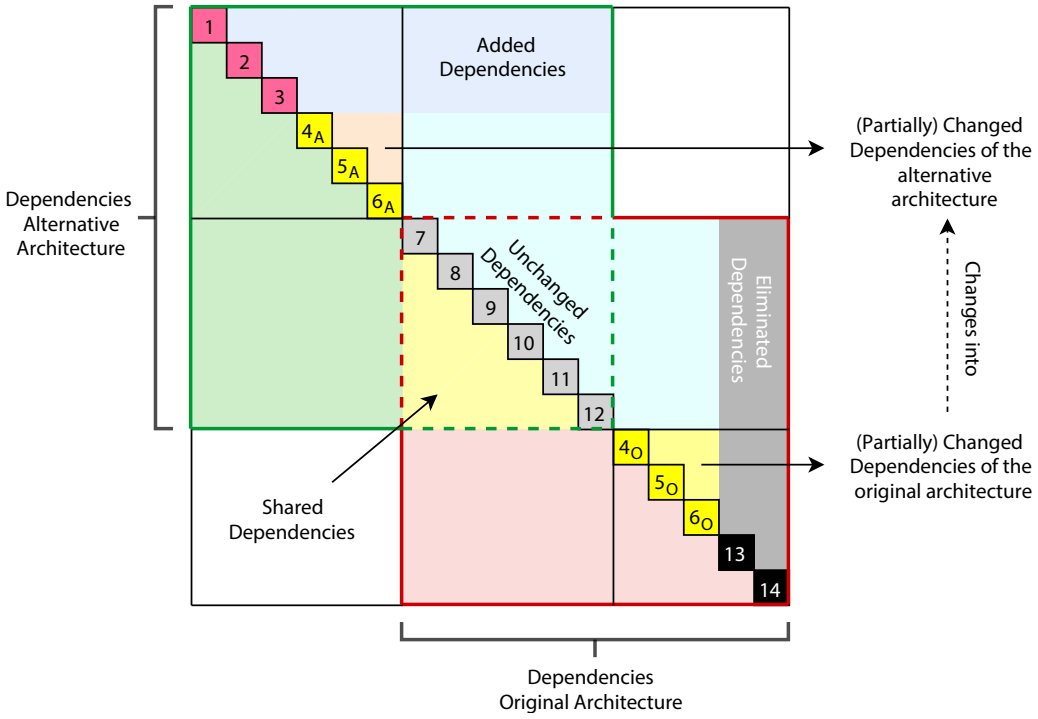


Figure 3.4: Schematic layout of the Changeover-Delta DSM

- *Pink* (1 till 3) - New elements with respect to the original architecture
- *Yellow* (4_A till 6_A) - Mutual elements with changed dependencies as in the alternative architecture
- *Grey* (7 till 12) - Mutual elements with no changes with respect to the original architecture
- *Yellow* (4_O till 6_O) - Mutual elements with changed dependencies as in the original architecture
- *Black* (13 and 14) - Removed elements with respect to the alternative architecture

Changed elements are defined as elements between which dependencies have changed, but none of the elements are added or removed in either design. This means that when a dependency changes between existing elements, both affected elements are marked as changed. Dependencies between added or removed elements, are not categorized as changed. In a system where only the network of dependencies change, the elements remain identical; no added or removed elements. To detect differences between both system architectures, changed elements are needed. For changed elements exist two situations, namely an old situation and a new situation. To display both situations, the elements are display twice in the CD-DSM, but only once for each system architecture.

It should be noted, that the CD-DSM contains a lot of information and to fully understand the CD-DSM may require effort. Yet, two system architectures can be evaluated in a single figure, regardless the difference in number of elements. Shared (or unchanged), added and removed dependencies and elements can be seen and changes of internal networks are included. Therefore, the CD-DSM is used within this project to visualize the differences between two system architectures.

Chapter 4

ESL Evaluation and Additions

In this chapter, the usage of ESL during this project is evaluated. ESL is a language (presented in Wilschut [20]). An ESL specification automatically derives dependencies between system components, functions, and (design) variables. This network of dependencies can be visualized in a DSM or a Multi Domain Matrix (MDM). ESL is still in development, in this report the version of ESL is used as presented in [20].

First, a review of the usage of ESL in this project is given. During this project additions to ESL are created. In section two of this chapter, these additions are presented. The description of the system architecture and meaning of the component abbreviations are explained in Chapter 5.

4.1 ESL Evaluation and Remarks

In this project, the NXE:3400C WH is used as a case study. Based on the design documents of the ASML NXE-series, a DSM of the NXE:3400C WH is created. The design documents used are:

- *Engineering Design Specification (EDS)* [37] and [38] - Documents describing the internal modules, interfaces and processes of the WH.
- *Engineering Performance Specification (EPS)* [39] - Document describing requirements of the WH. Includes external requirements between WH and environment (Waferstage (WS) and Track (TK)).

Within the design documents, natural language, figures and tables are used for the description of the NXE:3400C WH.

Tables provide mainly information regarding, sub-modules (of the treated module), quantities of internal flows and references to more detailed documents which are not used in this project. The natural language descriptions and figures from the documents are used to describe the NXE:3400C WH in ESL. From [38], the following natural language description of the Robot Vacuum Chamber (RVC) or Vacuum Robot Chamber (both names are used in [38]) is taken as example:

> "The function of this Vacuum Robot Chamber assy is to support mainly the Robots (SLR and SUR), VPA, LL and to provide a thermal station in vacuum. The VRC is supported by the WH Frame and includes several tool and facility interfaces."

Based on this natural language, ESL functional requirements regarding physical connections are formulated. These functional requirements can be seen in Listing 4.1.

```
goal-requirements
  gfrVWH-f1 : RVC must absorb LL-pos-force from LL
  gfrVWH-f2 : RVC must absorb PS-pos-force from PS
  gfrVWH-f3 : RVC must absorb SLR-pos-force from SLR
  gfrVWH-f4 : RVC must absorb SUR-pos-force from SUR
  gfrVWH-f5 : RVC must absorb VPA-pos-force from VPA
  gfrVWH-f6 : vWH-fr must absorb RVC-pos-force from RVC
```

Listing 4.1: goal-requirements as derived from [38]

Due the confidentiality of the EDS files [37] and [38] and the EPS [39], no figures are copied to illustrate the figures within these documents. However, Sander [40] presents a similar and non-confidential figure. Figure 4.1

presents the wafer flows as in the older generation (non-vacuum) Waferhandlers. From similar figures are among others, the cooling water paths, gas supplies and wafer flows derived and translated into ESL components, goal requirements and transformation-requirements.

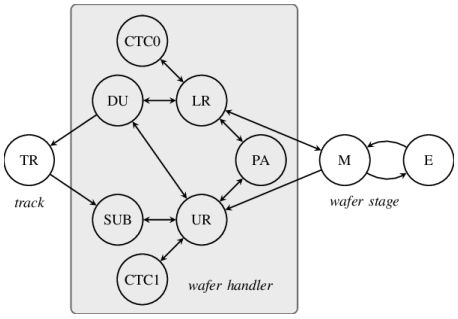


Figure 4.1: Example schematic of wafer routing (aWH only), from [40]

For the creating of the ESL specification, the following conclusions regarding the use of ESL can be made:

1. Defining the containment of a vacuum (in a vacuum chamber) is not straightforward, since a vacuum is not a flow. This is solved by stating that a gas-mixture is to be removed instead of a vacuum to be contained.
2. Apart from 1), the functional basis function set of Hirtz [21] provides sufficient verbs to describe the Waferhandler in ESL.
3. Apart from 1), the functional basis flow set of Tilstra [22] provides sufficient flow types to describe Waferhandler processes in ESL.
4. Goal requirements can have inverted flow directions related to the definition order. For this an ESL improvement is suggested in the following section.

4.2 Proposed language extensions

4.2.1 Flow directions

ESL flows are directional, a flow is defined from an active component (the sender) to a passive component (the receiver). Flow definitions are stated by goal-function requirements or goal-function constraints.

The direction of the flow can be opposite to the definition used within ESL. If the action, described by the flow, is preformed by the passive component. In Listing 4.2, the robot is the performing component, delivering and collecting a wafer at the pre-aligner. The robot is the active component and provides a wafer to the pre-aligner in the first goal requirement. In gfr-2, the pre-aligner is the active component and the robot the passive one. The (material) flow of the wafer is from the robot, to the pre-aligner and vise versa in the second goal requirement. In both case the robot is the performing component, delivering and collecting a wafer at the pre-aligner.

```
1 # goal-requirements
2 gfr-1: Robot must provide wafer-x to Pre-aligner
3 gfr-2: Pre-aligner must provide wafer-y to Robot
4 gfr-3: Robot must collect wafer-y from Pre-aligner
5
6 # transformation-requirements
7 tfr-4: must convert flow-f into signal-s
8 tfr-5: must generate signal-s from flow-f
```

Listing 4.2: Example of goal and transformation requirements

In gfr-3 of Listing 4.2, a more natural way is used to describe the same flow as gfr-2. The robot is the active component, but the flow is from the Pre-aligner to the robot, so opposite of direction. In Listing 4.2, the robot is the performing component, specifying so within ESL is a more natural way of describing the system. Using this more natural way, all flows still fulfil the functional basis proposed by Tilstra [22] and Hirtz [21].

Prepositions are used to define the direction. As can be seen in Listing 4.2 (and 4.1), the preposition `from` is used to imply a inverted flow. This modification is implemented into the ESL compiler for this project.

The same holds for transformation functions, as seen in Listing 4.2. Technically, a sensor generates a signal (based on a measured flow), instead of converting a flow into a signal. Transformation function (tfr-5), is a more natural description of the process. Within ESL, the description of a transformation is always from an old to a new flow. Yet, using opposed directions can describe the system in a more natural way.

4.2.2 KPI declarations

The functionality of ESL is sufficient for creating MDMs. As presented in Wilschut [16] and Brady [9], weight projections to the generated DSMs can be created using respectively the Pagerank algorithm [41] or Brady's method. To combine ESL with methods of ranking requires extension to ESL.

For ranking, there is chosen to define the values within the ESL specification file. This way only a single specification file needs to be specified. In later stages, automatic derivation of the KPI-values based on the ESL goal and transformation requirement subclauses can be done.

Listing 4.3 shows a component definition and a goal requirement definition with specified KPI values. Depending on the method ofnranking, KPI-values have to be assigned to goal-requirements and/or components.

```
# variable definition
define variables
const = 1.5
compvar = 1
gfrvar = 2

components
  # component definition(s)
  TK is a track with arguments
     wafer-TK-in, wafer-TK-out, ...
     ASP-ssig, ARR-ssig, ...
     TRR-ssig, TSA-ssig, ...
     tk-wh-alignment
     Comp_value = [ 0.01 const compvar ]

goal-requirements
  # goal requirement with respect to wafer transport
  gfrW-w0: TK must deliver wafer-TK-in to WH with subclauses
     c-0: wafer-temp must be at least 21 Celsius
     c-1: wafer-temp must be at most 23 Celsius
     c-2: exchange-time must be at most 7 seconds
     end
     KPI = [ 9 const gfrvar ]
```

Listing 4.3: component and goal requirement declarations with KPI variables

In both values declarations, the first values is a number assigned to the component and goal requirements. The second values, are constant values defined at the beginning of the ESL specification. The third values, are pre-defined values as well, but can be assigned to a (sub)class of components or goal requirements. The number of KPI values to be listed is unlimited.

4.2.3 subfiles

A third addition, related to the compiler, but specified in the ESL specification is the inclusion of subfiles. The definition used to include subfiles is presented in Listing 4.4.

```
\include{ filename.esl } # reference to file
\include{ .path/filename.esl } # reference to file in other directory
```

Listing 4.4: subfile inclusions statement

Subfiles are implemented to create smaller (sub)specifications to work with. When adding layers of decomposition, the ESL definition of the parent component needs to be adjusted and the child components need to be defined. Scrolling between both sections may occur frequently during programming. When using a single file, the scrolling can be time consuming. By using multiple files, both sections can be opened and edited simultaneously. Additionally, usage by multiple users (working on different sections) is possible without conflicts.

4.3 ESL Functionality Wish-list

Some functionalities are not included, neither added to the ESL compiler. A short list of suggested additions to ESL compiler is presented:

- *Dynamic behaviour* - Not all dependencies might be active at each time interval. The addition of time interval, can provide insight at which dependencies are active at which time interval. When allocating resource budgets, this feature can help to determine the budget needed for the system to operate.
- *Derivation of KPI values from subclauses* - ESL subclauses are used to add information regarding a specific goal or transformation requirement. Instead of defining the KPI values separately, the added information can be used to (automatically) derived KPI values. A suggestion for the automated derivation of KPI values is given in Section B.3 of the appendix.

Chapter 5

Current System Evaluation

The current system architecture of the Waferhandler (WH) is presented and analysed in this chapter. A DSM is created of the system architecture using ESL. This DSM is used for analysis of the current system architecture, using the methods presented in Chapter 3.

The WH consists of two main modules, the atmospheric WH (aWH) and vacuum WH (vWH). Whereby the vacuum of the vWH is maintained by the Robot Vacuum Chamber (RVC). The LoadLock (LL),part of the vWH, is the transition from the aWH to the vWH. Figure 5.1 depicts a schematic overview of the internal WH wafer-flows. The Track (TK) supplies a wafer to the aWH at the discharge unit (DU). The Atmospheric Load Robot (ALR) transports the wafer from the DU to the Atmospheric Pre-Aligner (APA). At the APA, the wafer is thermally conditioned and aligned to match the WS orientation. Next, the ALR loads the wafer into the load bay of the LL. The LL creates a vacuum and the wafer is unloaded by the Stage Unload Robot (SUR), which transports the wafer to the Vacuum Pre-Aligner (VPA). The VPA reconditions and realigns the wafer as compensation for the atmospheric-vacuum transition. The Stage Load Robot (SLR) collects the wafer and holds it in the Park Station (PS), until the Waferstage (WS) is ready to receive it. After lithography, the SUR places the wafer in the unload bay of the LL where the wafer is brought back to atmospheric conditions. The Atmospheric Unload Robot (AUR) transports the wafer to the DU where any static charge is removed. Finally, the TK then collects the wafer from the DU for further processing.

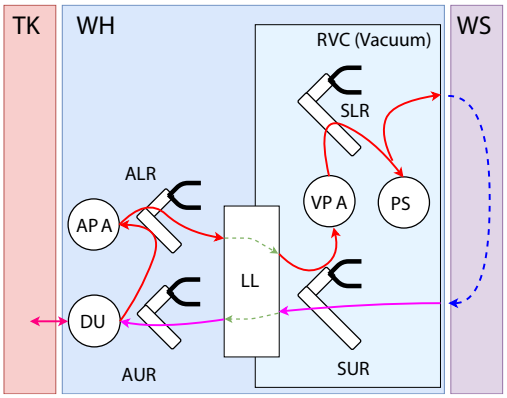


Figure 5.1: Schematic overview of the ingoing (red) and outgoing (purple) wafer flows of the WH

Figure 5.1 shows that the ingoing and outgoing wafer paths have common elements. Both the SUR and LL are used in both the in- and output path. In the aWH these paths are separated and only two moves of the wafer are needed to pass through the input-path of the aWH. The input path of the vWH requires three moves to get the wafer from the LL to the WS. This is caused by the shared elements of the in- and output paths. The SLR needs to remove a wafer from the VPA before the SUR can place a new wafer. The SUR places a wafer on the VPA before removing a wafer from the WS. As a result, the SLR needs to wait for the SUR before placing a wafer on the WS. The PS is used for thermal conditioning, while the SLR waits for the SUR to finish it tasks.

To support the thermal conditioning, alignment, vacuum and transport of the wafers, a number of other processes are taking place (not shown in Figure 5.1). Among these processes are: power feeds, 9 cooling water circuits, multiple gaseous flows (like N_2 and H_2) and control signals. A detailed system description of these circuits can

be found in [37] and [38].

The outline of this chapter is a as follows: section one explains how the WH system architecture is modelled based on the documents provided. Additionally, the resulting dependency structure matrix is shown, analysed, and discussed. Section two, presents the Ranked DSMs with respect to the KPIs of the current system architecture. Critical area's with respect to the KPIs are defined and discussed. Finally, conclusions are drawn regarding the current system architecture.

5.1 Architecture Modelling and Analysis

Three levels of decomposition are used in this report. In Table 5.1, all elements defined in the first, second and third level of decomposition are presented.

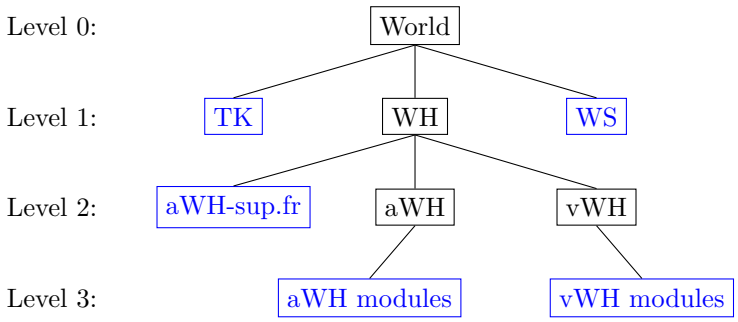


Figure 5.2: Decomposition tree of the original system architecture

Figure 5.2, shows the decomposition tree of the original system architecture. Non-decomposed elements (like the TK), continue to exist in the lower levels of decomposition. The figure shows the decomposition tree of the WH up to the third level of decomposition, whereby all elements present in the third level are depicted blue. For overview, the aWH and vWH modules are presented as one.

The base of decomposition is, by definition, the world (level 0) and can be seen as a litho-cluster. The world is a single element and acts as parent element to all elements in the decomposition tree. The environment of the WH consists of the TK and WS.

To model the paths of the wafer, power, cooling water, gas and control flows, the environment of the WH is modelled. The TK provides wafers to the WH and takes them away after lithography. The WS performs the lithography and, from a WH point of view, connects the ingoing wafer path to the outgoing wafer path (as seen in Figure 5.1). The WS is also the source and drain of all liquid and gaseous flows used by the WH. Finally, all control regarding system integration is preformed by the WS.

The second and third level of decomposition are related to the modules within the WH. On the second level, both main modules are defined: the aWH and the vWH. The aWH is placed on a support frame (aWH-sup-fr) to match the height of the vWH. The support frame defined on the second level of decomposition to illustrate the situation at that level. The aWH is a copy of the ASML NXT WH and is placed on this support frame. The third level of decomposition defines the modules within the aWH and vWH.

Mainly, the third layer of decomposition will be used. The second level of decomposition has an insufficient granularity to model the actions preformed with/on the wafer. A fourth level of decomposition yields too much detail due to the large number of components. The third level of decomposition has sufficient granularity, while not presenting too much detail. Figure 5.1 shows the wafer-flows and flow-related components in the third level of decomposition, with a sufficient amount of depth to follow the internal processes.

In Figure 5.3, of the third level of decomposition component DSM, a hierarchical clustering is displayed. The WH is decomposed into the bus-cluster, the aWH cluster and the vWH cluster. Each of those individual clusters is composed of several smaller clusters.

