5-2013

DECISION SUPPORT SYSTEMS FOR ASSEMBLY LINE PLANNING -MODULAR SUBSYTEMS FOR A LARGE SCALE PRODUCTION MANAGEMENT SYSTEM

Rahul Renu
Clemson University, rrenu@clemson.edu

DECISION SUPPORT SYSTEMS FOR ASSEMBLY LINE PLANNING: MODULAR SUBSYSTEMS FOR A LARGE-SCALE PRODUCTION MANAGEMENT SYSTEM

A Thesis
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Mechanical Engineering

---

by
Rahul Sharan Renu
May 2013

Accepted by:
Dr. Gregory M. Mocko, Committee Chair
Dr. Joshua Summers
Dr. Georges F. Fadel
Dr. Mary Beth Kurz

ABSTRACT

In the domain of assembly lines, process sheets are entities that carry assembly process descriptions (work instructions), assembly time estimations, product workspaces and other configuration management and control information. These process sheets get assigned to workstations during the process of assembly line balancing. In this research two tools have been developed to aid in the assembly line planning process. In order to ensure that assembly line workers do not intrude upon each other's workspace, two assembly processes operating on the same product workspace must not be assigned to the same station. Generating and maintaining product workspace information for every process sheet is automated by use of the Product Workspace Identification Tool developed in this research. This tool uses CAD data, custom built analysis software, and web-based databases to define, compute, and store product workspace information.

All assembly work instructions must have time studies. These time studies are used primarily in line balancing. Methods Time Measurement (MTM) is a set of charts that provide standard assembly time estimations based on the parts (and their surrounding space) that are being assembled. In the adapted version of MTM that is used for this research, there are twenty-two MTM tables and several pieces of information are required to arrive at an assembly time study. Using the MTM tables to generate assembly time estimates is cumbersome simply because the number of work instructions for a given product can run into the thousands. The MTM estimate generator presented in this thesis provides decision support to the process sheet author by presenting a reduced set of MTM tables and also by performing filtering within the MTM tables. Testing and validation of this tool showed that it has a mapping accuracy of 75%. These two tools are modular subsystems of a large production management system.

DEDICATION

I dedicate this thesis to my parents.

ACKNOWLEDGEMENTS

I thank BMW Manufacturing for providing funding for this research, with special thanks to Dr. Kilian Funk for his continued support. I would like to thank my advisory committee chair, Dr. Gregory Mocko for his support and guidance throughout the course of my research. I would also like to thank the other members of my advisory committee, Dr. Joshua Summers, Dr. Georges Fadel and Dr. Mary Beth Kurz for their support and guidance.

I would like to thank all the members of Clemson Engineering Design Applications and Research (CEDAR) for their help and feedback over the course of this work. I would like to give special thanks to Matthew Peterson and Aravind Shanthakumar, who have provided continued support and valuable feedback.

All my research would not have been possible without my parents who have always had belief in me. I would like to thank them for their unwavering love and faith.

TABLE OF CONTENTS

Table of Contents (Continued)

Page

Table of Contents (Continued)

Table of Contents (Continued)

Page

LIST OF TABLES

LIST OF FIGURES

List of Figures (Continued)

List of Figures (Continued)

List of Figures (Continued)

CHAPTER ONE
MOTIVATION AND RESEARCH OBJECTIVES

Chapter Objectives:

- Provide research objectives.
- Brief overview of each research objective.
- Provide an outline of the thesis.

The principal objective of the research is to automate product assembly time estimation analyses and provide decision support during the process of line balancing. The intent is to reduce the effort expended by planners during assembly line planning. The existing issues with assembly line planning are addressed by:

- Automating the identification of product workspaces.
- Providing decision support during the generation of assembly time estimates.

An assembly line is a series of workstations where a product is assembled. Assembly line balancing is the process of assigning assembly tasks to workstations such that the tasks do not break precedence, tooling, time, workspace and other constraints [1]. Workspace constraints ensure that assembly line workers do not intrude upon one another's working areas.

Assembly time estimates are linked to assembly work instructions. There are multiple work instructions on a single assembly process sheet (see Figure 0.2). There are several process sheets for every product. Time estimates are obtained from time study charts. Assigning a time estimate to every work instruction on every process sheet is cumbersome. Automation of this process will reduce the chance of errors and also reduce the effort expended.

These issues form the underlying motivation for the research objectives presented in Section 1.1. Figure 0.1 shows a general process flow of assembly line planning. The boxes with dotted borders are issues that this research addresses. An Automated Time Estimate Generator populates the process sheets with assembly time estimates based on work instruction text and part information. The Product Workspace Identification Tool (P.W.I.T.) is used to formalize and define the workspace occupied by a part.

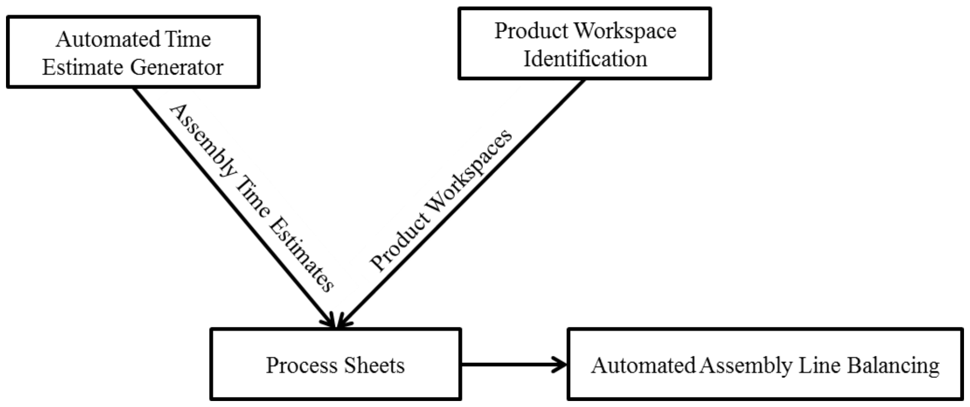

Automated Time Estimate Generator
Product Workspace Identification
Assembly Time Estimates
Product Workspaces
Process Sheets
Automated Assembly Line Balancing

Figure 0.1: Process flow of assembly line planning

1.1 Research Objectives and Motivation

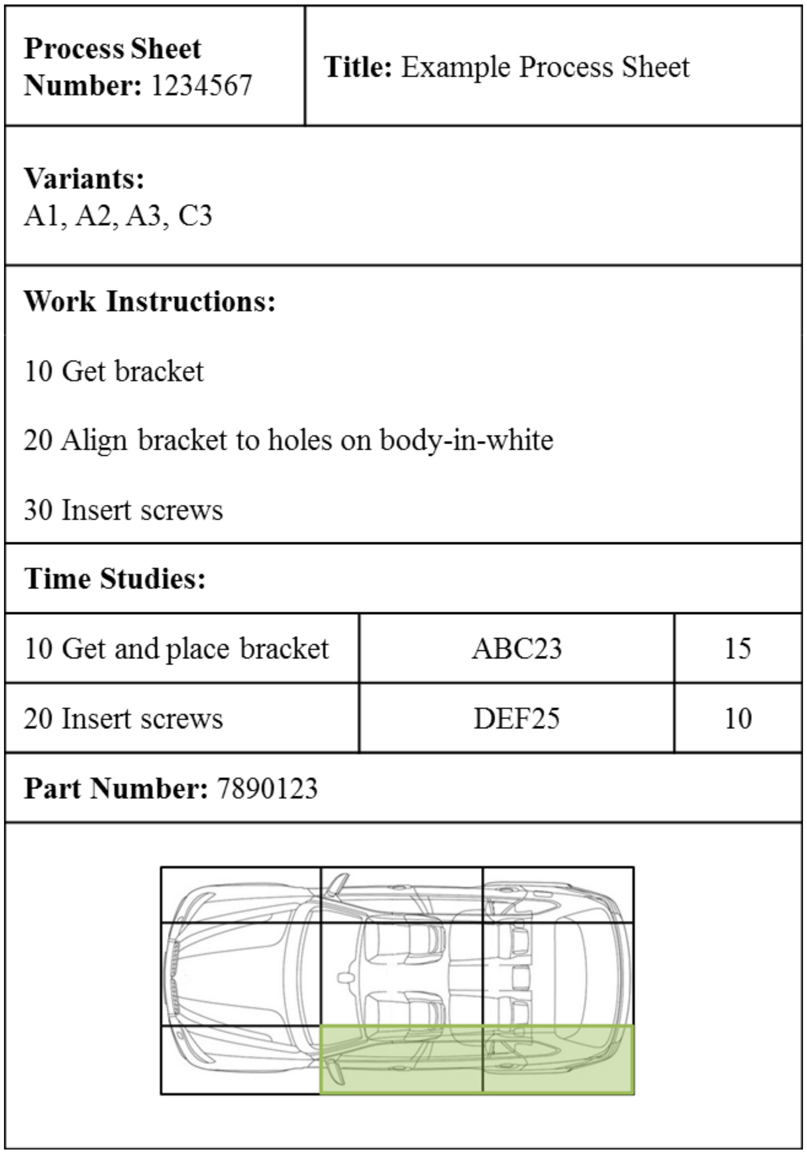
This section presents the two research objectives and a brief overview related to each of the questions.

1.1.1 Research Objective One

With respect to the automation of identifying product workspaces, the first research objective is presented.

Create a tool that will automatically identify product workspaces.

Process sheets are information packets used to communicate the assembly process instructions to assembly line workers. Its other purposes range from documentation to assembly line balancing. Process sheets contain work instructions, time estimates and other meta-information (see Figure 0.2) [2]. It has been observed with a large scale automotive original equipment manufacturer (OEM) that this meta-information includes diagrammatic representation of the approximate location of the part/parts (associated to that process sheet) within the vehicle. An example of this is shown in Figure 0.2.



Process Sheet Number: 1234567	**Title:** Example Process Sheet	
Variants: A1, A2, A3, C3		
Work Instructions: 10 Get bracket; 20 Align bracket to holes on body-in-white; 30 Insert screws		
Time Studies:		
10 Get and place bracket	ABC23	15
20 Insert screws	DEF25	10
Part Number: 7890123		

Figure 0.2: Example of an assembly process sheet

Figure 0.2 shows an example of a process sheet and some of the typical data that can be found in them. The diagrammatic representation of part location (last row of Figure 0.2) is manually generated. Automation of this process will reduce the time and effort expended by planners when they author process sheets. Also, this information of the part location is used during the line balancing process to ensure a line side associate's work space is not intruded upon [3].

Figure 0.3 shows the issue relating to research objective one in the darkened box.

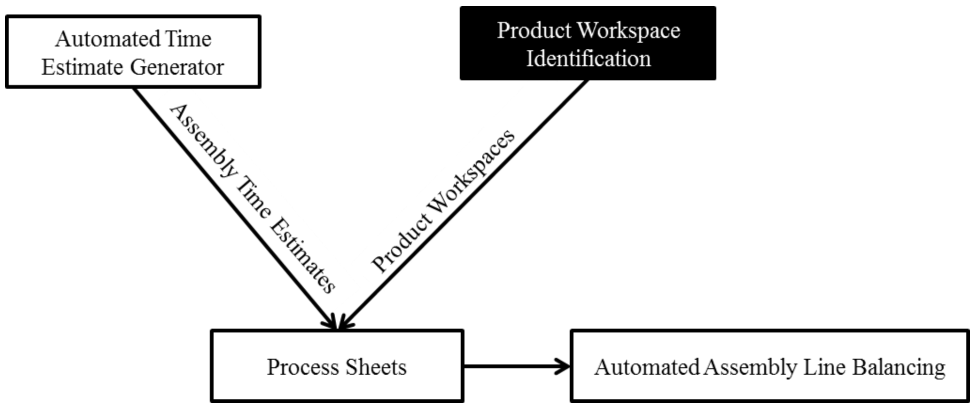

Automated Time Estimate Generator
Product Workspace Identification
Assembly Time Estimates
Product Workspaces
Process Sheets
Automated Assembly Line Balancing

Figure 0.3: Issue Research Objective One addresses

1.1.2 Research Objective Two

In context of providing decision support during time study analysis, the second research objective is presented below.

Extend existing part-process coupling to provide decision support to planners during the creation of MTM time estimates.

The time estimates for the work instructions within a process sheet is obtained from time study standards. These standards are tabulated compilations of time estimates based on historical data. A time estimate is obtained from these standards by traversing the tables based on part-related information.

Adapted Method-Time Measurement (MTM) [4] tables are used by large multi-national automotive producers. Selection of one MTM table from a pool of twenty-two and subsequently performing down selection to obtain one MTM code can be an error-prone process. These issues can be addressed through the use of data-processing and a computational approach. Figure 0.4 shows the issue relating to research objective two in the darkened box.

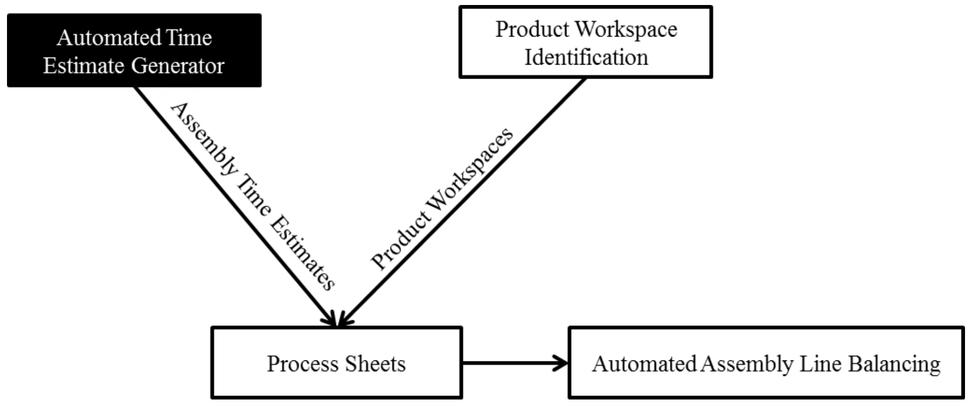
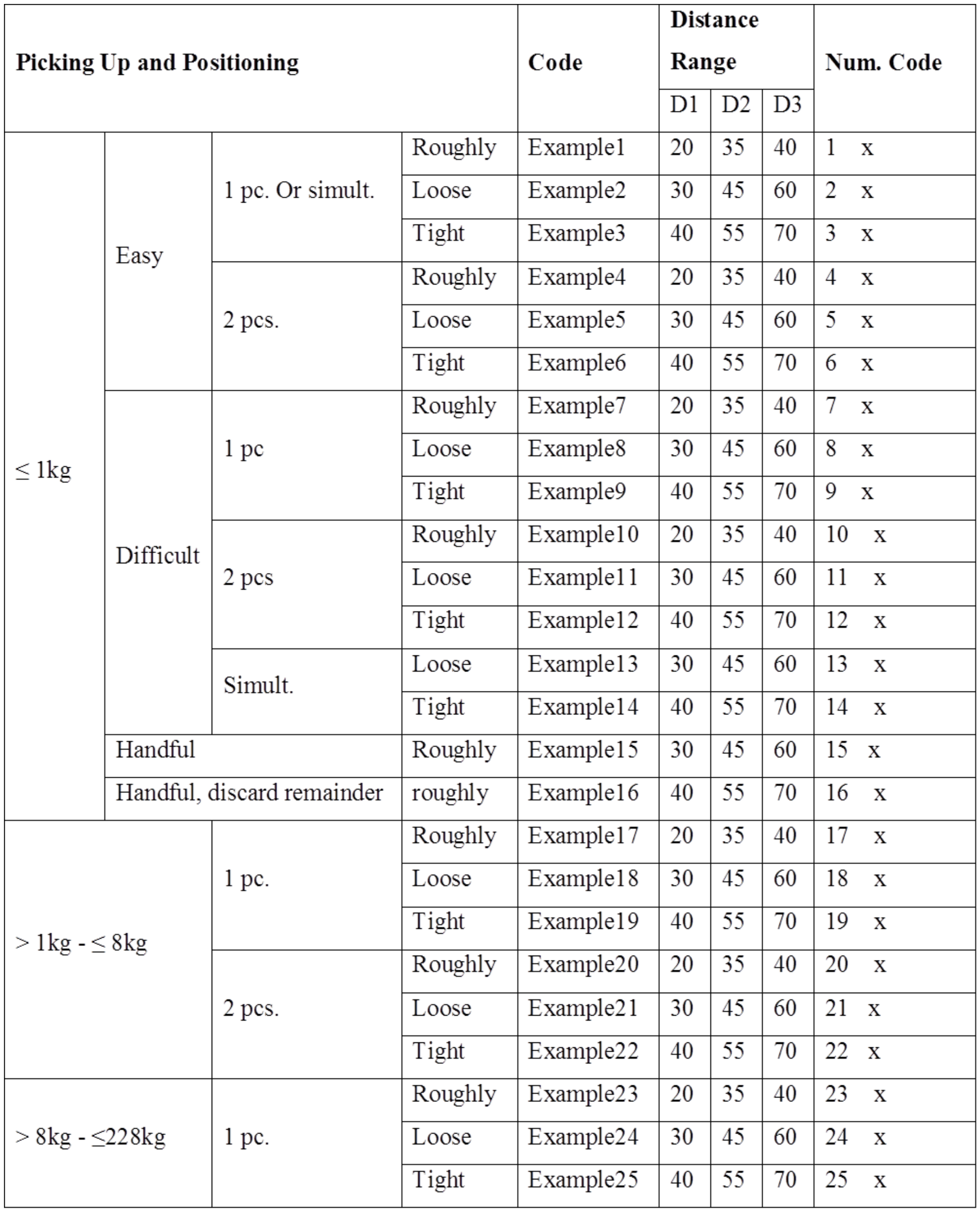

Automated Time Estimate Generator
Product Workspace Identification
Assembly Time Estimates
Product Workspaces
Process Sheets
Automated Assembly Line Balancing

Figure 0.4: Issue Research Objective Two addresses

The following figure shows an example of an MTM table.



Picking Up and Positioning				Code	Distance Range			Num. Code
					D1	D2	D3	
≤ 1kg	Easy	1 pc. Or simult.	Roughly	Example1	20	35	40	1 x
			Loose	Example2	30	45	60	2 x
			Tight	Example3	40	55	70	3 x
		2 pcs.	Roughly	Example4	20	35	40	4 x
			Loose	Example5	30	45	60	5 x
			Tight	Example6	40	55	70	6 x
	Difficult	1 pc	Roughly	Example7	20	35	40	7 x
			Loose	Example8	30	45	60	8 x
			Tight	Example9	40	55	70	9 x
		2 pcs	Roughly	Example10	20	35	40	10 x
			Loose	Example11	30	45	60	11 x
			Tight	Example12	40	55	70	12 x
		Simult.	Loose	Example13	30	45	60	13 x
			Tight	Example14	40	55	70	14 x
	Handful		Roughly	Example15	30	45	60	15 x
	Handful, discard remainder		roughly	Example16	40	55	70	16 x
> 1kg - ≤ 8kg		1 pc.	Roughly	Example17	20	35	40	17 x
			Loose	Example18	30	45	60	18 x
			Tight	Example19	40	55	70	19 x
		2 pcs.	Roughly	Example20	20	35	40	20 x
			Loose	Example21	30	45	60	21 x
			Tight	Example22	40	55	70	22 x
> 8kg - ≤228kg		1 pc.	Roughly	Example23	20	35	40	23 x
			Loose	Example24	30	45	60	24 x
			Tight	Example25	40	55	70	25 x

Figure 0.5: Example of an MTM table

1.2 Outline of the thesis

This section provides a comprehensive outline of this thesis. Figure 0.6 is a pictorial representation of the outline.

