Logistics modeling and analysis for chuck exchange using Product Line Engineering

Citation for published version (APA):

Document status and date:
Published: 24/10/2018

Publisher Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher’s website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

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

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

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.
Logistics Modeling and Analysis for Chuck Exchange using Product Line Engineering

Berihun Fekade Yimam

Where innovation starts
Logistics Modeling and Analysis for Chuck Exchange using Product Line Engineering

Eindhoven University of Technology
Stan Ackermans Institute / Software Technology

Partners

ASML
ASML Netherlands B.V.

TU/e
Technische Universiteit Eindhoven
University of Technology

Steering Group
Berihun Fekade Yimam
Umut Uyumaz
Julien Schmaltz
Yanja Dajsuren

Date
October 2018

Document Status
Public

SAI report no.
2018/060

The design described in this report has been carried out in accordance with the TU/e Code of Scientific Conduct.
Abstract

ASML is developing a Wafer Logistics Specification and Analysis Tool (WLSAT) framework and tools allowing controllers to be formally specified, automatically analyzed and even optimized. This methodology has already been applied successfully to model and analyze the input and output paths of the wafer flow logistics. This graduation project enhances the WLSAT methodology for modeling the behavior of Chuck Exchange (CEX) sequence.

Keywords

Chuck Exchange, Modeling, DSL, WLSAT, ASML, TU/e, Software Technology, PDEng

Preferred reference


Partnership

This project was supported by Eindhoven University of Technology and ASML Netherlands B.V..

Disclaimer

Reference herein to any specific commercial products, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the Eindhoven University of Technology or ASML Netherlands B.V.. The views and opinions of authors expressed herein do not necessarily state or reflect those of the Eindhoven University of Technology or ASML Netherlands B.V., and shall not be used for advertising or product endorsement purposes.

Disclaimer Liability

While every effort will be made to ensure that the information contained within this report is accurate and up to date, Eindhoven University of Technology makes no warranty, representation or undertaking whether expressed or implied, nor does it assume any legal liability, whether direct or indirect, or responsibility for the accuracy, completeness, or usefulness of any information.

Trademarks

Product and company names mentioned herein may be trademarks and/or service marks of their respective owners. We use these names without any particular endorsement or with the intent to infringe the copyright of the respective owners.

Copyright

Copyright © 2017. Eindhoven University of Technology. All rights reserved. No part of the material protected by this copyright notice may be reproduced, modified, or redistributed in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the Eindhoven University of Technology and ASML Netherlands B.V..
Foreword

The chuck exchange sequence is in the hearth of ASML’s wafer scanners. The chuck exchange sequence is crucial for logistics and throughput performance of the systems. Over the years, the sequence adopted new behavioral variations to optimize the system performance and became complex and hard to test. ASML looks for opportunities to use domain specific languages and model driven engineering to simplify the design and validation of complex problems like chuck exchange.

In this thesis, Berihun describes how he extended Wafer Logistics Specification and Analysis Tool (WLSAT), MDE based tool to model wafer logistics within ASML’s wafer scanners, to model chuck exchange sequence and its variations. His contribution made it possible to bridge the gap between the functionality provided by WLSAT and the functionality needed for modeling chuck exchange and all its variations. With extensions made by Berihun, WLSAT tool supports modeling different variations of chuck exchange, generating timing and critical path analysis for given chuck exchange variation, and code generation. Berihun showed with his proof of concept that validation of a chuck exchange design can be done at design time and development cycle can be shortened.

Umut Uyumaz, PDEng
Software Architect, ASML
Veldhoven, 26th September 2018
Preface

This design report represents the results of the graduation project for the Software Technology (also known as Ontwerpersopleiding Technische Informatica (OOTI), Dutch) post-master program at Eindhoven University of Technology.

The project titled “Logistics modeling and analysis for chuck exchange using Product Line Engineering” was executed under the supervision of ASML Netherlands B.V. in Veldhoven. The project is implemented in ten months and is conducted in the Embedded Software Department within the Hoods team. The goal of the project is to model the Chuck Exchange sequence using an in-house Model Driven Engineering tool and enhance the tool with the new domain behaviors.

The report is intended for readers with technical background in system modeling or software engineering, but it also targets readers with nontechnical background, such as managers. In general, readers should refer to Chapters 1 and obtain the general information about the project to grasp the contents of other chapters.

Readers with a knowledge about modeling domain can read Chapter 2 and 8. Readers with an interest in the requirement, design and implementation of the solution should read Chapters 4 to 10. Project managers can refer to Chapters 3, 11, 12, and 13.

This report is a public version. A separate private version is available at ASML.

October 2018
Acknowledgements

I would like to express my sincere gratitude to all the people who have helped me, guided me, and supported me during this ten-month project. This project was an amazing experience and the success would not be possible without them.

First of all, I would like to express my deepest gratitude to Umut Uyumaz PDEng, my supervisor at ASML Netherlands B.V. You have been continuously supporting and guiding me through the project. I appreciate your consistent support, insightful observation, and direction during the project.

I would like to extend my thanks to my university supervisor, dr. Julien Schmaltz, for assessing my work, for his critical thinking, and for being an important pillar of my project steering group meetings. Additionally, I would like to thank our program directors, dr. Yanja Dajsuren PDEng; the management secretary, Desiree van Oorschot; and all the technical and professional development coaches.

I would like to acknowledge the help provided by Yuri Blankenstein, your help was crucial for me and for the success of the project. I would like to thank also Wim Roos, the domain expert who helped me to review the models. A special appreciation to my colleagues in the Immersion Hoods team, who made my time at ASML counted.

Furthermore, I am thankful to my fellow colleagues from the Software Technology, Mechatronics, and Automotive PDEng programs for sharing our experiences during the two years of our program.

Last but not least, I would like to thank my fiancé, Yodit Zewdie, for her enduring love, support, and encouragement.

Berihun Fekade Yimam
October 2018
Executive Summary

ASML is the world's leading provider of complex lithography systems for the semiconductor industry. In these machines, plates of silicon material, so called wafers, are transported using several robots. Optimal logistics controllers are required for this transport to make system throughput as high as possible. To optimize throughput, ASML is developing a Wafer Logistics Specification and Analysis Tool (WLSAT) methodology and the technologies allowing controllers to be formally specified, automatically analyzed and even optimized. This methodology has already been applied successfully to model and analyze the input and output paths of the wafer flow logistics.

The goal of the project was to enhance the design of WLSAT methodology with the concept of product line engineering to allow succinct specification of families of products. The enhanced methodology was able to model the Chuck Exchange (CEX) logistics. In addition, WLSAT is extended with a code-generation functionality to automatically derive implementation running on the TWINSCAN machine.

An enhancement was designed for the WLSAT methodology to support the modeling of the CEX domain and its code-generation. The results from the CEX model are checked against the machine traces. The timing analysis generated from the CEX model are within the required specification. In addition, the generated sample code follows a template from the existing implementation. However, the generated code only captures the main features from the CEX sequence that are modeled by the framework.

In conclusion, the design and implementation of the enhancement in WLSAT allows modeling of the new CEX domain behaviors. One of the CEX sequence is modeled using the enhanced tool to validate the timing analysis against machine traces. The order of the sequence and the execution time of actions are validated against the traces. It is recommended that ASML continue to use the enhanced WLSAT framework to model the CEX sequences and get earlier prototype analysis results.
Table of Contents

Foreword .................................................................................................................. v
Preface ....................................................................................................................... vii
Acknowledgements ................................................................................................... ix
Executive Summary ................................................................................................... xi
Table of Contents ..................................................................................................... xiii
List of Figures ........................................................................................................... xv
List of Tables ............................................................................................................ xvii

1. Introduction ........................................................................................................... 1
   1.1 Context ............................................................................................................. 1
   1.2 Chuck Exchange .............................................................................................. 2
   1.3 Wafer Logistics Specification and Analysis Tool ............................................. 2
   1.4 Limitation of WLSAT ..................................................................................... 3
   1.5 Outline ............................................................................................................. 4

2. Background ............................................................................................................ 5
   2.1 Introduction ..................................................................................................... 5
   2.2 WLSAT framework ......................................................................................... 5
   2.3 Product Line Engineering .............................................................................. 9
   2.4 WLSAT and CEX ......................................................................................... 10

3. Stakeholder Analysis ............................................................................................ 11
   3.1 ASML Netherlands B.V. ............................................................................... 11
   3.2 Eindhoven University of Technology (TU/e) ................................................. 12
   3.3 Stakeholder analysis map ............................................................................. 13

4. Problem Analysis .................................................................................................. 15
   4.1 Problem description ...................................................................................... 15
   4.2 Envisioned solution ...................................................................................... 16
   4.3 Project scope ................................................................................................. 17

5. Domain Analysis ................................................................................................ 19
   5.1 Introduction ................................................................................................... 19
   5.2 Chuck Exchange (CEX) ............................................................................... 19
      5.2.1. Scan types ........................................................................................... 21
      5.2.2. Scan options ....................................................................................... 21
      5.2.3. Conditions and defectivity switches ..................................................... 22
   5.3 CEX Logical Sequence ................................................................................. 23
   5.4 CEX types ..................................................................................................... 25

6. Feasibility Analysis ............................................................................................... 27
   6.1 Issues and Challenges ................................................................................... 27
   6.2 Risks .............................................................................................................. 27

7. System Requirements ........................................................................................... 29
   7.1 Requirements ............................................................................................... 29
      7.1.1. Functional requirement ..................................................................... 29
      7.1.2. Non-Functional requirement .............................................................. 30
   7.2 Design criteria ............................................................................................... 30

8. System Design & Implementation ......................................................................... 31
   8.1 Introduction ................................................................................................... 31
   8.2 Enhanced WLSAT design ............................................................................. 31
      8.2.1. Machine specification ....................................................................... 32
      8.2.2. Activity specification ....................................................................... 32
      8.2.3. Code generation ............................................................................... 33
      8.2.4. Symbolic position with PLE ............................................................... 34

9. Deployment ........................................................................................................... 37
   9.1 Deployment view ........................................................................................... 37

10. Verification & Validation ..................................................................................... 39
List of Figures