Between the aWH and vWH clusters are only two material flow dependencies (wafer exchanges). Taking the bus-elements into account, the aWH and vWH are mechanically dependant (fixed) to one other. Excluding the

Level 1	Level 2	Level 3
Track (TK)		
Wafer Handler (WH)	aWH-support-frame	
	Atmospheric WH (aWH)	Atmospheric Load Robot (ALR)
		Atmospheric Unload Robot (AUR)
		Atmospheric Pre-Aligner (APA)
		aWH Electronics Cabinet (Ecab-NXT)
		Charier handler / FOUP (FOUP)
		Discharge Unit (DU)
		aWH-frame (aWH-fr)
		LCW & CA conditioning system (LCW-cond.)
		Pneumatic unit (PU)
		Safety Handling Box (SHB)
	Vacuum WH (vWH)	Stage Load Robot (SLR)
		Stage Unload Robot (SUR)
		Vacuum Pre-Aligner (VPA)
		Robot Vacuum Chamber (RVC)
		Load Locks (LL)
		vWH Electronics Cabinet (Ecab-NXE)
		Robot Gravity Control (RGC)
		Gas & Fluids Distribution Box (GFDB)
		SLR-Differential Seal Control Box (SLR-DSCB)
		SUR-Differential Seal Control Box (SUR-DSCB)
		Back Gas Delivery System (BGDS)
		Park Station (PS)
		vWH-frame (vWH-fr)
Wafer Stage (WS)		

Table 5.1: Decomposition of the system up to Level 3

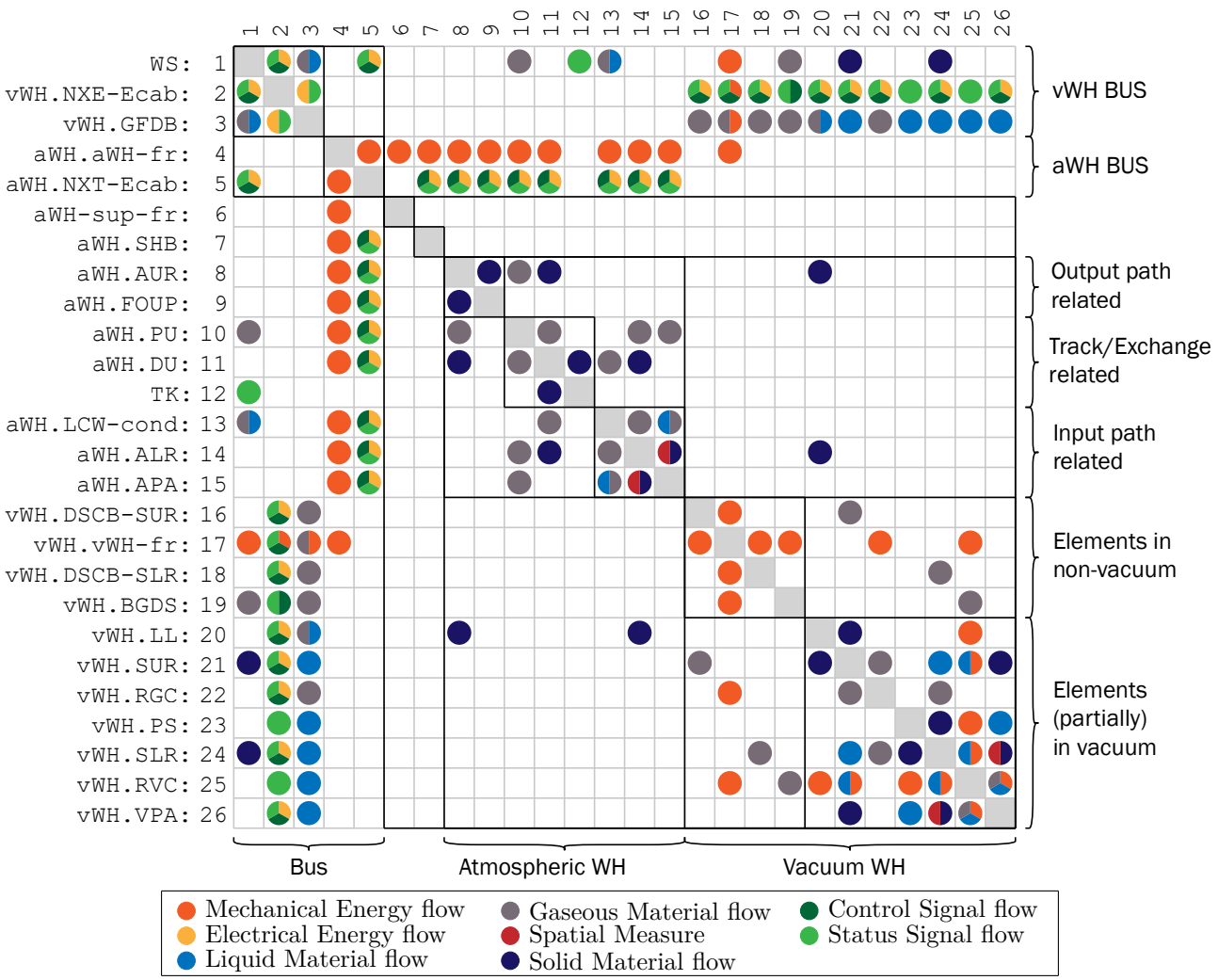


Figure 5.3: Component DSM, 3rd level of decomposition

wafer-flows and mechanical connection, the aWH and vWH are independent of each other. All dependencies regarding (liquid, gas and power) supplies and control-signals relate to the wafer scanner.

The Bus, aWH and vWH clusters are subdivided in a number of smaller clusters. The bus cluster is divided in an aWH and vWH part. Due to larger number of dependencies with the vWH, the WS is placed in the vWH bus.

In the aWH cluster, the smaller clusters reflect the internal wafer flow. Elements related to the wafer-track exchange are centred around the DU. After the track exchange, the wafer is transported by the ALR and processed by the APA. These are combined in the input-path cluster. The output path cluster, contains elements (only) used in the output path. The AUR handles the wafer only after lithography or (by exception) when the FOUP (not shown in Figure 5.1) is used. The FOUP is a removable wafer storage container and can be used for (small batch) in- and output.

The vWH cluster shows no clear wafer paths. Instead, all components related to the vacuum of the vWH are combined. A second cluster contains components positioned outside the vWH vacuum and connected by mechanical dependencies. The LL and SUR are used in both the vWH in- and output wafer paths, this is reflected in the clustering. No clear distinction in can be seen between the in- and output wafer paths.

5.2 KPI Ranking and Analysis

The KPI ranking is done twofold, based on the non-fixed KPIs and on the fixed KPIs (both presented in Chapter 3). Fixed KPI mappings will provide a baseline, KPI values are not allowed to deteriorate beneath this baseline. Non-fixed KPIs are used to define critical areas within the system architecture.

Availability and reliability are expressed in fail-rate (λ) and Mean Time To Repair (MTTR), as presented in Chapter 3. The MTTR and λ values used, are based on ASML reliability predictions for the NXE:3400C WH. For each component, the availability impact is calculated using Equation (3.1) and (3.2). Values for the aWH are not specified in the NXE:3400C WH reliability predictions. Values of the NXT:2000 WH (atmospheric only) are taken instead. The NXT:2000 WH is identical in performance and configuration to NXE:3400C aWH. Figure 5.3 shows that the aWH and vWH are independent of each other in therms of (liquid, gas and power) supplies and control, therefore it is assumed that the reliability predictions of the aWH and vWH modules are independent as well.

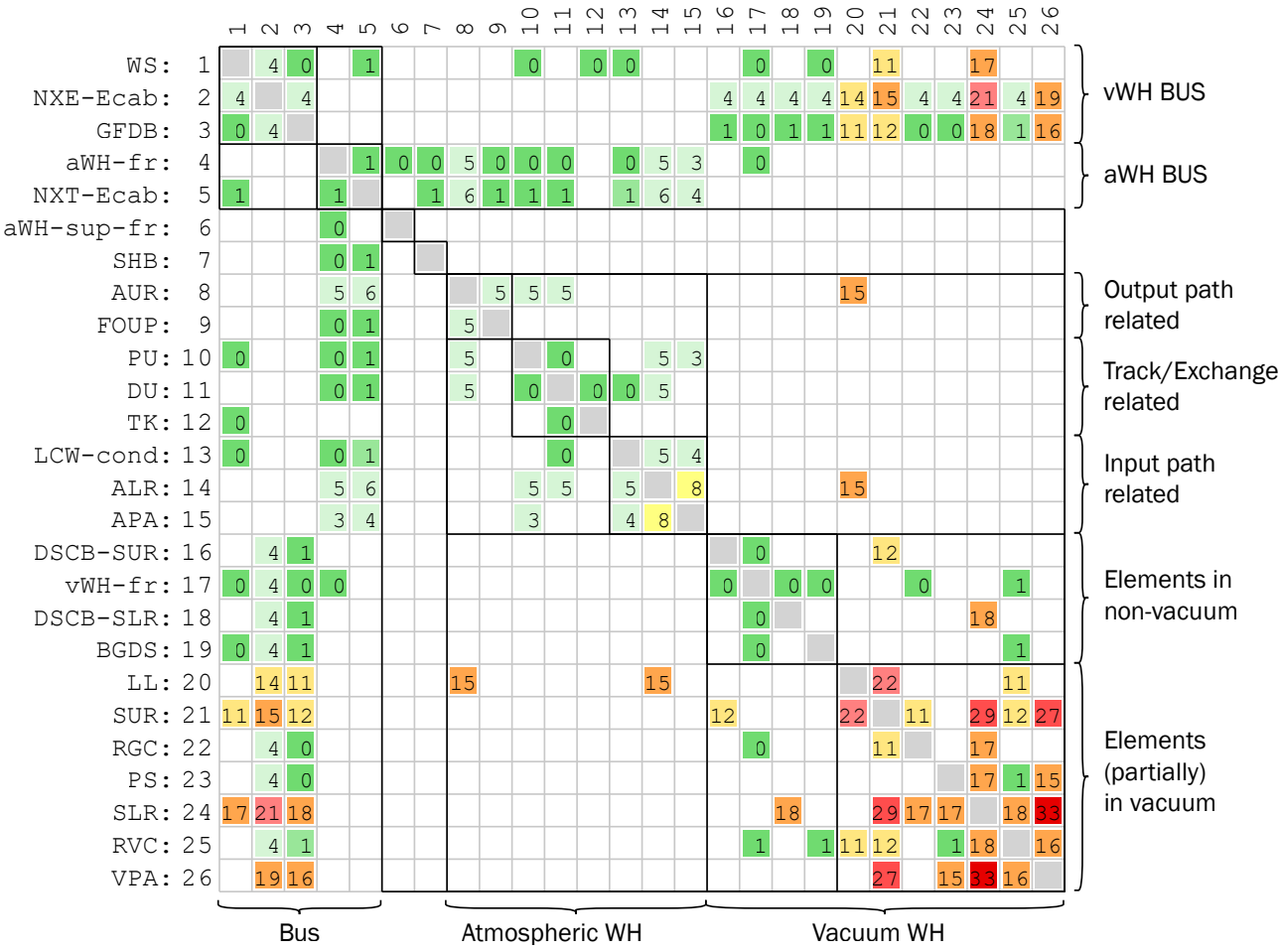


Figure 5.4: Availability Impact DSM, 3rd level of decomposition

Figure 5.4 displays the availability impact of the dependencies for the third level of decomposition. The clustering of the RDSM of Figure 5.4 is identical to Figure 5.3. The clusters, therefore have the same meaning as in Figure 5.3.

The dependencies show the criticality of the combination of elements, with respect to availability impact. The value of the entries show the availability impact, from the combination of elements. If two unreliable elements have a dependency, a failure of the dependency (or one of the elements) can have significant consequences on both elements. The total availability impact of the current system architecture is 79.3 hours per machine per year.

A critical area can be seen in the vWH cluster regarding elements in (partial) vacuum. Both the SLR, SUR, VPA and LL can be marked as critical with respect to availability impact. In the aWH, the APA and ALR combination can be seen as a local point of attention, but do not stand in contrast to the vWH critical points. Any improvements to the availability impact of the vWH have the preference over aWH improvements.

For each of the fixed KPIs a RDSM is created. These RDSMs are different to the RDSM of the non-fixed KPIs. Not all components of the WH have influence on the Fixed KPIs, therefore only the dependencies that are of influence (direct or indirect) are shown. Direct dependencies are shown in squares, indirect dependencies are shown in circles. For the (fixed) KPIs, the following is mapped to the DSM:

- *Throughput* - Allowable run-out time in relation to the performed process and the available time (expressed in a order of magnitude).
- *Temperature* - Allowable deviation (expressed in a order of magnitude) to a local nominal temperature.
- *Alignment* - Allowable deviation (expressed in a order of magnitude) to a local target value.
- *Contamination* - Allowable contamination (expressed in a order of magnitude).

In Figure 5.5, the RDSM for the throughput can be seen. It can be seen that not all relations are of influence on the throughput, neither are there any dependencies of indirect influence. Figures of the other fixed KPI can be found in Appendix C. These figures are only used to establish a baseline of the fixed KPIs, not for analysis. The generated alternative are not allowed to go below the baseline.

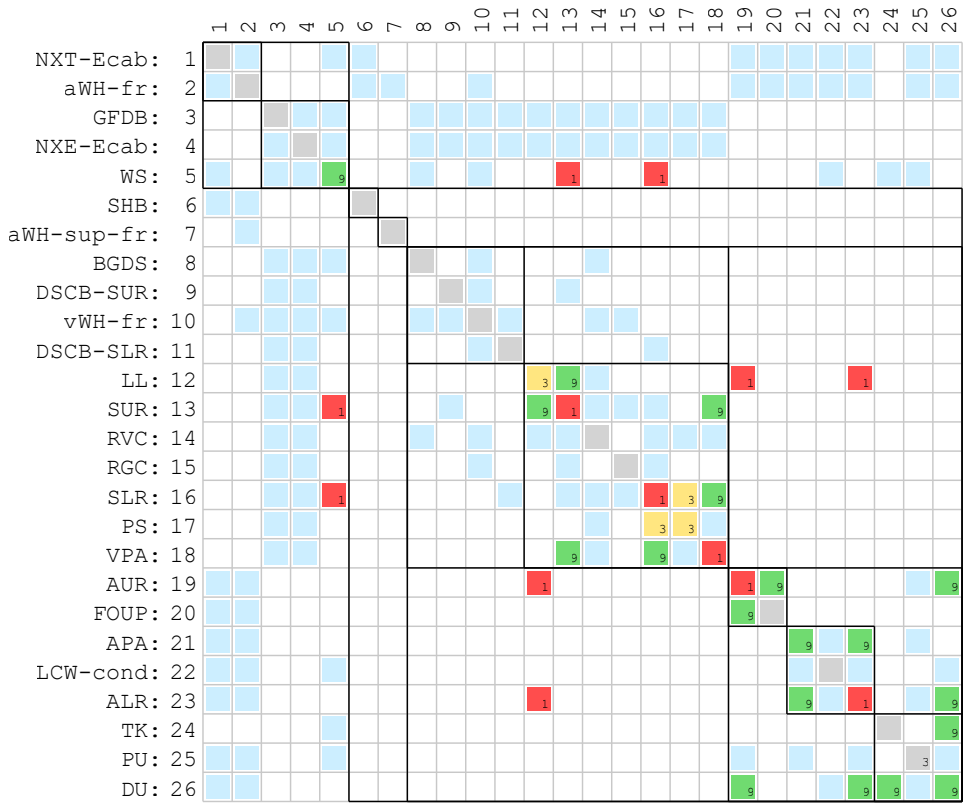


Figure 5.5: Throughput RDSM, 3rd level of decomposition

5.3 Conclusions

This chapter presents and analyses the generated DSM of the current system architecture. In addition, KPIs are mapped onto the DSM. Based on the non-fixed KPIs critical areas are defined. These areas are used as inuts for the generation of the alternative system architecture.

In the current system architecture, a clear distinction between the aWH and vWH can be seen (Figure 5.3). A

generated alternative must take the few existing dependencies between aWH and vWH into account, but can be altered otherwise. Internal changes in the aWH or vWH can be done without influencing the other.

Figure 5.4 shows that dependencies related to the vWH are marked as critical. Particularly, the dependencies between SLR, SUR, VPA and the LL. Failure of dependencies between these elements, or the elements themselves, can have significant impact on the (unscheduled) downtime of the system.

Dependencies related to the ALR and AUR have above-average criticality. Within the aWH, these dependencies can be seen as the (local) critical area's. Failure in these dependencies can cause intermediate downtime.

It makes sense that the vWH components and their relations are more critical regarding downtime, since vacuum has to be restored after repairs. Time to restore vacuum in the both the WS and vWH has significant impact on the downtime and therefore the availability impact.

Alternative designs can be created twofold. First is to change the system architecture such that the lifespan increases (decrease of the λ). Without changing the components, this can be achieved by reducing the number of cycles (if applicable). The second way, is change the design such that replacements can be easier performed.

The focus point of the system architecture design changes should be on the vWH. Specifically, the SLR, SUR, VPA and the LL should be examined, all related to the path of the wafer in the vWH.

Next chapter, the defined critical area and the analysis of the current system architecture, are used as import for TRIZ to generate design alternatives.

Chapter 6

Alternative System Evaluation

This chapter presents the generated alternative WH system architecture. The alternative system architecture is based on the original WH system architecture. In the first section, the alternative WH system architecture is generated using the 40 inventive principles of TRIZ. The changes to the system specifications and elements are presented and motivated. Section two, presents the component DSM of the generated alternative. To this DSM the availability impact and fixed KPIs are mapped. Finally, the critical areas within the alternative WH system architecture are determined.

6.1 Generation of Design Alternative

TRIZ is used for the generation of the design of alternative system architectures. Critical areas of the current system architecture are to be improved with respect to the non-fixed KPIs. The fixed KPI are not allowed to deteriorate. Both sets of KPIs are translated to the TRIZ domain in terms of the 39 technical parameters. The list of KPIs and the technical parameters in the TRIZ domain (from [27]) are:

- Mean Time Before Failure (MTBF)
  - *Reliability*: A system's ability to perform its intended functions in predictable ways and conditions.
- Mean Time to Repair (MTTR)
  - *Ease of Repair*: Quality characteristics such as convenience, comfort, simplicity and time to repair faults, failures or defects in a system.
  - *Duration of actions of a stationary / moving object*: The time that the object can perform the action. Service life.
- Wafer Throughput
  - *Speed*: The velocity of an object; the rate of a process or action in time
  - *Productivity*: The number of functions or operations performed by a system per unit time.
- Wafer Temperature
  - *Object affected / generated harmful factors*: A harmful effect that that reduces the efficiency or quality of the functioning of the object or system.
- Wafer Alignment
  - *Measuring Accuracy*: Closeness of the measured value to the actual value of a property of a system.
  - *Object affected / generated harmful factors*: (See above)
- Wafer Contamination
  - *Stability of the objects decomposition*: Wear, chemical decomposition and disassembly are all decreased in stability.
  - *Object affected / generated harmful factors*: (See above)

The contradiction matrix is used to solve the contradictions of improving the non-fixed KPIs while not affecting the performance of the fixed KPIs. The contradiction matrix is presented in Appendix A. The parameters to improve are listed vertically, the parameters not to get worse are listed horizontally. The matrix, which act as a lookup table, suggests a maximum of four inventive principles to solve the contradiction. All combinations between the parameters to be improved and those not to get worse are checked. The four most suggested inventive principles are selected. These inventive principles need to be translated back to the WH domain. The selected principles, with translation from [27], are:

- *1. Segmentation* - Divide into independent parts or segment even further.