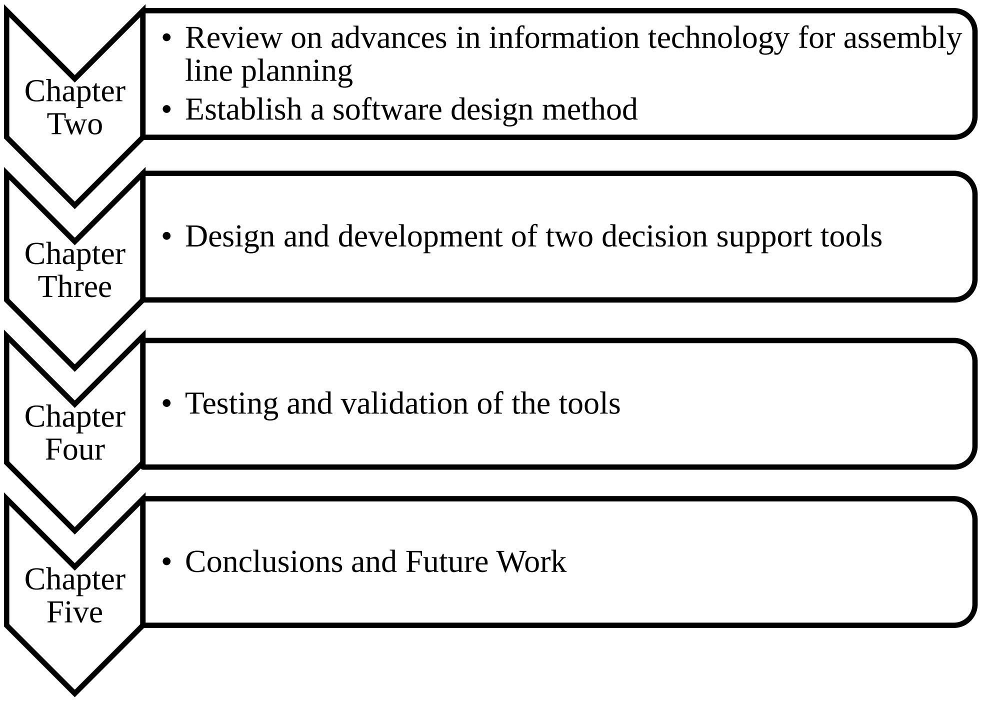

Chapter Two
Review on advances in information technology for assembly line planning
Establish a software design method
Chapter Three
Design and development of two decision support tools
Chapter Four
Testing and validation of the tools
Chapter Five
Conclusions and Future Work

Figure 0.6: Outline of the thesis

The development and use of Information Technology in the field of automotive manufacturing is reviewed in Chapter 2. With respect to each of the research objectives, the potential sources of information that can be leveraged to provide answers are identified. A design process associated with software systems is established.

Chapter Three explains the use of the design process (presented in Chapter Two) to guide the design and development of the two decision support tools. These tools are intended to provide an answer to each of the research objectives.

Chapter Four concerns the testing and validation of the tools that have been presented in Chapter Three. First, an individual testing plan for each of the tools is presented. This is followed by identification and presentation of data that will challenge and validate the effectiveness of the tools with respect to their requirements.

Chapter Five presents the answers to the research objectives that the tools provide. This is followed by a discussion of the larger impact of the tools that have been developed in perspective of a large production management system. Following this is a section that gives direction for improvement and extension of the developed tools.

CHAPTER TWO
LITERATURE REVIEW AND ESTABLISHMENT OF A SOFTWARE DESIGN METHOD

Chapter Objectives:

- Review relevant literature.
- Identify and select sources of information that will be used to develop the decision support tools.
- Provide an outline of the design process with regards to information systems.

1.3 Assembly Line Balancing

The process of assembly line balancing involves assigning process sheets to workstations with the intent of optimizing predetermined factors. Some of the common factors that are considered for optimization are number of stations, throughput time, equal distribution of work load and number of people required to complete the assembly [5]. The most significant assignment constraint comes in the form of precedence. Precedence relations dictate tasks that have to be completed in order for a certain task to be performed.

An assembly line consists of a number of workstations that can be arranged in different ways. Some of these arrangements include U-shaped assembly lines, parallel assembly lines and two-sided assembly lines [5]. A base product is placed on a moving transportation mechanism, such as a conveyor, at the first station. At every workstation, activities are performed on this base product. These activities include assembling parts on the base product, checking the parts that have been assembled and placing parts on the base product for subsequent assembly. The activities that are to be performed at workstations are dictated by assembly process sheets [8].

Market trends, strong competition and shorter product lifecycles have forced companies into producing products that are highly customized [5,6,9,10]. Another requirement for companies is to reduce lead times and maintain the Just-In-Time (JIT) manufacturing principles [11]. This necessitates a reduction of batch sizes. It also implies a need to move away from mass production strategies and move towards mass customization strategies. A mixed model assembly line has proven to be an efficient method to produce mass customized products. With mixed-model assembly lines, product platforming is consciously practiced. All the "customized" products are built on the same base product and true customization involves the inclusion/exclusion of optional features [5,12]. This is considered to be a combination of collaborative, adaptive and cosmetic mass customization [13]. The BMW 7 series of 2009 had over 200 options in addition to twelve exterior colors and twelve trim colors. This theoretically adds up to 3.5 E30 vehicle configurations [14]. Chul Ju Hyun et al. define mixed model assembly lines to be "a type of production line where a variety of product models similar in product characteristics are assembled" [15].

Boysen [5] identifies the problems associated with balancing mixed-model assembly lines within the JIT environment:

- Different variants imply varying cycle times and the possibility of reduced worker utilization.

- Tasks have to be assigned to workstations such that there is no intrusion of line side associate workspaces.

These complexities make line balancing a challenging and time consuming task and warrant the need to have computer supported line balancing. Attempts have been made to completely automate the line balancing process [5–7]. However the process of line balancing has been hard to completely automate. Capturing and codifying expert knowledge has proven to be a challenge [16]. Researchers have found it difficult to include all necessary constraints in line balancing algorithms in order to eliminate human input. In [16] it has been recognized that the assembly line balancing problem has "inherent problem complexity and diverging problem settings". A gap that remains to be filled is in the form of a decision support tool that will assist in performing semi-automated line balancing.

It is important to ensure that reconfiguring the line does not result in a line side associate intruding upon another line side associate's workspace. This constraint has to be handled by the line balancing algorithms as well. This constraint is considered by the assembly line balancing heuristic presented in [3]. The product workspace information is manually input in this heuristic. For a completely automated algorithm to consider this constraint, the definition and recognition of workspace that every part requires must be formalized and automated.

There is a need to leverage and apply information from the line balancing algorithms as well as information from assembly line balancing experts. It has been observed that manually editing outputs of line balancing algorithms is unintuitive and error-prone. This further warrants the need to have a decision support tool for semi-automated assembly line balancing.

1.4 Time Study Estimation Methods

The two most basic elements of assembly process sheets are assembly work instructions and their corresponding time study estimates. These time estimates are obtained from predetermined time standards. Some of the commonly used time standards are Methods-Time Measurement (MTM), Modular Arrangement of Predetermined Time Standards (MODAPTS) and Boothroyd and Dewhurst time estimates [4,8,17,18]. Of these, the MTM time standard is of particular relevance to the research presented in this thesis. Ford's production management system [8] uses MODAPTS. Their production system has the capability to parse through all work instructions and assign MODAPTS codes to each. A similar system is required that ties structured work instructions together with MTM estimates.

The MTM tables were developed by Maynard et al. [4] to eliminate subjectivity in the time analyst's judgment and evaluation of time required to perform a certain task. The MTM tables achieved this goal by creating a chart of time estimates and relating them to: physical attributes of the part to be handled/assembled, required handling precision, use of tools, number of parts etc. All these parameters have been defined in an objective manner to eliminate chances of inconsistency. Each MTM table relates to an action verb. The number and types of parameters associated with each table varies.

Assigning an MTM estimate to every work instruction involves:

1. Selecting the most appropriate MTM table.
2. Identifying sources of information to perform filtering.
3. Applying the necessary filters to arrive at one MTM estimate.

This is a cumbersome process. Peterson [2] has provided product-process coupling by linking work instructions to part numbers. This product-process coupling can be leveraged and enhanced to automate the selection of an MTM time estimate for every work instruction. Miller and colleagues [19] used artificial neural networks to relate work instruction text to time estimates. Several models were created that used verb, verb+object, verb+object+quanity, and verb+quantity. In this method, the estimated time is normalized to the number of work instructions within a process sheet. However, the results from the research were inconclusive with an average error of 271%. The proposed work differs from the approach by Miller and colleagues. First, the proposed approach provides a directed, filtered relationship work instruction text and specific MTM tables. Seconds, the time estimates for a work instruction are not generated by a neural net, but instead they are obtained from the MTM tables themselves.

1.5 Gaps identified from literature

The following table summarizes the gaps that have been identified and the domain to which these gaps belong.

Table 0.1: Gaps identified from literature

Domain of Problem	**Gap Identified**
Assembly Time Estimation	Automated method to choose MTM time estimates
Assembly Line Balancing	Formalizing and automated recognition of product workspaces
Assembly Line Balancing	Decision support tool to support assembly line balancing

1.6 Collaborative Process Planning

The need for collaborative process planning stems from the globalization of manufacturing and the complex nature of balancing mixed model assembly lines. Liu and Young [20] use the term Global Manufacturing Co-ordination (GMC) to address issues that pertain to integration of the planning process within and across plants.

One of the primary purposes of JIT manufacturing is to reduce lead time. JIT manufacturing must be supported by an able information system to ensure minimal lead times [21]. Howard et al. in [21] have studied the information system changes that are required within and between part suppliers and car manufacturers, in order for them to cope with the transition from mass production to JIT manufacturing. They have recognized the barriers to the changes within the information systems.

Frohlich et al. [22] review the viability of Demand Chain Management (DCM) in today's world (with the increased use of the Internet and the World Wide Web). DCM is the concept that drives JIT manufacturing. In DCM the end customers drive the entire production process, from manufacturing of components to producing subassemblies and to final assembly of these subassemblies. Although the benefits of DCM were recognized "many years" ago, the lack of an able information system prevented its implementation.

The Internet and the World Wide Web allow for databases to be setup, populated, manipulated and accessed from locations throughout the world. Noh et al. utilize an Internet based database system to allow for collaborative assembly process planning [1]. A requirement for their system is that the database must include all the information pertaining to the products, assembly processes and resources of the assembly line. It is shown that, when compared to manual assembly line balancing, the collaborative assembly line balancing method results in a better balance efficiency. A better balance efficiency implies reduced worker requirement and higher worker utilization. It is also shown that balancing the assembly line using the collaborative method requires less effort in terms of time for the planners. Noh et al. do not emphasize the need for product process coupling. Product process coupling is the driving force behind a collaborative process planning system. It ensures consistent data management, documentation of product lifecycle and enables continuous improvement of processes.

Many global manufacturers have a centralized location where process sheets are populated with work instructions, time studies and other information. These process sheets are then deployed to global locations. Collaborative process planning via the internet will enable interactions between the centralized locations and the peripheral plants, and will allow for continuous improvement of the processes being employed at all interacting locations (see Figure 0.1).

Rychtyckyj et al. [8] recognize the need for plant level optimization by considering (among other factors) walk times of the workers. Walking is considered to be a non-value added task and under the JIT manufacturing principle any non-value added task is considered a waste and must be eliminated [23,24]. In order to minimize the non-value added operation of walking, the plant layout will have to be reconfigured or the sequence of work instruction steps will have to be altered. Such local plant level modifications should be made available on a centralized database for perusal by other plants (see Figure 0.1). This is another situation that warrants the need for collaborative process planning.

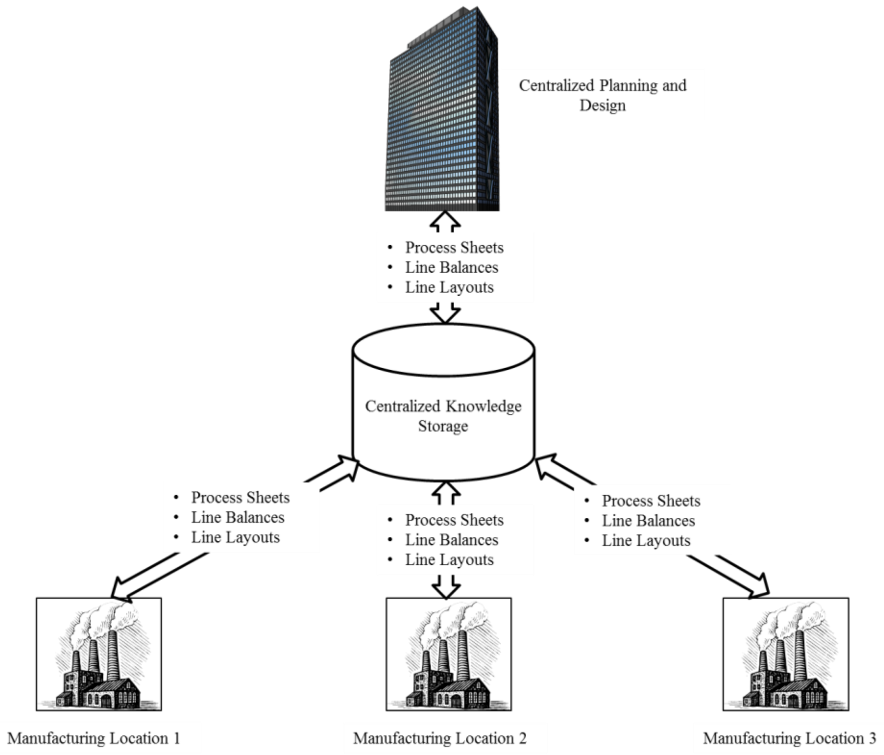


Figure 0.1: Example of decentralized manufacturing and the need for collaborative process planning by use of databases

The situations described above are advanced uses of a collaborative production management system. It is imperative to build a strong foundation for such a system. The research objectives look to contribute towards this end.

1.7 Leveraging information sources to answer research objectives

The following subsections explain the use of information sources that are employed to develop tools in order to answer the two research objectives.

1.7.1 Research Objective One

The first research objective concerns the recognition and application of data to automate the recognition of product workspaces. Computer Aided Design (CAD) has been used extensively by manufacturing companies. One of the most elementary pieces of information associated with, and available in CAD are the Cartesian space coordinates of the part that is being modeled.

CAD assemblies are created, most often, in one of the two following ways: first, by creating virtual mates to relate the relative location of parts or; second, by positioning parts relative to a fixed coordinate system by means of a homogenous transformation of coordinates. In either case, the coordinate information of the parts that constitute the assembly can be obtained. This information can be used to derive the recognition of location of parts within a vehicle.

Application Programming Interface (API) is a feature that is available in most commercially available CAD modeling software. API's allow users to develop customized functions that they want to see the software perform. An API can be developed to extract the relative location of parts from a CAD assembly of the vehicle. This information can then be stored on an online database for later use.

A standard method of diagrammatically representing the location of the part within the vehicle was found to be highlighting appropriate cells on a nine cell grid superimposed on the top view of the vehicle. An example of this can be seen in Figure 0.2. With knowledge of the geometric coordinates of the vertices for every part (relative to the assembly of the vehicle), the cells that every part occupies can be determined and highlighted. The highlighted cells represent the Product Workspace. This diagrammatic representation is used in the process sheets of a leading automotive manufacturing company.

The knowledge of the Product Workspace is used during the line balancing process. Parts are assigned to process sheets. Process sheets are assigned to workstations during line balancing. If two process sheets, that have different parts with the same Product Workspaces, are assigned to the same station, then there is a high chance of a line side worker intruding upon the working area of another line side worker. Product workspace information is used in line balancing to avoid this possibility.

In addition to extracting coordinate information, the API can be used to extract volume and density information of parts. This information is employed to automate the parsing of MTM tables.

To answer the first research objective a Product Workspace Identification Tool was developed. The design and implementation of the tool is explained in the following chapter.

1.7.2 Research Objective Two

Research objective two tackles the need to reduce the effort and error associated with generation of time estimates from MTM tables, by leveraging product process coupling. It is well established that this support is going to come from a network of sources connected within a developing production management system.

The first step that must be taken towards automation of the generation of MTM time estimates is to have an online version of the MTM tables in a form that enables them to be parsed programmatically. It is necessary to coordinate the various information sources and organize them. Database development and application techniques must be employed. A semi-automated MTM parsing tool has been designed and implemented which has been explained in the following chapter.

1.8 Design Process followed for the tools developed

The implementation of any tool must be performed in an organized manner. In order to properly define an organized method of implementation a design process was followed (see Figure 0.2). The following section describes the various stages of this design process.

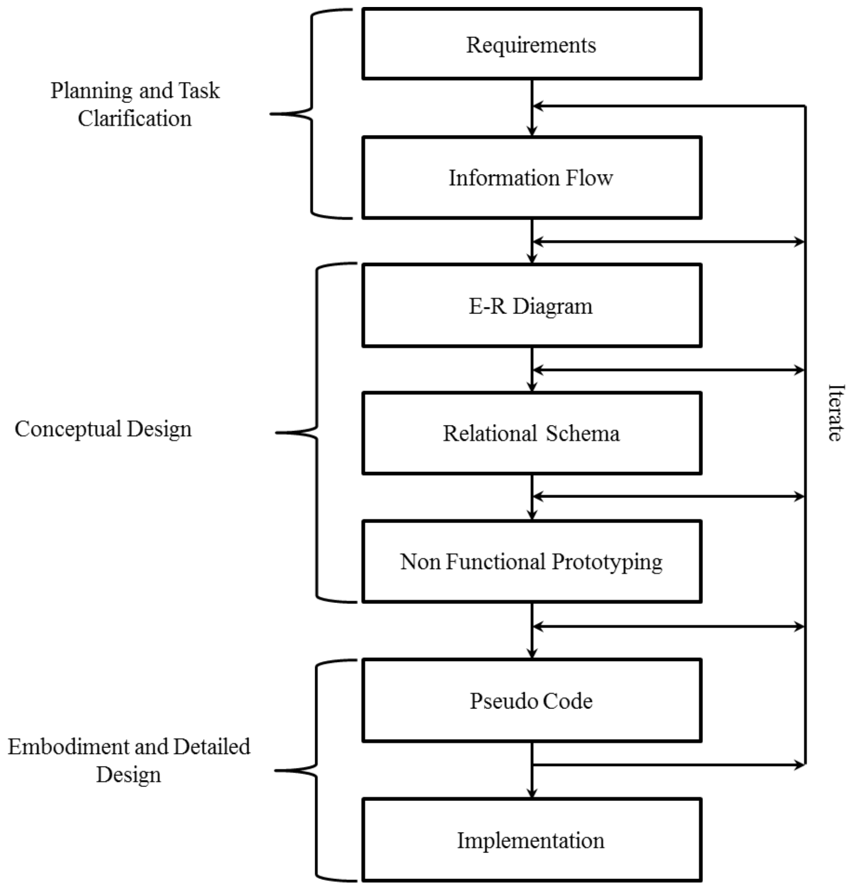


Figure 0.2: Software design process employed

The process shown above is greatly inspired from the design process prescribed by Pahl et al. [25]. Although their design process was prescribed for engineers attempting to design and develop products within the mechanical engineering domain, it was found that the process was adaptable to prescribe the design and development of knowledge management systems as well.

1.8.1 Planning and Task Clarification

The first step in the design and development of any product is to ascertain the required and the desired functionalities. This will be enabled by eliciting requirements from the customer of the product. Some requirements can be obtained from the problem statement directly. However, customer expectations are not always directly available and elicitation is required to bring them to the fore. The requirements presented used in this research are primarily to guide and the design and implementation of the tools.

Once the requirements have been obtained and understood, the next step is to identify information flow between the system modules. This will help to identify system module interactions with the database. It provides a high-level visual plan of action for the implementation of the tools. Figure 0.3 shows an example of the type of information flow representation that will be used in this thesis.

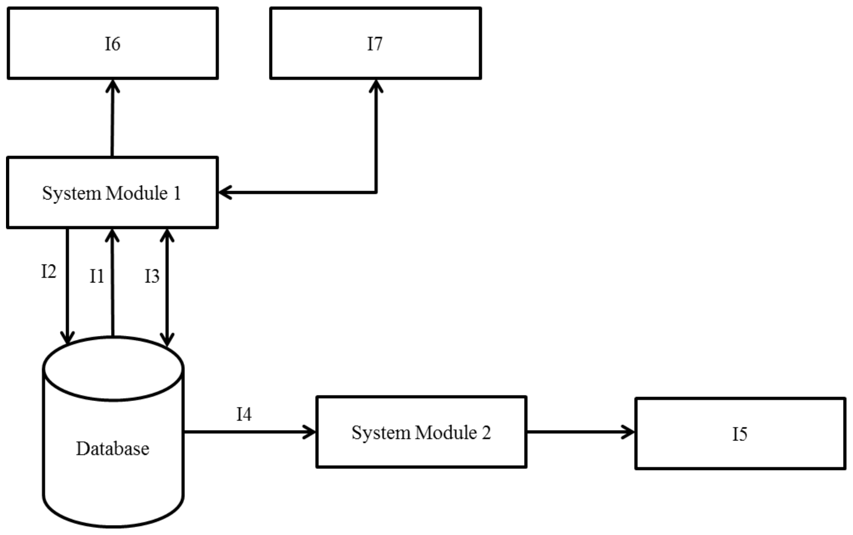


Figure 0.3: Sample Information Flow Representation

Table 0.2: Explanation of information flow representation elements

Information flow element	**Explanation**
System Module 1, System Module 2	These represent the modules of the system that are a communication medium between the database and the user. These hold the algorithms for intelligent retrieval and use of information.
I1, I4	These are packets of information that the system modules need as input and are provided by the database.
I2	This is information that is generated by the system module and stored in the database.
I3	This is information that is used as input by the system module, manipulated by it and then stored in the database.
I5, I6	This is information that is generated by the system module and presented to the user.
I7	This is information that is presented to the user and his/her feedback is used by the

Information flow element	Explanation
	system module.