Figure 1 – TWINSSCAN machine ................................................................. 2
Figure 2 – WLSAT vs CEX activity sequences ........................................... 3
Figure 3 – Relation of Directed Acyclic Graphs with activity sequence .......... 6
Figure 4 – Symbolic positions .................................................................. 7
Figure 5 – Movable action representations ............................................... 7
Figure 6 – Code snippet for action .......................................................... 8
Figure 7 – Activity sequence diagram .................................................... 8
Figure 8 – Gantt chart ............................................................................ 9
Figure 9 – Sample feature diagram ........................................................ 9
Figure 10 – Stakeholder analysis map ..................................................... 13
Figure 11 – CEX design and implementation flow .................................... 15
Figure 12 – Wafer Logistics of NXT Twinscan Machine ......................... 19
Figure 13 – Measure and Expose side in Twinscan Machine .................... 20
Figure 14 – Scan types .......................................................................... 21
Figure 15 – ALAP/ASAP scan types ....................................................... 22
Figure 16 – Sample CEX sequence ....................................................... 23
Figure 17 – CEX mechatronics sequence .............................................. 24
Figure 18 – Trajectory of the M2E and E2M chucks ................................. 24
Figure 19 – WLSAT’s current component diagram .................................... 31
Figure 20 – A snippet of the enhanced machine specification language ....... 32
Figure 21 – Sample activity specification DSL ......................................... 32
Figure 22 – A branching activity sequence ............................................ 33
Figure 23 – The code generation process ................................................ 34
Figure 24 – Symbolic position with available variant ............................... 34
Figure 25 – A prototype of the symbolic position with PLE ..................... 35
Figure 26 – Deployment view ............................................................... 37
Figure 27 – Flow of modeling CEX using WLSAT .................................... 39
Figure 28 – The variability points and variants for different CEX sequences .. 41
Figure 29 – Work-breakdown structure of the project ............................... 45
Figure 30 – Initial project plan ............................................................... 47
Figure 31 – Final project plan ............................................................... 48
List of Tables

Table 1 – Comparison of WLSAT and CEX domain ................................................. 10
Table 2 – List of stakeholders from ASML ............................................................. 11
Table 3 – List of stakeholders from TU/e ............................................................. 12
Table 4 – Functional requirements .................................................................. 29
Table 5 – Non-Functional requirements ............................................................ 30
Table 6 – Resources and the associated peripherals ............................................ 40
Table 7 – Final project plan ............................................................................. 49
Table 8 – Risk tasks involved with the project ..................................................... 50
1. Introduction

This chapter gives a brief introduction to the project’s context and the domain in which the project is conducted. It also gives an outline that shows the structure of this design report.

1.1 Context

ASML is a world leader in the manufacture of advanced technology systems for the semiconductor industry. ASML offers an integrated portfolio for manufacturing complex integrated circuits (also called ICs or chips). It designs, develops, integrates, markets and services photolithography machines used by customers – the major global semiconductor manufacturers – to create chips that power a wide array of electronic, communications and information technology products[1]. ASML continuously improves the manufacturing process by shrinking line widths (reduced resolution), thereby enabling customers to reduce the size and cost to add more functionality to future generations of integrated circuits. In addition, the finer line widths allow electricity to move across the chip faster, improving the chip's performance.

The machines developed by ASML, which are called TWINCAN, use a photolithography system that involves creating a circuit pattern on the silicon wafer by exposing the pattern and the wafer with a light source. The image of the pattern is focused through a projection lens onto the silicon wafer. This requires measuring of the wafer before exposing it so that the pattern is printed at the right position to eventually have multiple layers of pattern on the wafer. Thus, measuring and exposing of the wafer are the two important processes involved in the system.

In photolithography machines, plates of silicon material (wafers), are transported, conditioned, aligned, and exposed using several robots. Optimal logistic controllers are required for this transport to make system throughput as high as possible. To optimize throughput, ASML has developed a Wafer Logistics Specification and Analysis (WLSAT) methodology and tools allowing controllers to be formally specified, automatically analyzed, and even optimized. This methodology has already been applied successfully to model and analyze the input and output paths of the wafer flow logistics.

In the domain of wafer logistics, the Chuck Exchange (CEX) sequence is one of the operations performed within the machine. It moves two main pieces of hardware, which are called chucks, closer and positions them in a synchronized manner at a defined location. It is executed twice in the lifecycle of a wafer. The CEX sequence shares the common domain behaviors in the wafer logistics with some variations.

The goal of this project is to use an existing modeling language, which is WLSAT, to model the CEX sequence and enhance the tool to support the variation points with the new domain variations. A previous related work[6] to model the CEX shows that the WLSAT framework can model only partial actions. The need for making an enhanced model of the CEX sequence comes after this promising result.

The next sections give a brief description about the CEX sequence and the WLSAT framework. In Section 1.2 the Chuck Exchange domain is explained briefly. Section 1.3 discusses the WLSAT framework. Next, in Section 1.4, the current limitation of the WLSAT methodology is given, Section 1.5 gives an overview of the report structure.
1.2 Chuck Exchange

The sequence of CEX is one of the critical operations in the TWINSCAN machines. The CEX operation uses different hardware resources to perform one chuck swap. Based on the steps taken among different hardware resources in the CEX sequence, the throughput of wafers processed per hour is determined in the machine. Since neither of the wafers are getting processed during the CEX operation, timing to finish one sequence is an integral part of the machine throughput.

![Figure 1 – TWINSCAN machine](image)

*Figure 1 shows a TWINSCAN NXT machine, a Step-and-Scan (two-Wafer stage) system with dual-stage immersion lithography machine designed for volume production of 300-mm wafers[14]. It introduces a much-improved version of the existing dual-stage concept in which each stage can operate concurrently and independently. A two-wafer stage was originally designed to maximize throughput in the machine and it is the module responsible for carrying and positioning the wafer during its exposure and measure phases. The Wafer Stage also includes the moving part of the position module carrying the wafer (which is the chuck), other hardware resources, and various actuators and sensors. There are two chucks in the Wafer Stage so that the wafers are measured and exposed concurrently. In the Measure cycle various checks and alignments are done on the wafer and in the Expose cycle the reticle pattern is put on the wafer. After a wafer has been exposed and the other has been measured, a chuck swap is performed. This operation is called Chuck Exchange (CEX). This is the domain in which the project is conducted. A more detail analysis for the CEX sequence is given in the coming chapters.*

1.3 Wafer Logistics Specification and Analysis Tool

The Wafer Logistics Specification and Analysis Tool (WLSAT) is based on analysis model approach to specify requirements. WLSAT analysis models depicts user needs with a combination of diagrams and structured texts such as tables. The underlying architecture for creating this Model Driven Engineering (MDE) tool is the domain of wafer logistics in TWINSCAN machines[8]. In WLSAT, systems are decomposed into different components (concerning, e.g. peripherals, activities, settings, and sequences) and requirements regarding safety and timing. These components give an abstract representation for hardware and system behavior for the TWINSCAN machine. WLSAT is used to model all the deterministic operations of the logistics flow
of wafers as a set of activities. Given a set of activities, it is possible to construct an individual sequence that represents one valid logistics flow of a wafer.

The CEX domain is one of the steps in the wafer logistics path in the TWINSCAN machine. The hardware present in the domain of the CEX follows similar characteristics to move a wafer and change position when compared with the domain of wafer flow. In addition, the synchronization principle among different hardware is also similar in both wafer flow and CEX. The hardware/robots that carry the wafer in both domains move in a synchronous manner to move from one position to another with defined speed. These common behavioral similarities bring the idea for this project, which is to model the logistics of the CEX sequence using the WLSAT framework.

1.4 Limitation of WLSAT

The WLSAT framework models the logistics of wafer flow; however, it needs an enhancement to fully support the CEX domain. The WLSAT sequence of activities, which are synchronized movements within different hardware, follows same path to move a wafer from one position to another; however, the CEX sequence is determined based on conditional guards that are controlled by the user. An expression is created using the conditional guards that executes to one boolean value. Based on the selected expressions boolean value, the CEX is executed. Figure 2 shows the difference activity flow in the WLSAT specification language and in the CEX domain. In addition, models designed using the WLSAT does not have a code generation functionality in which implementations can be derived easily for sets of activities.

In WLSAT, the sequence of actions is synchronized with other action/s to satisfy safety requirements in the machine. Actions are executed sequentially in their order without leaving a time gap from their predecessors’ synchronized action; however, In the domain of CEX, synchronized actions can start at the same time or others with a delay to make their hardware peripherals to start or finish tasks within a defined time respectively. Such a domain variation exists in the CEX sequence and it will be an additional functionality to the WLSAT framework.

The above-mentioned limitations (conditional guards and code generation) with an enhancement to support the different kind of synchronization types will be the main contribution to existing WLSAT framework.
1.5 Outline

The structure of the report is given below:

- Chapter 2 (Background) introduces the WLSAT framework.
- Chapter 3 (Stakeholder analysis) states the stakeholders in this project, together with their goals and interests.
- Chapter 4 (Problem analysis) gives an insight into the problem that the project is trying to solve.
- Chapter 5 (Domain analysis) explains the domain in which the project is conducted.
- Chapter 6 (Feasibility analysis) lists the issues and risks expected at the beginning of the project and the mitigation strategies for each of them.
- Chapter 7 (System requirements) lists the functional and non-functional requirements.
- Chapters 8-10 (System design & Implementation, Deployment, and Verification & Validation) provides detailed explanation from design to its implementation and finally validation of results.
- Chapter 11 (Conclusions) lists the results, lessons learned, and experience gained during the project.
- Chapter 12 (Project management) gives detailed overview of the planning, risks, and milestones in the project.
- Chapter 13 (Project retrospective) looks back on the successfullness of the project as well as the evaluation of the design and design process.

■
2. Background

This chapter aims at introducing the concept of Model Driven Engineering (MDE) and providing background information on the WLSAT framework in relation with the CEX domain. The WLSAT modeling framework is used to model the CEX variation points in this graduation project.

2.1 Introduction

Many high-tech systems consist of numerous hardware and software components connected based on various interfaces. The size of such software systems evolves as new functionalities are added. The increasing system size, functionality, and complexity also increase the effort needed for the integration and testing phases. The need for earlier integration and testing of the behavior of the system through models before developing the real software is a new approach that makes the requirements be validated with ease. This new approach of modeling the behavior of the functional domain and testing requirements makes the development cycle less error prone and shortens the software development lifecycle. This new approach is addressed using Model Driven Engineering (MDE).

MDE involves a systematic use of domain models as essential artifacts throughout the software development cycle. MDE focuses on creating and exploiting abstract domain models for an application domain. At the core of MDE is the concept of domain-specific languages (DSL), which formalize the given application structure, behavior, and requirements for a business domain[15]. A DSL’s syntax is tailored closely to the business domain at hand so that the users will be at ease in modeling the system. In addition, using DSLs only requires knowledge about the problem domain, but not about the solution domain.

In the MDE vision of software development, models are the primary artifacts of development and the developers rely on computer-based technologies to transform models to implementation code for the systems. The major goal of the MDE approach is to produce technologies that shield software developers from the complexities of the underlying implementation platform[13]. MDE enables simulation and analysis, thereby resulting in an earlier identification of design defects than prototyping.

ASML uses a domain-specific language and model-driven engineering environment to model and generate an analysis and optimal logistics flow of wafers in ASML’s TWINSCAN machines. The system built using MDE methodology combines the concept of functional implementations that are later verified against the given requirements while constructing the specification language. One of the tools that is being used by ASML design engineers to model the logistics of wafers in TWINSCAN machines is the Wafer Logistics Specification and Analysis Tool (WLSAT). In this chapter, we introduce the functionalities of the tool in modeling the logistics behavior of wafer flow.