- *2. Taking Out* - Take out a problem, or get only what you need.
- *11. Cushion in advance* - Prepare emergency means beforehand to compensate for low reliability.
- *35. Parameter change* - Change temperature, volume, pressure or degree of flexibility.

Ease of repair and reliability are selected to be improved. Thus the purpose of the change is improvement of these parameters. A list of all applied changes can be found in Appendix D.

Segmentation is applied first, whereby the LL and RVC are split. An input path and an output path is created. The input path contains LL1 and RVC1, the output path LL2 and RVC2 as seen in Figure 6.1. The idea of the segmentation is that smaller parts of the vWH have to be removed during repairs. The ease of repair is improved, reducing repair times.

By adding gate valves, with the with the aim of disconnecting the vacuum chambers off the WS, there is a improvement in start-up time. Only the repaired RVC has to be brought back to vacuum conditions instead of the entire WS. Segmentation creates smaller sections and the problem of having to bring the WS to vacuum conditions is taken out (principle 2). This further improves restart times after repairs.

The applied changes can also be seen as cushioning in advance. Repairs are made easier to compensate for the lower reliability. The splitting of the RVC can be seen as change in volume and degree of flexibility. Both RVCs have (each) a smaller volume (parameter change) than the original RVC and can be easier removed for repairs. A smaller volume means, a smaller vacuum to be created and maintained.

Changes to the (strict) temperature regime and vacuum conditions in the vWH, make that parameter changes in these variables are undesirable. Therefore no further parameter changes are applied.

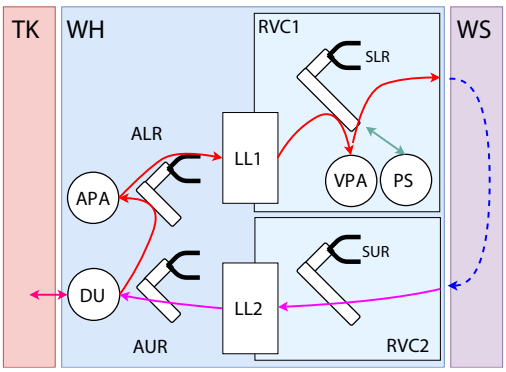


Figure 6.1: Schematic overview of the changed wafer flow

The changed wafer flow (shown in Figure 6.1), has a positive effect on the SUR. In the original wafer flow (Figure 5.1), each wafer is handled by the SUR twice. Splitting the LL, allows the SLR to unload LL of the ingoing wafer flow. Each wafer is still handled by the SLR twice, but the SUR only handles each wafer once. The SUR has a reduction in cycles, which can lead to a longer lifespan compared to the original design.

The PS is not used for thermal wafer conditioning, the SLR directly loads the wafer on the WS. The PS is not removed, it is still used for thermal conditioning of the SLR wafer-clamps.

The SLR now unloads and loads the LL and VPA respectively, causing the wafer to spend less time on the VPA. The NXE:3400C has a rate of 170 wafers per hour (WPH) or 21.7 seconds. A robot move takes about 3 seconds [38] and the minimal VPA time is 8.5 seconds (based on discussions with VDL ETG domain experts) [*R. Meulenbroeks, personal conversation 19-12-2018*]. This means the wafer rate of 170WHP is not affected. Additionally, this improves thermal conditioning of the wafer [*R. Meulenbroeks*].

Alignment is unaffected by the applied changes. The number of wafer transports before the wafer is loaded onto the WS is equal to the original design. Furthermore, there is still one wafer transport after the VPA (the final alignment). In the original design, the wafer is not released (from clamping) during thermal conditioning in the PS. No gain, neither loss in alignment is achieved.

Contamination might slightly improve for the generated design, but the measure is unknown. Due to segmentation, particles for RVC1 as less likely to reach RVC2. If so, the particles have to travel via the vacuum chamber of the WS. Any particles created by the components in RVC1 are less likely to reach RVC2, and the other way

round. Static charge, created by the wafer (electro magnetic) clamps might reduce too. Thermal conditioning in the PS is removed. As a result the clamping time is reduced, which can lead to a lower static charge. A lower static charge, attracts less particles.

6.2 Architecture Modelling and Analysis

The design changes are implemented into the ESL specification requirements. From the specification, a DSM is generated for each level of decomposition. The DSM for the third level of decomposition can be seen in Figure 6.2. The DSMs for the first and second level of decomposition are identical to the original system architecture DSMs (presented in Appendix C).

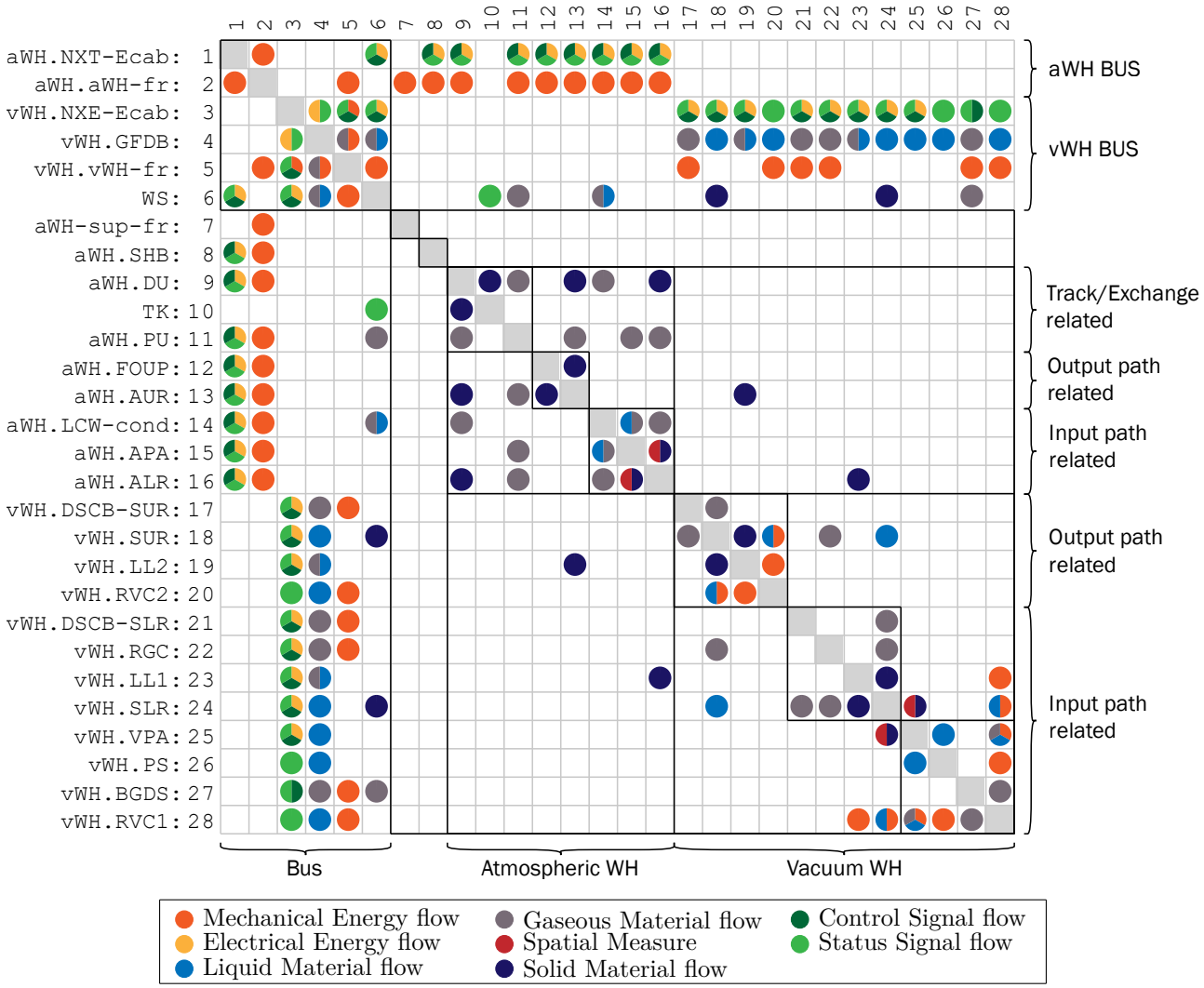


Figure 6.2: Component DSM, 3rd level of decomposition

In the DSM of Figure 6.2, a hierarchical clustering can be seen. A clear distinction between the aWH and vWH is seen on the first level of clustering depth (highlighted below the DSM). On the right of the DSM, the second level of clustering depth is shown. For both the aWH and vWH, clusters containing elements of the input path and output path can be detected. Furthermore, the clustering of the aWH is identical to the original system architecture.

The clustering of the vWH is changed compared to the original system architecture. The load path of the vWH is divided over two clusters. The clustering algorithm does not provide a clustering where both clusters are

combined. Since clustering is a way to analyse the system and not the solution of the analysis, both clusters of the vWH input path are combined.

6.3 KPI Ranking and Analysis

Similar to the current system architecture, the availability impact is mapped to the component DSM. The availability impact of two elements are projected on the positions where dependencies exist between these elements. The result can be seen in Figure 6.3. The off-diagonal positions display the downtime (SD and USD) of the combination of elements per machine per year.

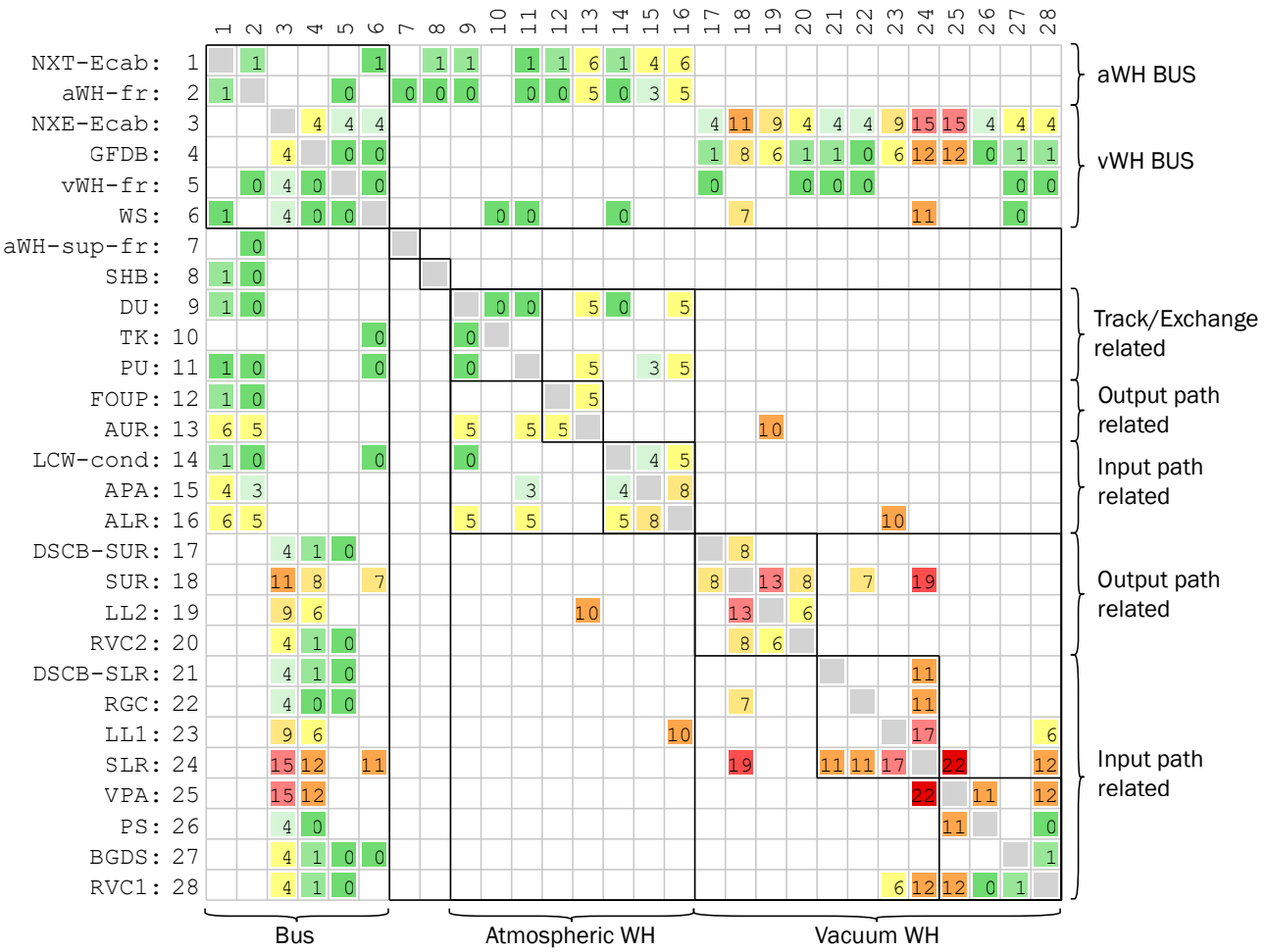


Figure 6.3: Availability Impact DSM, 3rd level of decomposition

The alternative system architecture has structural changes and therefore a different availability impact. Segmentation of the LL and RVC into the LL1, LL2, RVC1 and RVC2 respectively, results in changes to the availability impact data. These changes are the result of discussions with a reliability engineer (and domain expert) [J. Wijn, Personal conversation 04-02-2019]:

- IVR (In Vacuum Robots, SLR and SUR) - 4 hours reduction in restart time.
- Clamps (part of IVR) - 6 hours reduction in restart time.
- VPA - 5 hours reduction in restart time.
- LL1 and LL2 - Preventive maintenance times added (SD).

Restart time, part of the MTTR (see Appendix A.1), is the time needed to return to operating conditions. The reduced restart times are based on conversations with VDL ETG reliability engineers whereby the effect of the

segmentation, the smaller volume to be brought to vacuum conditions, and easier access are combined into the reduced restart time. The overall expected downtime for the generated alternative is due to the reduction in restart time 66.2 hours per machine per year (SD and USD combined).

The availability impact of the new components is calculated based on the availability impact of the original components. New components are created due to segmentation, no internal working principles have changed. The availability impact data is specified for sub-assemblies of the defined components. All relevant sub-assemblies are combined, to calculated the availability impact of the new components.

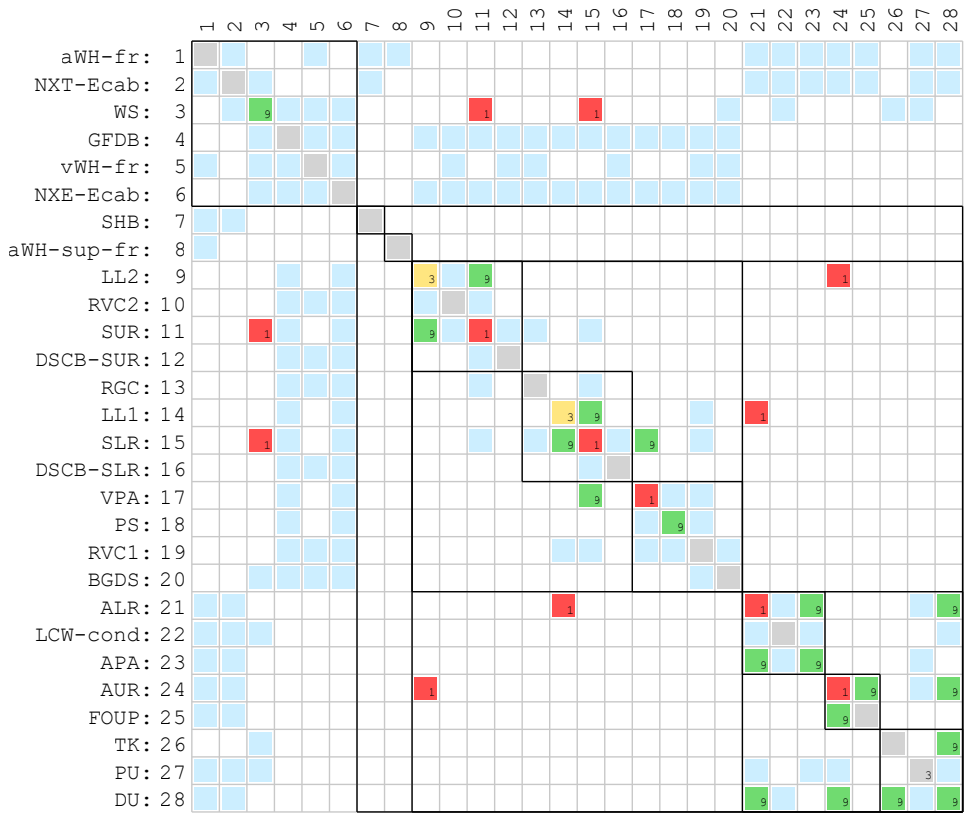


Figure 6.4: Throughput RDSM, 3rd level of decomposition

Similar to the original system architecture, fixed KPIs are projected to the DSM. In Figure 6.4 the fixed KPI ranking of the throughput can be seen, other fixed KPI mappings can be found in Appendix D. KPI values are based on the original design, the internal processes of the LLs and RVCs have not changed. Therefore the same values are taken. The wafer flow is changed, for the changed wafer flows the KPI values are based on conversions with VDL ETG domain experts.

6.4 Discussion

By applying the inventive principals of TRIZ, an alternative system architecture is generated. The inventive principles used are: 4

- Segmentation
- Taking Out
- Cushion in advance
- Parameter change

Segmentation of the LL and RVC result in a changed wafer flow. The wafer input and output paths are separated and do not have common elements any more. This is shown in Figure 6.2. RVC1 contains all elements related

to the input path and RVC2 contains all elements related to the output path. The separation in wafer paths is not only in clustering, but also in space and time. The changed wafer flow requires a different sequence of operation. Given the minimal process times of the elements, this is possible. Additionally, a reduction in SUR cycles is achieved. The SUR only handles each wafer once while the SLR still performs two moves with a wafer.

A second result, are two RVCs of the vWH that can be removed separately from the WH for maintenance or repairs. As a consequence, smaller volumes have to be restored to vacuum conditions after repairs. Effectively, the vacuum chamber of WS is taken out of the volume to be brought to vacuum conditions. The result is also a cushioning in advance, to compensate for the lower reliability of the vWH components. The flexibility to repair or replace parts of the vWH is also increased.

The critical areas within the machine regarding availability impact, relate to elements within the vWH. The goal to reduce the MTTR and therefore the availability impact, is partially reached. The SLR and VPA are the most vulnerable elements followed by the SUR. Since only segmentation is applied to the LL and RVC, it makes sense that the critical areas within the WH are not changed. The overall expected downtime is reduced to 66.2 hours per machine per year (SD and USD combined).

TRIZ is recommended to be used in similar projects, but alternative methods for the generation of system alternative must be examined. In this project, fixed KPI are not allowed to deteriorate, making TRIZ an ideal method. The TRIZ contradictions, where one parameter must remain constant while improving another, suits the criteria of the fixed KPIs of this project. If no fixed KPIs are present, research is needed to find if TRIZ is still sufficient to generate alternative system architectures. A reason for additional research in these cases, is that some inventive principles are contradicting, for example; segmentation and merging. Therefore it is also advised, that the designer (using TRIZ) takes into account that inventive principles suggested by TRIZ may contradict.

Chapter 7

System Architecture Comparison

In this chapter the current system architecture of the WH, presented in Chapter 5, is compared with the alternative system architecture, presented in Chapter 6. The Analytical Hierarchy Process (AHP) is used for comparison and selection of the optimal system architecture.