1.8.2 Conceptual Design

With the requirements understood and an information flow identified, the next step is to identify and formalize a logical structure between the data sources. Chen et al. provide a method to perform this type of data modeling by means of an Entity-Relationship (ER) diagrams [26]. In an ER diagram an entity can be described as an "aspect of the real world which can be distinguished from other aspects of the real world" [27]. Relationships between the entities define the logical structure being captured. Entities have attributes that describe its properties. In some cases, relationships can have attributes as well. An example of an ER diagram is shown in Figure 0.4.

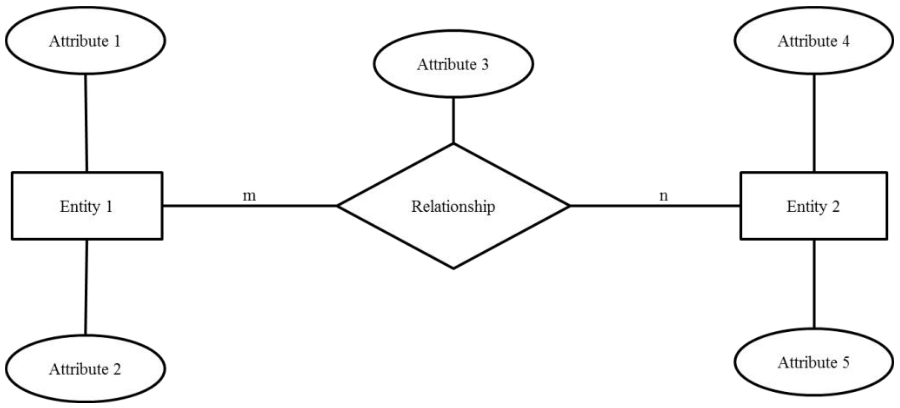


Figure 0.4: Sample Entity-Relationship Diagram

Table 0.3 explains the elements of the ER diagram.

Table 0.3: Explanation of elements of ER Diagram

Element	Explanation
Entity 1, Entity 2	The aspects of the real world that fall within the boundaries of the system being modeled.
Relationship	A descriptive relationship between the two entities.
Attribute 1, Attribute 2, Attribute 4, Attribute 5	Properties of the respective entities.
Attribute 3	Property of the relationship.
m, n	The cardinality of the relationship.

The next step is to develop a relational schema based on the relational data model [27]. This will result in the tables that will be stored within the database.

This method of data modeling allows for easy maintainability and extensibility of the system being designed. Entities and the corresponding attributes can be added to the ER diagram and related to existing entities. The tables on the database can then be added based on the additions to the ER diagram.

After the logical structure and organization of the data has been formed, the next step is to create a non-functional prototype of the user interfaces for the system modules under development. This allows for critical design and implementation decisions to be made at an early stage.

1.8.3 Embodiment and Detailed Design

With conceptual design completed, the next step is to create all the user interfaces that were designed in the “Non-functional Prototyping” stage. Pseudo codes or flow charts are two common methods employed to give high level descriptions of the algorithms that will drive the interaction between the user interface and the data sources. The intelligence and automation that the tools provide all come from the algorithms that they employ.

In this research pseudo codes have been employed as opposed to flow charts. The final step of the design process is to translate the pseudo code to the coding language of the platform on which the tool is being developed.

1.8.4 Iteration and Refinement

As progress is made in the design and implementation of the system, more often than not, changes will have to be made to the design of the system. The stage at which this change will need to be made is tough to predict. However, this change must percolate down to all the subsequent design stages and the affect that this change has on the antecedent stages must be analyzed. The implementation of changes to software systems consumes relatively less time and effort as compared to mechanical, hardware systems. The next chapter applies this design process to the development of two tools that each answers one research objective.

CHAPTER THREE
DESIGN AND IMPLEMENTATION OF THE DECISION SUPPORT TOOLS

Chapter Objectives:

- Introduce the tools that have been developed.
- Explain the design process for each of these tools.
- Present the developed tools.

The two research objectives presented in the previous section are each answered by means of specific decision support tools. This section introduces these tools and discusses their design, development and implementation.

1.9 Product Workspace Identification Tool

Product workspace information is entered on process sheets manually. Product workspace information is used during line balancing to avoid intrusion of line side worker's workspace by another line side worker [3]. Automating the recognition of product workspaces will reduce the work load on the process sheet authors. This will also ensure consistency and accuracy of product workspace information for use in line balancing algorithms. To this end, the Product Workspace Identification Tool (PWIT) will answer the first research objective presented in 0.

1.9.1 Requirements

The following table lists the requirements for the PWIT.

Table 0.1: Requirements for PWIT

1.	Must identify product workspaces with respect to a nine cell grid as defined by the automotive OEM that supported this research (see Figure 0.1).
2.	Must be designed such that it is flexible with respect to product workspace definitions.
3.	Must identify all product workspaces for every part.
4.	The PWIT must maintain part-process coupling by relating product workspaces to their respective part numbers.
5.	Assembly operations can be performed on a part only when its product workspace has entered the workstation.

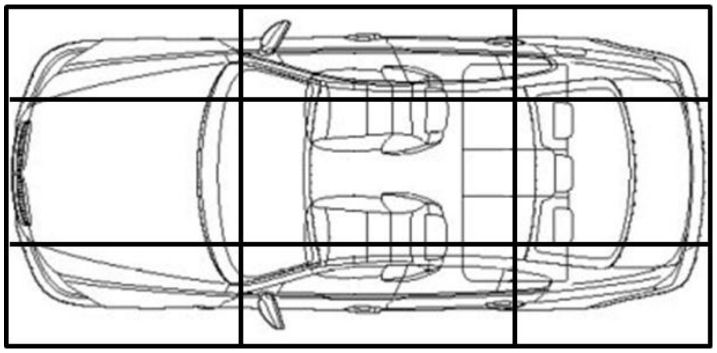


Figure 0.1: Nine cell grid on the top view of a vehicle

A direct customer to the product workspace information is the process sheet authorship tool developed by Peterson [2]. This authorship tool uses online databases to support the authorship of work instructions. All the sentence elements are stored on the online database. All the work instructions on every process sheet are authored for a specific part. One degree of part-process coupling is achieved here by linking the process sheet to the part number that it is operating on (see Figure 0.2). Using these part numbers, the degree of part-process coupling can be increased by linking part information (product workspaces, part geometry and mass properties) to the process sheets.

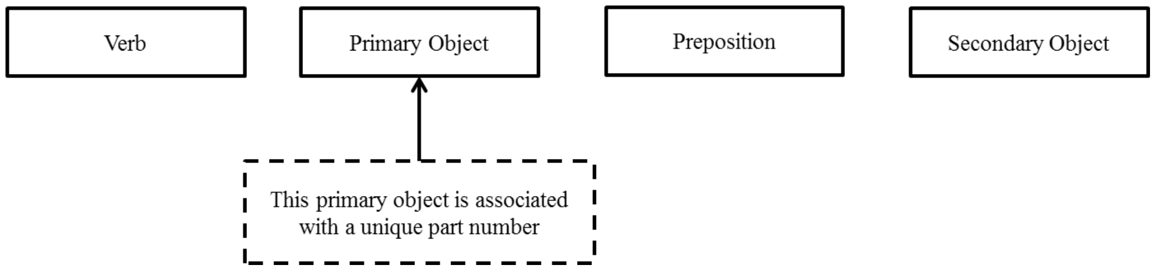


Figure 0.2: Standard work instruction structure

The authorship tool supports the use of online databases to automate the population of product workspace information in process sheets. Requirement three ensures that the authorship tool's database is provided with the necessary product workspace information. Requirement one is based on observations made from a large scale OEM's process sheets.

1.9.2 Information Flow Diagram

The Product Workspace Identification Tool essentially requires knowledge of part information. The tool also requires a high level of automation. Both of these requirements can be achieved by creating a tool within the CAD development environment.

As mentioned previously, CAD assemblies are most often created in one of two ways; first, by creating virtual mates between parts in order to define their spatial location, orientation and degrees of freedom; or second, by positioning parts relative to each other through a homogeneous transformation of coordinates. In both of the methods, information of the part's location relative to a fixed origin is available and can be extracted by developing API's within a CAD environment. In order to implement this approach an information flow diagram was created (see Figure 0.3).

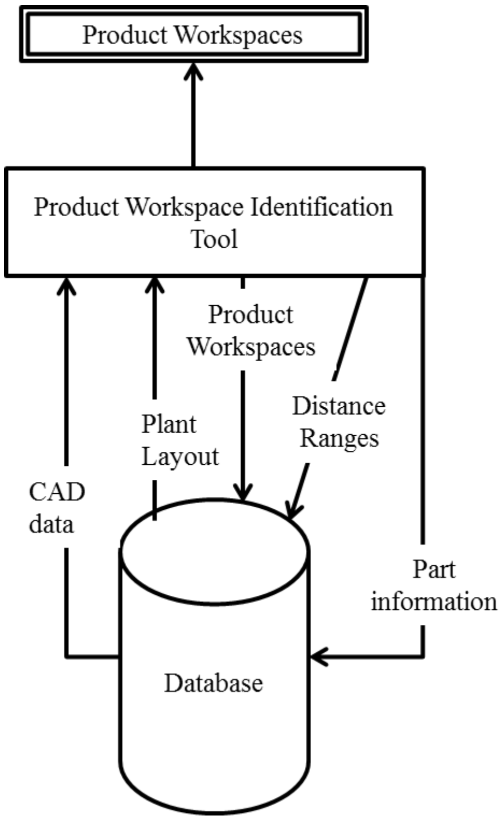


Figure 0.3: Product Workspace Identification Tool - Information Flow Diagram

Table 0.2: Explanation of Product Workspace Identification Tool Information Flow elements

Information flow element	Explanation
Product Workspace Identification Tool	SolidWorks API tool that is needed to identify product workspaces.

Information flow element	Explanation
CAD data	SolidWorks assembly file of the final product.
Plant Layout	Manually input plant layout that describes the location of part bins.
Product Workspaces	For each part on the assembly file, product workspaces are computed and shown to the user and stored on the database.
Part Information	Mass, volume and dimensional information of every part is stored on the database.
Distance Range	Distance between the part's product workspace and the bin where the part is located.

Plant layout, Part Information and Distance Ranges are used by the automated MTM Time Estimate Generator (research objective two).

1.9.3 ER Diagram

After the flow of information is recognized and laid out, the relationships between the entities of the system have to be laid out. Figure 0.4 shows a concise ER diagram for the Product Workspace Identification Tool. The attributes of the entities have not been included in the ER diagram due to lack of space.

Table 0.3 lists out each entity of the ER diagram, its description and its associated attributes. The entities and attributes used are a subset of all possible entities and attributes. Only the information being captured and generated by the Product Workspace Identification Tool has been modeled here. An ER diagram is extensible and additional information can be modeled and captured when required. For example, Work Instruction comprises of an action verb, primary part description, preposition and secondary part description. In the ER diagram, the relationship between the work instruction and the primary part is the only one that is captured.

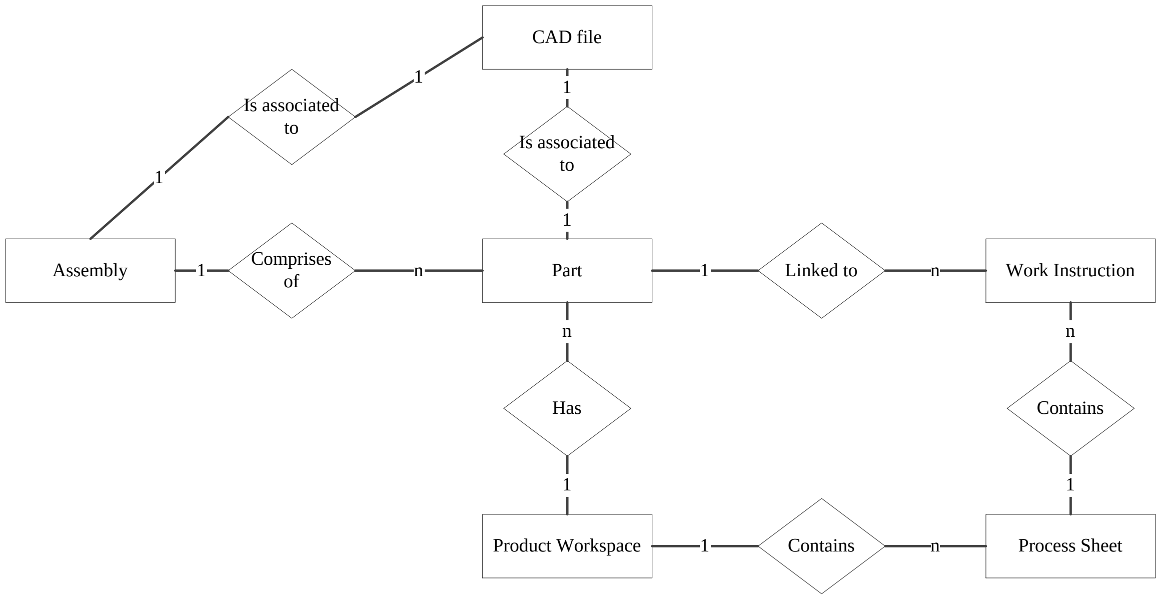


Figure 0.4: Concise ER Diagram for Product Workspace Identification Tool

Table 0.3: Details of components of ER diagram for PWIT

Entity	**Description**	**Attributes**
Process Sheet	It is a document that contains work instructions, time study analyses, product workspace information and other meta information.	<u>ID</u>, Number, Title

Entity	**Description**	**Attributes**
Work Instruction	A set of work instructions dictate the assembly operations that the process sheet is associated to.	ID, Process_Sheet_ID, Step_Number, Verb_ID, Object1_ID, Prep_ID, Object2_ID
Part	The object whose state is directly affected by the work instruction. Every work instruction operates on one primary object.	ID, Part_Number, Description

Entity	**Description**	**Attributes**
CAD file	Every part and assembly is associated to a CAD file. This CAD file holds information regarding the geometric size and mass of part.	ID, Part_Number, Dimensions, Mass, Volume
Assembly	It is a virtual entity that is an aggregation of the constituent parts.	ID
Product Workspace	It is the space that each part occupies within a nine cell grid superimposed on the top view.	ID, Part_Number, LM, LV, LH, RM, RV, RH, MM, MV, MH (nine cell names)

The table above lists each of the entities of the ER diagram for the PWIT. The relationships of the ER diagram are explained in Table 0.4. The cardinality of a relationship between tables defines the type of interaction between them. This concept is best explained by means of an example.

Table 0.4: Explanation of relationships in the ER diagram

Relationship	**Description**	**Cardinality**
Process_Sheet_contains Work_Instruction	One purpose of process sheets is to convey assembly operations to line side associates. This is done by means of work instructions contained within the process sheet.	1:n
Process_Sheet_contains_Product_Workspace	Process sheets also contain information regarding the product workspace of the part that it operates on.	1:n
Work_Instruction_linked_to_Part	Every work instruction on a process sheet is linked to a part.	n:1

Relationship	**Description**	**Cardinality**
Part_has_Product_Workspace	Every part is associated with its (product) workspace.	1:n
Part_associated_to_CADfile	Every part has its own CAD file.	1:1
Assembly_associated_to_CADfile	It is the assembly of parts in CAD environment.	1:1

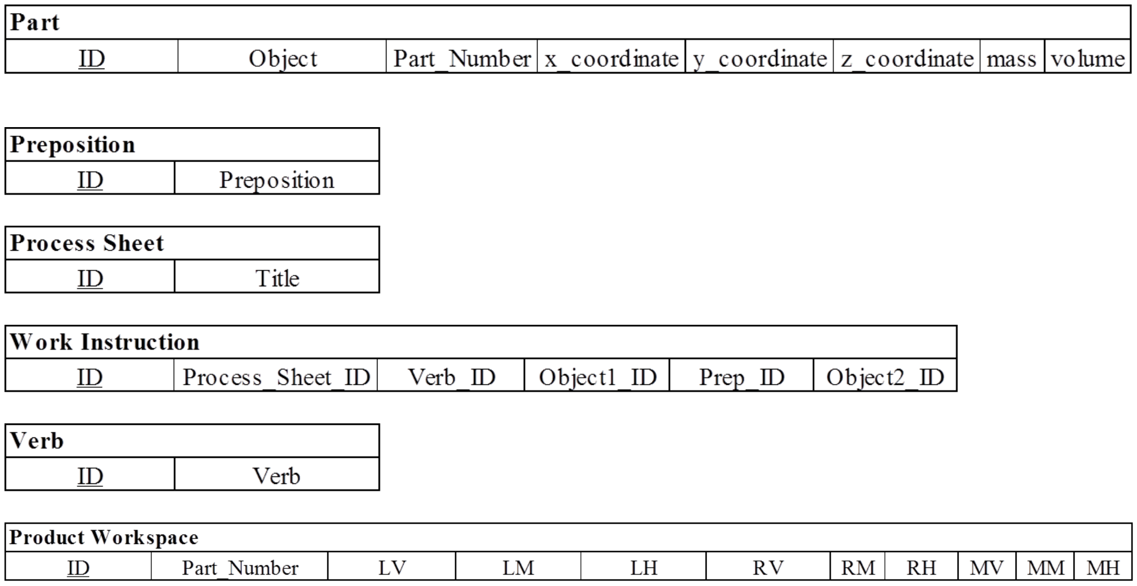
The “Assembly” entity is essentially a CAD file in which the constituent parts are placed relative to each other. This entity helps in laying out a sensible logical structure between information elements, but it is not modeled beyond this point because it is a virtual entity.

The “Product Workspace” entity has, as its attributes, the nine product workspace cells. These attributes assume binary values based on the presence/absence of the part in each cell. This type of modeling does not follow conventional modeling schemes, but is kept this way to assist in implementation during process sheet authoring. It is for the same reason that the “Product Workspace” shares a “1:n” relationship with “Part”.

Similar to the “Assembly” entity, “CAD file” is a virtual entity that is linked directly to a “Part Number”. Beyond this point, the “CAD file” entity is merged with the “Part” entity. Therefore, the “Part” entity inherits all the attributes of the “CAD file” entity.

1.9.4 Relational Schema

The ER diagram is then used to construct the database tables. This is done within the MySQL environment. The tables that are created are presented in the remainder of this section. The code for the creating these tables can be found in Appendix A (A.1).



Part

ID	Object	Part_Number	x_coordinate	y_coordinate	z_coordinate	mass	volume

Preposition

ID	Preposition

Process Sheet

ID	Title

Work Instruction

ID	Process_Sheet_ID	Verb_ID	Object1_ID	Prep_ID	Object2_ID

Verb

ID	Verb

Product Workspace

ID	Part_Number	LV	LM	LH	RV	RM	RH	MV	MM	MH

Figure 0.5: Database tables created for Product Workspace Identification Tool

1.9.5 Non-Functional Prototype

A non-functional prototype allows the implementation of the system to take more shape. User interfaces are conceptualized at this stage. The Product Workspace Identification Tool is a CAD based tool and this tool has been implemented through a CAD system's API. The non-functional prototype of the PWIT can be seen in Figure 0.6.