2.2 WLSAT framework

The Wafer Logistics Specification and Analysis Tool is being developed by ASML in collaboration with TNO-ESI and TU/e. WLSAT is designed to be used to model a much wider range of logistics systems[8], such as the wafer handler and wafer stage subsystems. It has both textual expression and graphical representation for specifying requirements.

In this modeling framework, System actions are modeled as activities. Activities are defined as Directed Acyclic Graphs (DAGs), which consists of a set of actions...
executed on peripherals and set of dependencies among them. The system requirements are specified in terms of these actions. Then the requirements in turn enforce the product safety based on their specification. Figure 3 shows the relation between DAG and sequence of actions executed per peripheral. The DSL representation in (b) describes the set of actions per peripheral and their sequence. Each action is represented as a Node and their sequential order is represented as Edge as shown in (a).

WLSAT has several graphical (such as graphs, charts, and tables) representations and textual representation language, which is a domain-specific language, developed using the Xtext framework. Xtext is a framework for the development of programming languages and domain-specific languages[16]. The graphical representation for each of the layers is developed using the graphical modeling workbench, Sirius, that allows users to create custom graphical representation by leveraging the Eclipse Modeling technologies.

![Directed Acyclic Graphs with activity sequence](image)

There are four textual specification languages in WLSAT. These are the machine, activity, settings, and dispatching languages. A system user has to create all the four languages to exploit the benefit of the tool. Machine specification language is the root of the model in WLSAT[8][9]. It contains resources and peripheral types. Resource corresponds to resources in a machine, e.g., a robot. Each resource contains peripherals, e.g., clamp of a robot. Each peripheral refers to Peripheral type that corresponds to the type of the peripheral, e.g., linear motor is of a type motor. Each peripheral type defines the action a peripheral can perform using action type. These actions are typically called 'simple actions', which are defined as actions which are atomically executed and consumes a particular amount of time.

The positions a peripheral can take in its coordinate system is represented by Symbolic Positions contained in its Peripheral. The allowed moves by a peripheral between the symbolic positions are represented by paths. Paths can be Uni-direction, Bi-directional, or Full-Mesh (path where all moves from a location to another location are allowed).
Once the Machine specification language is constructed to represent the physical machine, the movable hardware peripherals defined within a resource can be graphically visualized. In Figure 4, the symbolic position diagram for the target machine shows specific information about the positions and motion profiles within each of the allowed paths. This helps to understand how the movable peripherals move from one symbolic position to another. Figure 4 depicts how movable peripheral moves within the defined symbolic positions and which speed profile name is being used for the movement.

The activity specification language contains an activity sequence for peripherals defined in the machine specification language. An activity that is performed by a peripheral can be prep, scan, move (to-point scan), and simple action. These activities have a defined way of expression in the textual specification language. The prep, move (to-point scan), and scan are movable actions and they require a start and end symbolic position. However, simple actions have only a time to perform an action (for example: clamp and unclamp a wafer). The move action, which is a to-point scan, makes the peripheral be at standstill at the destination symbolic position. On the other hand, a scan action moves the peripheral with a constant speed within the start-end positions. The third type of movable action, which is the prep phase, makes peripherals be descending/ascending to have a specific speed at the destination symbolic position. Prep phase is always a predecessor for a scan action so that it adjusts the speed required by the scan.

The movable action types with respect to speed at a given time is shown in Figure 5. In the figure, a simple scan with its prep phase is shown. A to-point scan is executed after the scan phase to bring the hardware into standstill position. The specification language for the movable and simple actions is shown within the code snippet in Figure 6.
The activity defined in an activity specification language has a graphical representation as shown \textit{Figure 7}. The graphical activity diagrams are graphical representations of workflows of stepwise actions. In the context of WLSAT, the Activities module composes high level actions that a resource can perform into activities. The diagram shows these activities in their defined order sequentially. A resource is claimed first and executes the defined actions per peripherals and at the end, it is released. The brown blocks represent actions for peripherals and the long black bars represent a synchronization point among different peripheral actions.

The setting specification language contains all the numerical values that will be set for each symbolic position, motion profile, timing for simple actions. The setting values are the one used for calculating the time at execution.

The last group is the dispatching specification language. It refers the available activities and tries to schedule them in the given order. Scheduling an activity sequence generates a Gantt chart that represents the system execution in time. Within an activity sequence, one or more activities can be included. In \textit{Figure 8}, the same activity is scheduled to be executed twice with the name MoveRobot and MoveRobot2. These activities are scheduled as parallel as possible adhering to the claim/release plan. An activity
typically starts with claiming a Resource; hence this activity can only start if its prede-
cessor activity has released this Resource.

Figure 8 – Gantt chart

2.3 Product Line Engineering

In this subsection, the project introduces the basic concepts related to product line engineering. The methodology is used to capture similarities and differences among products.

In recent years Model Driven Engineering (MDE) has emerged as a paradigm that al-

ows dealing with software artifacts with a high level of abstraction. Product Line Engineering (PLE) is a way to engineer a portfolio of related products in an efficient manner, taking advantage of products’ similarities while managing their differences. The basic requirement for enabling PLE is managing variability among family of products.

Variabilities of a product are represented using a feature diagram [5]. Figure 9 shows a typical example of a software product using a feature diagram. The model includes the variation point (VP) ‘security package’ and the two alternative variants (V) ‘basic’ and ‘advanced’. These variants come with different packages. Selecting the ‘basic’ package also selects the variant ‘motion sensors’ of the variation point ‘intrusion detection’ and the variant ‘keypad’ of the variation point ‘door locks’. If the ‘advanced’ package (variant) is selected, the variants ‘camera surveillance’ and ‘fingerprint scanner’ are chosen.

Figure 9 – Sample feature diagram

The Feature based products take the description of the variation out of all those different places and centralizes it using a well-defined, common language of features. The
features describe product variation which are the distinguishing characteristics. In this graduation project, we analyzed the variabilities within the CEX sequence for selected variability points and their respective variants.

The CEX sequence is performed in every wafer life cycle and there is multiple type of sequences which are designed only for one machine. These sequences are executed similarly but different from one another in the number of steps performed and the time taken for the execution. The differences among them can be expressed using the product line engineering principle while maintaining their similarity. The variability points among them were chosen with their variant from the CEX sequence design document. This brings another dimension to the WLSAT framework and as a result, users can use one model of the CEX sequence to generate the timing analysis based on selected CEX types.

2.4 WLSAT and CEX

The domain of CEX uses different terminologies in comparison with the WLSAT language to express similar behaviors within the TWINSCAN machine. The movable peripherals behavior shown in Figure 5 are common for WLSAT and CEX domain; however, the terms are expressed differently in WLSAT and CEX. Table 1 shows the relationship between the terms used in WLSAT and CEX.

<table>
<thead>
<tr>
<th>WLSAT domain</th>
<th>CEX domain</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Move -- to</td>
<td>To-point scan</td>
<td>Stops (stand-still) at the destination location</td>
</tr>
<tr>
<td>Move -- passing</td>
<td>Prep</td>
<td>Passes the destination with the current speed</td>
</tr>
<tr>
<td>Scan -- passing</td>
<td>Scan</td>
<td>Moves with a constant speed</td>
</tr>
<tr>
<td>Simple action</td>
<td>Simple action</td>
<td>Common for both and it takes time to complete an action</td>
</tr>
</tbody>
</table>

A detail domain analysis about the CEX sequence is described in Chapter 5. ■
3. Stakeholder Analysis

This chapter discusses the interests, concerns, and involvement of each stakeholder involved in the project. The main parties involved in the project are ASML Netherlands B.V. and Eindhoven University of Technology.

3.1 ASML Netherlands B.V.

ASML Netherlands B.V. is responsible for providing the domain knowledge that is related to the Chuck Exchange behavior in NXT systems. The team members that are involved in the current design and implementation of Chuck Exchange and Wafer Logistics Specification and Analysis tool are the main stakeholders from ASML. ASML also provides the necessary materials and work place for the trainee to conduct the project. Stakeholders’ interest and involvement to the project are described in Table 2.

<table>
<thead>
<tr>
<th>Stakeholder</th>
<th>Position</th>
<th>Role/Interest</th>
<th>Involvement</th>
</tr>
</thead>
</table>
| Umut Uyumaz | Supervisor (ASML) | • Main contact person from ASML  
• Provide detailed guidance on the project for the trainee  
• Agree on project scope and requirements with the trainee  
• Evaluate project progress and review report  
• Interested to model the CEX sequence variation points using WLSAT and code generated from the models | Throughout the project via regular meetings, weekly progress meetings and monthly PSG meetings |
| Maurice Sloots | Group lead | • Approval of the project result  
• Ensures the project brings added value to the team and company | Through short update meetings whenever necessary |
| Jeroen Hoefnagels | Team lead | • Provides Chuck Exchange domain knowledge of NXT systems  
• To have a modeling tool that captures the Chuck Exchange sequence behavior for immersion machine types | Through weekly team meetings |
| Yuri Blankenstein | WLSAT team member | • Provides the trainee with WLSAT related resources  
• Evaluates the design for extending the WLSAT framework  
• He wants to have an extensible WLSAT tool which includes new functionality to support CEX sequence | Through on demand-based meetings |
3.2 Eindhoven University of Technology (TU/e)

The Eindhoven University of Technology is responsible for the educational aspect of the project. TU/e provides supervisor/s to control the educational aspects for the trainee and provides expert knowledge on relevant aspects of the problem domain. The educational aspects of the project are related to the standards that validate the software design methodology, project plan, and risk management procedures for the project. The stakeholders from TU/e are listed below in Table 3.

Table 3 – List of stakeholders from TU/e

<table>
<thead>
<tr>
<th>Julien Schmaltz, Associate Professor</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Position</strong></td>
</tr>
</tbody>
</table>
| **Role/Interest** | • Monitor project progress  
  • Ensures quality of the design and the scientific aspect of the project  
  • Guide the trainee with technical and non-technical skills  
  • Review the final report |
| **Involvement** | Throughout the project via regular meetings, progress meetings and monthly PSG meetings |

<table>
<thead>
<tr>
<th>Dr. Yanja Dajsuren</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Position</strong></td>
</tr>
<tr>
<td><strong>Role/Interest</strong></td>
</tr>
<tr>
<td><strong>Involvement</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Berihun Fekade Yimam</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Position</strong></td>
</tr>
</tbody>
</table>
| **Role/Interest** | • Analyze, design, and implement the project  
  • Manage the project  
  • Schedule regular meetings with supervisors  
  • Match project results with company and university standards |
| **Involvement** | Throughout the project |
3.3 Stakeholder analysis map

Based on their interest and the power to influence the project’s progress, a stakeholder analysis map is shown Figure 10.

Figure 10 – Stakeholder analysis map

Figure 10 shows how each of the stakeholders is compared among others based on how much they are interested in participating in the project versus the power with which they should influence the project. The interest axis shows how much detail the stakeholder wants to be informed throughout the project. The higher the interest value, the more detail the information needed and vice versa. On the other hand, the power axis describes how much the stakeholder decisions influence the project direction and progress.