Proper AHP, as presented in [31], can applied if an original and more then 1 alternative are to be compared. For a single alternative, AHP reduces to a direct comparison and consistency checks between preferences can longer be performed. In both cases, the Changeover-DSM and Changeover-Delta-DSM can provide an overview of commonalities and differences between the system architectures and KPIs, which supports conceptual decision making in engineering design.

The outline of this chapter follows the AHP steps. First, the KPIs are ranked by means of pairwise comparison. Secondly, the candidate system architectures are compared by using Changeover and Changeover-Delta DSMs. Finally, the results of AHP are discussed en evaluated.

7.1 Criteria ranking

Ranking the criteria, In this case the KPIs, is the first part of AHP. Each KPI is pairwise compared to each other according to a scale of 1 to 9 [42]. Table A.1 presents the meaning of the comparison values. All pairwise comparisons are preformed with domain-experts at VDL ETG. The listed order of the KPIs is presented in Equation (7.1).

$$KPIs = \begin{bmatrix} \text{Machine MTBF} \\ \text{Machine MTTR} \\ \text{Wafer Alignment} \\ \text{Wafer Throughput} \\ \text{Wafer Temperature} \\ \text{Wafer Contamination} \end{bmatrix} \tag{7.1}$$

Equation (7.2) shows the result of the KPI ranking. In matrix $A_{Combined}$, the result of the pairwise KPI comparisons is presented. Diagonal values have a value of 1, off-diagonal values are the result of a pairwise comparison. The value 4 of entry (2,1) means that MTBF is weak to moderately favoured over MTTR. Meaning a long lifespan of components is favoured over repair time. Entry (1,2), compares the same KPIs (in reverse order) and therefore gets the inverse value assigned by definition [31][42]. All AHP calculations are further explained in Appendix A.

$$A_{Combined} \begin{bmatrix} 1 & 3 & \frac{1}{8} & \frac{1}{2} & \frac{1}{2} & 1 \\ \frac{1}{3} & 1 & \frac{1}{5} & 1 & 1 & 1 \\ 8 & 5 & 1 & 5 & 5 & 5 \\ 2 & 1 & \frac{1}{5} & 1 & 1 & 3 \\ 2 & 1 & \frac{1}{5} & 1 & 1 & 3 \\ 1 & 1 & \frac{1}{5} & \frac{1}{3} & \frac{1}{3} & 1 \end{bmatrix}, V_{KPIs} = \begin{bmatrix} 0.0924 \\ 0.0828 \\ 0.5045 \\ 0.1271 \\ 0.1271 \\ 0.0661 \end{bmatrix}, CR = 0.0742 \tag{7.2}$$

The normalized eigenvector V shows the weighted importance of each KPI with respect to the others and is the result of the KPI ranking. The order of V is identical to the order of (7.1). Wafer alignment is the main KPI

of the WH according to the criteria ranking. This is in line with the domain experts, wafer overlay (alignment) is extremely critical for the performance of the entire wafer stepper. Wafer alignment is one of the main and important tasks of the WH and therefore the main KPI of the WH. If the wafer alignment fails, all wafers processed by the lithography machine are considered faulty and therefore useless. If other (fixed) KPIs fail to meet the performance, then some (by definition not all) wafers will be faulty.

It is important to note that the wafer alignment KPI is a fixed KPI. In the current system architecture this KPI is achieved. The goal of this project is to improve non-fixed KPIs. Deterioration of fixed KPIs is not allowed. Wafer alignment is the most important KPI, but is not aimed to improve in the generated alternative system architecture.

Finally, in (7.2), the consistency ratio (CR) is shown. AHP has a consistency check to prevent inconsistent choices [32]; like loops. For example, the following comparisons are not consistent: A is preferred over B, B is preferred over C, and C is preferred over A. Small degrees of inconsistency are allowed [31]. Inconsistent choices allow for expertise to be included into the decision process. Expert knowledge can contradict with the decisions made, the consistency ratio take this expert knowledge into account. The CR is aimed to be below 0.10 [31], in (7.2) can be seen that this is the case. The formulas for the CR calculation are presented in Appendix A.

7.2 System Architecture comparisons

Candidate comparison is the following step of AHP. The candidates are the original system architecture and generated alternatives. Comparison between the current system architecture and the generated alternative are based on the regular DSMs, the availability impact DSMs and fixed KPI DSMs.

Figure 7.1, shows the CD-DSM of the generated alternative system architecture (green box) and the original system architecture (red box). The layout of the CD-DSM is identical to the layout of Figure 3.4. The LL and RVC elements are eliminated compared to the alternative system architecture. The RVC1, RVC2, LL1 and LL2 are added elements compared to the original system architecture. The clustering of the current system architecture is used as a basis for the CD-DSM. The group of mutual (and unchanged) elements shows the (remainders) of the clustering. If an element is eliminated or changed, it is removed from the cluster such that the CD-DSM can be created

Added dependencies are highlighted in rows 1 till 4 of the Δ of the CD-DSM. Rows 5 till 8 display (above the diagonal) all network changes as in the alternative system architecture situation. Likewise, columns 33 and 34 highlight the eliminated dependencies in the Δ of the CD-DSM. Changed dependencies as in the alternative system architecture are shown in columns 29 till 31 (again above the diagonal). All blue squares in the CD-DSM indicate mutual dependencies between both system architectures.

Differences in the wafer path, can be detected in the Δ of the CD-DSM (above the diagonal). Solid material dependencies between the LL and SUR (entry 33,29) are eliminated due to the splitting of the LL and RVC. Added dependencies show a solid material dependency between the LL1 and SLR (entry 7,4) and LL2 and SUR (entry 5,2), depicting the new wafer flows. Additionally, the solid material flows between the PS and SLR (entry 31,30) are removed, as is the dependency between the SUR and VPA (entry 32,29). Instead, an added dependency is shown between the SLR and VPA (entry 8,7). The changeover part of the CD-DSM shows that the original system architecture contains a spatial and a solid material dependency between the SLR and VPA (entry 31,32). The alternative system architecture shows a changed ratio between the spatial and a solid material dependencies between the SLR and VPA (entry 7,8).

The availability impact DSMs are compared in Figure 7.2. In addition to network changes, a check is added to find differences in availability impact (of the elements). The set of elements with changed availability impact is (in this case) equal to the set of elements with changed dependencies. Therefore, no additional elements are added to the set of changed components. Since the dependency values rely on the component values (as described in Equation (3.2)), all dependencies related to the set of elements with changed availability impact have changed availability impact values.

All values, of both system architectures, are colour-coded and scaled to the highest availability impact. Off-diagonal values are the sum of the availability impact of the corresponding elements, as described in Equation (3.2). The assigned colours of the off-diagonal values are coupled to the magnitude. For colour contrast, the

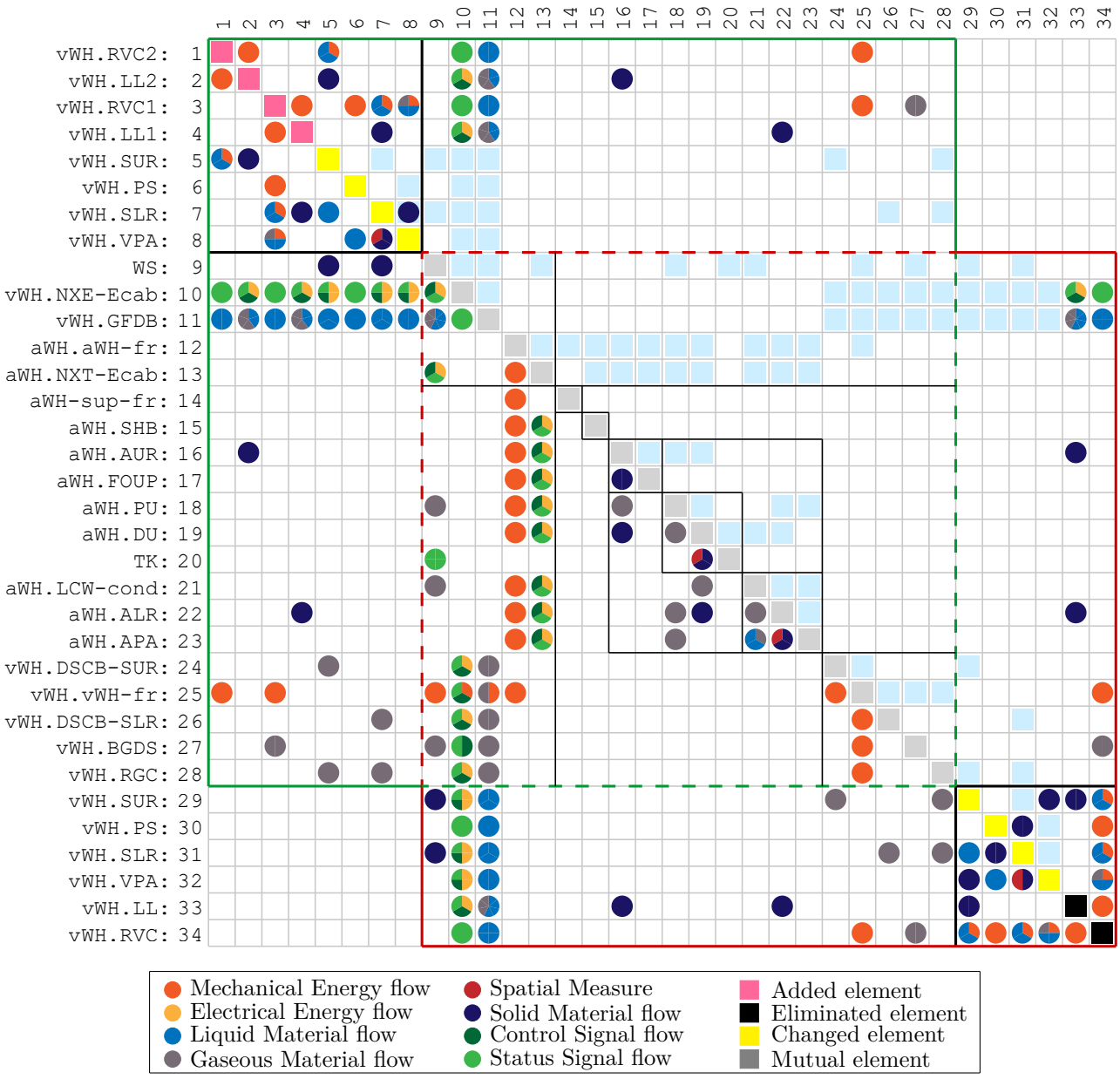


Figure 7.1: CD-DSM, 3rd level of decomposition

changed elements have been assigned a new colour; blue instead of yellow. The other element colours are identical to Figure 7.1.

The defined critical areas of both system architectures (in Chapter 5 and 6 respectively), can both be seen in Figure 7.2. In both system architectures, the SLR, SUR, VPA and LL(s) are critical with respect to the availability impact. The availability impact of the alternative system architecture are lower compared to the original system architecture. Segmentation of the LL and RVC achieves a significant reduction in availability impact between the SLR and VPA (entry 5,6 versus 29,30). These critical elements can, due to the segmentation, be repaired faster then the current system architecture.

Mutual (unaffected) elements, show that the aWH is unaffected by the changes. The availability impact of the aWH is still lower then the vWH, leaving the vWH the part of the WH to be further improved. Availability impact improvements related to the wafer exchange between the aWH and the LL(s) relate to the faster repair times of LL1 and LL2, not to the aWH.

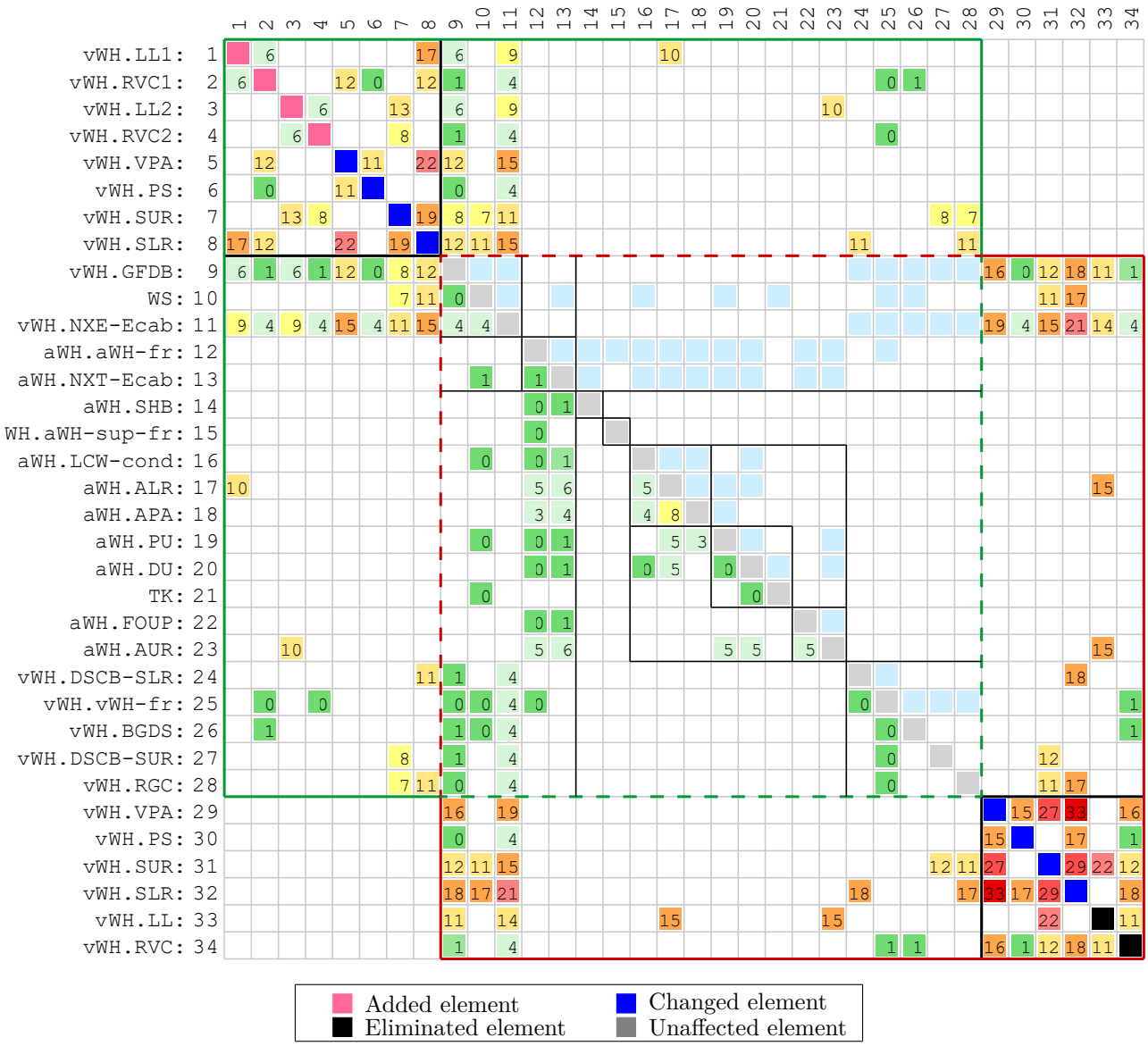


Figure 7.2: Availability impact CD-DSM, 3rd level of decomposition

It can be concluded that the alternative system architecture is an improvement with respect to the non-fixed KPIs (MTBF and MTTR). Due to the changes in the system architecture, the availability impact values of the alternative system architecture are decreased. Availability impact values of changed and added elements (alternative system architecture), are all lower or equal compared to the eliminated and changed elements (original system architecture). The overall availability impact of the alternative system architecture is better (lower) compared to the original system architecture (66.2 versus 79.8 [*hours/year*], SD and USD combined).

Both system architectures are also pairwise compared with respect to the fixed KPIs (wafer alignment, temperature, throughput and contamination). Figure 7.3 shows the Changeover DSM of the wafer temperature for both system architectures. The figure shows the smallest allowable deviation with respect to a local nominal temperature. Due to the segmentation of the RVC and LL, all wafer handling elements related to the output path (SUR, LL2), have a large allowable margin compared to the original design. In the original design, these elements also had to handle the wafers of the ingoing path with a more strict temperature margin.

No deterioration in the wafer temperature deviation (to a local nominal temperature) can be seen in Figure 7.3. Improved values can be seen in the outgoing wafer path of the vWH, this relates to the segmentation of

the RVC and LL. It can be concluded that the wafer temperature regime in RVC2 (outgoing wafer path) is less strict (entry 3,5). However, during operation of the WH, both RVC's contain the same vacuum as they are connected by the vacuum chamber of the WS. So, allowing the temperature to fluctuate more, still affects the ingoing wafer path.

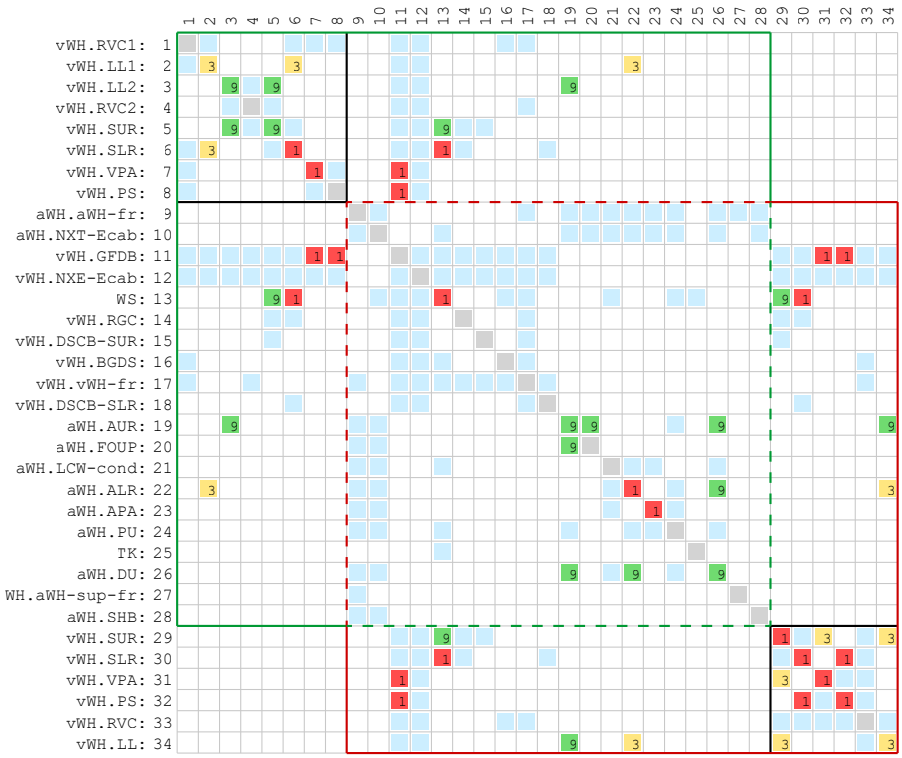


Figure 7.3: Changeover RDSM of the Temperature, 3rd level of decomposition

A second improvement relates to the PS. The original system architecture requires the wafer to be held in the PS by the SLR, while the SUR loads the VPA and unloads the WS. In the alternative design, the PS is skipped due the changed wafer flow of the WH. A critical (value 1) dependency with respect to temperature is eliminated as a result (entry 8,6 versus 32,30). Therefore, based on figure 7.3 and after discussion with a domain expert, an improvement in wafer temperature is to be expected when the wafer is directly transported from the VPA to the WS. However, the magnitude of the improvement is hard to determine.