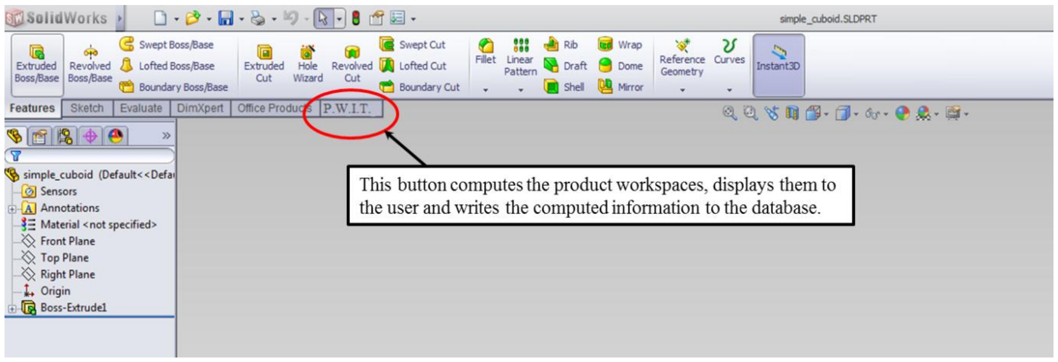


Figure 0.6: Non-Functional Prototype of Product Workspace Identification Tool

1.9.6 Pseudo Code – Guide for implementing PWIT

The pseudo code of this tool is divided into two sections: the first section deals with the extraction of part information from the CAD models of the part; and the second deals with the generating output of the product workspaces for the user and the database.

For the extraction of part information the pseudo code is the following:

```
1. Get active document
2. Get assembly
3. Get components of assembly
4. For each body of every component
5. Get bounding box
6. Get name, mass and volume
7. Write all information to CSV files
```

A bounding volume is a closed volume that contains the geometries under consideration completely within it. There are several types of bounding volumes. Bounding spheres are bounding volumes that are characterized by a center point and a radius. A bounding cylinder is a bounding volume that is characterized by a height, a center point and a radius. A bounding box is the smallest cuboid that can completely contain the geometry being analyzed. There are two types of bounding boxes. The first is an axis-aligned bounding box. Here the axes of the bounding box are aligned with the global co-ordinate system axes in which the assembly is created. The second type is the oriented bounding box. This type of bounding box has its coordinate axes aligned arbitrarily with respect to the global coordinate system axes. For the application required in PWIT, axis-aligned bounding boxes are more suited. From this point on axis-aligned bounding boxes will be referred to just as ‘bounding boxes’.

The coordinates of the bounding box can be compared with the coordinates of the nine cell grid and can therefore be used to determine the product workspaces of every part. Figure 0.7 shows an example of a bounding box for a cylinder.

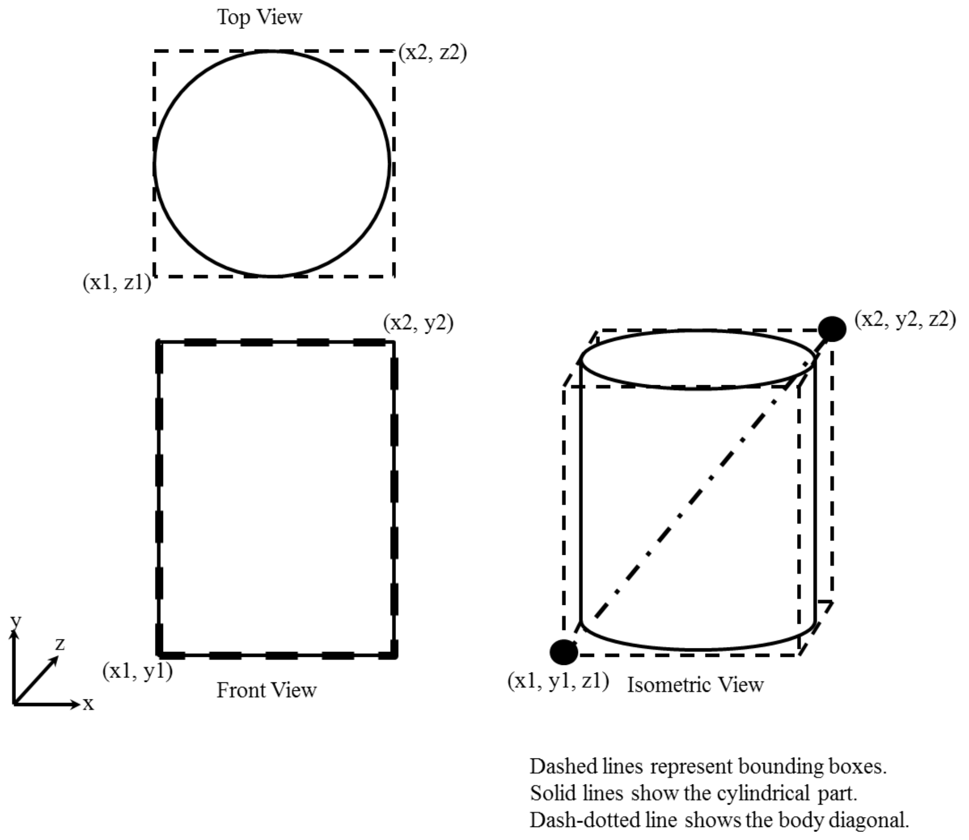

Top View
(x2, z2)
(x1, z1)
(x2, y2)
(x1, y1)
Front View
y
z
x
(x2, y2, z2)
(x1, y1, z1)
Isometric View
Dashed lines represent bounding boxes.
Solid lines show the cylindrical part.
Dash-dotted line shows the body diagonal.

Figure 0.7: Example of bounding box

An important assumption that this pseudo code makes is that the constituent parts of the assembly each have their corresponding part numbers as their name in the assembly tree (see Figure 0.8). This ensures part-process coupling. This naming convention is used by a large scale automotive OEM.

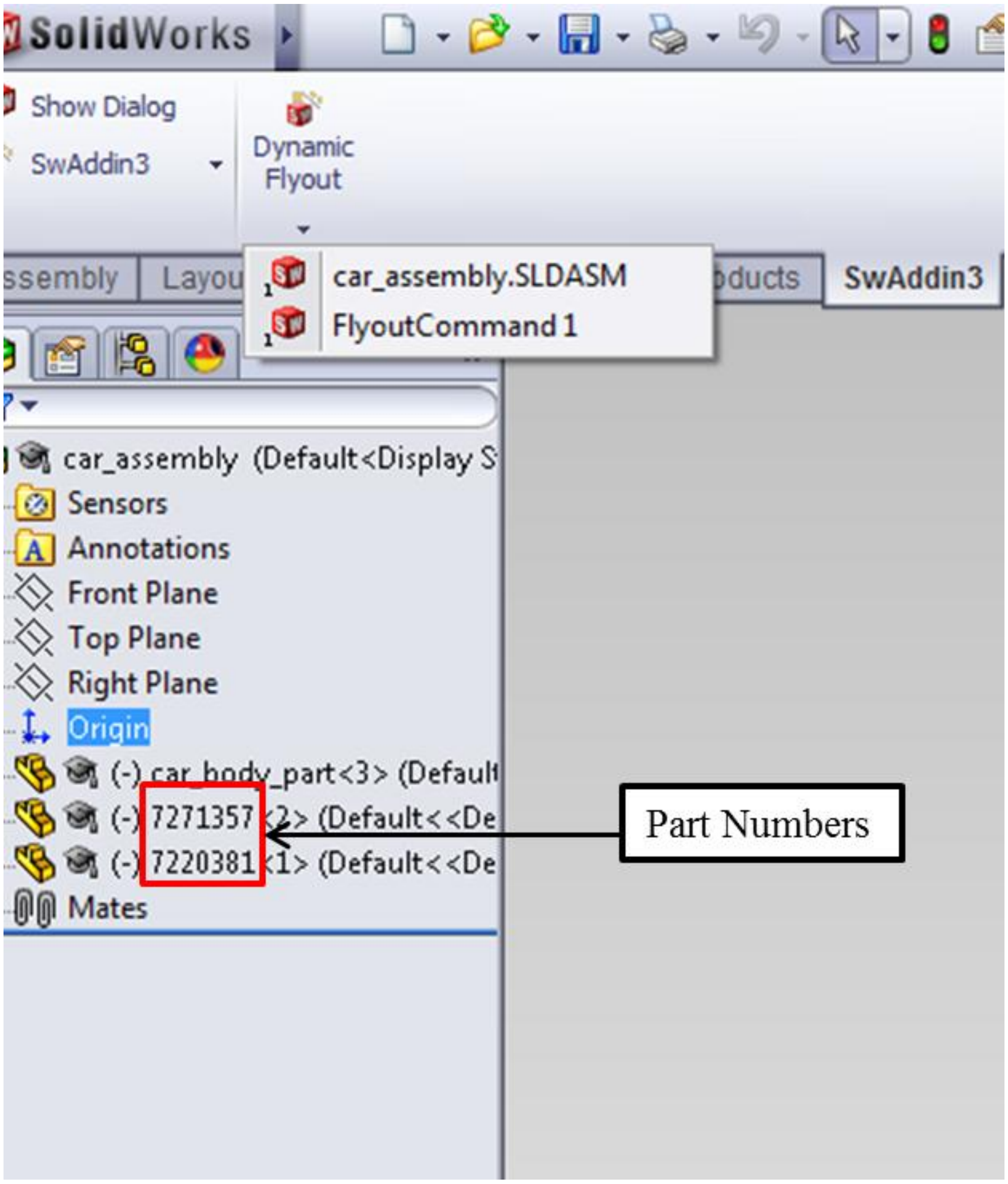


Figure 0.8: Assembly tree with part numbers as component names

For the use of the bounding boxes to obtain product workspaces, the pseudo code is the following:

```
1. Get bounding boxes for every part
2. Calculate and store coordinates of the nine cells
3. For every part
4. Compare the x and z coordinates of the part to
   the x and z coordinates of the nine cells
5. Record the product workspaces
```

1.9.7 Implementation

SolidWorks API's were used to create a part information extraction tool within the SolidWorks modeling environment. The API works on 3D assembly drawings of the final product whose constituent parts' product workspaces are to be obtained. The structure of a SolidWorks assembly model (boundary representation) must be understood (see Figure 0.9) in order to access and extract required information.

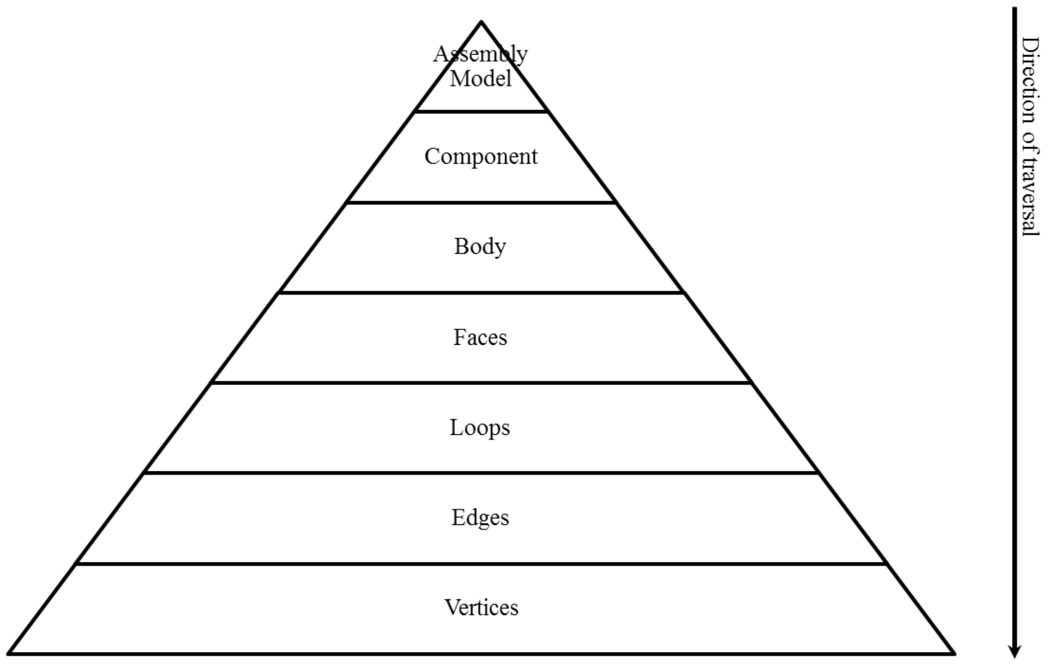


Figure 0.9: Structure of assembly models in SolidWorks and the required direction of traversal

A vertex is the boundary of an edge and is formed where two or more edges meet. In a well-designed solid model there will never be any open geometry. All vertices are the meeting point of two or more edges. In Figure 0.10, Edge 1 and Edge 2 coincide to form Vertex 2. Vertex 1 and Vertex 3 are formed where other edges (not shown in the figure) meet Edge 1 and Edge 3 respectively.

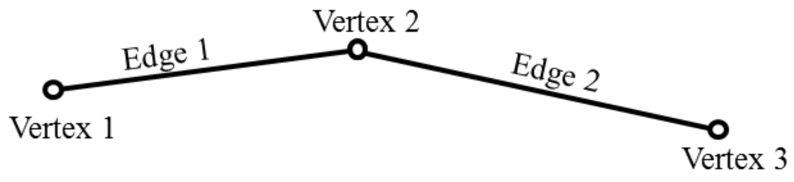


Figure 0.10: Diagrammatic representation of vertices

An edge is formed where two or more faces meet. In Figure 0.11, Face 1 and Face 2 meet to form Edge 1.

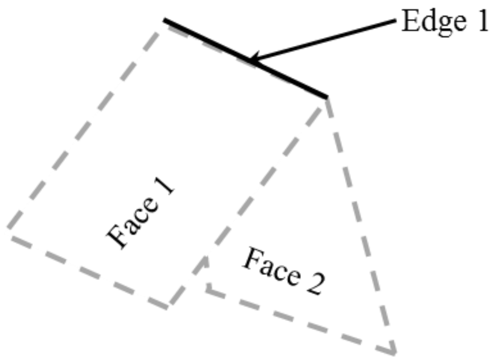


Figure 0.11: Diagrammatic representation of an edge

Faces are the bounding surfaces of a closed solid. Figure 0.12 shows a cuboid and three of its face (Face 1, Face 2 and Face 3). The other three faces are hidden in this view of the cuboid.

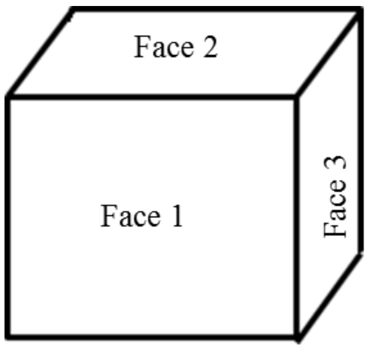


Figure 0.12: Diagrammatic representation of faces

A loop is an ordered, connected, closed set of edges. Every face will have only one outside loop. A face can have any number of inside loops. If the number of inside loops on a face is zero, it implies that there are no features that exist on that face (see Figure 0.13). If there is one or more inside loops on a face, it implies that there exists one or more features on that face (see Figure 0.14).

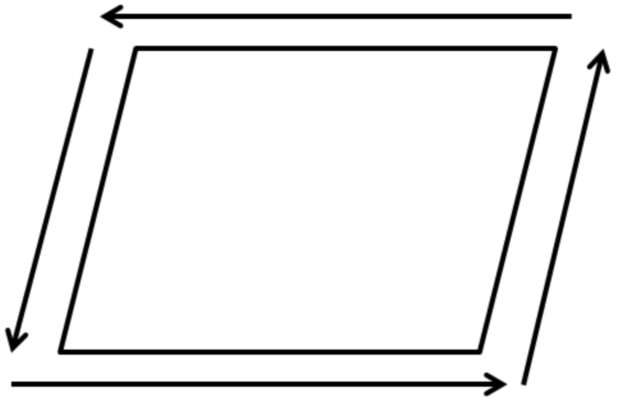


Figure 0.13:Face with one outside loop and no inside loops

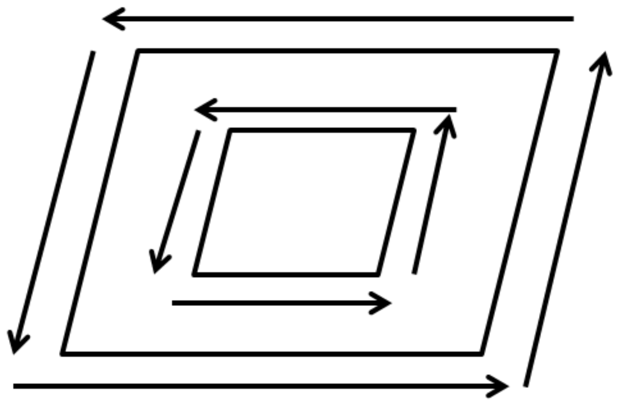


Figure 0.14: Face with one outside loop and one inside loop

For the application in the Product Workspace Identification Tool, the information that is required can be found at “Body”. The information extracted here (part dimensions and mass properties) are stored on CSV files that are then used by a program developed in MATLAB for the purpose of recognizing and visualizing the Product Workspaces. This MATLAB program also writes the information from the CSV files to the online database. The MATLAB program was converted to an executable standalone file by use of MATLAB’s Compiler Toolbox. This executable was linked to the SolidWorks API to ensure a seamless transition from SolidWorks to MATLAB environment. Therefore, a one button click on the SolidWorks (with the assembly file open) will generate and visualize product workspace information for every part (see Figure 0.15). It will also write the generated information to the online database.

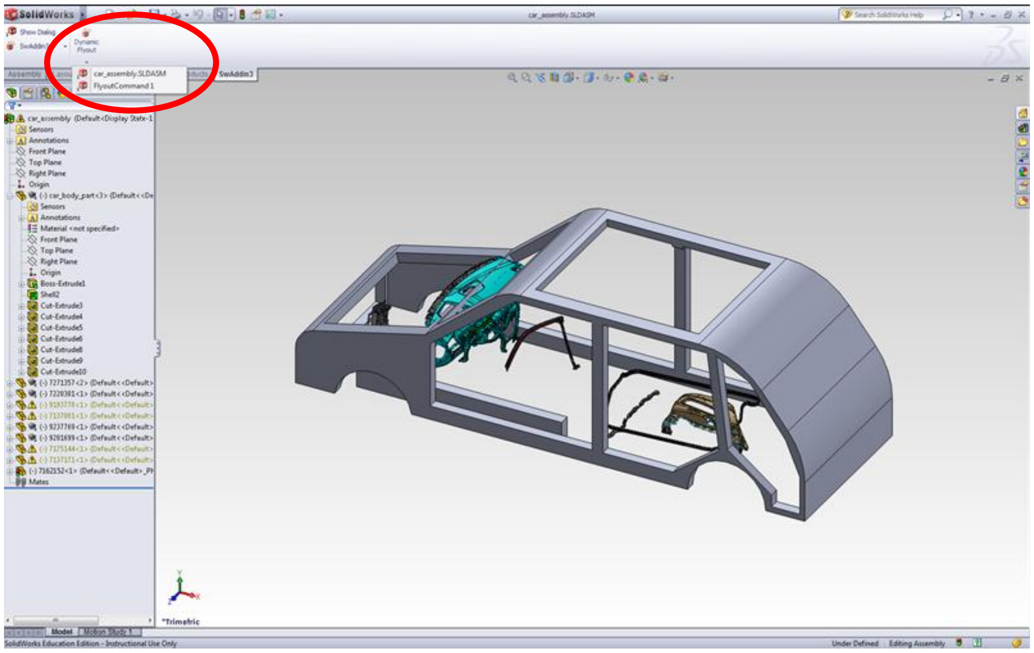


Figure 0.15: Snippet of PWIT button in SolidWorks

Since the bounding boxes provided information in all three dimensions, the Product Workspace Identification tool's pseudo code was extended to capture product workspaces in the side view of the product as well (see Figure 0.16). This information is currently not being used however its potential uses have been discussed in 0.

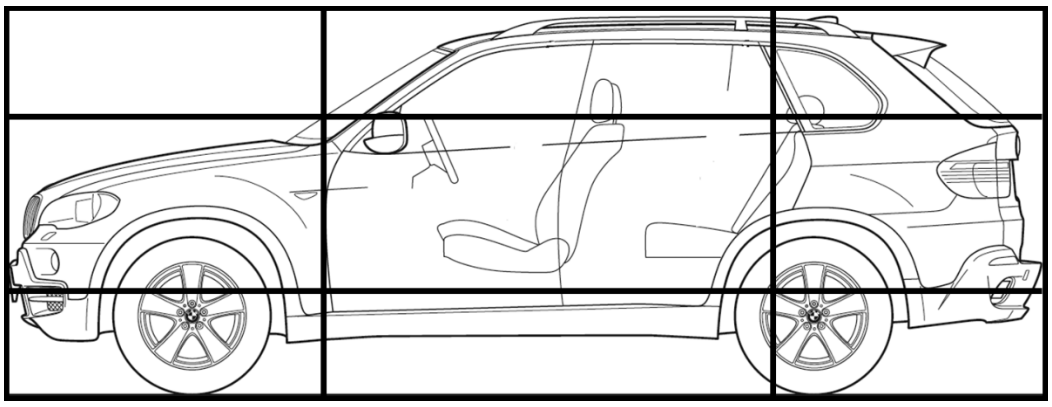


Figure 0.16: Side view of Product Workspaces

1.10 Automated MTM estimate generator

In order to maintain ensure consistency assembly of time estimations, predetermined time standards are used [4,8,17,18]. This research focuses on automating the use of Methods Time Measurement (MTM) tables. Figure 0.17 is adapted from [2] and shows the source of time studies at different stages in the production cycle of an automotive.