The stakeholders placed in the above four quadrants describe how the communication method and content varies among them. For example, the stakeholders placed at the top-right (high interest/high power) need a detailed report on project status and demonstration of results through face-face meetings. On the contrary, the stakeholders positioned at the bottom-left (low interest/low power) are the domain knowledge experts for the CEX and they are interested in the result of the project and want to be informed occasionally such as on achieving a milestone. The stakeholders placed on top-left (high power/low interest) are the managers who control and view the project from a higher-level perspective. The other quadrant in the map, which contains stakeholders with high interest and low power, has the person who is a potential supporter with his knowledge in achieving the project scope.

The stakeholders that are placed on the right side of the map are the key players and show the highest interest in every aspect of the project. In addition, they are also high contributors to the project. I kept informing and consulting them on their area of expertise to help me achieve the goal of the project. However, the group of stakeholders that are placed on the left part of the plot engage in the project less often. They have interest on the result of the project and only provide high level guidance related with the project plan and risk. Most of them have also expertise in the CEX domain but not on the modeling framework. The crucial part of the project scope involves enhancing the WLSAT framework and these stakeholders are too new to the modeling domain to give input to the project. Throughout the period of this project, my plan was to inform and engage them as much as I could so that they have more interest in the domain my project involves and finally move to on the right-hand side of the map.
4. Problem Analysis

This chapter focuses on analyzing the current state of the Wafer Logistics Specification and Analysis Tool (WLSAT) in relation to modeling the Chuck Exchange (CEX) for NXT systems. It also pinpoints the functionalities in CEX that are not part of the WLSAT framework.

4.1 Problem description

The WLSAT modeling framework described in Chapter 2 can model deterministic operations of the CEX as a set of activities. Given a set of activities, it is possible to construct an individual activity sequence that represents one valid CEX execution. However, there are multiple ways by which the CEX can be executed. The action execution depends on system inputs such as motion profiles, positions, action completion time, and the executing/skipping of a specific action. These system parameters are controlled by predefined conditional guard value. Based on chosen conditional guards, the flow of the CEX sequence is determined. For every combination of conditional guards, the CEX timing varies while maintaining its requirement.

The goal of this graduation project is to extend the WLSAT methodology using the Product Line Engineering (PLE) concept to capture the variations that exist in families of products. Enhancing the WLSAT framework with these variations will help to model different CEX sequences.

The current procedure in designing a CEX sequence to deploying the code into a TWINSCAN machine is shown in the Figure 11 (a). Functional design Engineers are the those who are responsible for designing the CEX sequence. They use the MATLAB based timing analysis library for executing of each action in CEX and Microsoft Visio to draw sequences. Embedded software engineers implement this CEX sequence design and perform testing in the virtual TWINSCAN environment before they can start testing on the real machine. The design, development, and implementation phases take time to have a fully functional CEX code on the machine. In addition, the CEX sequence has many variation flows based on its type, peripherals involved, and conditional guards chosen for execution per one swap in the machine. It is cumbersome to try all the possible CEX sequence execution in the machine.

The models of various CEX sequences are implemented in Data Definition File (DDF). The DDF is an input to the software that controls the TWINSCAN machines and one
of the subsystems reads this file to execute the Chuck Exchange sequence. Based on the configuration on the TWINSCAN machine, a CEX sequence is executed and a timing analysis is generated with a Gantt chart from the machine traces. The timing analysis shows the required time to perform an action per peripheral, the start and end time of each action, and the synchronization points among actions. Validation of these outputs against the CEX model is done to check either the CEX sequence is executed as expected or not. The expected time for a movable action is determined from a MATLAB based tool, which is designed to calculate the setpoints followed and the time needed to finish an action, based on its position and profile.

The DDF file contains all the different CEX sequences for one machine. Users of the TWINSCAN machine choose the type of CEX sequence to execute and conditional guards that determine the flow of the sequence. The CEX sequence is then executed based on the selected configuration. However, all combinations of conditional guards and their associated action flow vary based on the selection, and the timing is also affected in CEX execution.

The process of translating the CEX sequence into implementation and finally deploying it on the machine for testing takes resource and time to see the final CEX execution result. Applying MDE to CEX domain shortens the time for such a process and gives other advantages. Such advantages include:

- Creates improved earlier prototypes
- Shortens the development cycle
- Error free implementation through models
- Easier customization of the CEX sequence

Applying the MDE concept using the WLSAT framework brings new design opportunities. Figure 11 (b) shows the additional changes to the current way of designing the CEX sequence. Through models, it is easier to show the CEX timing analysis without deploying a single line of code on the machine. It also helps software and functional design engineers to work closely to get accurate prototypes, which will be used to drive error free implementations.

### 4.2 Envisioned solution

The CEX operation relies heavily on physical movement profiles and locations of peripherals. This graduation project uses and refines the WLSAT modeling framework described in Chapter 2. The CEX sequence movement types are already part of the specification language with minor deviations. However, the new functionality that controls the flow of execution based on conditional guards is needed to fully design the CEX sequence using the WLSAT framework.

The solution for modeling CEX sequences using the enhanced WLSAT tool will help design engineers easily test and generate timing analysis for different models before implementing and deploying them on TWINSCAN machines. The CEX model designed using the tool generates timing analysis based on selected conditions at execution time. This reduces the time for running a specific CEX sequence based on selected configuration values on the TWINSCAN machine. The functional and software design engineers can easily check the timing analysis result and enhance the CEX model before starting their implementation. The enhanced WLSAT methodology will have the following functionalities:

- It will include the domain behaviors of CEX on the existing PLE concept to be able to model CEX sequence.
- It will have a code generator that captures the sequence of activities.

The deliverables from the project include:

- Design document (report) for the extension of WLSAT.
- A model (prototype) of the production CEX sequence using the enhanced WLSAT.
- Timing and critical-path analysis for the prototype CEX model.
4.3 Project scope

This section defines the scope of the project. Below there is a list of items related to scope of the project:

- This graduation project duration is ten months.
- The enhanced WLSAT supports scan options.
- The enhanced WLSAT supports conditional flows.
- The enhanced WLSAT can model the NXT systems’ CEX sequence.
- The enhanced WLSAT supports code generation.
- The enhanced WLSAT supports backward compatibility for modeling wafer logistics.
- Prototype the product line engineering concept to support variations among different CEX sequences.
5. Domain Analysis

This chapter presents the domain in which this graduation project is conducted. It discusses the different aspects that make up the Chuck Exchange (CEX) sequence. We introduce the CEX domain and go through the different variation points that constitute multiple CEX types.

5.1 Introduction

The wafer routing scheme in the TWINSCAN is shown in Figure 12. Wafers are fed into the system from a track. The system itself consists of two operational areas: The Wafer handler and the Wafer stage[2]. The wafer handler is responsible for carrying out several pre-exposure steps on the wafers and routing wafers from the track to the wafer stage and vice-versa. The wafer handler also consists of two robots, the load and unload robots. There are other participating hardware in the wafer handler stage which are not shown in the Figure 12. In the wafer stage, wafers are put on a chuck to perform the measurement and exposure operations. The wafer is first measured (on the measured side) and then exposed (on the expose side). There is one chuck for each side at any given time. For maximizing the throughput of the machine, the two chucks perform their task in parallel. There are always two wafers being processed at the same time. A set of wafers, which is called lot, is processed together in a fabrication plant from start to finish and all wafers receives the same procedure. Since lots can be “streamed”, which means last wafer of lot n-1 is processed (exposed) at the same time with first wafer of lot n (measured). After the measure and expose side finishes, the two chucks swap their position. The CEX is in the critical path of the system. The faster the CEX is, the more throughput the machine will achieve. This swapping of chucks is the domain that this graduation project is addressing, which is the Chuck Exchange (CEX) process.

![Figure 12 – Wafer Logistics of NXT Twinscan Machine](image)

5.2 Chuck Exchange (CEX)

The CEX sequence is a set of actions to bring the measure chuck, which has the measured wafer, to the expose side, so it can start the expose sequence. In the meantime, the expose chuck, which has the exposed wafer, should come to the unload position at the measure side so it can unload the exposed wafer for the wafer handler subsystem.
In Immersion$^1$ lithography, the exposed side contains a liquid medium that has a reflective index greater than one for resolution enhancement. The gap between the final lens and the wafer surface is covered with a highly purified water. One important note in the wafer stage is a wafer must always be present on the exposure chuck to avoid disruption of the film of water below the lens. The measure and expose sides in the Twinscan machine are shown in the Figure 13.

![Figure 13 – Measure and Expose side in Twinscan Machine](image)

The CEX sequence controls a set of peripheral movements or actions. The measure and expose chucks are the main peripherals that participate in the CEX sequence. However, there are other hardware resources that move within their defined positions or actuate an actuator (on/off) and set a value for sensor in each of the chucks. The balance mass, the immersion hood (only for immersion machines), and Bubble Extraction Seal (BES)$^2$ are the peripherals that also participate in the CEX sequence. The balance mass is the hardware that resides underneath the two chucks. Its main purpose is to balance the movement of the two chucks by creating an equal amount of momentum in the opposite direction.

Each peripheral performs actions such as turning a heater on/off or moving a certain resource in a specific axis. The two chucks have six degrees of freedom (DOF) and the Balance Mass has only one DOF. Based on these axes the peripherals have defined positions called symbolic positions. Each symbolic position has a value in each of its axes. For example, for a symbolic position that has three axes is described as:

\[
\text{Symbolic\_position\_name} = (\text{position}_{\text{axis\_1}}, \text{position}_{\text{axis\_2}}, \text{position}_{\text{axis\_3}})
\]

The BES is another type of peripheral that does the so-called simple actions. The BES actions are executed for defined time while the peripheral does not move in respect to its attached resource, such actions are called simple actions. Simple actions are events that atomically executed with other synchronized action and consumes a particular amount of time. Since BES it a stationary peripheral attached to a resource, it has no axis, symbolic position, and motion profiles associated with it.

One other aspect of movable peripherals is the scan performance ID (which are also called motion profiles). A motion profile is a set of velocity, acceleration, and Jerk. Motion profiles are being used to calculate the time it takes for movable peripherals to

---

$^1$ In immersion lithography, light travels down through a system of lenses and then a pool of water before reaching the photoresist on top of the wafer.

$^2$ The BES around the wafer is used to avoid bubbles of air rising into the immersion water into the light path.
move from one symbolic position to another. Since symbolic positions are a set of points in each axis, motion profiles are defined for each axis.

5.2.1. Scan types

A scan move is a move with a constant speed, preceded by a preparation (prep) phase to get up-to-speed and succeeded with either a coda\(^3\) phase to slow down, or another prep phase for a next scan. Peripherals move within symbolic positions using a sequence of scans and/or a prep phase in between two scans. Even though Scans were designed to expose and measure wafers, they are used in a generic way to move the chucks around at the wafer stage. In the Chuck Exchange sequence two types of scans are used. The two types of scans are depicted also in the Figure 14.