The comparison of the most important KPI, wafer alignment, can be seen in Figure 7.4. In the alternative system architecture, the PS is excluded from the vWH input path. However, in the original system architecture the SLR holds the wafer in the PS and does not release it. This results in no losses, nor gains in wafer alignments.

Other fixed KPI figures can be found in Appendix E. From these figures can be concluded, that no deterioration in the remaining fixed KPIs exists.

The fixed KPI pairwise candidate comparison between both system architectures is set to 1 (equal importance) for all fixed KPIs, since the comparisons between the system architectures show no deterioration. Only for the wafer temperature a small improvement can be achieved. Because no substantiated value can be given for the temperature improvement. It is assumed the temperature improvement is negligible, any improvements are considered a safety factor to the wafer temperature.

Differences in non-fixed KPIs can be seen in Figure 7.2. It is known that the reduction of the availability impact is due to the reduction of the repair times. A small improvement in the MTBF can be achieved, due to the smaller number of cycles the SUR has to perform as explained in Chapter 6. The SLR does not have an increase in cycles. Therefore, the alternative system architecture is preferred over the current system architecture with

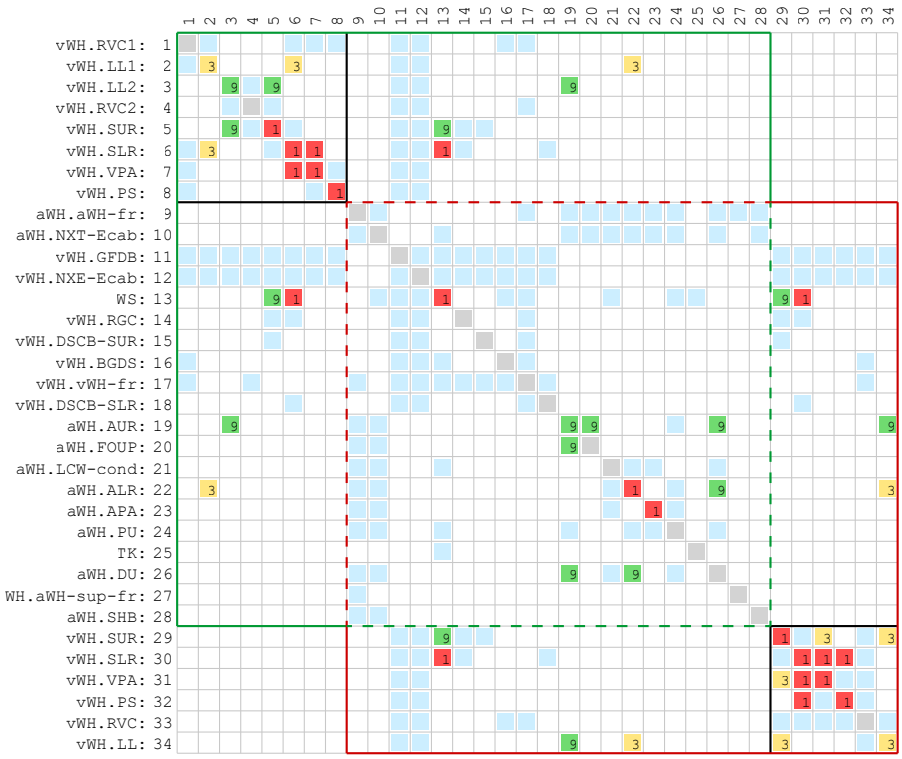


Figure 7.4: Changeover RDSM of the Alignment, 3rd level of decomposition

a very small preference of 2 regarding MTBF. The comparison is shown in Equation (7.3).

$$A_{MTBF} \begin{bmatrix} 1 & \frac{1}{2} \\ 2 & 1 \end{bmatrix}, V_{MTBF} = \begin{bmatrix} 0.333 \\ 0.666 \end{bmatrix} \tag{7.3}$$

Major improvements in availability impact are achieved by reducing the MTTR. The alternative system architecture is preferred over the current system architecture. Since not all elements are affected, a moderate preference is given (value of 5). The comparison is shown in Equation (7.4).

$$A_{MTTR} \begin{bmatrix} 1 & \frac{1}{5} \\ 5 & 1 \end{bmatrix}, V_{MTBF} = \begin{bmatrix} 0.200 \\ 0.800 \end{bmatrix} \tag{7.4}$$

When all comparisons are combined, the candidate ranking is equal to $A_{Candidates}$ of (7.5). This matrix is multiplied with the normalized eigenvector V_{KPIs} of (7.2). The result is the eigenvector V_{final}, which shows that the generated alternative system architecture is slightly preferred over the original system architecture.

$$A_{Candidates} \begin{bmatrix} 0.333 & 0.200 & 0.500 & 0.500 & 0.500 & 0.500 \\ 0.666 & 0.800 & 0.500 & 0.500 & 0.500 & 0.500 \end{bmatrix}, V_{final} = \begin{bmatrix} 0.46 \\ 0.54 \end{bmatrix} \tag{7.5}$$

7.3 Conclusion and Discussion

This chapter compares the original system architecture with the generated alternative system architecture. For the selection of the optimal system architecture a subset of AHP is used. AHP is an existing method of which the working principles have been developed and proved in other research [32][42]. The pairwise comparison between system architectures, visualized by (comparison) DSMs, supports the decision making between two designs. If only two design are available, the pairwise comparison reduces to direct comparison without consistency checks to the decisions.

Insight into the differences and commonalities between the system architecture is provided by the Changeover-Delta DSM. The dependency changes between designs can be seen, as are added and eliminated dependencies and elements. It can be stated that the overlap between two systems is a measure of commodity. Or that the non-overlap between systems is a measure of change. The measure relates to the number of changed, added and removed dependencies. If an element is added, all related dependencies are added. If a network of dependencies changes, the elements do not have to change. Therefore, dependencies are leading in a measure of change with respect to the system architecture.

The CD-DSM is not limited to the comparison of system architectures. If the alternative system architecture would to be applied to WH and become the real system architecture. The CD-DSM of Figure 7.1 can be send to the designers and engineers, who (by the delta-part of the CD-DSM) can apply the changes to the internal structures of the machine.

The Changeover DSM provides a basis for the projection of fixed KPIs. As seen in Figure 7.3. KPIs which apply to all, or a selection of elements, and can be mapped to the Changeover DSM. This provides the engineer with a single view in which both systems, and the KPI values, can be compared.

The generated alternative is an improvement in comparison with the original system architecture. The expected total downtime is dropped from 79.3 to 66.2 hours per year (SD and USD combined). This is achieved by segmentation of the LL and RVC, allowing smaller sections of the vWH to be moved out of the WH.

No deterioration in the fixed KPIs and (very) small improvements may be present in the generated alternative system architecture. Since the effect of the improvements can not be substantiated, it is assumed the improvements are negligible. This way, a safety margin is created to support the claim that no deteriorations are present.

Fixed KPI projections onto the DSMs can be improved. A margin (expressed in an order of magnitude) is projected with respect to a KPI. This method is not optimal. Preferably, the actual contribution or deviation is projected on the dependency or element. Yet, if none or limited information is available regarding the KPIs, the used method allows estimation by the engineer. If more information is available, the used method is subject for future improvements.

In addition to the improvements of the fixed KPI projections, investigation to the orthogonality of KPIs can be preformed. If two KPIs both have effect on certain dependencies, the effects of design changes can have a larger effect (positive or negative) throughout the system than decoupled KPIs. Visualization of KPI orthogonality with the use of projection DSMs can provide a clear overview to the system engineer if, where, and which dependency influences other KPIs.

AHP puts the availability impact improvements into perspective. Although the alternative system architecture is moderately preferred (with respect to MTTR), lack of fixed KPI improvements suppress MTTR improvements in the overall preference values. However, the goal is to find improvements for just the non-fixed KPIs. Therefore, it is concluded that the generated alternative system architecture is a proper alternative to improve the availability impact of the WH.

An important notice is that system cost and redesign effort are not taken into account. Cost is not a selected KPI in the case study. It would be interesting to see if the alternative system architecture can reduce operational losses (due to unscheduled downtime) to compensate for the redesign effort. This consideration is an important factor in deciding whether to apply the design change.

Chapter 8

Conclusions

The goal of this project is to investigate whether ESL and DSM can support the evaluation of conceptual design choices of high-tech engineering systems. To this purpose, the NXE:3400C Waferhandler is used as a case-study.

A four step method is presented which combines ESL, DSM, TRIZ, and AHP. The first step is to model and evaluate the current system architecture using ESL and DSM. The second step consists of the generation of design alternatives using TRIZ. The TRIZ generated alternative is modelled and analysed in step three, again by ESL and DSM. The final step compares the alternative to the original system architecture using AHP and comparison DSMs. Chapter 3 explains these steps in detail.

The proposed method is applied to the case study, where Chapter 4 presents the evaluation of ESL, used for design specification of system architectures. Analysis and modelling of the current system architecture is performed in Chapter 5. Chapter 6, covers the generation and analysis of the alternative system architecture. Comparison between system architectures is applied in Chapter 7.

This chapter concludes this report. In the first section, the presented method and its application to the case study is concluded. Section two, concludes the usage of ESL to specify system architectures. In the third section, the outcome of the proposed method is concluded. Finally, some recommendations and points for future work are presented.

8.1 Method

In this report, a method is proposed for generation and comparison of alternative system architectures. The method integrates three existing methods, Design Structure Matrices (DSMs), Theory of Inventive Thinking (TRIZ) and the Analytic Hierarchy Process (AHP). DSMs are used for the analysis and visualization of the system architectures. TRIZ is used for the generation of alternative system architectures. And finally, AHP is used for the selection of the optimal system architecture.

Based on system specifications, an ESL specification is created of the current system architecture. From this ESL specification, a DSM is generated which is also the basis for projection of Key Performance Indicators (KPIs) onto the DSM. This provides an overview regarding the system's performance with respect to KPIs. From the KPI projections, critical areas and elements are defined. These are used as input for the generation of an alternative system architecture.

The theory of inventive problem solving (TRIZ) is selected as a method for generation of an alternative system architecture. The critical areas of the original system architecture are translated to the TRIZ domain. By translating the KPIs into the TRIZ technical parameters, inventive principles are suggested by the contradictions matrix. Selected inventive principles are translated back to the WH-domain, where they are applied to the WH system architecture.

Comparison of system architectures, is the final step of the method. Pairwise comparison, part of the Analytic Hierarchic Process (AHP), is used to compare system architectures. To support pairwise comparison, Changeover-Delta DSMs (CD-DSMs) are used. The CD-DSM combines two DSMs and shows additions, eliminations and changes in a single figure for both elements and dependencies.

Comparing the effects of the system architectures on the fixed KPIs, is done by Changeover-DSMs *[S.D.Eppinger, personal conversation 27-11-2018]*. These allow for a pairwise comparison, where changes in KPI values can be

compared. This allows a design team to pairwise compare system architectures and support conceptual design choices.

Improvement to the proposed method are improved KPI rankings and the investigation of different methods for generation of design alternatives. The KPI ranking used (for fixed KPIs) was selected due to a lack of information. A different KPI ranking, if equal or more information is available, can improve the proposed method.

TRIZ is not necessary for generation of alternatives designs, other methods or random (non method related) ideas are possible as well. TRIZ is used to generate and support conceptual decisions. That is, the TRIZ inventive principles have been created based on proven concepts [11]. Thus using the inventive principles increases the chance of a successful alternative. It is therefore advised to use TRIZ in similar cases as the WH case study.

Pairwise comparison is aided by the CD-DSM and changeover DSMs, the latter with projections of the KPIs. If more then one alternative is generated, AHP can be used properly, to it's full extend (including the consistency checks). This in contrast to the subset used in this case. Pairwise comparison of the KPIs is performed, even for two system architectures. KPI comparison can point out the most important KPI and put gains of other KPIs into perspective.

The CD-DSM, created in this project, is not limited to the scope of the project or presented method. The CD-DSM can be used for finding differences and commodities between two system architectures. Above the diagonal; changed, added or eliminated dependencies between two system architecture DSMs are displayed. Below the diagonal all dependencies of both system architectures are shown. This creates a single overview of the total situation (below diagonal) and the changes (above diagonal). One can see exactly which dependency is changed and which other dependencies exist between two elements. This combined view can inform a design team or customer concerning design changes. The CD-DSM can be used for finding differences, but also commodities. For example; comparison of evolutionary systems to find changed dependencies and for finding commodities within a product family.

The case study shows that the proposed method can be used to support conceptual design choices. Therefore, it confirms that ESL and DSM can be used in the early conceptual design stage. The presented method generates alternative designs, based on TRIZ supported conceptual design choices. Pairwise comparison of AHP is used to determine whether an alternative is an improvement. The Changeover DSM and Changeover-Delta DSM, both constructed from the system architecture DSMs, are used to support pairwise comparison.

8.2 ESL

In Chapter 4, the use of ESL is evaluated and changes and additions to the language are presented. In Appendix B the additions to the compiler are presented.

ESL is considered a helpful tool to create system specifications in a structured way. Furthermore, ESL helps to gain insight into the system architecture by allowing system decomposition. By decomposing the system, parts of the system can be modelled up till the required amount of granularity. This allows for smaller sets of functional requirements to be defined per layer of decomposition and maintain overview.

Another advantage of ESL is the automatic derivation of dependencies. Applying changes to the functional requirements has a direct effect on the derived dependencies and corresponding DSM. Removal or addition of functional requirements results in elimination or addition of all (and correct) corresponding dependencies. Compared to manual creation of a DSM, ESL proves to be more detailed and systematic [19].

Translation of natural language and figures into ESL functional requirements is done by the functional basis function set Hirtz [21] and functional basis flow set of Tilstra [22]. Both functional basis provide sufficient verbs and flows to describe the WH in terms of ESL goal-requirements, transformations requirements and components.

In addition to the ESL evaluation, some additions to the ESL language are proposed and implemented during this project. Proposed improvements are:

1. *Use of the preposition* "`from`" *to indicate an inverted flow for both goal- and transformation requirements.*
   - An inverted flow is defined as a flow whereby the target component in the flow definition is the preforming

component.
2. *Introduction of KPI declarations, located at the goal requirements and component definitions.* - There is chosen for declarations in the ESL specification, to allow for automatic value derivation in the future. Meaning, that KPI values can be derived from the contents of the subclauses (already part of ESL).
3. *Introduction of subfiles.* - `\include`-statements are used to refer to subfiles. This allows for smaller files to work with and for parallel work on the ESL specification.

8.3 Wafer Handler

A DSM of the NXE:3400C Waferhandler's system architecture is created and analysed in Chapter 5. Analysis relieved that the atmospheric WH (aWH) and vacuum WH (vWH), the two main modules, are independent of each other. The only dependencies between the modules are a physical connection and wafer exchanges. All control and (liquid, gas and power) supplies are taken care of by the wafer-scanner. Within the aWH, clustering revealed a clear input and output path of the wafer. For the vWH, no such distinction could be made regarding wafer paths. This caused by the Stage Unload Robot (SUR), which participates in both the ingoing and outgoing wafer paths.

Projection of availability impact to the DSM, shows that both vWH robots (SLR and SUR), the vacuum pre-aligner (VPA) and LoadLock (LL), are critical elements within the WH. The cluster containing the vWH elements positioned in vacuum is identified as critical area within the WH.

The alternative system architecture is generated by TRIZ. The selected TRIZ inventive principles and the implementations on the system are;

- *Segmentation* - Segmentation of the LL and Robot Vacuum Chamber (RVC), the wafer input path and output path of the vacuum WH are split as a consequence. All components related to the input-path are placed in one vacuum chamber and the components related to the output path are placed in the second vacuum chamber.
- *Taking Out* - Addition of a gate valve to close off the vacuum chamber from the Waferstage (WS). The WS can remain vacuum while the WS is repaired. Effectively the (vacuum chamber of the) WS is taken out, reducing the volume and time to restore vacuum conditions.
- *Cushion in advance* - By segmentation and taking out, smaller parts of the WH can be removed for easier repair. The smaller vacuum to be restored also reduces restart times and therefore shorter repair time.
- *Parameter change* - Segmentation results in a change in volume. A smaller volume must be brought and maintained at vacuum conditions.

As a result of segmentation, the wafer input and output paths are split. Clear wafer in- and output paths are defined within the machine. The clusters of the alternative system architecture DSM confirm and reflect the separated wafer paths.

Due to the improved ease of repair, the availability impact of the alternative system architecture is 16,7% lower compared to the original system architecture. Critical areas within the WH remain the same components. Only repair times are reduced, not the fail-rate. Although improvements to the fail-rate of the SUR might be present due to a reduced amount of cycles per wafer.

Pairwise comparison between the system architectures show that the generated alternative system architecture is preferred over the original system architecture. No deterioration in the fixed KPIs (Alignment, Contamination, Temperature and Throughput) is present. Non-fixed KPIs (MTTR and MTBF) are both improved. Due to segmentation of the WH, smaller volumes have to be brought to vacuum conditions after repairs and modules are easier accessible. This results in an improvement in repair times compared to the original system architecture. Separation of the wafer paths reduce the amount of cycles per wafer for the SUR, while the cycles for the SLR remain identical. Hence, an improvement in lifetime of the (vacuum unload) robot is to be expected.

8.4 Recommendations and Future Work

For the proposed method the following points of future work are defined:

- Apply method for multiple alternatives, preferably on a smaller and simpler system. - The method is only tested on a single case study. Case studies with multiple alternatives can prove if the method is sufficient or modifications are needed.
- Generate alternatives using different methods instead of TRIZ - Multiple methods for the generation of alternatives exist and can be examined for usage within the method.
- Investigate alternative comparison methods instead of AHP - Multiple methods for decision making exist and can be examined for usage within the method.
- Use different Ranking methods - KPI projections for fixed-KPIs are not optimal. Different methods to project the system properties to the created DSMs are needed.
- Research for the orthogonality visualization of KPIs - Investigation whether KPI projections on DSMs can determine if KPIs are orthogonal to each other, as described in axiomatic design [43]. If two requirement are orthogonal, both can be modified without affecting the other.

The CD-DSM, created in this project, is also subject to future work. As stated before, the CD-DSM can be used outside the scope of this project. Therefore these possibilities can be investigated.

- Use of the CD-DSM to find commonalities between system architectures - The CD-DSM is used to find and compare differences between system architectures. However, the CD-DSM can also be used for finding commonalities within (for example) product families.
- Include (re)design effort complexity measures and map these to the CD-DSM for comparison between two generated alternatives. A suggestion for (re)design effort complexity measures is given in Hölltä [44], where DSMs are used for analyse of redesign effort.
- Possible solution for the orthogonality of KPIs - The commonalties of the CD-DSM can be used to detect dependencies which are marked in both KPI projections. This means that a system architecture with two orthogonal requirement results in a CD-DSM where only the dependencies which do not participate in either KPI are displayed in the commonality section.

The proposed changes to ESL, can be implemented into the ESL language syntax. To this purpose the compiler has to be adjusted as well. A suggestion for the implementation of these changes into the compiler is presented in Appendix B. Additions made to the compiler, used in this project, consist of the algorithms to create the projection DSMs, Changeover DSMs and CD-DSM. If these functionalities are added to the newest compiler version, parts of the created algorithms can be reused.

The created model of the WH, is a high level model. However, the improvements in availability impact of the generated alternative system architecture are promising. It is advised to further investigate the proposed improvements for the WH, by creating a DSM with more (detailed) levels of decomposition. Research for lifespan improvements of the SUR, as element of the alternative system architecture, is recommended. If improvements can be achieved, the availability impact of the SUR is further reduced.