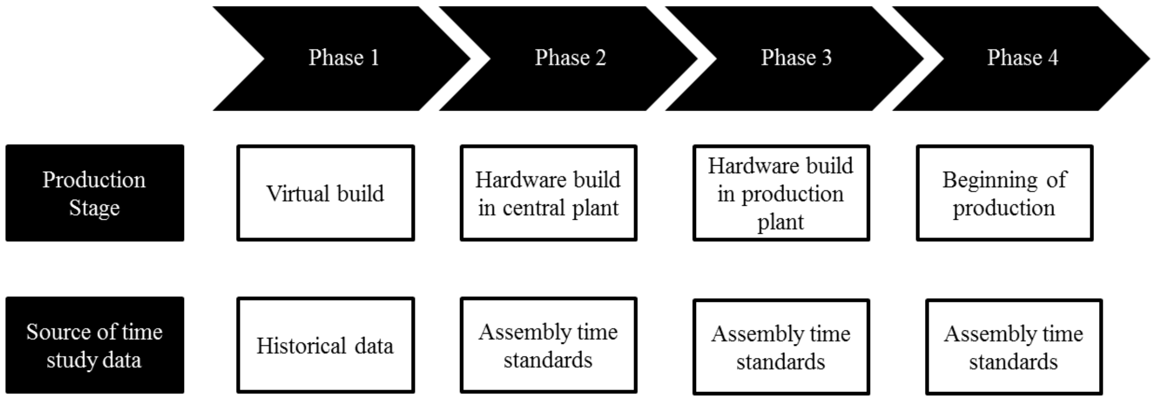


Figure 0.17: Source of time study data at different stages of production of an automobile

For applications where assembly process sheets use MTM estimates each estimate is associated to work instructions contained within the process sheet. Each assembly process sheet can contain several work instructions and for a product (such as a car) there can be several hundred process sheets [2]. Assigning an assembly time estimate to work instructions within each process sheet manually is a demanding task. An automotive OEM uses an adapted version of MTM which has twenty-two tables. This research is based on this adapted version of MTM tables. However, it must be noted that the decision support tool developed in this research can be easily modified to work on any version of the MTM [4]. Ambiguity can arise while selecting an MTM table. Assuming that the correct MTM table is chosen, filtering down to one MTM estimate can give rise to errors as well. To perform this filtering requires multiple sources of information. Automation of the generation of MTM estimates based on the work instructions within a process sheet will reduce the planner's effort required and also reduce the chances of error occurrence.

As mentioned previously, a large range of information sources are required for filtering down onto one assembly time estimate. The objective of this portion of the research is to reduce the ambiguity in selection of an MTM table based on the elements of the work instruction text and provide decision support during the filtering that is performed thereafter. This will lay the ground work to build a completely automated time estimation system.

1.10.1 Requirements

The following requirements were recognized as essential for the effective performance and use of the MTM estimate generator.

Table 0.5: Requirements for the MTM estimate generator

1.	Must guide the user to the most appropriate MTM table based on work instruction text.
2.	Must be based on the work instruction structure and vocabulary proposed by Peterson.
3.	Must not affect the MTM tables generated bu the MTM estimate generator.
4.	Must allow MTM codes to be assigned to process sheets.

5.	Must allow MTM codes to be decoupled from the process sheets that they have been previously assigned to.

The standard sentence structure ensures that every work instruction has only one action verb. The standard vocabulary provides a finite number of verbs to the process sheet author. Each MTM table is related to an action. This predefined action can relate to one or more of the action verbs from the standard work instruction authorship vocabulary. This presents an opportunity to create a link between standardized work instructions and MTM tables. It is observed that the filtering parameters of the MTM tables require part information amongst other pieces of information. PWIT enhances the part-process coupling provided by Peterson [2]. This enhanced part-process coupling can be utilized to drive the filtering within an MTM table.

1.10.2 Information Flow Diagram

The MTM estimates are required to be generated based off information provided by the work instructions from process sheets. With this constraint and the vision of a production management system that supports collaborative process planning in mind, it is decided that the MTM estimate generator will be implemented as a web-based application. The information flow for this system is shown below in Figure 0.18. Table 0.6 explains the elements of the information flow diagram for the MTM estimate generator.

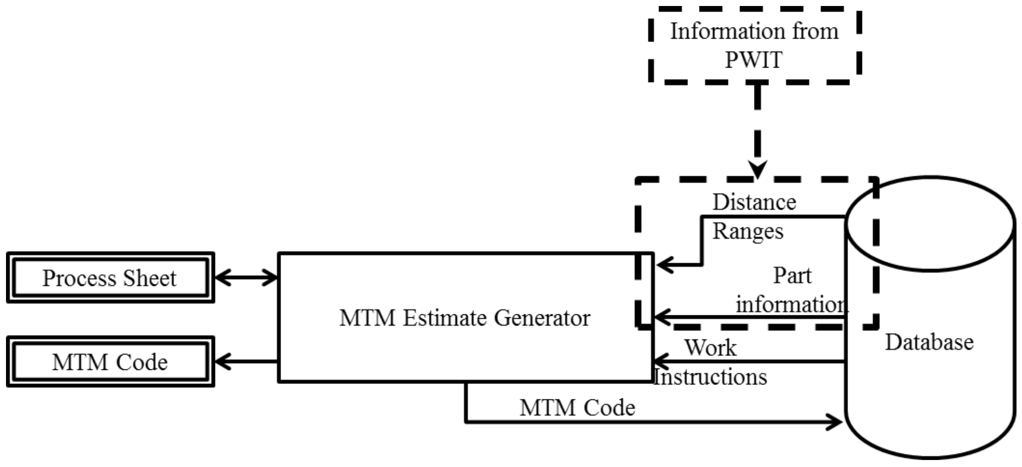


Figure 0.18: Information Flow Diagram for MTM time estimate generator

Table 0.6: Explanation of MTM estimate generator's information flow diagram elements

Information Flow Element	Explanation
Process Sheet	Process sheet for which MTM estimates are to be generated.
MTM Code	Ideally, the system will parse through the work instructions, perform MTM table selection and all the filtering thereafter and present the user with the MTM estimates for the process sheet and also record these time estimates on the database.

Information Flow Element	**Explanation**
MTM Estimate Generator	This performs intelligent parsing (based on predefined rules) of work instructions from the user defined process sheet to generate most applicable MTM estimates.
Distance Ranges	The distance between the part's product workspace and its bin location.
Work instructions	The work instructions contained within the user-defined process sheet.
Part Information	Part numbers.

The MTM estimate generator takes as an input from the user, the process sheet title. This information is used to retrieve all the work instructions contained within that process sheet. The work instructions provide information regarding the part that the process sheet corresponds to. This allows for relevant part information to be extracted. Along with part information, distance of the part's product workspace to the bin containing the part can be retrieved. These two information pieces (part information and distance ranges) are extracted/calculated by PWIT.

PWIT creates virtual bin locations around the product workspace diagram (see Figure 0.19). The straight line distance between the center point of these bins and the center point of the product workspace is calculated by PWIT and output as "Distance Range". Each part has a corresponding distance range after it is run through PWIT. Two access product workspaces are defined - RM, LM. An access product workspace is used to account for cases where parts have their product workspaces in "M" (i.e. MV, MM and/or MH). In order for an assembly worker to reach these product workspaces he/she must move through the access product workspace. Therefore, the distance range for a part that has product workspace in "M" is calculated by computing the distance from the part's bin to the nearest access product workspace and adding it to the distance between the access product workspace and the part's product workspace.

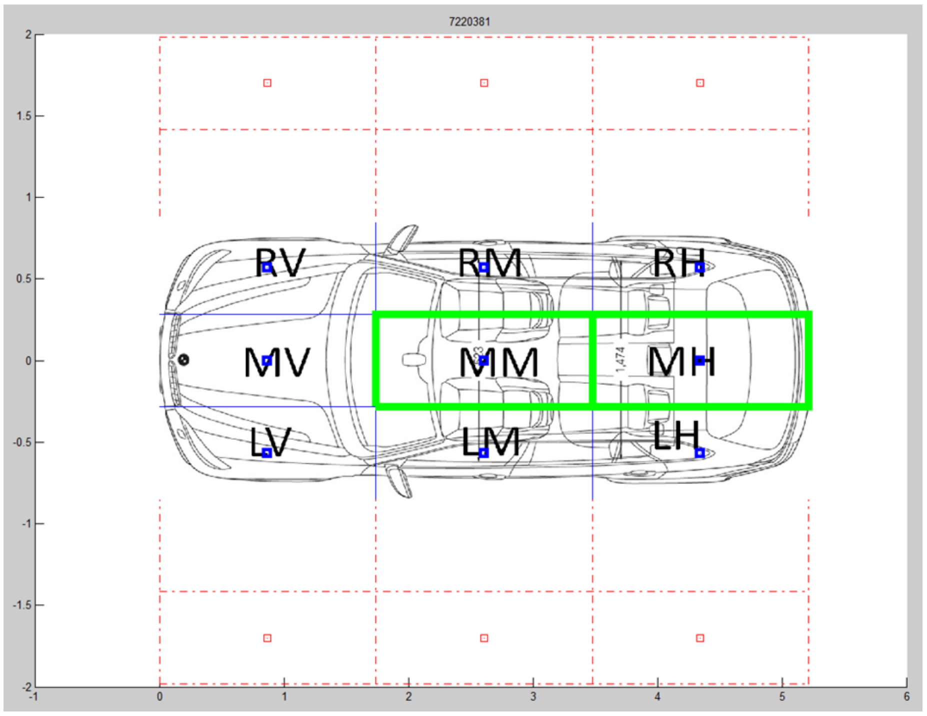


Figure 0.19: Example of the bin locations created around the product workspace diagram

The information flow diagram presented in this section is used to develop the ER diagram.

1.10.3 ER Diagram

The ER diagram shown in Figure 0.20 does not show any of the attributes of the entities. These entity attributes are listed in Table 0.7. It was observed that some MTM codes are not exclusive to one MTM table. The work instructions in a process sheet can warrant the need to have MTM codes from multiple tables. Standardized work instructions have the potential to provide a mapping between themselves and MTM tables. Part information from the part-process coupling provided by work instructions can be used to filter down to one MTM code after a MTM table has been selected.

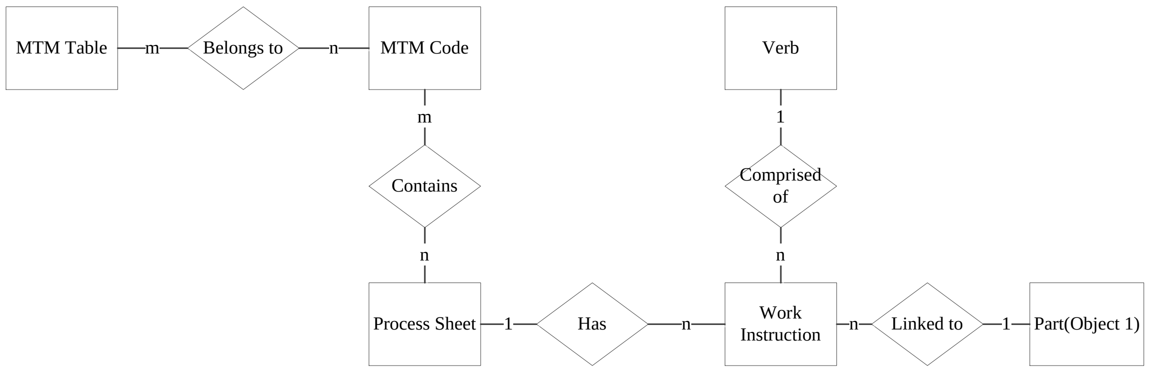


Figure 0.21: ER diagram for MTM estimate generator

Table 0.7: Details of components of ER diagram for MTM estimate generator

Entity	**Description**	**Attributes**
MTM Table	Methods Time Measurement tables that are used to obtain standard time estimates for assembly processes	ID, Table_name
MTM Code	These codes are used to represent the standard times (TMU) associated to assembly processes. Every table has several codes that can be obtained by performing suitable filtering.	ID, Code, Table_ID, TMU
Process Sheet	It is a document that contains work instructions, time study analyses, product workspace information and other meta information.	ID, Number, Title

Entity	Description	Attributes
Work Instruction	A set of work instructions dictate the assembly operations that the process sheet is associated to.	ID, Process_Sheet_ID, Step_Number, Verb_ID, Object1_ID, Prep_ID, Object2_ID
Part	The object whose state is directly affected by the work instruction. Every work instruction operates on one primary object.	ID, Part_Number, Description
Verb	Every work instruction describes an assembly action. This action is directly related to the verb that the work instruction uses.	ID, Verb

This ER diagram is an extension of the ER diagram of the PWIT. This is expected because information from PWIT is used to drive automation in the MTM estimate generator. It is important to note that “TMU” – Time Measurement Unit is the unit of time that the MTM table uses and one TMU is 0.036 seconds [4].

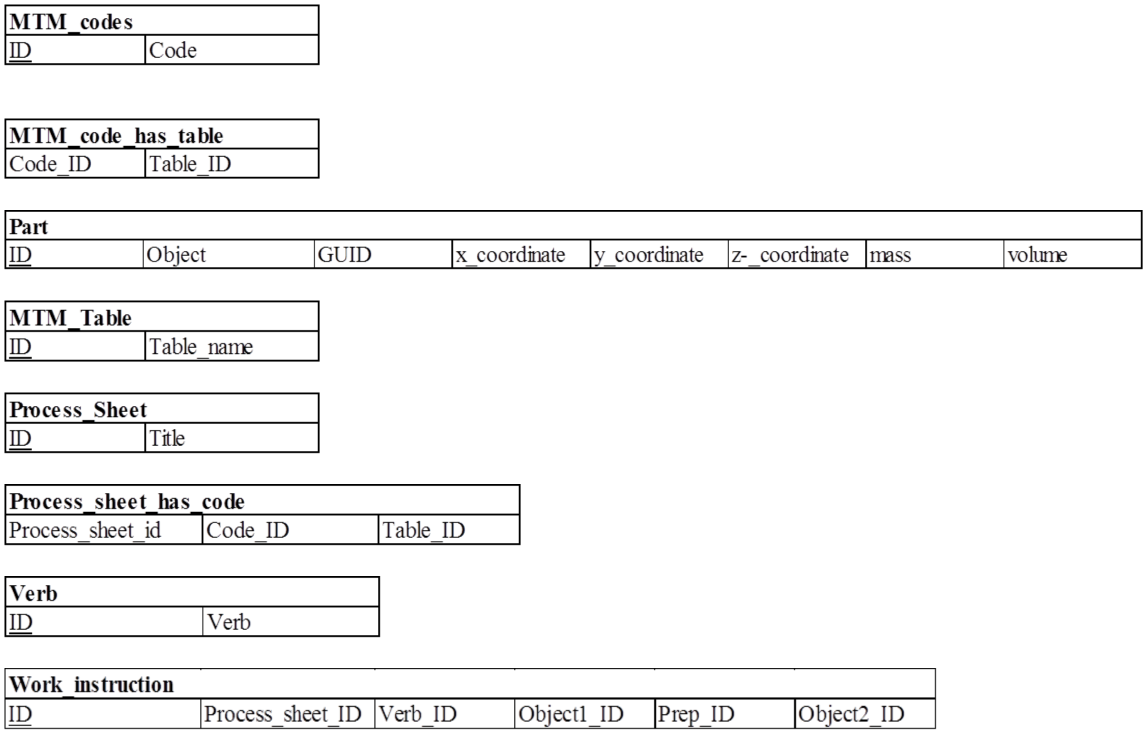
Table 0.8: Explanation of relationships in ER diagram for MTM estimate generator

Relationship	Description	Cardinality
MTM_Code_belongs_to_MTM_Table	Every MTM code can be found in one or more MTM tables	m:n
Process_Sheet_contains_MTM_Code	Every process sheet has MTM time estimate associated with it. This estimate is described by the MTM code	m:n
Process_Sheet_has_Work_Instruction	One purpose of process sheets is to convey assembly operations to line side associates. This is done by means of work instructions contained within the process sheet.	1:n
Work_Instruction_linked_to_Part	Every work instruction on a process sheet is linked to a part.	n:1
Work_Instruction_comprises_of_Verb	The action which the work	n:1

Relationship	Description	Cardinality
	instruction is describing is defined by the verb it uses	

1.10.4 Relational Schema

The ER diagram is converted to tables that are used within the online database to store information. The tables that are created for the MTM estimate generator have overlap with the tables that were created for PWIT. The SQL code for the creation of these tables can be found in Appendix A (A.2). Figure 0.23 shows the tables that have been created to store information that will be used and generated by the MTM estimate generator.



MTM_codes	
ID	Code

MTM_code_has_table	
Code_ID	Table_ID

Part							
ID	Object	GUID	x_coordinate	y_coordinate	z-_coordinate	mass	volume

MTM_Table	
ID	Table_name

Process_Sheet	
ID	Title

Process_sheet_has_code		
Process_sheet_id	Code_ID	Table_ID

Verb	
ID	Verb

Work_instruction					
ID	Process_sheet_ID	Verb_ID	Object1_ID	Prep_ID	Object2_ID

Figure 0.23: Tables created for MTM estimate generator

1.10.5 Non-functional Prototype

The next step in the design process calls for non-functional prototyping. This will allow user interfaces to be conceptualized. In Figure 0.24, it conceptualized that the user will select the process sheet that he/she wants to compute the MTM estimates for. The webpage will then run queries to retrieve and display the work instructions for the selected process sheet.

The webpage will run inference engines in the background to map the work instruction text to MTM tables. The webpage will also extract relevant part information from the database and apply them to the parameters of the table in order to perform filtering and eventual selection of a single MTM estimate. The automatically generated MTM estimates can then either be modified or removed (decoupled) from the process sheet.