1. **Micro-scan**: It is a very short scan from Scan\(_{start}\) to Scan\(_{end}\) position with a constant scan velocity. Its prep is the trajectory to go from the previous position to the Scan\(_{start}\) position and it gains the speed needed for the scan velocity. Its main purpose is to give direction and speed for executing an action. However, during the prep phase the velocity can vary until it reaches the scan velocity. Figure 14 (a) shows how a micro-scan.

2. **To-Point scan**: It is a scan where the Scan\(_{start}\) is equal to the Scan\(_{end}\) position. The time required to perform the scan is zero and the end velocity for the chuck is zero. In other words, the chuck moves from the start position to the Scan\(_{start}/\)Scan\(_{end}\) position and will stand still at the end. It only has a prep phase when it is compared with the micro-scan. Figure 14 (b) shows the different types of to-point scans. To-point scan can follow a micro-scan or can start from a standstill position.

![Figure 14 – Scan types](image)

In Figure 14, the micro-scan is with constant velocity and the action that occurs after the micro scan is a coda unless there is no concatenated scan, which brings the chuck into a standstill state. All the physical actions in the CEX are concatenated and they do not have a coda move after a micro scan. In the WLSAT modeling framework, a coda move is not handled and every peripheral’s movable actions, which are scans, are concatenated unless they are a to-point scan.

5.2.2. Scan options

In CEX sequence, peripherals are working in an orchestrated manner to achieve an action within different subsystems. A subsystem requires synchronization with other subsystems. Synchronized actions among subsystems consists of a negotiation, preparation, and execution phases.

Scans are used to synchronize different peripherals to move at the same time. A scan trajectory needs to be followed with a defined velocity at a defined time interval. Generally, multiple peripherals (which are controlled by their own subsystems) are

---

\(^3\) Coda (Italian for "tail", plural code) is a term used in music to designate a passage that brings a piece (or a movement) to an end.
involved in a scan, it is important that they all start at the same moment and perform their actions in the same time interval. In order to optimize throughput, subsystems are designed to determine what to do next while they execute their current subsystem action. This is implemented by a queueing mechanism. Telling subsystems what to do next opens another advantage called concatenation. In concatenation, the amount of time needed for two consecutive scans is less than making two movements with an intermediate stop.

A scan moves in several peripherals is one synchronized action and the individual prep phases per peripheral for these scans are stretched out such that they match the longest of them (to match the start time of the synchronized scan [7]). There are two kinds of stretching out the time for the prep phase, the As Late As Possible (ALAP) and the As Soon as Possible (ASAP) prep phases. In cases when the scan option is set to ASAP the prep phase is executed as soon as possible and the next synchronized scans start. On the other hand, when the scan option is set to ALAP, the prep phase is executed as late as possible so that the scan of all the synchronized peripherals could start at the same time. These two cases are depicted in the Figure 15.

![Figure 15 – ALAP/ASAP scan types](image)

In Figure 15, the prep phases in peripheral B and C are ASAP while ALAP for peripheral A. The Scans after the prep phases are synchronized and start at the same time. Peripheral A needs the speed to start Scan1 and its prep phase starts as late as possible (ALAP), which means to get the speed needed for Scan1, to match the prep phase ending time in peripheral B. The prep phase of peripheral B and C starts as soon as its predecessor move is finished. The two ASAP types are visualized with the peripheral B and C prep phases.

### 5.2.3. Conditions and Defectivity switches

Most physical actions for the CEX are executed sequentially. However, some physical actions per peripheral or per set of peripherals are executed if a condition or a combination of conditions is true. Due to confidential reasons, the names and their purposes of these conditions are discussed in the confidential part of this report.

There are also CEX settings, so-called Defectivity switches (which are also conditions), that can differ per customer depending on how each customer uses the machine. The purpose of each Condition or Defectivity switch is to alter how the CEX sequence is executed. The actions that have a true conditional guard value put into the flow of the CEX logistics. On the other hand, opposite actions are not executed till their condition becomes valid. Based on a condition’s (a combination of conditions) value, the steps taken for each sequential flow varies. This variation in turn affects the time required for one CEX sequence to complete. These sets of conditions that are associated with physical actions create many CEX sequences depending on the users of the machine.
The sample in Figure 16 shows all aspects of the conditional scans with their associated symbolic positions and profiles.

*Figure 16* shows a synchronized scan for peripheral 1 (green line and shade) and peripheral 2 (blue line and shade). The scans have conditional guards that are mutually exclusive. The condition is constructed using two defective switches, which are the *Condition1* and *Condition2*. Whenever the result of this set of condition is true, the first scans with its motion profile are executed. On the other hand, when the opposite condition is true, the second synchronized scans are executed with its slow-motion profile. The scans for peripheral 2 are labeled as ASAP and their prep phase starts as soon as the predecessor actions are finished. However, the prep phases in peripheral 1 are labeled as ALAP, which means they are executed as late as possible after its predecessor action is executed to match the start time of the scan with peripheral 2.

The time required for executing the first and second synchronized scan is $t_1$, and $t_2$ respectively. These timing requirements are part of the CEX model and they are generated from a MATLAB based API, so-called setpoint generator. These API is used within the TWINSCAN system hence timings are exactly same for scans. The setpoint generator uses the symbolic positions of the start/end location by applying the provided motion profile parameters. The start/end symbolic positions for the scan are shown for each of the peripherals. For example, the physical movement in peripheral 1 from its last known position to the start position of the scan is to the location [$x_1, y_1$] and when the scan is finished the peripheral should be in the setpoint [$x_1, y_2$]. These locations are given for only two axes with similar measurement units. The synchronized scans are expected to finish within the expected time, for the above example $t_1$ is for the first synchronized scan and $t_2$ for the slower synchronized scan.

### 5.3 CEX Logical Sequence

In this section, we try to give a brief introduction to the mechatronics sequence of the CEX domain. The mechatronics sequence contains a flow of actions that need to be done to get from a finished expose and measure cycle to the start of a new expose and
measure cycle. This requires all the peripherals to be at specific synchronization states (to avoid collusion of peripherals).

Figure 17 – CEX mechatronics sequence

Figure 17 shows the steps that need to be taken to have one complete CEX. Each of the steps has a set of synchronized actions for their used peripherals. The detail movements in each step can be referred in the confidential part of this report. The combined flow of one CEX sequence is shown in Figure 18. The figure shows the path of E2M and M2E during one of the chuck exchange operation. The trajectories represent the center of chuck position during the CEX. The full trajectory is a subset of each of the mechatronics sequences.

Figure 18 – Trajectory of the M2E and E2M chucks
5.4 CEX types

The CEX sequence follows the above stepwise logical sequences to perform one swap between the measure and exposed chucks. While executing the sequence, the machine is designed to achieve various requirements within the CEX operation and/or on other part of the machine. There are several CEX requirements, but most important and critical of all requirements are machine safety and timing. The safety of the machine is controlled by the Machine and Material Damage Control (MMDC) subsystem. MMDC deals with unwanted interactions (such as collisions, scratching, and touching) between moving parts in the machine.

Depending on the participating peripherals, there are three major variations of the Chuck Exchange sequence that are designed satisfying the above requirements. The main difference within each type is the presence or absence of the immersion water. Based on the immersion water, the CEX types are determined as dry or wet CEX sequence. The detail difference among the three types of CEX are part of the confidential report and We will not discuss them in this version of the report.
6. Feasibility Analysis

In the previous chapters, the problem and domain that this project desires to solve has been identified. This chapter covers the issues and challenges that may arise in the domain of the project. In addition, some risks that were identified during the project also listed in the chapter.

6.1 Issues and Challenges

The following section presents the issues and challenges that occurred during the project.

The design of the CEX sequence in NXT machines contains the physical movement operations in each of its peripherals. In addition, it also contains attributes that control how the software stack manages these movements so that the safety of the system is maintained. The software stack then takes these attributes to control different sub-systems inside the machine to perform a CEX operation. The WLSAT framework is designed to replicate the physical/hardware operations in the machine but not the software stack that controls the hardware. Since all the attributes that are present in the CEX sequence are not modeled, it limits the functionalities that will be generated. The functionalities that are only modeled using the tool will be generated for use in the machine and design engineers need to add those missing attributes into the generated file.

The WLSAT framework is an ongoing project. New functionalities and enhancements are added to the tool to model the logistics of the wafer in the machines. The source code for the tool is maintained in the Git repository within the ASML network. The master branch of the WLSAT framework is the stable version and that is the starting point for the project. However, there are other ongoing branches that contain new research ideas that will eventually merge into the master branch. This brought a challenge in this graduation project if new major design changes are incorporated in the master branch. In this graduation project, I used the master branch at the start of the project and continue to commit my enhancements locally in my workstation and finally merge them with the master branch or create a new branch for additional enhancement.

6.2 Risks

In this section, the risks that were anticipated during the project are listed with the impact on project scope.

The WLSAT framework uses the Setpoint generator API for its movable peripheral actions. The API takes a set containing start/end symbolic positions and motion profile for the move and returns the execution time with its trajectory path. The API generates a correct timing analysis for actions that move only in one axis. However, some actions move in more than one axis (such as rotating at an edge) and it shows incorrect timing result and also order of actions is affected. This risk has an impact on modeling the CEX sequence and one mitigation plan is to change those movable actions into a simple action, which has a defined time to execute an action.

The other API that is being used inside the WLSAT tool is the Scheduler. It takes all the actions and generate a directed graph which contains a sequence of all actions. This

---

4 Git is a version control system for tracking changes in computer files and coordinating work on those files among multiple people.

5 API is the acronym for Application Programming Interface, which is a software intermediary that allows two applications to talk to each other.
scheduler is designed thinking all the actions start one after the other, which is the ASAP scheduling type. However, the new type of scheduling in CEX, which is ALAP, starts an action at a delayed time to adjust its ending with the start of the next action. Such behavior is not managed in the current version of the used API and needs design changes. The mitigation plan used was to continue modeling the CEX using the existing scheduler API and label the actions if they are an ALAP type action.
7. System Requirements

In this chapter, the system requirements for enhancing the existing WLSAT to incorporate the CEX behavior is analyzed. These analyses are described in the aspect of functional and non-functional requirements.

The functional requirement describes what the enhanced WLSAT should do and it describes the behavior of the system as it relates to the system's functionality. On the other hand, the non-functional requirements place constraints on how the system does its main functionality and its evolution through time (e.g., extensibility and maintainability).

7.1 Requirements

This section describes the list of requirements that should be satisfied, taking into consideration the critical project goals and the project duration. We define the requirements that need to be fulfilled in the project. Requirements are prioritized based on the following categories:

- **Essential**: The product is not acceptable unless these requirements are satisfied.
- **Conditional**: It would enhance the product, but the product is acceptable if absent.
- **Optional**: Functions that may or may not affect the success of the project.