It is expected that the proposed method will be sufficient for most systems. If all KPIs need to be improved the method may be unsuitable. Yet, in this case, one may wonder if the existing system architecture is any good at all. Hence, the usage of DSM, TRIZ and AHP combined can prove to be a successful combination. Within this report it is already shown that conceptual decisions can be evaluated by DSMs, alternative system architecture can be generated by TRIZ and compared by AHP (supported by the CD-DSM and changeover DSM). Overall, the proposed method can be seen as the basis for future use, with some opportunities for improvement.

Bibliography

[1] G. Pahl, W. Beitz, J. Feldhusen, and K. Grote, *Engineering Design, A Systematic Approach.* Springer-Verlag London Limited, third english ed., 2007.

[2] L. Wang, W. Shen, H. Xie, J. Neelamkavil, and A. Pardasani, "Collaborative conceptual design - state of the art and future trends," *Computer-Aided Design*, vol. 34, no. 13, pp. 981 – 996, 2002.

[3] P. Couturier and A. Imoussaten, "A qualitative method for evaluation in conceptual design," *IFAC-PapersOnLine*, vol. 48, no. 3, pp. 721 – 726, 2015.

[4] D. V. Steward, "The design structure system: A method for managing the design of complex systems," *IEEE Transactions on Engineering Management*, vol. EM-28, no. 3, pp. 71–74, 1981.

[5] T. R. Browning, "Design structure matrix extensions and innovations: A survey and new opportunities," *IEEE Transactions on Engineering Management*, vol. 63, pp. 27–52, Feb 2016.

[6] S. D. Eppinger and T. R. Browning, *Design Structure Matrix Methods and Applications.* MIT-press, 1st paperback ed., May 2016.

[7] T. Wilschut, *System specification and design structuring methods for a lock product platform.* PhD thesis, Department of Mechanical Engineering, University of Technology Eindhoven, 2018.

[8] VDL Enabling Technologies Group. `www.vdletg.com`. Website, [Online] Accessed: 2019-03-18.

[9] T. K. Brady, "Utilization of dependency structure matrix analysis to assess complex project designs," *14th International Conference on Design Theory and Methodology, Integrated Systems Design, and Engineering Design and Culture*, vol. 4, pp. 231–240, 2002.

[10] G. S. Altshuller and P. B. Shapiro, "Psychology of inventive creativity," *Voprosi of Psihologi (Problems of Psychology)*, vol. 6, 1956.

[11] G. S. Altshuller, *40 principles, TRIZ keys to technical innovation.* No. 1 in Triz tools, Worcester, Mass.: Technical Innovation Center, 1. ed ed., 1998.

[12] T. Saaty, *The Analytic Hierarchy Process.* McGraw-Hill international, New York, 1980.

[13] K. Ulrich, "The role of product architecture in the manufacturing firm," *Research Policy*, vol. 24, no. 3, pp. 419 – 440, 1995.

[14] E. Crawley, O. L. de Weck, S. Eppinger, C. S. P. Magee, J. Moses, W. J. Seering, J. E. Schindall, D. Wallace, and D. Whitney, "The influence of architecture in engineering systems," tech. rep., Massachusetts Institute of Technology (MIT), The ESD architecture committee, 2004. monograph.

[15] H. Jaakkola and B. Thalheim, "Architecture-driven modelling methodologies.," *Frontiers in Artificial Intelligence and Applications*, vol. 225, pp. 97–116, 2010.

[16] T. Wilschut, J. Rooda, L. Etman, and J. Vogel, "A dsm based method for the ranking of system components w.r.t. system reliability and availability," *19th International Dependency and Structure Modeling Conference, DSM 2017, 1-13 September 2017, Espoo, Finland*, pp. 1–10, 2017.

[17] T. Wilschut, L. Etman, J. Rooda, and I. Adan, "Multi-level flow-based markov clustering for design structure matrices," *Journal of Mechanical Design : Transactions of the ASME*, vol. 139, no. 12, 2017.

[18] R. Thebeau, "Knowledge management of system interfaces and interactions for product development processes," master thesis, Massachusetts Institute of Technology, 2001.

[19] T. Wilschut, L. F. P. Etman, J. E. Rooda, and J. A. Vogel, "Generation of a function-component-parameter multi-domain matrix from structured textual function specifications," *Research in Engineering Design*, Feb 2018.

[20] T. Wilschut, A. Hofkamp, L. Etman, J. Rooda, and J. Vogel, "A language for multi-level system specification and automated derivation of component, function and variable dependencies." Submitted.

[21] J. Hirtz, R. B. Stone, D. A. McAdams, S. Szykman, and K. L. Wood, "A functional basis for engineering design: Reconciling and evolving previous efforts," *Research in Engineering Design*, vol. 13, pp. 65–82, Mar 2002.

[22] A. Tilstra, C. Seepersad, and K. Wood, "A high-definition design structure matrix (hddsm) for the quantitative assessment of product architecture," *Journal of Engineering Design*, vol. 23, no. 10-11, pp. 767–789, 2012.

[23] T. Wilschut, L. Etman, J. Rooda, and J. A. Vogel, "Multi-level function specification and architecture analysis using esl: A lock renovation pilot study," *Proceedings of the ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 7, pp. 1–10, 2018.

[24] T. U. Pimmler and S. D. Eppinger, "Integration analysis of product decompositions," Working papers 3690-94., Massachusetts Institute of Technology (MIT), Sloan School of Management, 1994.

[25] G. M. Bonnema, "Triz for systems architecting," *Procedia Engineering*, vol. 9, pp. 702 – 707, 2011. Proceeding of the ETRIA World TRIZ Future Conference.

[26] L. Haines-Gadd, *TRIZ for dummies.* John Wiley & Sons, Ltd, 2016.

[27] K. Gadd, *TRIZ for Engineers: Enabling Inventive Problem Solving.* John Wiley & Sons, Ltd, 2011.

[28] G. M. Bonnema, "Function and budget based system architecting," *6th International Symposium on Tools and Mehods of Competitive Engineering, TMCE 2006, TMCE*, 4 2006.

[29] L. Chechurin, "Triz in science. reviewing indexed publications," *Procedia CIRP*, vol. 39, pp. 156 – 165, 2016. Structured Innovation with TRIZ in Science and Industry: Creating Value for Customers and Society.

[30] C. C. Tseng, C. C. Torng, and S. C. Lin, "Prioritization of product design tasks using qfd, triz and dsm," in *2010 IEEE 17Th International Conference on Industrial Engineering and Engineering Management*, pp. 871–875, 2010.

[31] R. Saaty, "The analytic hierarchy process - what it is and how it is used," *Mathematical Modelling*, vol. 9, no. 3, pp. 161 – 176, 1987.

[32] E. Triantaphyllou and S. H. Mann, "Using the analytic hierarchy process for decision making in engineering applications: Some challenges," *Industrial Engineering: Applications and Practice*, vol. 2, no. 1, pp. 35 – 44, 1998.

[33] M. Darwish and E. Shehab, "Framework for engineering design systems architectures evaluation and selection: Case study," *Procedia CIRP*, vol. 60, pp. 128 – 132, 2017. Complex Systems Engineering and Development Proceedings of the 27th CIRP Design Conference Cranfield University, UK 10th-12th May 2017.

[34] J. F. Maier, C. Eckert, and P. J. Clarkson, "Experimental investigation of the implications of model granularity for design process simulation," *Journal of Mechanical Design*, p. (In Press), 2019.

[35] P. Baker and J. T Whalen, *Survey of Trade Study Methods for Practical Decision-Making.* NASA Independent Verification & Validation Facility, 2012.

[36] E. S. Suh, M. Furst, K. J. Mihalyov, and O. de Weck, "Technology infusion for complex systems: A framework and case study," *Systems Engineering*, vol. 13, pp. 186–203, 01 2009.

[37] ASML, "EDS Module X," tech. rep., Document owner: Z. Alberti, 2009. Confidential document, Doc Id: Unknown.

[38] ASML and VDL ETG, "EDS Wafer Handler NXE mark1," tech. rep., Document owner: R. Visser, 2011. Confidential document, Doc Id: 127187 / 07.

[39] ASML and VDL ETG, "EPS NXE Wafer Handler mk1," tech. rep., Document owner: J. Franssen, 2010. Confidential document, Doc Id: 126907 / 05.

[40] B. van der Sanden, M. Reniers, M. Geilen, T. Basten, J. Jacobs, J. Voeten, and R. Schiffelers, "Modular model-based supervisory controller design for wafer logistics in lithography machines," *2015 ACM/IEEE 18th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pp. 416–425, 09 2015.

[41] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web.," Technical Report 1999-66, Stanford InfoLab, November 1999.

[42] N. Aziz, S. Sorooshian, and F. Mahoud, "Mcdm-ahp method in decision makings," *ARPN Journal of Engineering and Applied Sciences*, vol. 11, no. 11, pp. 7217 – 7220, 2006.

[43] N. Suh, *Axiomatic Design: Advances and Applications*. MIT-Pappalardo series in mechanical engineering, Oxford University Press, 2001.

[44] K. M. Hölttä and K. N. Otto, "Incorporating design effort complexity measures in product architectural design and assessment," *Design Studies*, vol. 26, no. 5, pp. 463 – 485, 2005.

Appendix A

Extensions to Chapter 3

A.1 Key Performance Indicators

In this appendix, the Key Performance Indicators (KPIs) used in this project are explained in a more detail way compared to the main report. First a short overview of the KPI categories is given. Then each KPI is defined and explained why this KPI is taken into account.

A Key Performance Indicator (KPI) is a measurable value of the system that demonstrates the effectiveness of the system. The KPIs used in this project can be divided into three main categories.

- *Uptime* - Reliability, Availability and Serviceability
- *Performance w.r.t. productivity* - Wafer throughput
- *Performance attributes* - Accuracy, thermal and contamination

By means of KPIs, the system can be ranked using either brady's method or the PageRank method. The goal of the ranking is to find, based on the KPIs, critical components or areas within the machine.

Since wafers are sensitive products, a set of performance attributes is defined by ASML. These performance attributes are split into temperature, positional accuracy and contamination KPIs. The wafers provided to the wafer stage must have a certain temperature and positional accuracy. Furthermore, the number of added particles to the wafer (contamination) must be within bounds.

The KPIs are based on the high level specifications provided by ASML to VDL ETG. The KPI values are specific for the machine version covered in this project, namely the WH of the NXE:3400C.

For this project, the KPIs are divided into two sets: fixed and non-fixed KPIs. The fixed set of KPIs is not allowed to change in a negative way. Meaning the overall performance and quality can not go down. The non fixed set is allowed to change, so values may improve or get worse to investigate the effects of different system architectures.

A.1.1 Non fixed KPIs

The non fixed KPIs are the KPIs related to Reliability, Availability and Serviceability. Indirectly all three sub categories are related to each other. If reliability increases and / or serviceability is improved, availability improves as well.

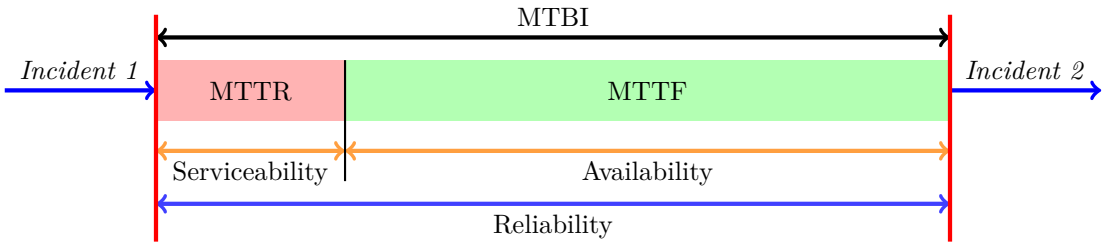


Figure A.1: Reliability and Availability definitions

As can be seen in Figure A.1, the time between two incident is defined as the Mean Time Between Intervals (MTBI). Sometimes MTBI is also referred to as Mean Time Between Failures (MTBF). The MTBI can be split into a number of sub-sets.

Reliability

The Mean Time To Failure (MTTF), is the average time before the next failure occurs. The MTTF is calculated based on the fail-rate. The fail-rate is the average number of times, a(n) (sub)assembly or part fails per year. MTTF (in this project) is measured in years, as can be seen in Equation A.1.

$$MTTF = \frac{1}{failrate}[year] \tag{A.1}$$

The average time it take to repair the machine, or parts of the machine, is the Mean Time To Repair (MTTR). The MTTR does include; time to diagnose, time to access the parts, time for the actual repair and time to restart the machine. MTTR does not include time for the spare parts to arrive, if not available. The MTTR, in this project, is measured in hours.

The MTTR is calculated over one or multiple parts or assemblies. Simply adding the average repair times does not hold for the MTTR calculation. Equation (A.2) shows how the MTTR is calculated. The MTTR is the sum of the individual repair times divided by the individual fail-rate. This gives the average hours per year needed to repair the components / assemblies. Dividing by the sum of the fail-rates (in years), gives the MTTR in hours. This way, the MTTF and MTTR are independent of each other in therms of units of measurement.

$$MTTR = \frac{\sum_{i=1}^{n}(\frac{repairtime}{failrate})_i}{\sum_{i=1}^{n} failrate_i}[hr] \tag{A.2}$$

Availability

Availability is analysed in two ways. The first is related to the MTTR and MTTF. The second related to the Scheduled Down time (SD) and Unscheduled Down time (USD). Part of the availability is the availability-impact, which is equal to numerator of Equation (A.2). Hence, the average time per year the machine needs repairs.

USD is the time the machine is down (not working when it should), due to a failure. If the repairs are the result of machine (or component) failures, the availability impact is part of the USD time.

SD is the time, the machine is down for maintenance. Where USD of a machine randomly occurs, SD is planed beforehand. Hence, the moments the machine is down can be optimized (e.g. at night). Replacing components or assemblies before they fail, effectively shifts work from USD to SD. Although replacement times remain the same, the access times and restart times are only counted once. This results in an overall reduced total down time (USD and SD) if done properly.Furthermore, the lower USD times result in a more reliable machine to operate.

$$\text{Av-impact} = \sum_{i=1}^{n}(\frac{repairtime}{failrate})_i[hr/year] \tag{A.3}$$

Serviceability

The serviceability of the machine is of influence on the MTTR. The time needed for diagnosis and replacement is part of the serviceability. Shorter diagnose, replacement and start-up time bring down the time to repair a failure, but not the number of times a failure occurs. As seen in Equation A.2, reducing repair times is partially a solution to reduce the MTTR.

A.1.2 Fixed KPIs

KPIs related to the performance attributes and productivity of the machine are fixed. These KPIs influence the entire wafer scanner of they are not met. If the WH can not deliver the required amount of wafers per hour, it becomes the bottleneck regarding wafer throughput, affecting the entire machine.

Throughput

The number of wafers processed in a hour, is one of the most important KPIs. If the quality KPIs are not achieved, the number of wafers processed is irrelevant. Only if the other KPIs are met, the throughput can be increased. And if the throughput is increased, the other KPI need still to be met. The main KPIs regarding throughput are the wafers per hours [wph]. Second is the wafer exchange time (WEX), this is the time that is available for the WH module to load and unload the wafer stage during a cycle.

Positional Accuracy

Positional alignment of the wafer is one the main tasks of the wafer handler. The wafer provided by the track can have any orientation and can be off centre by $\pm 2mm$. The WH must orientate the wafer such that the wafer is positioned correctly for the lithography process. The positional accuracy KPIs are related to the rotational and lateral position of the wafer. In both x and y-direction the offset to the centre may be $7.5\mu m$. The allowed rotational offset is $75\mu rad$.

Temperature

The temperature KPIs are related to the temperature of the wafers provided to the wafer stage. The wafer must have a specific temperature with respect to the WS temperature in order for the lithography process to succeed (wafer must not expand or shrink during exposure). The wafers are provided by the track with a temperature between 21 and $23°C$. Besides alignment, the WH is responsible for conditioning the wafer temperature to $22°C$. The lithography process requires the wafers to be close to the target temperature. Temperature differences are therefore strictly bounded by the KPIs. The variation in wafer temperature (uniformity) is allowed to be $\pm 5.6mK$. The internal temperature difference (non-uniformity) is allowed to be $\pm 1.0mK$.

Contamination

Wafer contamination consists of chemical contamination and added particles. Chemical contamination is related to the deposit of atoms to the wafer surface. The added particles are related to deposits of molecules (e.g. dust). The maximum size of the molecules is limited to $40nm$ and the number of added particles is limited to 7.5 particles, on average, per wafer pass through the WH. In this project, chemical contamination is not taken into account. The way of manufacturing of the components has influence on the chemical contamination. Since the manufacturing of the components is not part of this project, chemical contamination is excluded from analysis.

A.2 TRIZ Contradictions Matrix

Figure A.2 shows the contradictions matrix taken from [26]. On the vertical axis (left of the matrix), the parameters to improve are depicted. Parameters not to get worse, are depicted on the horizontal axis. At the intersections of the parameters, the inventive principles suggested to solve the contradiction are listed. The corresponding inventive principles to the number is the matrix, are shown on the right side of the figure.

A.3 Balancing between non-fixed KPIs

The non-fixed KPIs can be influenced by changing the business process of the system. Due to the nature and interdependency of the non-fixed KPIs, balancing between the non-fixed KPIs is possible. Reliability and availability are a result of Scheduled Down time (SD) or maintenance and Unscheduled Down time (USD) or failure. By preforming preventive maintenance, USD can be decreased by prematurely replacing critical parts. SD will increase as a result, but can be planned in contrast to USD.