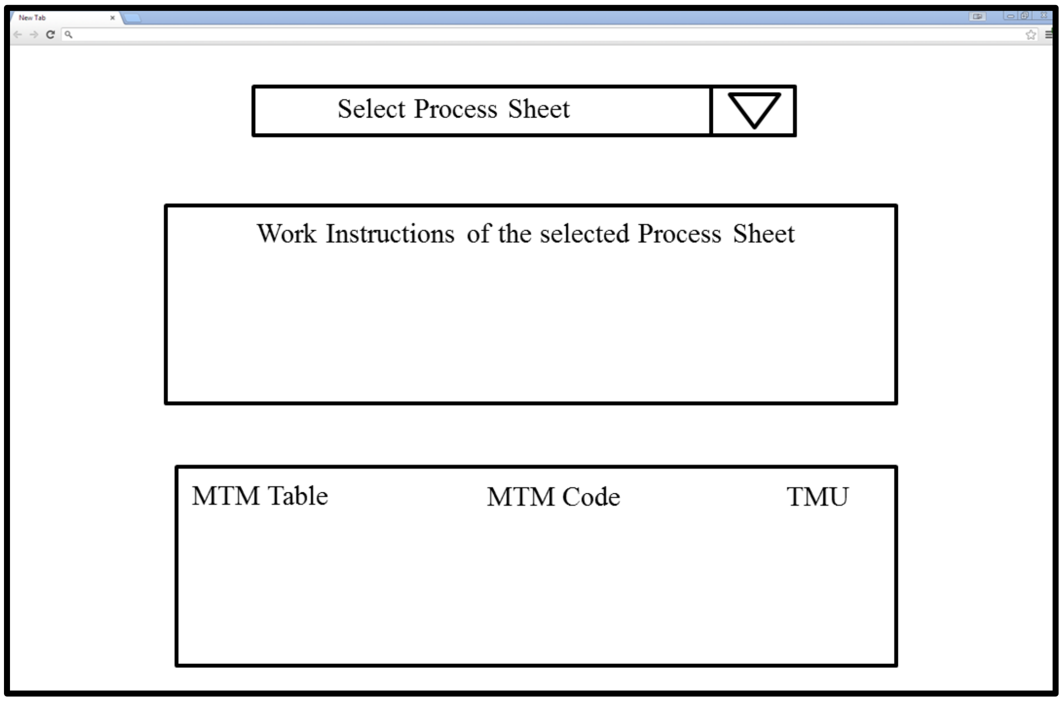
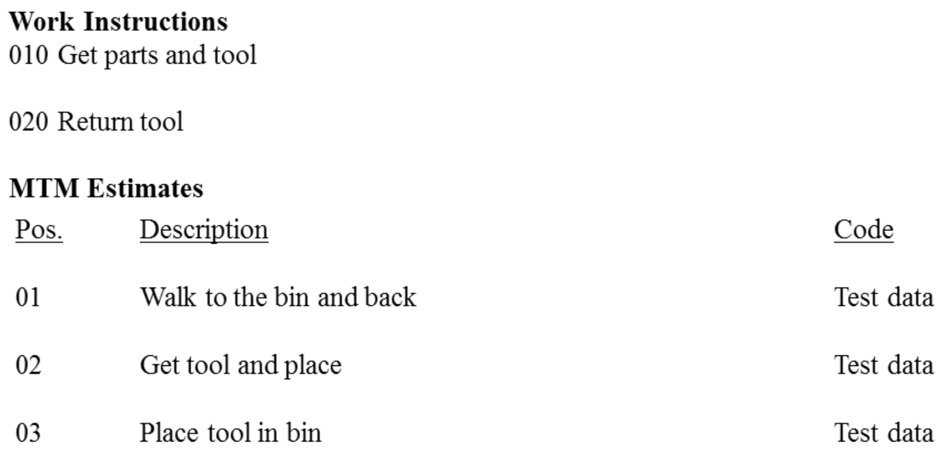


Figure 0.24: Non-functional prototype of the MTM estimate generator

1.10.6 Pseudo Code – Guide for implementing the MTM estimate generator

Mapping rules have to be created in order to relate work instructions to MTM tables based on historical data of a large scale automotive OEM. The challenge here lies in the fact that not every work instruction shares a one-to-one mapping with an MTM table. For example, the MTM table "Get and Place" clearly refers to two assembly operations; “get” a part and “place” it. Approximately 250 process sheets (that are used by a large scale OEM) were analyzed to recognize and formalize mapping rules.

The process sheets that were available for analysis did not use the standard sentence structure and vocabulary from [2]. The work instructions on these process sheets were analyzed by an MTM analyst who then created a set of work instructions of his/her own (see Figure 0.25). These MTM descriptions were analyzed by recasting them into the standard sentence structure and vocabulary.



Work Instructions

010 Get parts and tool

020 Return tool

MTM Estimates

Pos.	Description	Code
01	Walk to the bin and back	Test data
02	Get tool and place	Test data
03	Place tool in bin	Test data

Figure 0.25: Example of process sheet with work instructions and MTM estimate descriptions

The mappings that were recognized fell under two broad classifications: mappings that had process sheet level scope, and mappings that had work instruction level scope. The process sheet level mapping is for instances that related multiple work instructions to one MTM table (see Figure 0.28). The work instruction level mapping is for instances where there exists a one-to-one mapping between work instruction verbs and MTM tables (see Figure 0.26). It is also found to be relevant for instances where there exists a one-to-one mapping between work instruction verb + object-type and MTM table (see Figure 0.27).

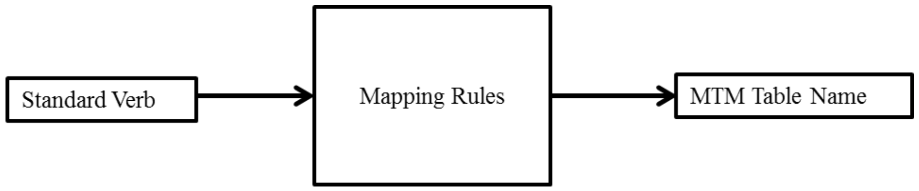


Figure 0.26: MTM mapping rules – Type 1

In order to analyze the work instructions with respect to the hypothesized mapping rules, there is a need to categorize parts into object types. If a single work instruction maps to one MTM table irrespective of the object type, it can be inferred that the verb used in the work instruction maps to the MTM table following Rule 1 (Figure 0.26). If the work instruction verb maps to multiple MTM tables depending on the object type that it operates on, it can be inferred that the verb and object type, together, map to an MTM table following Rule 2(Figure 0.27). With the help of members from Clemson Engineering Design Applications and Research (CEDAR), a list of object types was created (Table 0.9).

Table 0.9: Object types and their descriptions

Object Type	Description
Part	Any object that has a part number and appears on the bill of materials
Fixture	A device used to hold parts in a desired orientation
Plant Item	Any object that exists in the plant and is not any of the other object types, primarily part bins
Consumable	Object that is used during the assembly of the final product but does not figure on the bill of materials (protective tape, cleaning wipes, and so on)
Tool	Object that aids in installation assembly operations

Table 0.10: Results from MTM mapping analysis – Rule Type 1

Action Verb	MTM Table
Apply	Application of a medium
Connect	Laying Cables
Engage	Operate
Exchange	Handling containers
Get	Get and Place
Handstart	Working with screws/bolts
Lay	Laying Cables
Open(Preparatory)	Preparatory Activities
Place	Place
Read	Read
Remove	Get and Place
Restock	Parts supply
Restrict	Laying Cables
Scan	Marking and Documenting
Screw in	Handling auxiliary materials/ tools
Unscrew	Motion Cycles

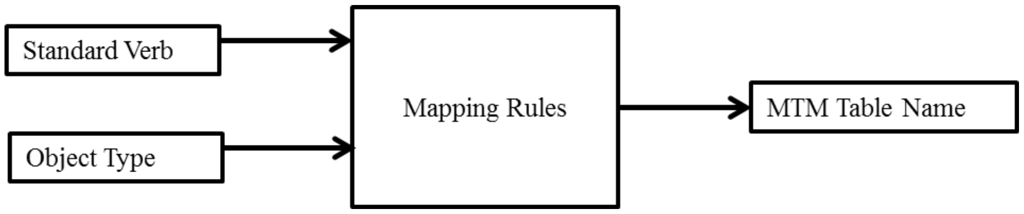


Figure 0.27: MTM mapping rules – Type 2

Rule 2	
Action Verb	
& Object Type	MTM Table Name
Align	
& Fixture	Place
& Plant Item	Place
& Tool	Motion Cycles
Attach	
& Consumable	Work with adhesives
& Fixture	Get and Place
& Plant Item	Get and Place
& Tool	Get and Place
Clean	
& Consumable	Cleaning

Connect	
& Part	Laying Cables
Disengage	
& Plant Item	Operate
& Tool	Motion Cycles
Engage	
& Fixture	Operate
& Part	Operate
Exchange	
& Plant Item	Handling Containers
Get	
& Fixture	Get and Place
Handstart	
& Part	Working with screws/ bolts
Inspect	
& Tool	Visual control

Lay	
& Part	Laying Cables
Move	
& Plant Item	Body Motions
& Tool	Body Motions
Open (Preparatory)	
& Consumable	Preparatory Activities
& Plant Item	Preparatory Activities
Operate	
& Part	Handle Tool
& Plant Item	Operate
Place	
& Consumable	Working with Adhesives
Press	
& Fixture	Operate
& Plant Item	Operate
Push	

& Fixture	Motion Cycles
& Plant Item	Get and Place
Read	
& Plant Item	Read
Remove	
& Fixture	Get and Place
& Tool	Get and Place
Remove (Preparatory)	
& Part	Get and Place
Restock	
& Part	Parts Supply
Restrict	
& Part	Laying Cables
Scan	
& Part	Marking and Documenting
Screw In	

& Part	Handling Auxiliary Material/ Tools
Secure	
& Fixture	Operate
Tighten	
& Tool	Handling Auxiliary Materials/Tools
Unscrew	
& Part	Motion Cycle
Walk	
& Fixture	Body Motions
& Part	Body Motions
& Tool	Body Motions

The third mapping rule deals with multiple work instructions. This rule has process sheet level scope and the logic of this rule can be seen in Figure 0.28. Verbs across multiple work instructions are extracted along with the object type of the primary objects of those work instructions. These were mapped to a single MTM table. This represents a one-to-one mapping between work instruction text and MTM tables.

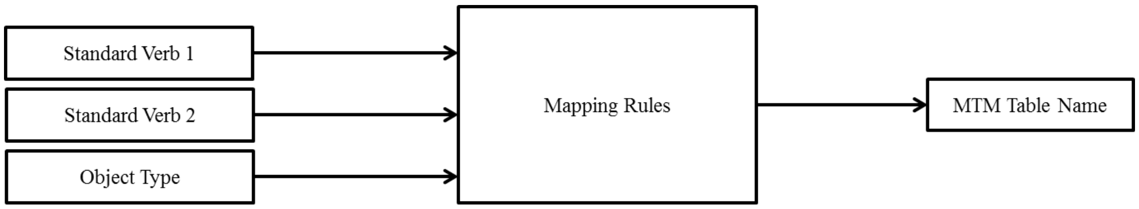


Figure 0.28: MTM mapping rules – Type 3

Table 0.11: Results from MTM mapping analysis – Rule Type 3

Verb 1, Verb 2, Object Type	MTM Table Name
Align, Place, Part	Place
Connect, Inspect, Part	Laying Cables
Get, Align, Part	Get and Place
Get, Attach, Part	Working with Clips
Get, Attach, Tool	Get and Place
Get, Connect, Part	Laying Cables
Get, Handstart, Part	Working with Screws/Bolts
Get, Insert, Part	Working with Screws/Bolts
Get, Insert, Part	Working with Clips
Get, Operate, Tool	Get and Place
Get, Operate, Tool	Handle Tool
Get, Place, Part	Working with Screws/Bolts
Get, Place, Part	Laying Cables

Get, Place, Part	Place
Get, Place, Part	Place
Get, Place, Tool	Get and Place
Get, Place, Tool	Operate
Get, Place, Tool	Handling Auxiliary Materials/Tools
Get, Place, Tool	Handle Tool
Get, Push, Part	Get and Place
Get, Remove, Fixture	Get and Place
Get, Scan, Part	Marking, Documenting
Get, Snap, Part	Get and Place
Get, Tighten, Part	Handling Auxiliary Materials/Tools
Move, Walk, Plant Item	Body Motions
Remove(Preparatory), Place, Part	Get and Place
Restock, Remove, Plant Item	Handling Containers

The use of object types warranted a change in the information flow and ER diagram. The object type information of the primary object in the work instruction needs to be extracted from the database. Object type was modeled to be an entity in the ER diagram with its attributes being ‘ID’ and ‘Description’. Figure 0.29 and Figure 0.30

show the revised information flow and ER diagram for the MTM estimate generator. This change also resulted in an update to the relational schema. The SQL code for the creation of these tables can be found in Appendix A (A.2).

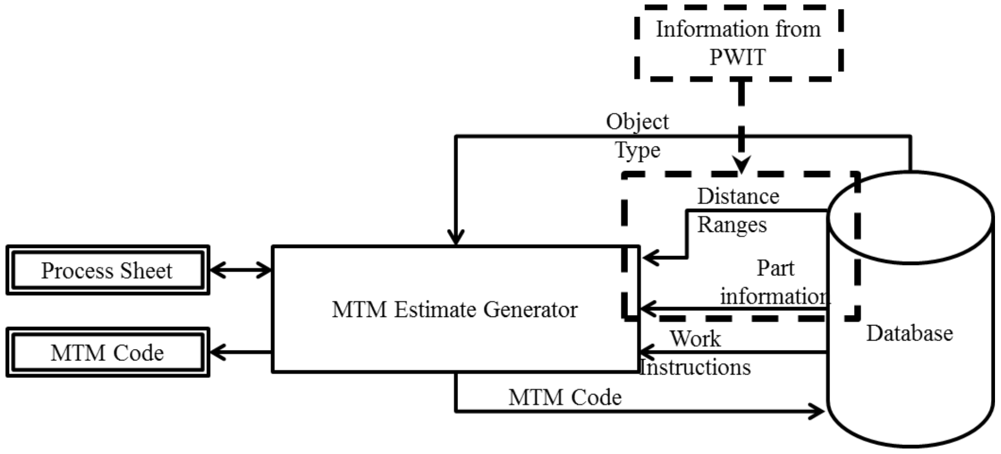


Figure 0.29: Revised Information Flow Diagram for MTM estimate generator

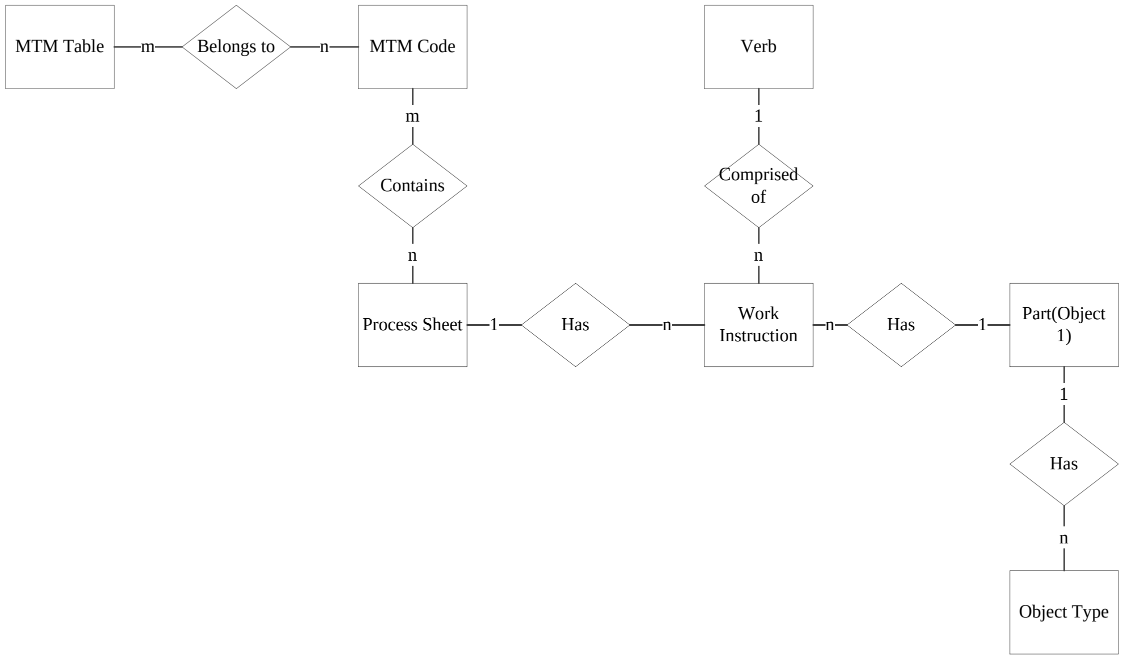


Figure 0.30: Revised ER Diagram for MTM estimate generator

With these mapping rules and revised relational schema, a pseudo code was developed to implement the non-functional prototype and to satisfy requirements. This pseudo code is shown below and discussed in the remainder of the subsection. The pseudo code operates such that the order of the work instructions does not affect the tables that are output.

```
1. Get work instructions for the selected process sheet
2. Get corresponding part information
3. Display work instructions and assigned MTM codes
4. Apply rule 3
```

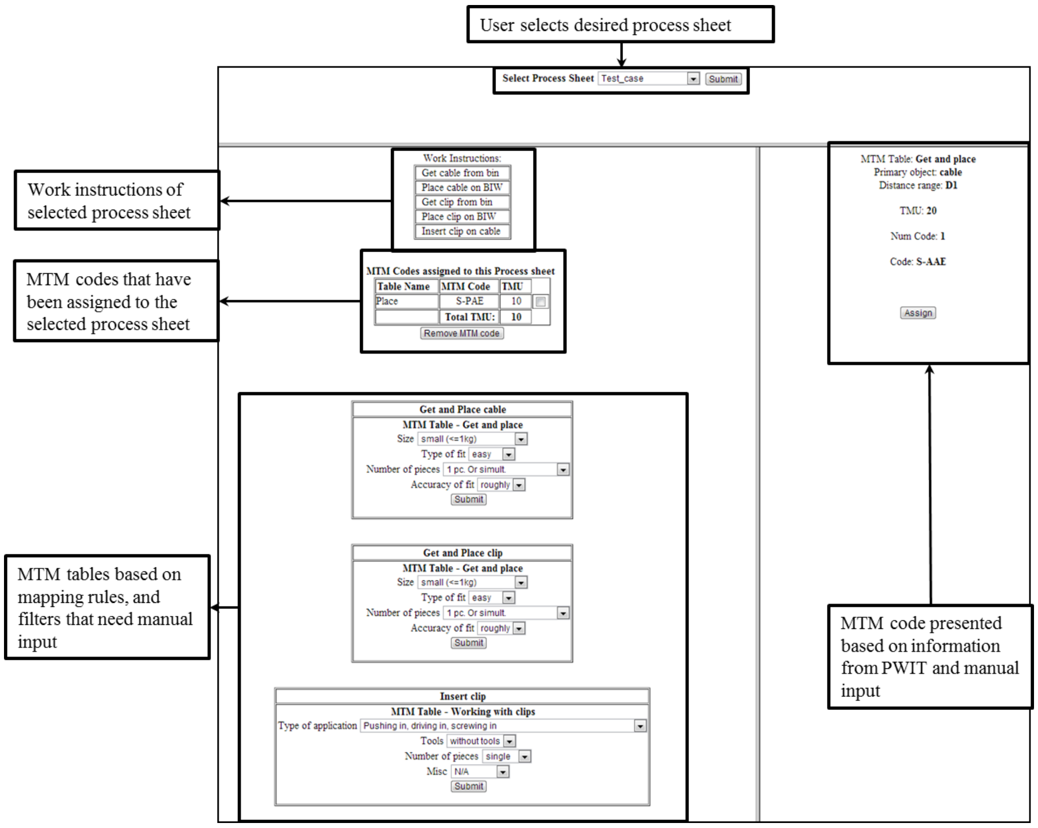
5. *If rule 3 is not applicable, apply rule 2*
6. *If rule 2 is not applicable, apply rule 1*
7. *Filter based on information from PWIT*
8. *Display resulting tables and remaining filters for manual down-selection*
9. *Assign MTM code to process sheet*

The first step is to obtain all the work instructions for the selected process sheet. These work instructions will also provide us with the part number of the primary object. With this information the part's distance ranges (from PWIT) and object type must be obtained. It is essential to inform the user of the work instructions contained within that process sheet and the MTM codes that have previously been assigned.

The first step of creating a link between the work instructions and the MTM tables is to apply Rule 3. This rule's scope spans across the entire process sheet and therefore it must be applied first. If, for a set of work instructions, Rule 3 does not apply at all then Rule 2 must be applied to every individual work instruction within that set. The last rule to be applied is Rule 3. Once an MTM table, or a small set of MTM tables, is narrowed down upon, filtering within this table can be performed based on the information obtained from PWIT. In most cases, the information from PWIT will not be sufficient to narrow down on one MTM code. Therefore, the user must be presented with the remaining filters to perform manual filtering and assign the MTM code to the process sheet.

1.10.7 Implementation

In order to implement the conceptualized system it is essential to have a virtual version of the MTM tables. In conjunction with other members of Clemson Engineering Design Applications and Research (CEDAR) at Clemson University, the MTM tables were entered into the online database. Each MTM table had a corresponding table on the database.


User selects desired process sheet
Work instructions of selected process sheet
MTM codes that have been assigned to the selected process sheet
MTM tables based on mapping rules, and filters that need manual input
MTM code presented based on information from PWIT and manual input
Submit
Remove MTM code
Assign
Submit
Submit
Submit

Select Process Sheet: Test_case

Work Instructions:
- Get cable from bin
- Place cable on BIW
- Get clip from bin
- Place clip on BIW
- Insert clip on cable

MTM Codes assigned to this Process sheet

Table Name	MTM Code	TMU	
Place	S-PAE	10	☐
	Total TMU:	**10**	

MTM Table: **Get and place**
Primary object: **cable**
Distance range: **D1**

TMU: **20**

Num Code: **1**

Code: **S-AAE**

Get and Place cable
MTM Table - Get and place
Size: small (<=1kg)
Type of fit: easy
Number of pieces: 1 pc. Or simult.
Accuracy of fit: roughly

Get and Place clip
MTM Table - Get and place
Size: small (<=1kg)
Type of fit: easy
Number of pieces: 1 pc. Or simult.
Accuracy of fit: roughly

Insert clip
MTM Table - Working with clips
Type of application: Pushing in, driving in, screwing in
Tools: without tools
Number of pieces: single
Misc: N/A

Figure 0.31: Screenshot of MTM estimate generator

The pseudo code presented in the previous section was implemented by using HTML and PHP script to create a webpage. A screenshot of this webpage is show in Figure 0.31. In this figure the top most frame lists out all the process sheets that are available. The user selects the desired process sheet. Submitting this choice triggers the retrieval of the process sheet's information from the database. The frame in the left displays the work instructions associated to that process sheet followed by the MTM codes that have already been assigned to that process sheet. The user is allowed to select and delete as many of the assigned MTM codes as desired. This deletion decouples the MTM code from the process sheet.