7.1.1. Functional requirement

The Functional requirements are shown in the Table 4 with their assigned priority.

Table 4 – Functional requirements

<table>
<thead>
<tr>
<th>Id</th>
<th>Description</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Req-01</td>
<td>The extended WLSAT environment shall support ASAP scan type.</td>
<td>Essential</td>
</tr>
<tr>
<td>Req-02</td>
<td>The extended WLSAT environment shall support ALAP scan type.</td>
<td>Essential</td>
</tr>
<tr>
<td>Req-03</td>
<td>The DSL shall express the conditional actions of the CEX sequence.</td>
<td>Essential</td>
</tr>
<tr>
<td>Req-04</td>
<td>The &quot;Production CEX(fast/improved)&quot; sequence shall be modeled using the enhanced WLSAT.</td>
<td>Essential</td>
</tr>
<tr>
<td>Req-05</td>
<td>The CEX sequence model output shall match the machine trace output’s timing analysis.</td>
<td>Essential</td>
</tr>
<tr>
<td>Req-06</td>
<td>The extended graphical representation shall include the textual DSL changes.</td>
<td>Optional</td>
</tr>
<tr>
<td>Req-07</td>
<td>The wet and safe production CEX sequence shall be modeled using the enhanced WLSAT.</td>
<td>Conditional</td>
</tr>
</tbody>
</table>
7.1.2. Non-Functional requirement

The non-functional requirements are shown in Table 5 with their assigned priority.

<table>
<thead>
<tr>
<th>Id</th>
<th>Description</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Req-08</td>
<td>The changes to WLSAT shall be backward compatible.</td>
<td>Essential</td>
</tr>
<tr>
<td>Req-09</td>
<td>The enhanced WLSAT shall be extensible to support future improvement.</td>
<td>Essential</td>
</tr>
<tr>
<td>Req-10</td>
<td>The enhanced WLSAT shall be easier to use.</td>
<td>Conditional</td>
</tr>
</tbody>
</table>

The current system is being used for modeling the logistics behavior of the wafer flow and it should continue to support those functionalities while adding new features for modeling the CEX behavior. The other non-functional aspect is considering the extensibility of the software. It should keep clean boundary condition where updating the software at some future date will only require modification or rebuilding a few components instead of creating a new structure.

7.2 Design criteria

In this section we introduce the design criteria that surround the software design of the project. The criteria are from a high-level point of view and based on the approach the project tries to use, which is an enhancement of the existing tool. The three important criteria for the design are:

Usability
The enhanced tool should improve Design Engineers the productivity and shorten their learning curve. Usability is addressed by means of making the product easy to understand by the domain expert.

Genericity
There are different types of CEX sequences within the machine which means there are different CEX model for each of those sequences. The enhanced WLSAT should be able to model all the different variations. Therefore, while making our design, genericity criteria should be considered.

Extensibility
The design is a solution to capture different variation points in CEX. The enhanced tool should be easily extendable with functionality that the software design engineers might require in the future.

Based on our design, the design criteria that is less applicable:

Simplicity
The CEX sequence is a complex set of actions that use different hardware resources. An experienced design engineer might require more time to model and visualize the different CEX variation.
8. System Design & Implementation

In the previous chapters the problem and the requirements are defined. This chapter elaborates on the design decisions based on system requirements. This chapter also enhances the current design with the new scope to achieve the envisioned solution.

8.1 Introduction

The WLSAT methodology is designed using Eclipse Modeling Framework (EMF) and Xtext framework. EMF is a modeling framework and code generation facility for building tools and other applications based on a structured data model[12]. The core EMF framework includes a meta model (Ecore) for describing models and runtime support for the models. In addition, the Xtext is used to represent the EMF models in textual form. Xtext uses the EMF models as the in-memory representation. While enhancing the WLSAT framework, our design improves the Ecore models and the Xtext specification languages.

8.2 Enhanced WLSAT design

WLSAT is designed to model the logistics of wafer flow by separating different concerns in a modular way. Typically, persistence of each domain language in WLSAT is implemented in Xtext and for graphical editing, Sirius representation is implemented. The current WLSAT metamodel is designed to solve different concerns within each component. Figure 19 shows the interrelation among each component. This graduation project scope is to extend two main functionalities of the current architecture so that modeling the CEX sequence can be realized. The first functionality is the ALAP/ASAP scan types within the activity sequence. This functionality enhances the activity component and adjusts the scheduler component to visualize the new functionality. The second functionality, which is conditional action, enhances most of the components.

The design that is presented in this chapter only covers the main inputs to the existing WLSAT design. The auto-generated model representations from the enhancement are not discussed here and the auto-generated model representations alters the existing codebase based on the new design.
8.2.1. Machine specification

The Machine’s Ecore-metamodel is enhanced to define conditional guard names. Each guard can be true or false. The combination of guards by operators also creates another guard as an alias. The Xtext specification language that represent the machine is also enhanced to include the new functionalities. The conditional guards are defined in the machine specification and used in the activity. It is also possible to use these guards in the setting specification for the PLE prototype which will be discussed later in section 8.2.4. Since both the activity and setting specification import the machine specification, the conditional guards are easily available in both components in this design. Figure 20 shows a sample model of the machine specification language with the conditional guards. The Ecore enhancements to achieve the intended result is part of the confidential report.

```plaintext
//Machine specification language
Conditional guards {
    HP = True
    SLOW = True
    HP_AND_NOT_SLOW = [HP AND NOT SLOW] //alias
}
```

Figure 20 – A snippet of the enhanced machine specification language

8.2.2. Activity specification

The editable graph component provides the interface for a directed graph with its node and graphs. The activity takes these directed graph interfaces to model the sequence of activities. Since the sequence of activities are enhanced with the conditional moves, a conditional edge concept is added based on the directed graphs’ edge concept. The conditional edge also has a boolean conditional guard that makes the edges active or not.

The enhanced metamodel Ecore adds the functionality to support scan types and conditional actions. The textual specification language for activity component is also enhanced to capture the metamodel changes. Edges have a new type to able model and capture conditional edges and each conditional edge have a set of guards that controls the flow of actions. A new type of Nodes is also incorporated to show the start and end of a conditional activity. Conditional Edges start with a `StartBranch` and end with `EndBranch` nodes.

```plaintext
activity SampleActivity {
    prerequisites {
        Robot.M at Above_Right
    }
    actions {
        C: claim Robot
        R: release Robot
        A1: move Robot.M to Above_Left with speed profile slow
        A2: move Robot.M to Above_Right with speed profile slow
        A3: move Robot.M to Below_Left with speed profile normal
    }
    action flow {
        C -> A1 -> <branch [HP] -> A2 -> branch_end> -> R
        <branch [NOT HP] -> A3 -> branch_end>
    }
}
```

Figure 21 – Sample activity specification DSL
Figure 21 shows a code snippet of a model for branching activities. As can be seen from the figure, action A1 has two successors (A2 and A3) and based on the conditional guard’s value (High Performance (HP) or NOT HP (i.e. Low performance)), the next action (A2 or A3) is selected.

![Diagram showing branching activity sequence](image)

Figure 22 – A branching activity sequence

Figure 22 shows the graphical representation of a sample activity diagram. The new conditional branch functionality is shown with its edges having a boolean guard. It shows the activity diagram sequence for only one peripheral and each action is executed based on the boolean value of the associated condition guard. The conditional guards are shown on the edges connecting the start branch node to its successor. The evaluated boolean values are also shown together and the conditional edge that will not be part of the sequence are drawn with red color.

8.2.3. Code generation

The CEX sequence is modeled using the enhanced WLSAT framework. The Ecore models are the in-memory representation of the CEX model. The Eclipse plugin project has the capability for creating a generator out of the Ecore models. The code generator implementations can be written in Java or Xtend. Xtend is a statically typed language and it uses the Java type system (including Java generics). Thus, Xtend and Java are completely interoperable[3]. The main goal of Xtend is to have a less “noisy” version of Java; indeed, in Java, some linguistic features are redundant and only make programs more verbose.

These are the reasons why Xtend is used in the code-generation design of WLSAT:

- It has multi-line template expressions, which are useful when writing code generators
- It has a nicer syntax compared to Java
- It allows debugging Xtend code as in Java implementations
- It is completely integrated with Java and all the Java libraries are accessible from within Xtend.

The syntax of the templates used for the code generation process are from the existing CEX sequence implementation. The template is written in a DDF file format and the generated code is in the same file format. The syntaxes are included in the implementation as multi-line template concatenated with the model attributes. *Figure 23* shows the flow of the code generation process. The DDF file for the CEX contains action definition, symbolic position, and the order of sequence. The exact data structure schema for the generated code is part of the confidential report and can be accessed within ASML network.

### 8.2.4. Symbolic position with PLE

In this section, the symbolic position was chosen as a variability point and its variants are expressed for each CEX sequence type.

The prototype designed in this project assumes the model diagram in *Figure 24*. WLSAT is extended with the prototype functionality to support the domain variability point for the symbolic position. The execution of the CEX sequence chooses its associated variants for the symbolic position based on selected CEX types.
// A peripheral symbolic position per each axis
Positions {
  Start { // Name of the position
    Option = 20.0 For [Type1]
    Option = 15.0 For [Type2]
  }
  Middle { // Name of the position
    Option = 10.0 For [Type1] //Value for type 1
    Option = 5.0 For [Type2] //Value for type 2
  }
  End = 0.00 //same for all the CEX types
}

Figure 25 – A prototype of the symbolic position with PLE

The prototype was designed to support the symbolic positions that have different values based on the type of CEX sequence. The Optional values are set at design time for each CEX sequence types. At execution time, the model selects the associated values for the selected CEX sequence type to generate the timing analysis. The enhanced WLSAT selects the associated CEX type and the execution begins by assigning the related variant values for each symbolic positions. Figure 25 shows a code snippet on how different symbolic position values are described using the variability point and its respective variant. The prototype assumes that the number of actions executed remain unchanged while the symbolic position values varies based on selected CEX type.

■
9. Deployment

This chapter explains on the deployment view of the enhanced WLSAT application, it gives a brief explanation about the technologies that are used for deploying an eclipse plug-in application and how the WLSAT artifacts are used within ASML.

9.1 Deployment view

The deployment view is used to describe the related hardware infrastructure in which the software artifacts are deployed. The deployment of the enhanced WLSAT framework for modeling the CEX is shown in Figure 26. The deployment diagram shows the two blocks involved for deploying the generated code into the TWINSCAN machine. The existing deployment view for the WLSAT framework is being used with an addition of the code generation. The design changes in the WLSAT has an impact only on the behavior of the system to incorporate the new domain, which is the CEX sequence. The deployment diagram shown below persists for the enhanced WLSAT framework and adds the related TWINSCAN components that will interact with the generated CEX sequence. The Immersion Handling Action Layer (IHAL), uses the code generated from its respective model. The IHAL components use the synchronization component (SN) in the TWINSCAN system. The deployment of the generated code inside the TWINSCAN environment was not done due to time limitation and prioritization of tasks. Only the generated DDF files are checked against the existing implementation of CEX sequence for the IHAL component.