Introducing preventive maintenance, will decreases failures and thus increases the reliability at a cost of increased SD. Replacing components causes down time, the total sum of downtime (SD and USD) might increase due

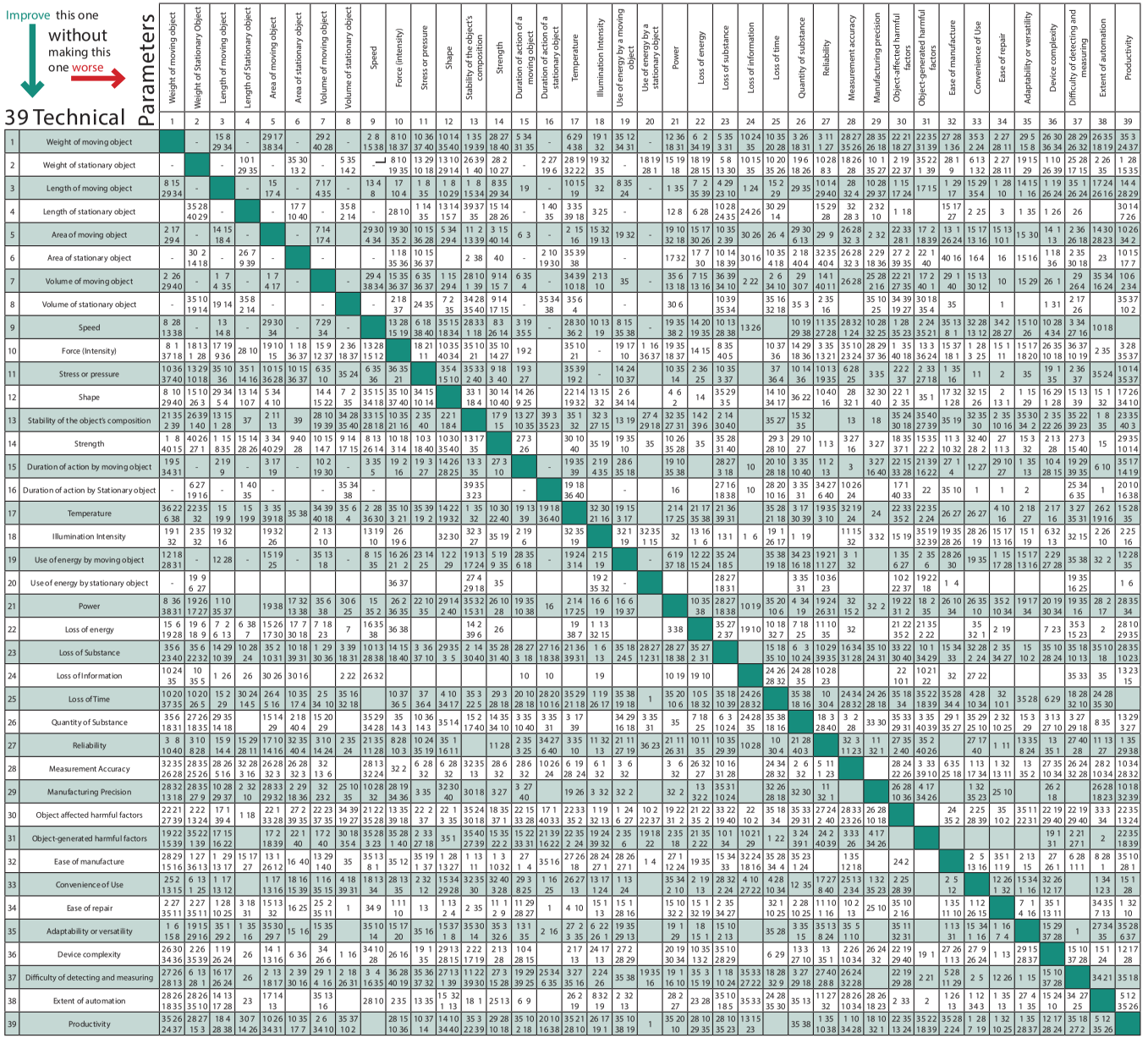


Figure A.2: TRIZ contradiction matrix [26]

40 Inventive Principles

1	Segmentation
2	Taking Out
3	Local Quality
4	Asymmetry
5	Merging
6	Universality
7	Nested Doll
8	Anti-Weight
9	Prior Counteraction
10	Prior Action
11	Cushion in Advance
12	Equipotentiality
13	The Other Way Round
14	Spheroidality - Curvature
15	Dynamics
16	Partial or Excessive Action
17	Another Dimension
18	Mechanical Vibration
19	Periodic Action
20	Continuity of Useful Action
21	Rushing Through
22	Blessing in Disguise
23	Feedback
24	Intermediary
25	Self-Service
26	Copying
27	Cheap Short-Living Objects
28	Replace Mechanical System
29	Pneumatics and Hydraulics
30	Flexible Membranes / Thin Films
31	Porous Materials
32	Colour Change
33	Homogeneity
34	Discarding and Recovering
35	Parameter Change
36	Phase Transition
37	Thermal Expansion
38	Accelerate Oxidation
39	Inert Environment
40	Composite Materials

to the replacement of components. Resulting in a decrease of availability, but parts of the downtime now can be planned. Shifting work from USD to SD creates variations with equal system architecture but different maintenance regimes. Each variation can be considered an alternative to the business process, whereby the degree of work shifted to SD is a conceptual design choice.

A.4 AHP calculations

For each pairwise comparison, a preference is determined according Table A.1. The result is stored in a $n \times n$ real matrix (with n the number of criteria or candidates). If criteria i is compared to criteria j and the decision is in favour of criteria i, the value of the comparison is stored in position (i, j). If the decision is in favour of j, the value is stored in position (j, i). When all pairwise comparisons are preformed, the other values within the matrix can be calculated such that $A_{ij} \cdot A_{ji} = 1$. Hence, Formula A.4 shows the relation between values within the matrix.

Value	Definition
1	Equal importance
3	Weak importance of one over another
5	Moderate importance of one over another
7	Strong importance of one over another
9	Extreme importance of one over another
2,4,6,8	Intermediate value between two adjacent judgements

Table A.1: Scale of relative Importance [42]

$$A_{ij} = \begin{cases} 1 & \text{for } i = j \\ \frac{1}{A_{ji}} & \text{for } i \neq j \end{cases} \tag{A.4}$$

$$V_k = \frac{v_k}{\sum_{i=1}^{n} v_i} \tag{A.5}$$

From the pairwise comparisons matrix, only the maximum eigenvalue and corresponding eigenvector are required. Additionally, the eigenvector needs to be normalized such that the sum of the values is equal to 1, as shown in (A.5). Where V_k is the normalized weight of criteria k and v_k is the original value of the eigenvector.

The (in)consistency is calculated as described by [31]. The consistency index (CI) has to be calculated first, using the maximum eigenvalue (λ_{max}) of the matrix with size ($n \times n$). The CI-formula is shown in (A.6).

$$CI = \frac{\lambda_{max} - n}{n - 1} \tag{A.6}$$

$$CR = \frac{CI}{RI} \tag{A.7}$$

The consistency ratio (CR) can be calculated as shown in (A.7). The RI value can be found in Table A.2, where n is the matrix size. The values of the RCI table are taken from [31].

n	1	2	3	4	5	6	7	8	9	10
RI	0.00	0.00	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.49

Table A.2: Random (Consistency) Index (RCI)

Consistency is achieved when $CR \leq 0.1$. CR values slightly over 0.1 (slightly inconstant) can be tolerated according to [31]. If inconsistency is not allowed, experience regarding a topic is not allowed. Experience

regarding a topic can increase of the CR value. Not allowing any margin within the CR value, would exclude knowledge and experience regarding a topic. Without the knowledge a non optimal outcome may be found and make the AHP method less effective.

Per KPI (in case of m KPIs), a matrix is created containing the results of the pairwise comparisons. This matrix has the same structure as the matrix in (A.4). For all matrices with comparisons, the normalized eigenvectors (w) are placed together in a general weight matrix as shown in (A.8). The matrix W (size $m \times n$) is multiplied with the normalized eigenvector of the criteria matrix V (size $n \times 1$), which results in a global weight vector W (size $m \times 1$). The candidate related to highest value of vector W, is the optimal design according the pairwise comparisons.

$$W = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m1} & w_{m2} & \cdots & w_{mn} \end{bmatrix} \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix} = \begin{bmatrix} W_1 \\ \vdots \\ W_m \end{bmatrix} \tag{A.8}$$

A.5 Remarks on the Delta DSM

The ΔDSM (or Delta DSM) is presented and used in several papers, like [36]. ΔDSMs presented in these papers are mostly symmetric of structure. The following can be detected in the ΔDSM of [36]:

- Eliminated elements are marked, but the corresponding eliminated dependencies are not not highlighted.
- Changed elements are marked, but it is not clear whether the corresponding dependencies are added or eliminated. This makes detecting the network changes in the ΔDSM harder.

Having a clear overview of the added, eliminated and changed dependencies (both before and after the change) can help to understand the system changes. Therefore, the presented format of the ΔDSM in this report uses an asymmetric format (schematically shown in Figure 3.3a). This format allows to:

1. Show eliminated elements and eliminated dependencies
2. Show added elements and added dependencies
3. Show changed elements
4. Show dependencies in both situations, so the situations can be detected.

Appendix B

Extensions to Chapter 4

B.1 Compiler Additions

The functionality of ESL is sufficient for creating MDMs. As presented in Wilschut [16] and Brady [9], weighted DSMs can be created using respectively the Pagerank algorithm [41] or Brady's method. To combine ESL with methods of ranking and quantification requires extension to the ESL compiler functionality. In addition to ranking algorithms, the implementation of subfiles and the visualization of the changeover-Delta DSM (presented in Section 3.4) is created.

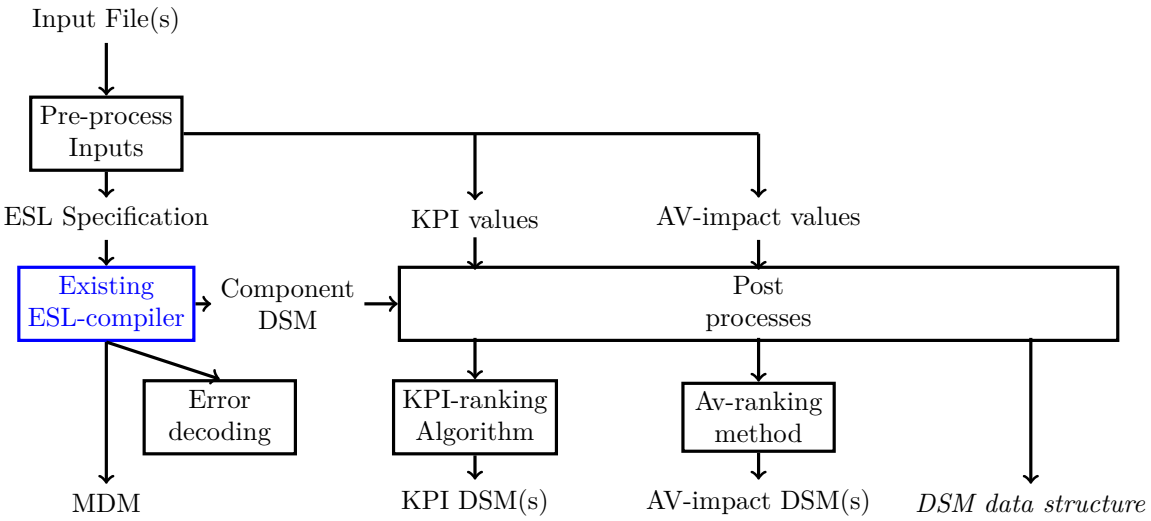


Figure B.1: The structure of the added ESL compiler functionality

In Figure B.1 the structure of the added ESL compiler functionality can be seen. The created extensions to the ESL compiler functionality can be seen as a toolbox. The toolbox is not integrated into the compiler, but created around the compiler with a pre-process and a post-process.

For overview and modifications to the decomposition tree, the ability to use multiple (ESL specification) files is created. A sub-branch of the decomposition tree can be replaced or closed (no further decomposition) by referring to a different file containing the desired ESL specification. Repairs in the code, due to programming mistakes can be located faster. Instead of a 3000 line file, the programmer can be directed to a 400 line sub file. This avoids the need to scroll up and down during debugging as errors are not provided in chronological order. The pre-process algorithm(s) preforms the following tasks:

- Combine sub-files into a single file, readable by the existing ESL compiler.
- Extract KPI values for each goal function.
- Extract AV impact values for each component.

For ranking and quantifying, there is chosen to define the values within the ESL specification file. The pre-process creates a correct ESL specification file for the ESL-compiler and stores the ranking values for use in the post-process. In case of a programming or language error in the ESL specification, the error provided by the ESL compiler is translated to refer to the corresponding file and line.

The post-process combines all ranking data with the component DSM derived by the existing ESL compiler. The KPI-ranking and AV-ranking algorithms calculate the values for the mappings can create the corresponding

figures. The component DSM is used as a basis for the ranked DSMs such that the RDSMs have the same structure and clustering as the component DSM. Additionally, the data is stored for usage in the CD-DSM.

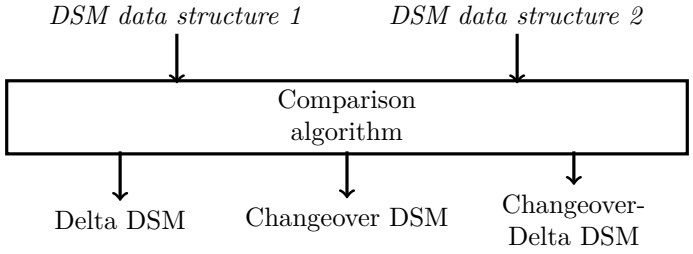


Figure B.2: structure of the DSM comparison algorithm

In Figure B.2, the structure of the DSM comparison algorithm is depicted. Using the DSM data structures of both system designs, the algorithm can create three types of comparison DSMs. It is important that the ESL specifications of both designs to be compared are processed by the ESL compiler and post-process of Figure B.1.

Based on the DSM-data structures, the algorithm searches for commodities, additions, changes and removals of elements and dependencies between the data structures. *DSM-data structure 1* is seen as the base structure and *DSM-data structure 2* as the alternative structure. Elements only existing in *DSM-data structure 2* are marked as added, elements only existing in *DSM-data structure 1* are marked as removed (with respect to *DSM-data structure 2*). Visualization of the results (the comparison DSMs) is included in the algorithm.

B.2 Clustering Algorithm Remarks

The Bus detection of the clustering algorithm [17], determines whether a element becomes a bus element based on a threshold. The threshold can be altered by a variable γ. The threshold determines the number of elements linked to an element based on the binary existence of dependencies.

During the determination of the bus size, it may therefore occur that a small change in γ will lead to a large increase or decrease in the number of bus elements. This is due to the equal number of element connected by dependencies. The result can be that a bus cluster of a certain size is not possible. This restricts the possibilities to create a DSM with that specific bus size.

The solution can be the addition of an additional variable in the clustering algorithm. The additional variable extends the threshold with the number of dependencies between an elements and the connected elements. The additional variable will be a (very) small factor to make a distinction between elements with a equal number of connected elements.

B.3 Automatic KPI derivation

In Listing B.1, a proposal for the automatic KPI derivation from ESL subclauses is presented. First a definition of the KPI is shown, in this case temperature. The (default) average value is defined, and the deviation with the corresponding KPI values. The temperature is defined in Celsius, this can be used for identification of temperature values. The variable `wafer-temp` can be used for the detection of statements related to a KPI.

In the goal-requirement definitions the boundaries of the temperature are given for the goal-requirement. In the case that `c-2` is not defined, the predefined average value will be taken. From `c-0` and `c-1` can be derived that the maximum deviation is equal to 1 degree Celsius. Therefore, the temperature KPI for `gfrW-w0` or variable `Wafer-TK-in` get assigned a value of 9 due to the allowed deviation. Similar to the goal requirement, such KPI values can be defined for the components (this is not shown in Listing B.1).

The second KPI definition is for availability impact. The within `AV-impact` there is no average or KPI value definition. This can be used for the direct assignment of values to components. So, instead of a pre-defined

KPI value (like the temperature KPI), a value of 1.5 is stored as KPI value for component TK.

```
define temperature{Celsius}
  average = 22 # used if not defined
  deviation:
    0.005 =< 1 # if deviation is smaller or equal to 0.005, assign value 1
    0.3 =< 3 # if deviation is smaller or equal to 0.3, assign value 3
    0.3 > 9 # if deviation is larger then 0.3, assign value 9

define AV-impact{hours}
  empty

define variable
  wafer-temp is a temperature

components
  # component definition(s)
  TK is a track with arguments
    wafer-TK-in, wafer-TK-out, ...
    ASP-ssig, ARR-ssig, ...
    TRR-ssig, TSA-ssig, ...
    tk-wh-alignment
    with subclause
    c-0: AV-impact is 1.5 hours
    end


goal-requirements
  # goal requirement with respect to wafer transport
  gfrW-w0: TK must deliver wafer-TK-in to WH with subclauses
    c-0: wafer-temp must be at least 21 Celsius
    c-1: wafer-temp must be at most 23 Celsius
    c-2: wafer-temp is nominal at 22 Celsius
    end
```

Listing B.1: component and goal requirement declarations with subclauses

Appendix C

Extensions to Chapter 5

C.1 Clustering variables

In Table C.1, the used variables for the clustering algorithm are presented. For each layer of decomposition, the variables are presented such that the clustering can be recreated.

Level of Decomposition:	1	2	3
α (expansion coefficient)	2	2	2
β (inflation coefficient)	3.5	3.5	3.2
μ (evaporation constant)	2.0	1.5	4.8
γ (Bus threshold)	1.3	1.3	1.6

Table C.1: Values used for clustering per level of decomposition.

C.1.1 1st and 2nd level

In Figure C.1, both the DSMs of the first and second level of decomposition can be seen. The first level of decomposition shows that the WH is fully depended of the WS for all supplies. Only a solid material flow and a spatial measure exist between the WH and TK. These are the wafer exchanges and the alignment for handover of the wafers.

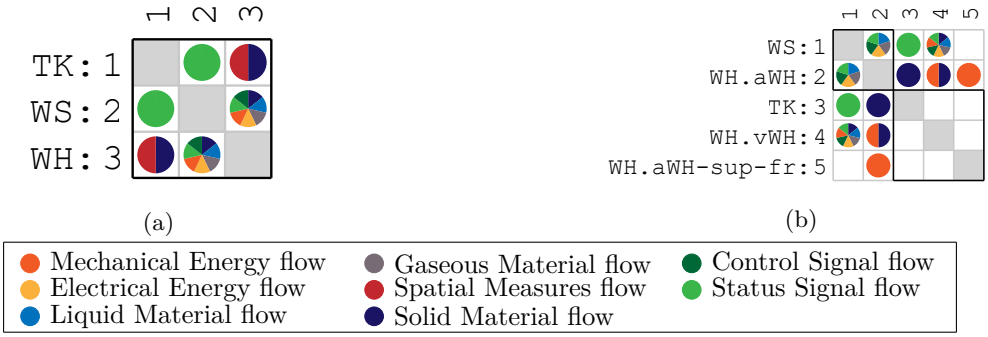


Figure C.1: Component DSMs, 1st and 2nd level of decomposition

C.2 KPI projections

In the Figures C.2, C.3 and C.4 the KPI projections of respectively the temperature, alignment and contamination are shown. More information regarding these figures can be found is Chapter 5.