The program then applies the rules for mapping work instructions to MTM tables. A MTM view of the work instructions is displayed. The user is also presented the filters that have not been applied. After performing manual filtering here, the user submits the information and this triggers the right frame to display the corresponding MTM code and TMU. The user is allowed to "Assign" the code to the process sheet. The testing and validation of this tool is presented in the next chapter.

1.11 Chapter Conclusions

The similarity between the elements of the information flow of the two tools was intended. There is need to have a web-based production management system. These two tools contribute towards this end. The interconnections between the two tools are shown in a combined information flow and ER diagram in Figure 0.32 and Figure 0.33.

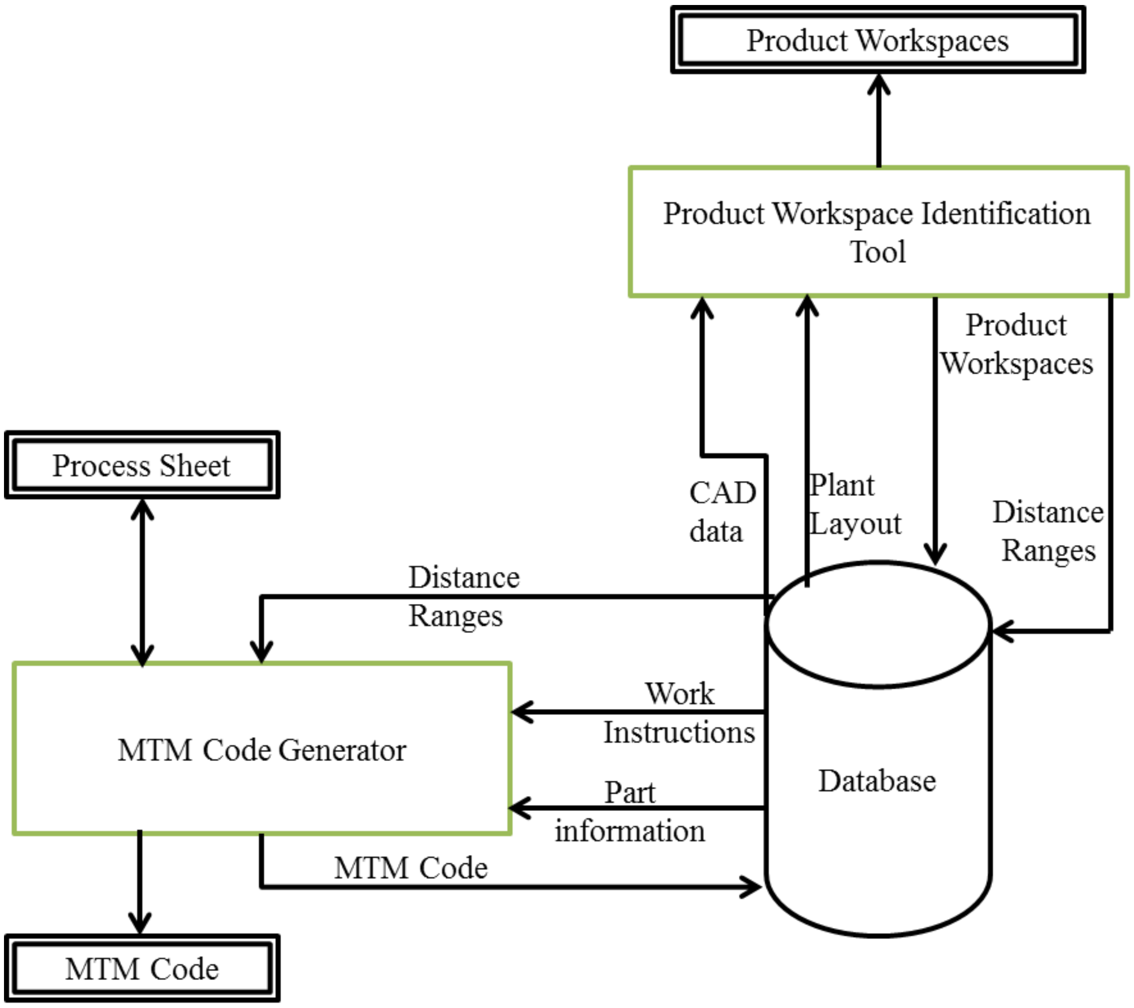


Figure 0.32: Aggregated information flow diagram – Production System model

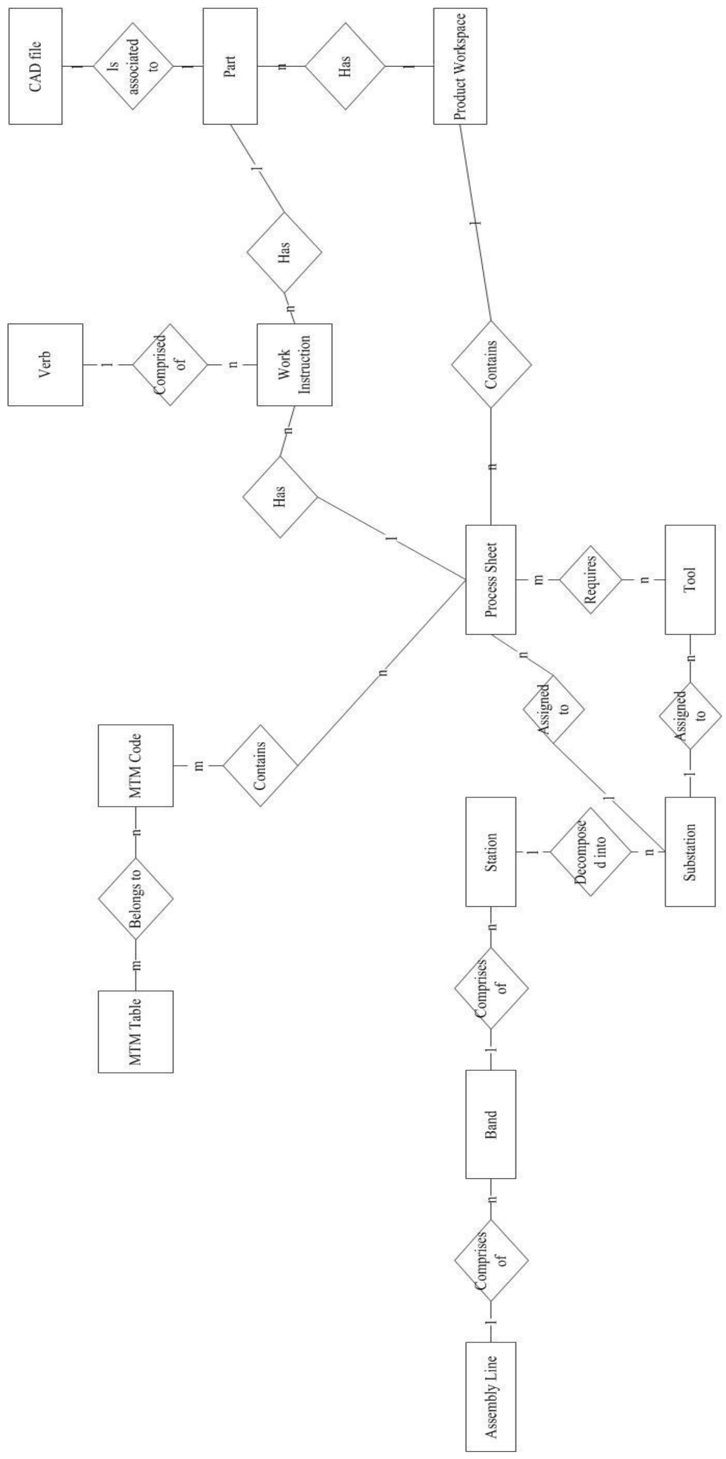

CAD file
Is associated to
Part
Has
Product Workspace
Has
Verb
Comprised of
Work Instruction
Contains
Has
Process Sheet
Requires
Tool
Contains
MTM Code
Assigned to
Assigned to
Belongs to
MTM Table
Station
Decomposed into
Substation
Comprises of
Band
Comprises of
Assembly Line

Figure 0.33: Aggregated ER diagram for the Production System

The information flow diagram and ER diagram shown above contribute towards one portion of a large production management system. The system shown must be expanded to include ergonomic evaluations of work instructions, precedence relations between tasks (process sheets and work instructions), layout of workstations and logistics. The inclusion of these elements will ensure that the production management system will be able to handle product information throughout the entire lifecycle of production, from production volume forecasting to complete assembly of the product and delivery.

CHAPTER FOUR
TESTING AND VALIDATION OF THE DECISION SUPPORT TOOLS

Chapter Objectives:

- Explain test cases for the tools that have been developed.
- Show results from the test cases.
- Identify potential areas for refinement.

Validation is performed in three stages (see Table 0.1). The information generated from the first stage (testing and validation of PWIT) is used during the second stage (testing and validation of MTM estimate generator). The test case for the PWIT is an assembly file of an automobile. This assembly file consisted of a body-in-white and nine parts. The parts were placed in the body-in-white relative to a fixed coordinate and no virtual mates were created. The Product Workspace Identification Tool was run on this assembly file. The extracted part information was stored on the database and product workspaces were displayed to the user.

Table 0.1: Stages for testing and validation

Stage 1	Testing and validation of PWIT.
Stage 2	Testing and validation of MTM estimate generator.

The MTM estimate generator is validated by re-authoring seventy work instructions from process sheets of a large automotive OEM. These work instructions were sent to the MTM estimate generator as input and a reduced list of MTM tables were presented to the user based on predefined mappings (see Section 1.10.6). The parameters within the user selected tables are filtered based on information from PWIT. The user is allowed to assign an MTM code to the corresponding process sheet.

1.12 Testing and validation of Product Workspace Identification Tool

The Product Workspace Identification Tool (PWIT) was tested with a body-in-white and nine constituent components of a car. Seven of these nine components have been obtained from a large scale automotive OEM. Figure 0.1 shows a screenshot of the isometric view of assembly model that was created in SolidWorks.

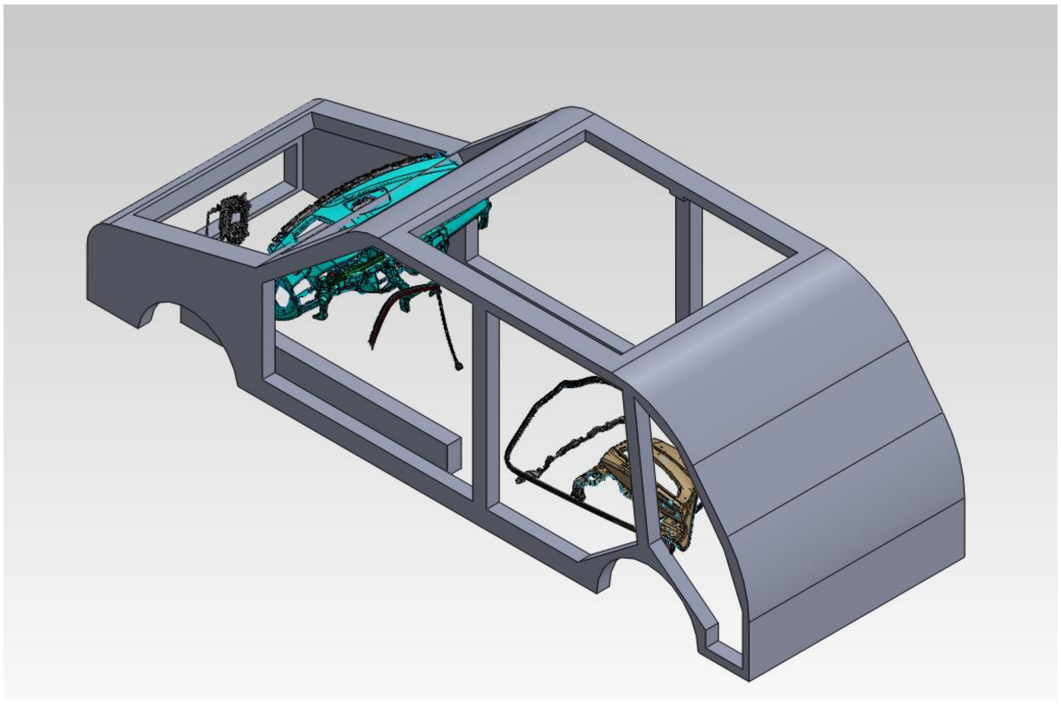
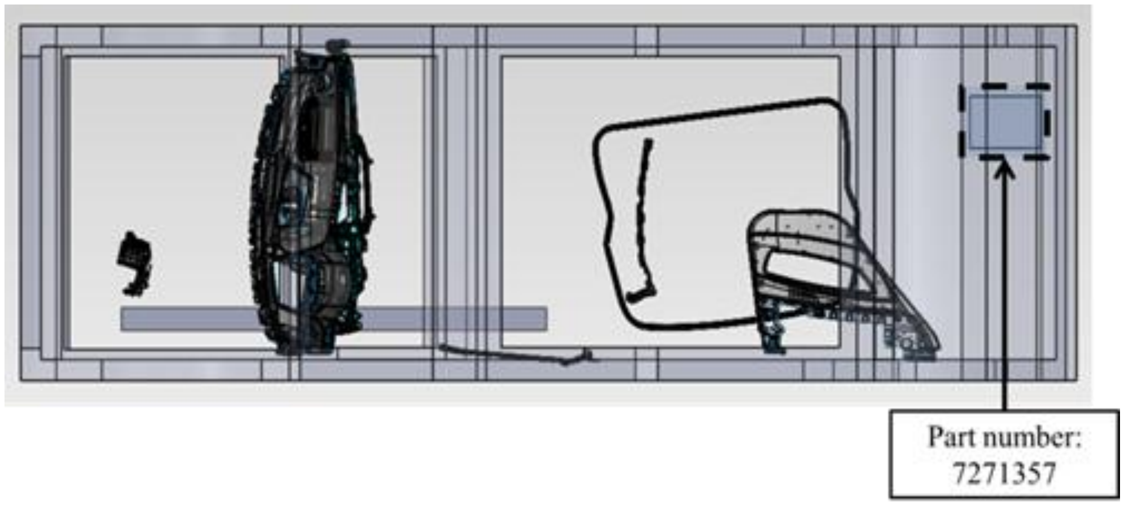


Figure 0.1: Snapshot of the isometric view assembly model that was created for testing PWIT

The following figures show the top view of the SolidWorks assembly model in Figure 0.1 with a different part highlighted each time. The parts that have been highlighted are annotated with their part numbers. Also shown are the product workspace representations generated by PWIT.



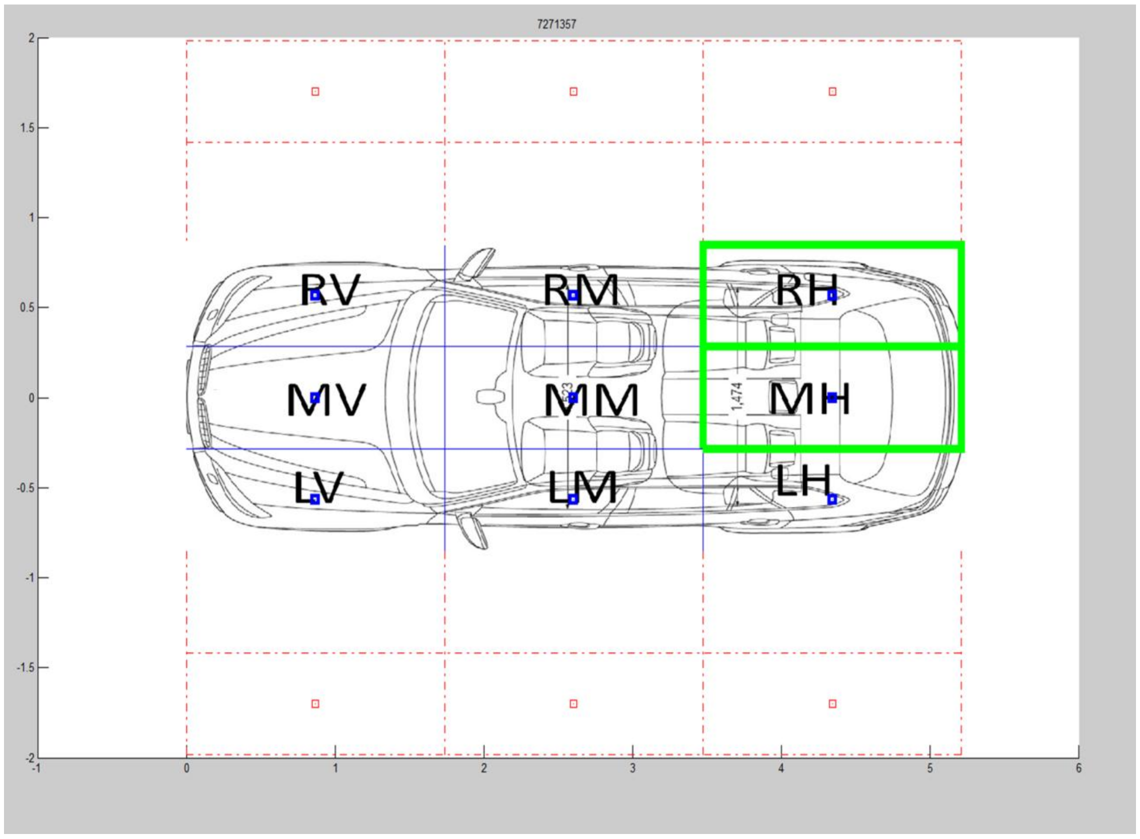
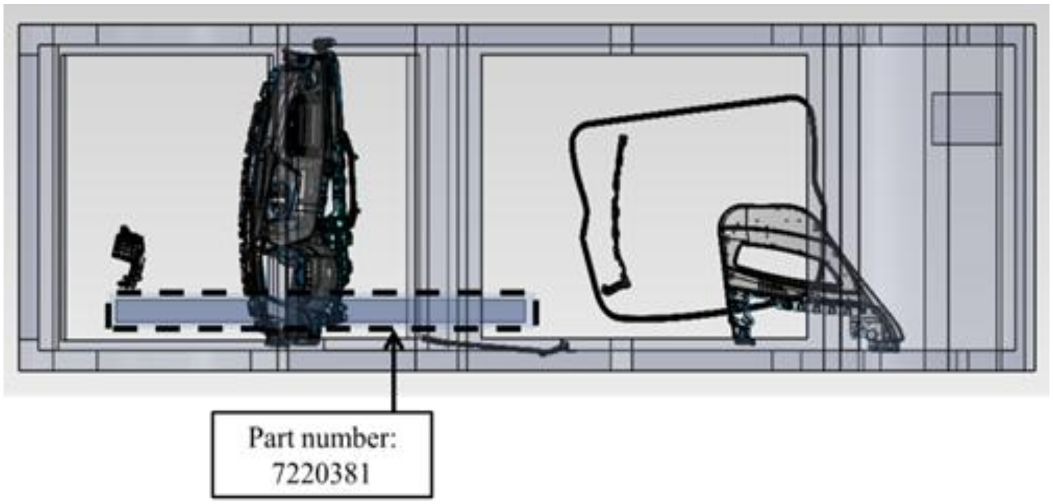