The existing repository for the WLSAT framework has all the require projects and files to deploy the application as a plugin project for eclipse IDE. The project used Maven as its build automation tool. Maven projects are configured using a Project Object Model (POM), which is stored in a pom.xml file[4]. The XML file describes the software project being built, its dependencies on other external modules and components, the build order, directories, and required plug-ins. Maven dynamically downloads the required Java libraries and Maven plug-ins from one or more repositories and stores them in a local cache.
The Eclipse platform is designed to serve as an open tools platform[10], it is architected so that its components could be used to build any client application. The minimal set of plug-ins needed to build a rich client application is collectively known as the Rich Client Platform (RCP). The WLSAT project has all the configuration projects for its deployment as an RCP application. These are the project types that are included:

- **Parent project:** It contains the configurations needed for the runtime environment and the associated plugin projects.
- **Plugin project:** It contains all the plugin projects in the eclipse workspace.
- **Feature project:** It describes a list of plug-ins and other features which can be seen as a logical unit, i.e., a set of related components.
- **Documentation project:** It includes the code documentation generator in Eclipse.
- **Branding project:** It include items such as the splash screen, about dialog, and the program executable.

The above sets of projects are built using Tycho[11], which is a set of Maven plug-ins for building Eclipse components via the Maven build system. Maven then creates the files that are necessary for a standalone RCP project, which are representation of the languages associated inside WLSAT. ■
10. Verification & Validation

In the previous chapter the system design and implementation of the product is defined. This chapter presents the validation and verification process used in order to confirm that the right CEX sequence is modeled. Most of the model results related to the CEX sequence are not discussed in this public report, please refer the confidential report for more detail.

10.1 Introduction

Modeling the CEX sequence using the enhanced WLSAT framework needs to follow the several steps. The production CEX sequence was selected at the beginning of the project to validate and verify the capabilities of the enhanced WLSAT framework. The steps needed to create a specification language is shown in Figure 27. After modeling each hardware peripheral, the validation of the production CEX sequence is checked against the machine traces. The timing analysis and the order of actions are checked with the machine traces for each peripheral. The hardware modeled for the production CEX sequence includes the two chucks, Balance Mass, Immersion Hood, Bridge-BES, BES, and the wafer table heaters. These peripherals are modeled in a resource that contains one or more peripherals. The name of the resources used within the CEX sequence and their respective peripherals are shown in Table 6.

![Figure 27 – Flow of modeling CEX using WLSAT](image-url)
Table 6 – Resources and the associated peripherals

<table>
<thead>
<tr>
<th>Resource</th>
<th>Peripheral/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>M2E and E2M chucks</td>
<td>The two chucks with their short and long stroke</td>
</tr>
<tr>
<td>Balance Mass</td>
<td>Balance mass</td>
</tr>
<tr>
<td>Immersion Hood</td>
<td>Immersion Hood</td>
</tr>
<tr>
<td>SI_E2M and SI_M2E</td>
<td>The stages and immersion resource, it contains the BES and the B-BES that are attached to chucks</td>
</tr>
<tr>
<td>SB_E2M and SB_M2E</td>
<td>The stages and the Bridges resources, it contains the swap bridges and the B-BES that are attached to chucks</td>
</tr>
</tbody>
</table>

10.2 Verification

Verification is important in order to check whether the project’s design and implementation is satisfying the requirements. The project has a model prototype and the design is developed through time with an iteration, the verification is done informally, every iteration, by discussions with project supervisors. Other stakeholders were also occasionally included in the verification process using demonstration and presentation.

10.3 Validation

The validation of the CEX model is checked against the machine traces from TWINSCAN machine. The timing analysis and the order of actions for selected CEX flows are generated to validate the model results. The main peripherals used for the check exchange are modeled using the enhanced WLSAT framework. The representative model includes the allowed paths among symbolic positions, the motion profiles used per peripheral, the flow of actions, and finally the timing analysis is generated from the model. The detail of the production CEX sequence model is discussed in the confidential report and all the necessary model attributes that includes symbolic positions, motion profiles, activity sequences, and timing analysis are discussed.

10.3.1. Activity sequence

The activity sequence for the CEX sequence is modeled using the enhanced WLSAT. The conditional flows are controlled by the guard values and it is determined at execution time. The activity specification language is enhanced in the textual and also in the graphical part to visualize the new features.

10.3.2. Timing analysis

The CEX model validation against the machine traces is tested using the results of timing analysis. Scheduling an activity sequence generates a Gantt chart that represents the system execution in time. The machine traces from the TWINSCAN shows similar Gantt chart representation and the order of the actions and the execution time are compared against the model results. In general, the timing analysis shows the execution of each CEX movement modeled with the WLSAT are within the design specification requirement. For more detail analysis of the production CEX sequence, refer the confidential report.
Figure 28 – The variability points and variants for different CEX sequences
10.4  CEX analysis using PLE

In this section, we will present the analysis of the CEX sequence using PLE variability model. Since the domain of PLE was a vast topic, we choose to concentrate on making a prototype on one of the variability point within the CEX sequence. The design part of the prototype is discussed in section 8.2.4.

Analyzing the different CEX sequences and choosing the variation points was the first step towards the prototype. Some of the variation points that exist among different CEX sequence includes:

- Symbolic positions used for similar actions.
- Motions profiles used for similar actions.
- Number of actions executed.
- Peripherals used in the CEX sequence.

The above listed variation points have their own variant based on selected CEX sequence type and conditional guards selected. The different CEX sequences variability points and their required variants are shown in Figure 28. The three CEX types are shown with their associated logical actions and the peripherals involved. The exact names of the CEX types are omitted for confidential reasons. The details inside each logical action are not described in the figure; however, the symbolic position variability point is prototyped with its variants. In the developed prototype, only the symbolic position within one CEX common sequence is expressed with its variants. ■
11. Conclusions

This chapter summarizes the results achieved with this project and explains the lessons learnt during the execution of the project.

11.1 Results

This section describes the conclusions with respect to the contents of the technical results of the project.

Inside AMSL’s TWINSCAN machine, the process of executing a CEX sequence is a complicated process and it moves the two hardware (Chucks) in synchronized manner. The time for executing one CEX sequence is one of the integral part of the machine throughput. To maximize the number of wafers processed with in the machines, an optimum CEX sequence should be executed.

Creating an illustrative model of the system using DSL helps the users to clearly define requirements easily. The model results are helpful to analyze the design before starting any implementation. The design engineers use similar domain language to model the CEX sequence and that in turn reduces the time for modeling and analysis. From the results, they can see earlier prototype of the CEX sequence without testing on the virtual TWINSCAN environment.

From this project, it is shown that the models created using the enhanced WLSAT framework are consistent with the CEX sequence. The timing analysis of the CEX model is within the range specified when executed. The different conditional flow of the production CEX sequence can be tested with ease in comparison with the actual implementation of the test environment within the TWINSCAN machine. From the results, ASML’s machine users can enhance their way of CEX execution based on their chosen machine usage.

11.2 Future work

This section elaborates on future works and improvements, as well as on features that were not done in this project due to constraints such as time and lack of knowledge.

Possible works that could improve the WLSAT framework and the CEX model:
- Enhance the WLSAT framework’s ASAP scheduler API to support the ALAP tasks.
- Enhance the WLSAT framework to support movements that use a different preparation speed in comparison with the successor scan.
- Enhance the WLSAT framework to support movements in multiple axis.
- Enhance the generated code output to incorporate other software stack aspects of the CEX implementation.
- Enhance the WLSAT tool to incorporate the full capability of features at peripherals, actions, and motion profiles levels.
- Model the remaining CEX sequence types using the enhanced WLSAT framework.

Analyzing the CEX sequence model using the enhanced WLSAT framework helps the design engineers to analyze the timing requirements for the sequences easily. This in turn helps to refine the CEX design, to get an earlier prototype of the model, and reduce implementation time.
11.3 Lessons Learned

During the past ten months, I have developed my technical and soft skills as a result of conducting the project at the client premises. From technological point of view, the project consisted of both familiar and unfamiliar topics. The domain of Xtext, Xtend, and Sirius graphical modeling tool were unknown topic before the project but is now part of my software experience.

These are the two perspectives in which I learn throughout the project:

- **Technical skills:**
  - Broadened my knowledge in the domain of domain specific languages and the technologies behind modeling using Xtext framework.
  - Learn the domain of ASML’s TWINSCAN machine CEX sequence.

- **Soft skills:**
  - Monthly PSG meetings help me to gather my thoughts for a presentation and discussion.
  - Planning and managing a project from its initial to its completion phase.

Working within ASML was a valuable and interesting experience. In addition, gaining new knowledge while interacting with different people and getting to know new technologies was a plus to my experience. In conclusion, I achieved the goals set at the start of the project and in the meantime learned a lot in process.
12. Project Management

This chapter elaborates on the project management process that was conducted during the lifetime of the project.

12.1 Work-Breakdown Structure (WBS)

In this section, the Work-Breakdown Structure of the project is discussed. The project is divided up into four major categories: Planning and Management, Analysis, Design & Implementation, and Documentation. Figure 29 shows the detail activities conducted in each category.

![Figure 29 – Work-breakdown structure of the project](image)

12.2 Project plan

The project is planned in accordance with the work-breakdown structure. However, the plan changed during the project because of earlier task completion. Therefore, the final project plan deviated from the initial planning. These two project plans are described in the following sections. The project plan was visualized using an Excel and I updated the timeline of tasks every week.

Every month, a Project Steering Group (PSG) meeting was held at TU/e and at ASML every other month. A project progress and/or a demonstration of new prototype was presented to the supervisors. These meetings allowed the stakeholders to have a clear view of the project progress and the supervisors were giving feedback about the progress.

12.2.1. Initial

The initial project plan was shown in Figure 30. This plan was laid out at the initial phase of the project. As the detail of the project scope was clear, some tasks were completed earlier than expected and others took longer than expected. Moreover, new additional tasks were added to realize the project.

12.2.2. Final

The final version of the project plan displays what actually happened during the project. Most of the tasks in the initial project plan were executed in their order and earlier
prototyping of the CEX model and its analysis from WLSAT for stakeholders was presented. *Figure 31* shows the timeline of the project plan and *Table 7* lists the actual tasks and their duration. Due to earlier prototyping of the DSL and the code generation more additional tasks were included in the project and the report writing took more time than expected. The additional analysis of the different CEX types using PLE concept was conducted. However, only the production CEX sequence was modeled, analyzed, and validated against machine traces.