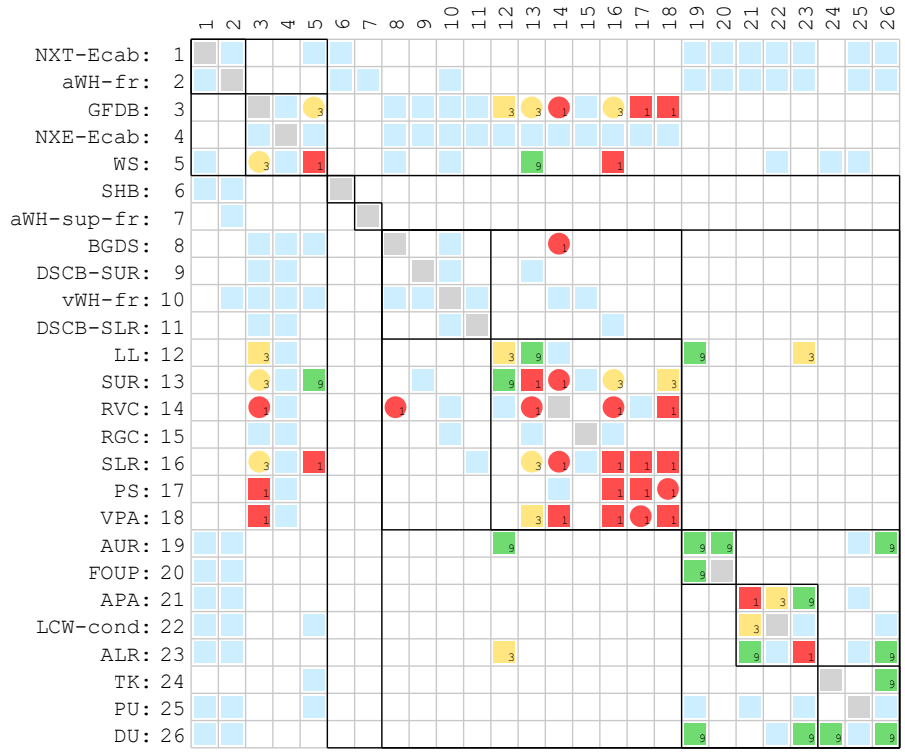


Figure C.2: Order of magnitude of the Temperature, 3rd level of decomposition

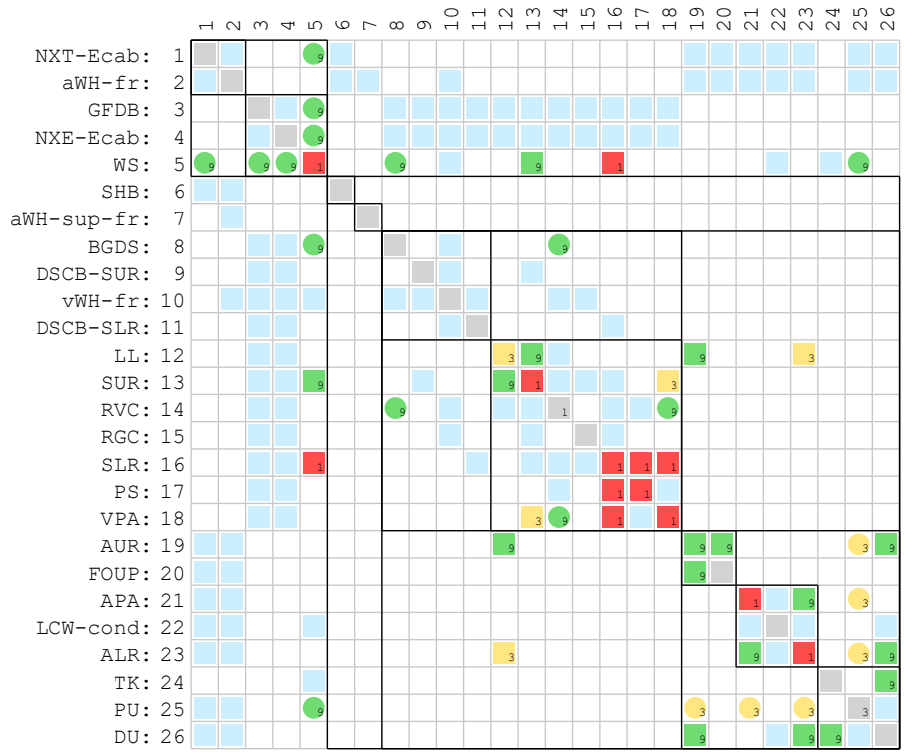


Figure C.3: Order of magnitude of the Alignment, 3rd level of decomposition

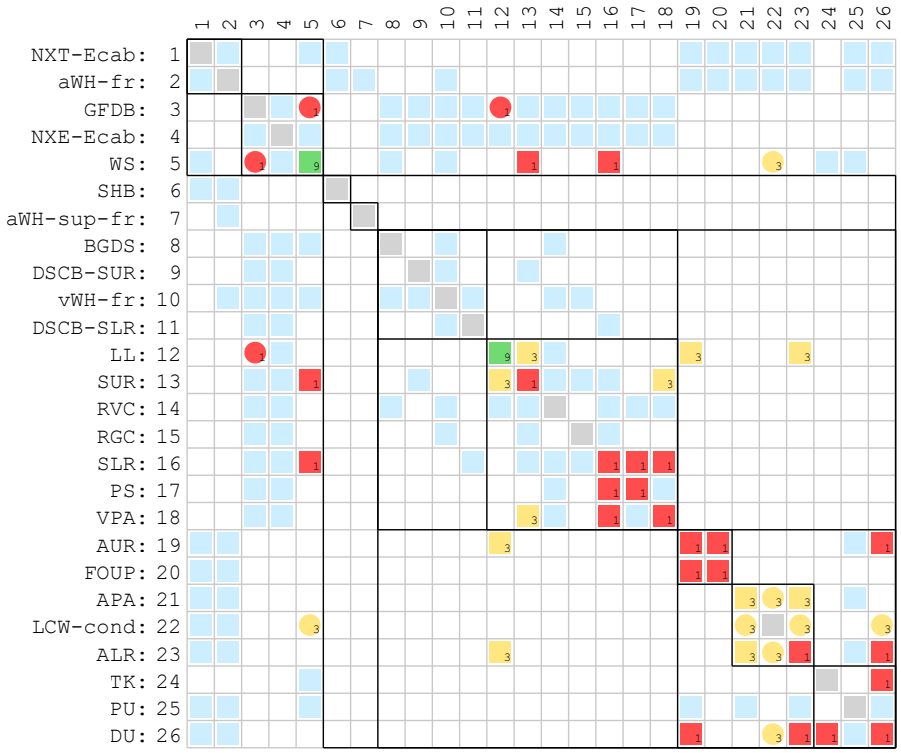


Figure C.4: Order of magnitude of the contamination, 3rd level of decomposition

Appendix D

Extensions to Chapter 6

D.1 Architectural changes

The following system architecture changes are applied to the original system architecture:

- Segmentation of the LL
  - LL1 created for input path
  - LL2 created for output path
  - Wafer flow changed to: ALR → LL1 → SLR
  - Wafer flow changed to: SUR → LL2 → AUR
- Segmentation of the RVC
  - RVC1 created for input path components
  - RVC2 created for output path components
  - SLR placed in RVC1
  - VPA placed in RVC1
  - LL1 placed in RVC1
  - PS placed in RVC1
  - LL2 placed in RVC1
  - SUR placed in RVC2
  - Wafer flow changed to: LL1 → SLR → VPA → SLR → WS
  - Wafer flow changed to: WS → SUR → LL2
- LCW flow redirection
  - LCW2 flow (removed): GFDB → RVC → GFDB
  - LCW2 flow (added): GFDB → RVC2 → SUR → RVC2 → GFDB
  - LCW3 flow (removed): GFDB → RVC→ SLR→ RVC → VPA→ RVC → SUR→ RVC → GFDB
  - LCW3 flow (added): GFDB → RVC 1→ SLR → RVC1 → VPA → RVC1 → GFDB
- RCW flow redirection
  - RCW5 flow (removed): GFDB → LL → GFDB
  - RCW5 flow (added): GFDB → LL1 → GFDB
  - RCW8 flow (removed): GFDB → LL → GFDB
  - RCW8 flow (added): GFDB → LL2 → GFDB

The segmentation of the LL and RVC, result into an input path and an output path. All relevant components are placed in the correct RVC and the wafer paths are defined.

The LCW (high accuracy Lens Cooling Water) and RCW (low accuracy Lens Cooling Water) both provide heating and cooling to the various components. The segmentation of the LL and RVC results in changed flows as well. The size of the flow is taken into account, as well as the type of flow (LCW or RCW). The changes LCW flows relate to the component heating and cooling. So the wafer is not directly cooled or heated by these flows. Additions to the flow paths are done such that both RVCs and LL are independent of each other in terms of liquid material (water) flows.

D.2 Clustering variables

In Table D.1, the used variables for the clustering algorithm are presented. For each layer of decomposition, the variables are presented such that the clustering can be recreated.

Level of Decomposition:	1	2	3
α (expansion coefficient)	2	2	2
β (inflation coefficient)	3.5	3.5	4.0
μ (evaporation constant)	2.0	1.5	3.0
γ (Bus threshold)	1.3	1.3	2.0

Table D.1: Values used for clustering per level of decomposition.

D.3 KPI projections

In the Figures D.1, D.2 and D.3 the KPI projections of respectively the temperature, alignment and contamination are shown. More information regarding these figures can be found is Chapter 6.

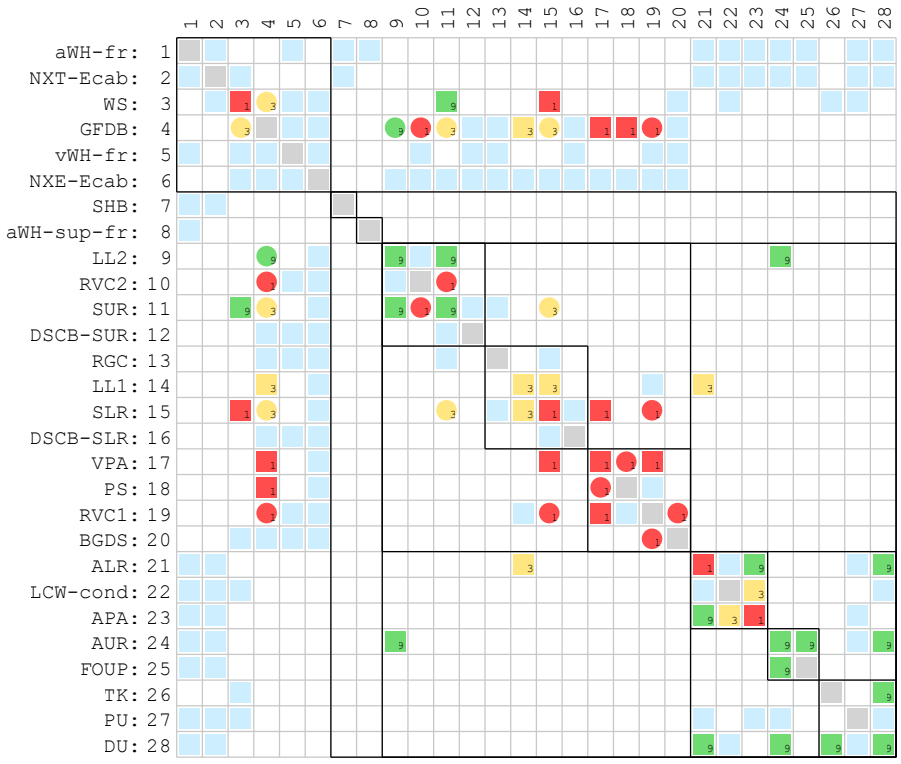


Figure D.1: Order of magnitude of the Temperature, 3rd level of decomposition

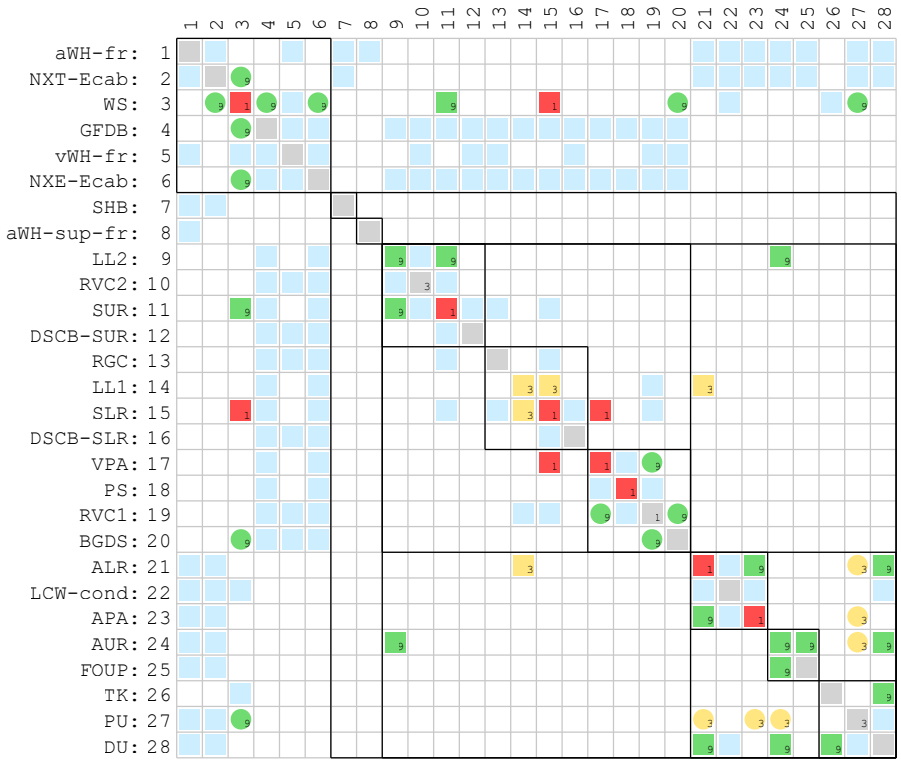


Figure D.2: Order of magnitude of the Alignment, 3rd level of decomposition

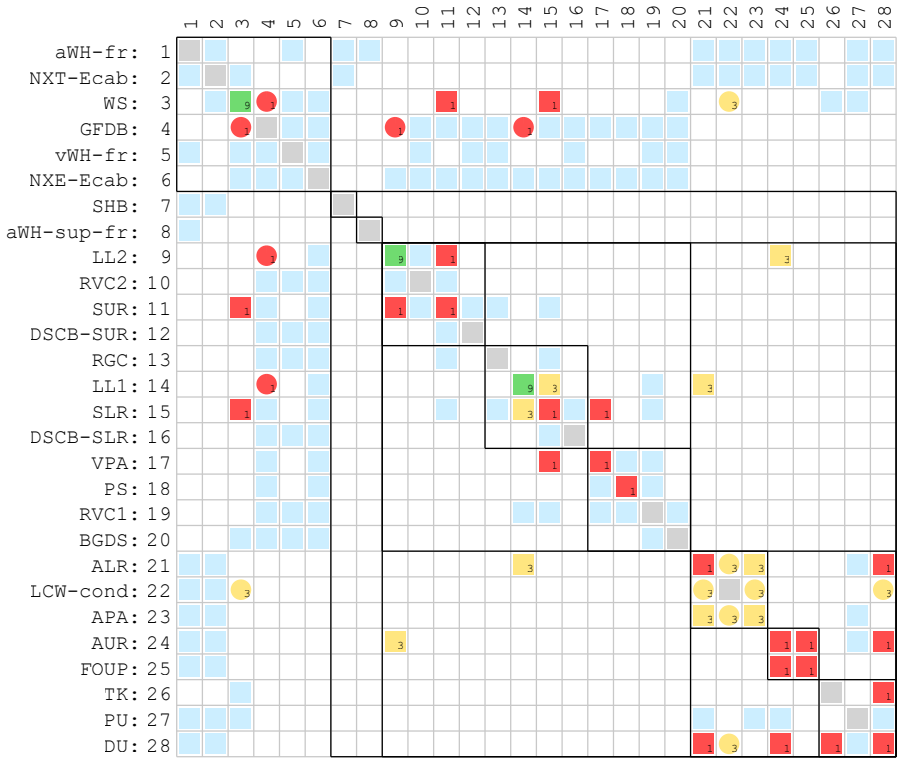


Figure D.3: Order of magnitude of the contamination, 3rd level of decomposition

Appendix E

Extensions to Chapter 7

E.1 KPI figures

In the Figures E.1 and E.2 the KPI comparisons of respectively the throughput and contamination are shown. More information regarding these figures can be found is Chapter 7.

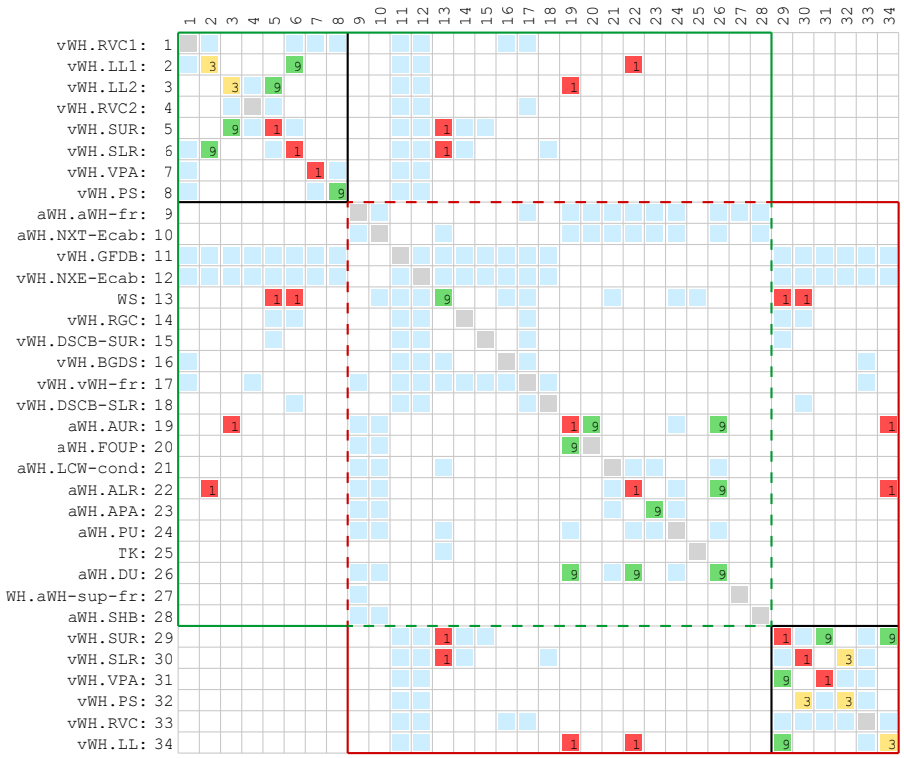


Figure E.1: Changeover RDSM of the Throughput, 3rd level of decomposition

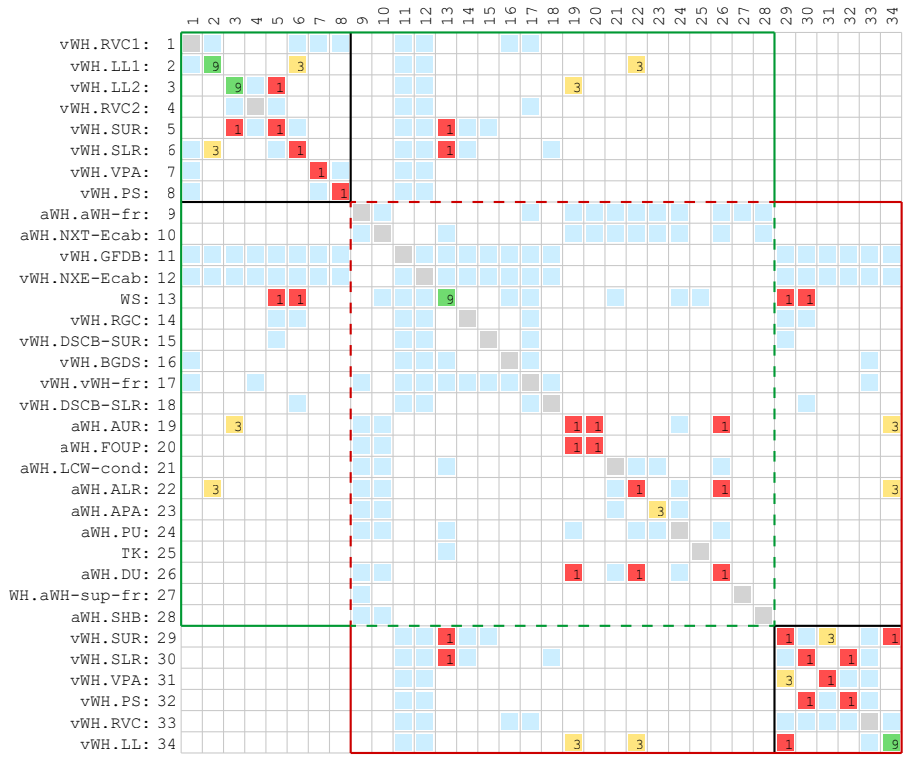


Figure E.2: Changeover RDSM of the contamination, 3rd level of decomposition