### 12.3 Risk analysis

In this section, the risks involved with the project are discussed. For each risk item, its impact and level are defined within a scale. In addition, a mitigation strategy is given with its triggering task. The list of risk items for the project are shown in *Table 8*. The impact of each risk item is labeled either high, medium, or low based on the severity of the effect it can cause to the project schedule and result.
Figure 30 – Initial project plan

- Domain exploration
- Project scope
- Requirement analysis
- Requirement analysis-Doc
- High level design
- Implementation
- Extending DSL
- Sample DSL demo
- Code Generation
- Sample code generation
- Modeling CEX
- Model presentation
- Testing
- Report writing & review
- Final report-Doc
- Final presentation

Milestone
Figure 31 – Final project plan
Table 7 – Final project plan

<table>
<thead>
<tr>
<th>Id</th>
<th>Tasks</th>
<th>Start date</th>
<th>End date</th>
<th>Duration (Days)</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task-01</td>
<td>Domain exploration</td>
<td>1-Jan</td>
<td>28-Feb</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>Task-02</td>
<td>Project scope</td>
<td>10-Feb</td>
<td>28-Feb</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>Task-03</td>
<td>Requirement analysis</td>
<td>15-Jan</td>
<td>28-Feb</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>Milestone-1</td>
<td>Requirement analysis-Doc</td>
<td>28-Feb</td>
<td>5-Mar</td>
<td>5</td>
<td>Milestone</td>
</tr>
<tr>
<td>Task-04</td>
<td>High level design</td>
<td>28-Feb</td>
<td>30-Apr</td>
<td>61</td>
<td></td>
</tr>
<tr>
<td>Task-05</td>
<td>Implementation</td>
<td>15-Mar</td>
<td>30-Jul</td>
<td>137</td>
<td></td>
</tr>
<tr>
<td>Task-05-01</td>
<td>Extending DSL</td>
<td>15-Mar</td>
<td>30-Jun</td>
<td>107</td>
<td></td>
</tr>
<tr>
<td>Milestone-2</td>
<td>Sample DSL demo</td>
<td>23-May</td>
<td>28-May</td>
<td>5</td>
<td>Milestone</td>
</tr>
<tr>
<td>Task-05-02</td>
<td>Code Generation</td>
<td>15-Apr</td>
<td>30-Jul</td>
<td>106</td>
<td></td>
</tr>
<tr>
<td>Milestone-3</td>
<td>Sample code generation</td>
<td>23-May</td>
<td>28-May</td>
<td>5</td>
<td>Milestone</td>
</tr>
<tr>
<td>Task-06</td>
<td>Modeling CEX</td>
<td>25-Apr</td>
<td>15-Aug</td>
<td>112</td>
<td></td>
</tr>
<tr>
<td>Milestone-4</td>
<td>Model presentation</td>
<td>23-May</td>
<td>28-May</td>
<td>5</td>
<td>Milestone</td>
</tr>
<tr>
<td>Task-07</td>
<td>Testing</td>
<td>23-May</td>
<td>31-Aug</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>Task-08</td>
<td>PLE analysis for CEX types</td>
<td>20-Jul</td>
<td>8-Oct</td>
<td>80</td>
<td></td>
</tr>
<tr>
<td>Task-09</td>
<td>Report writing &amp; review</td>
<td>1-Jul</td>
<td>1-Oct</td>
<td>92</td>
<td></td>
</tr>
<tr>
<td>Milestone-5</td>
<td>Final report-Doc</td>
<td>31-Aug</td>
<td>5-Sep</td>
<td>5</td>
<td>Milestone</td>
</tr>
<tr>
<td>Task-10</td>
<td>Final presentation slide</td>
<td>15-Sep</td>
<td>4-Oct</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>Task-11</td>
<td>Final presentation rehearsal</td>
<td>21-Sep</td>
<td>26-Sep</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Task-12</td>
<td>Final presentation</td>
<td>8-Oct</td>
<td>13-Oct</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Task-13</td>
<td>Final presentation-feedback</td>
<td>8-Oct</td>
<td>23-Oct</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>Risk Id</td>
<td>Description</td>
<td>Impact</td>
<td>Impact level</td>
<td>Trigger Task</td>
<td>Mitigation</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>--------</td>
<td>--------------</td>
<td>--------------</td>
<td>------------</td>
</tr>
<tr>
<td>Risk-01</td>
<td>Unable to extend WLSAT framework to support the variation points of the Chuck Exchange</td>
<td>It is not possible to deliver the full scope of the project</td>
<td>High</td>
<td>05</td>
<td>Model the CEX with the existing WLSAT framework using timing settings</td>
</tr>
<tr>
<td>Risk-02</td>
<td>Unable to identify and classify the variation points in the CEX behavior</td>
<td>It will not be possible to model a generic DSL on WLSAT for all CEX behavior</td>
<td>Medium</td>
<td>05-01</td>
<td>Design the DSL to support the common variation points in each CEX behavior</td>
</tr>
<tr>
<td>Risk-03</td>
<td>Unable to model every aspect of CEX on WLSAT</td>
<td>Unable to support all the functionalities of CEX in the generated code</td>
<td>Medium</td>
<td>05-02</td>
<td>Identify the main parts of the CEX model and code generate</td>
</tr>
<tr>
<td>Risk-04</td>
<td>Unable to generate similar Gantt chart</td>
<td>Similar visual analysis cannot be validated against machine traces</td>
<td>Medium</td>
<td>07</td>
<td>Cross check the order of the sequence manually</td>
</tr>
<tr>
<td>Risk-05</td>
<td>Unable to get an updated API to support new CEX behaviors</td>
<td>Unable to show graphical results easily</td>
<td>Low</td>
<td>02</td>
<td>Work with the current version of the API</td>
</tr>
<tr>
<td>Risk-06</td>
<td>Unscheduled / unplanned holiday by stakeholders</td>
<td>A delay project milestone</td>
<td>Low</td>
<td></td>
<td>Work on tasks that doesn’t require immediate assistance from stakeholders</td>
</tr>
<tr>
<td>Risk-07</td>
<td>Trainee is ill for a long period</td>
<td>Not possible to deliver the full scope of the project</td>
<td>Medium</td>
<td></td>
<td>Keep a buffer time at each task and negotiate requirements with based on priority</td>
</tr>
</tbody>
</table>
13. Project Retrospective

In this chapter, the experience that I gain during the execution of the project is discussed. Moreover, the design criteria that were chosen at the earlier phase of the project are revisited.

13.1 Project reflection

The project conducted during the past ten months brought challenging and interesting characteristics. I had the knowledge of Model Driven Engineering and experience with creating Domain Specific Languages with JetBrains MetaProgramming System (MPS) during my first year PDEng program. However, in this project, the process to get to know the Xtext framework, understanding the existing codebase of WLSAT and finally modeling the CEX sequence were not easy.

At the beginning of the project, I was introduced by company supervisor to my stakeholders and discussed on domain of CEX in relation with the WLSAT framework, the functionalities of the existing WLSAT framework, how to gain access to the codebase and how I can communicate with them to get their inputs on the project. It helps me to create a smooth relation with them and share my design ideas during project execution.

Throughout the execution of the project, I shared my design ideas with all the stakeholders and received their feedbacks. This helps me to refine the design and the CEX model that I created. In the meantime, I also shared part of the CEX model results with the team members that I work with and it gave them a perspective to my project. It also gave me a great experience in managing expectations.

In conclusion, the project was a great experience and opportunity for me to practice and broaden my horizon with the technical skill of creating domain specific languages and modeling. In addition, other skills that are related to software development were also exercised, such as managing and planning a project, communication with stakeholders, and analyzed results effectively. All these exposures and experiences in working with different teams on the domain and technology within AMSL will be a valuable lesson for my new role within ASML after this graduation project.

13.2 Design opportunities revisited

The design opportunities that have been introduced in Section 7.2 are revisited as follows:

**Usability:** The new enhancements in the WLSAT framework add new concepts that are related to the CEX sequence. A user who was familiar with the CEX domain understands and knows these terms and it makes the usability of them in modeling the sequence easier.

**Genericity:** The modeling concept introduced within the existing WLSAT framework is a common behavior in all the CEX sequences. As a result, the enhanced WLSAT tool can model the other CEX sequences.

**Extensibility:** The WLSAT framework is designed in a way to tackle different sets of concerns in modular way, this behavior is not changed, and the enhancements are done in each component to make future improvements.
Glossary

Activity  Activity composes high level stepwise flow of actions that a re-
source can perform.

ASML  Advanced Semiconductors Materials Lithography

API  Application Programming Interface

BES  Bubble Extraction Seal

B-BES  Bridge Bubble Extraction Seal

BM  Balance Mass

CEX  Chuck Exchange

DDF  Data Definition Format that provides a definition of data structures
in a language-independent way

Dispatch  A set of ordered activities scheduled to be executed

DSL  Domain Specific Language

EMF  Eclipse Modeling Framework

E2M  The chuck that moves from exposure to measure side, see M2E

IHAL  Immersion Handling Action Layer

LEX  Left Exposure chuck, see also REX

MDE  Model Driven Engineering

MMDC  Machine & Material Damage Control

M2E  The chuck that moves from measure to exposure side, see E2M

PDEng  Professional Doctorate in Engineering

Peripheral  Each resource contains peripherals, e.g., clamp of a robot.

Peripheral type  Each peripheral refers to Peripheral type that corresponds to the
type of the peripheral, e.g., linear motor is of a type motor.

PLE  Product Line Engineering

PSG  Project Steering Group

Resource  Resource corresponds to a set of hardware peripherals in a ma-
chine, e.g., a robot.

REX  Right Exposure chuck, see also LEX

SB  Swap Bridge

SB M2E  Stages and Bridges at the M2E chuck

SB E2M  Stages and Bridges at the E2M chuck

Scan  A scan move is a move with a constant speed, preceded by a prep-
oration phase to get up-to-speed and either a coda phase to slow
down, or another prepare phase for a next scan

SI  Stage and Immersion

SI M2E  Stages and Immersion at the M2E chuck

SI E2M  Stages and Immersion at the E2M chuck

Sirius  Eclipse plug-in for creating visual representations

Symbolic position  The physical positions a peripheral can take in its coordinate sys-


tem and it is expressed for each axis.

TWINSCAN  The photolithography machine that ASML is producing

TU/e  Eindhoven University of Technology

Wafer  A thin slice of semiconductor material, such as a crystalline sil-
icon, that serves as a substrate for the fabrication of integrated cir-
cuits

WLSAT  Wafer Logistics, Specification and Analysis Tool

XText  The framework used for creating DSL for WLSAT
References


Additional Reading


About the Author

Berihun Fekade Yimam received his BSc degree in Electrical Engineering from Arba Minch University, Ethiopia in 2007. After graduation, he worked for six years at a local software development firm in Ethiopia. Then he left to pursue his master’s program in South Korea. He received his master’s degree in Computer Science from Korea University in August 2016. He is interested in the architecture, design and implementation of multidisciplinary applications.

Berihun was part of the Software Technology PDEng program, generation 2016, at the Eindhoven University of Technology. During his graduation project, he worked for ASML Netherlands B.V., on a project focused on enhancing an in-house modeling tool to support the Chuck Exchange domain at the Embedded Software Department.