Figure 0.2: PWIT output for part number 7271357



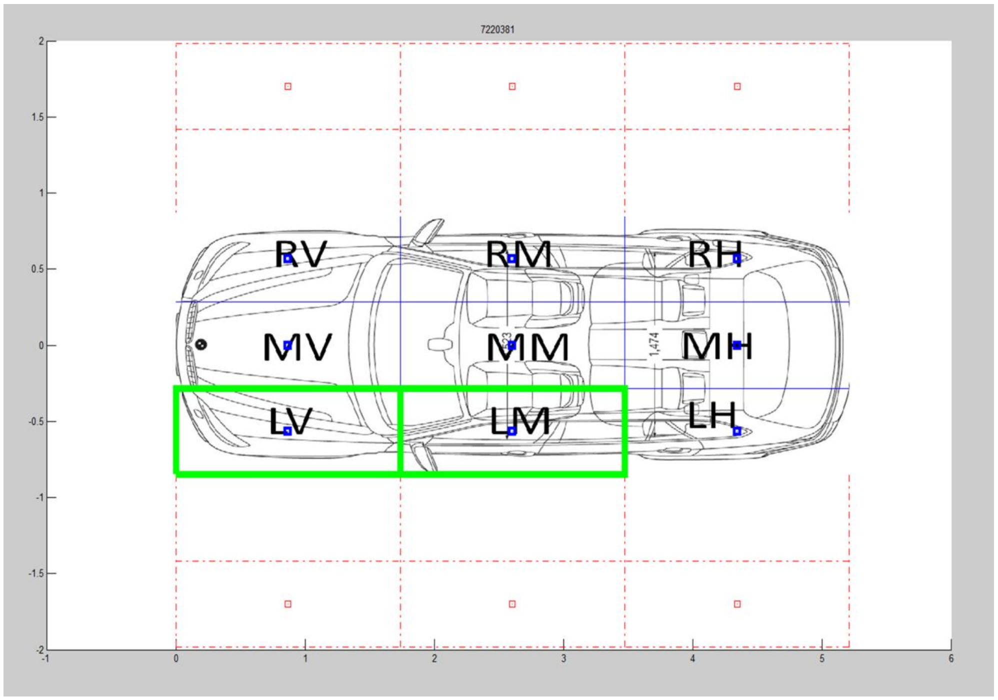
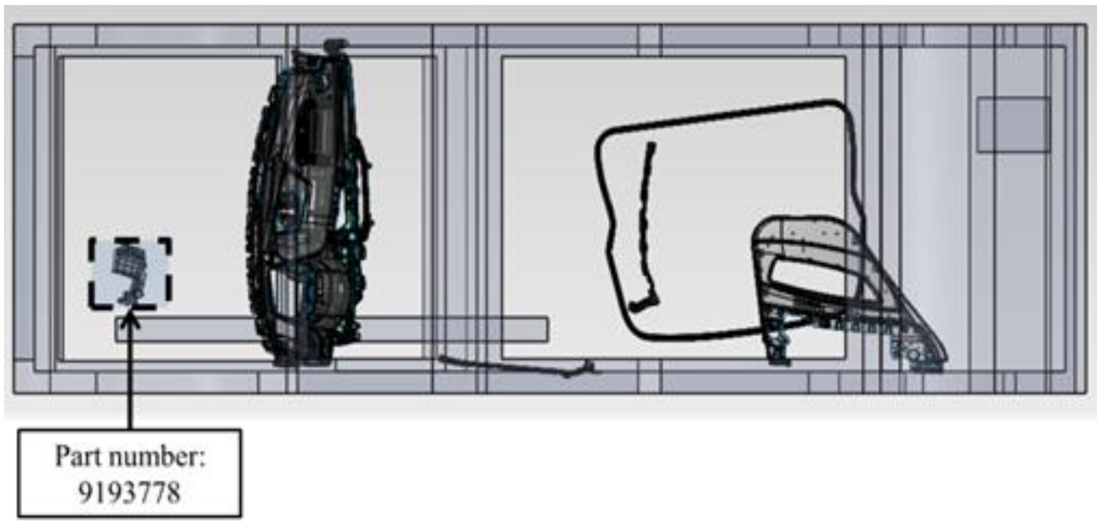


Figure 0.3: PWIT output for part number 7220381



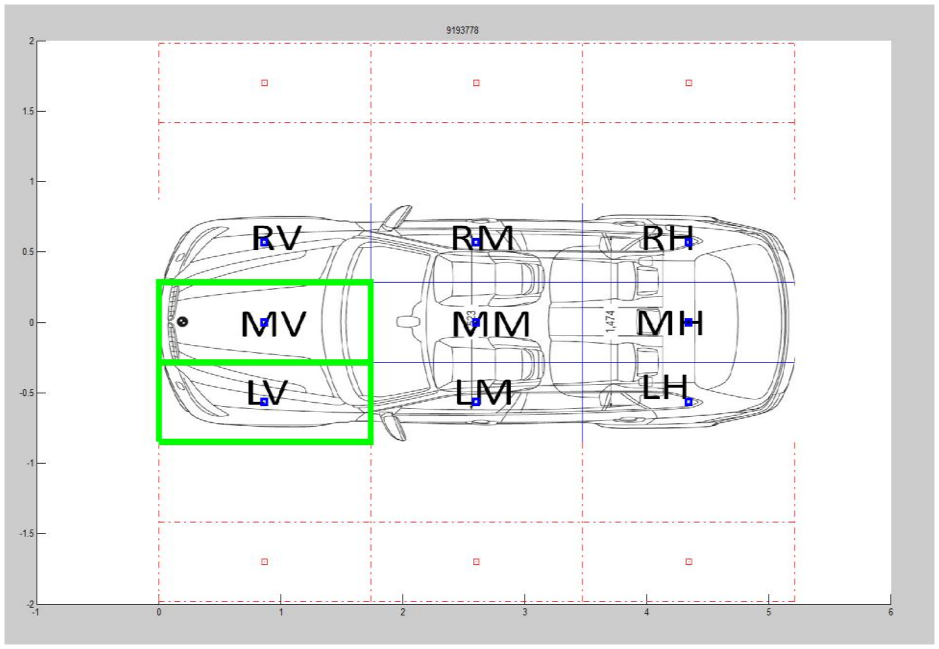
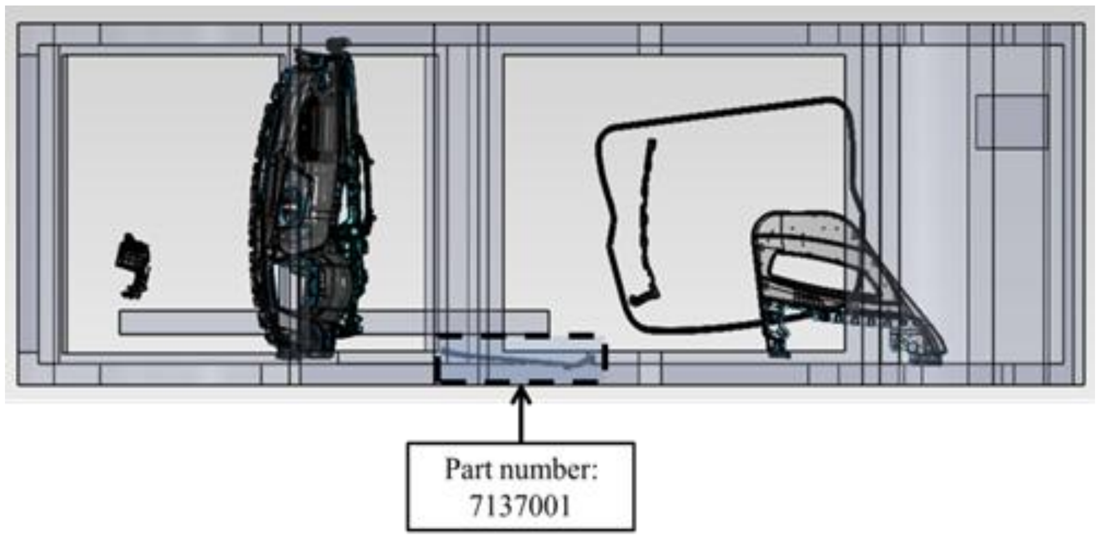


Figure 0.4: PWIT output for part number 9193778



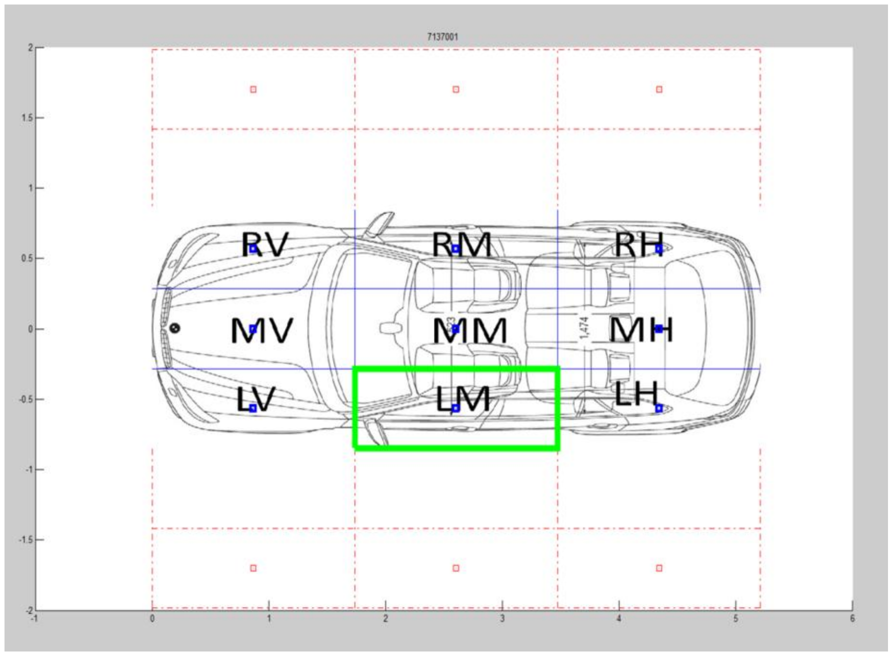
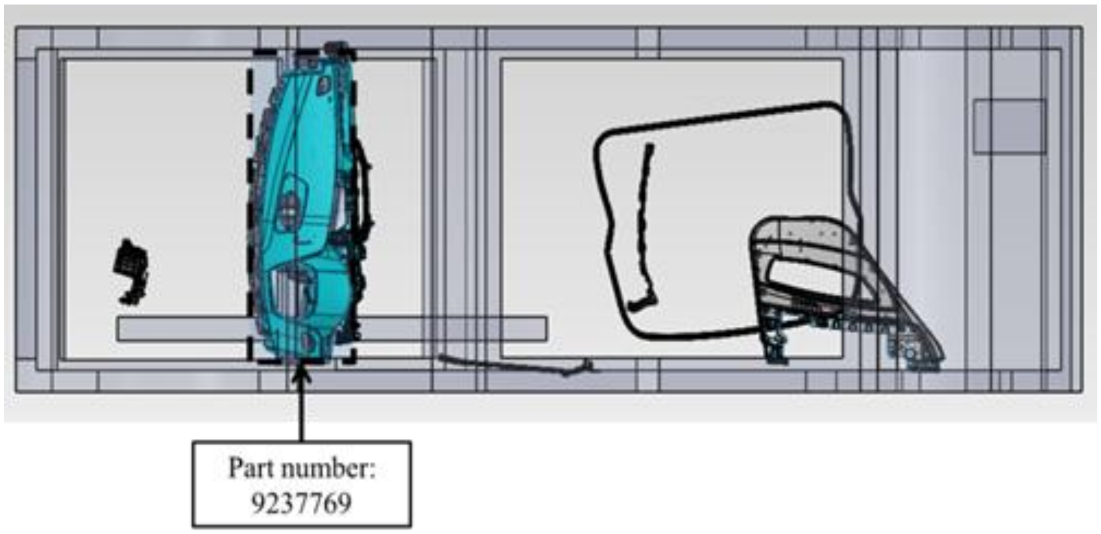


Figure 0.5: PWIT output for part number 7137001



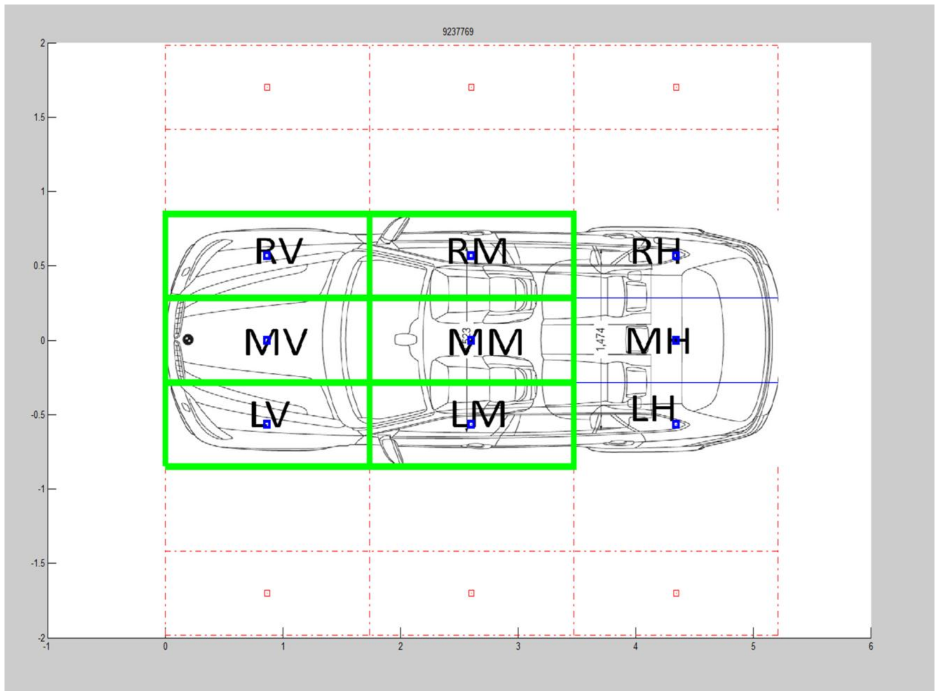
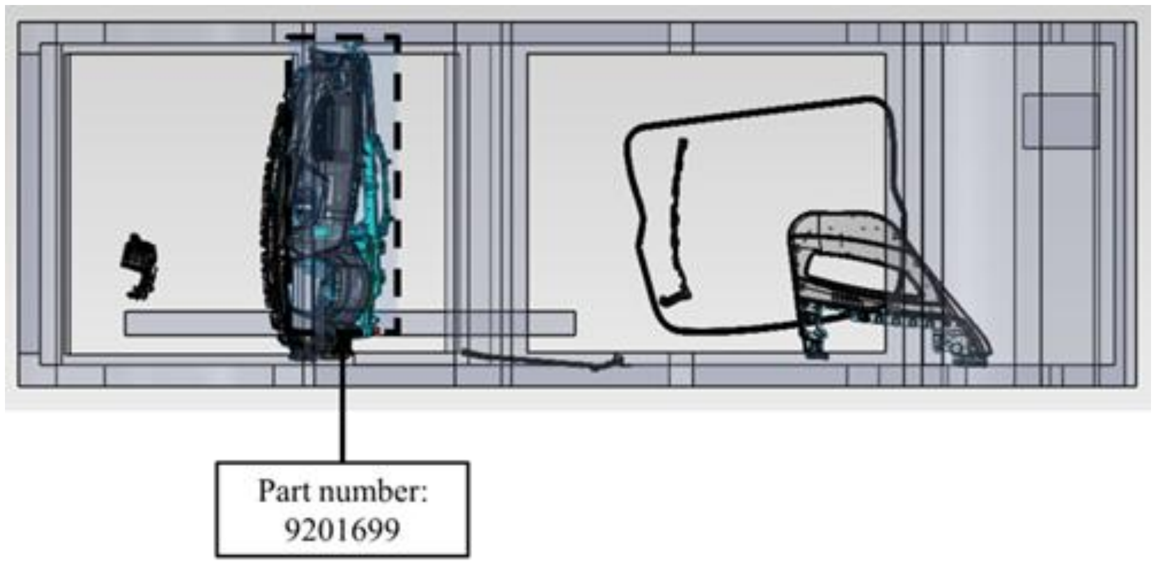


Figure 0.6: PWIT output for part number 9237769



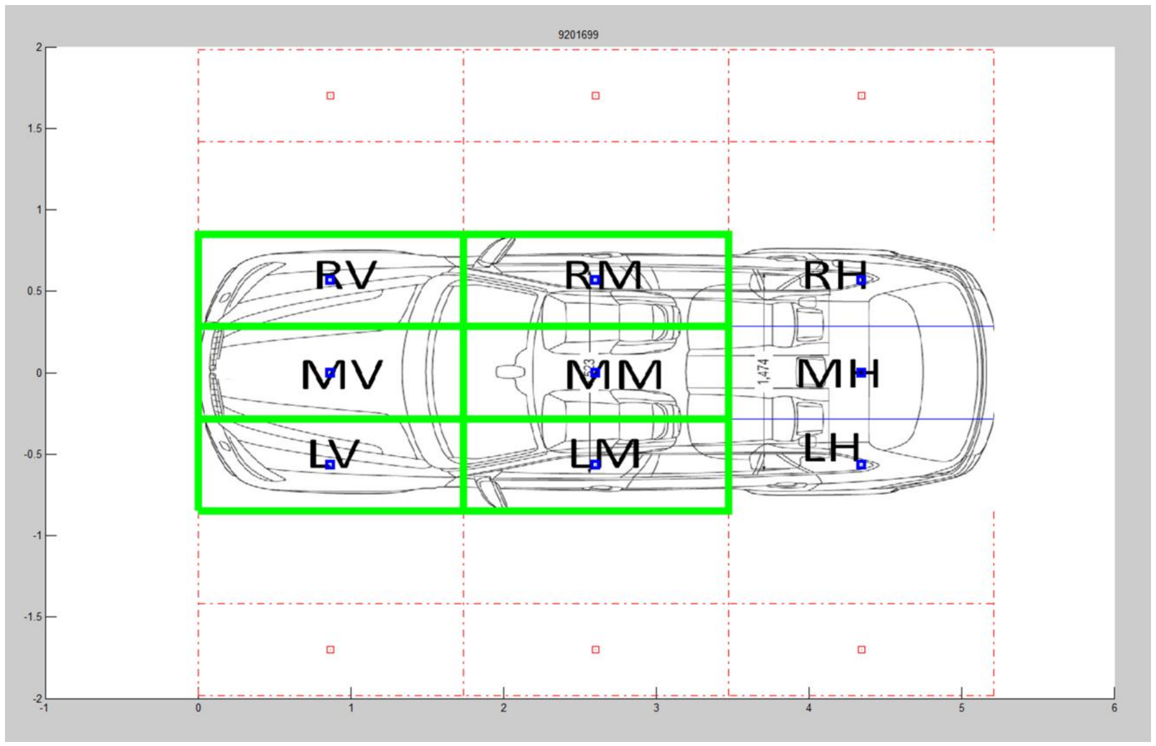
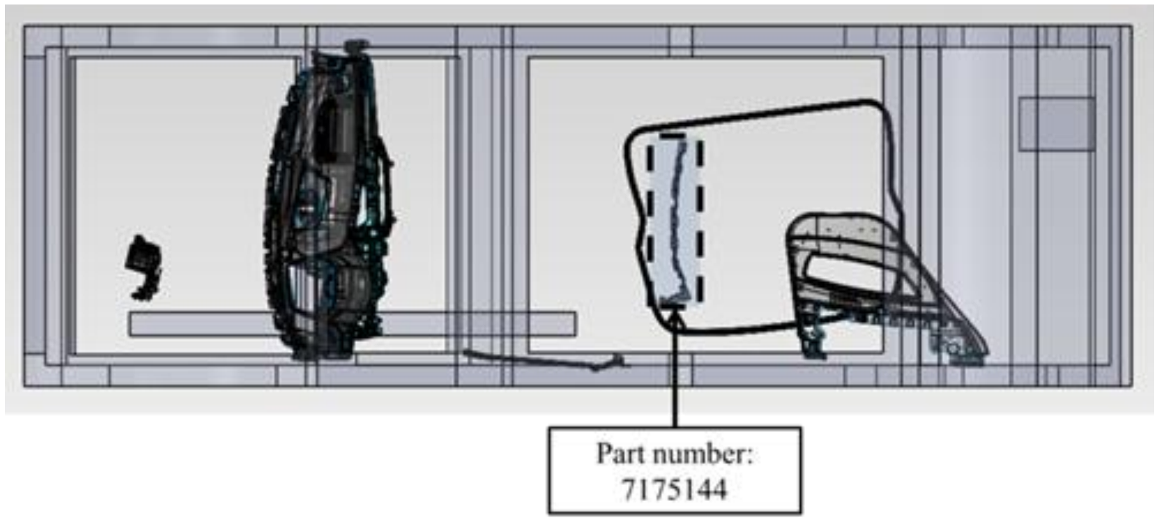


Figure 0.7: PWIT output for part number 9201699



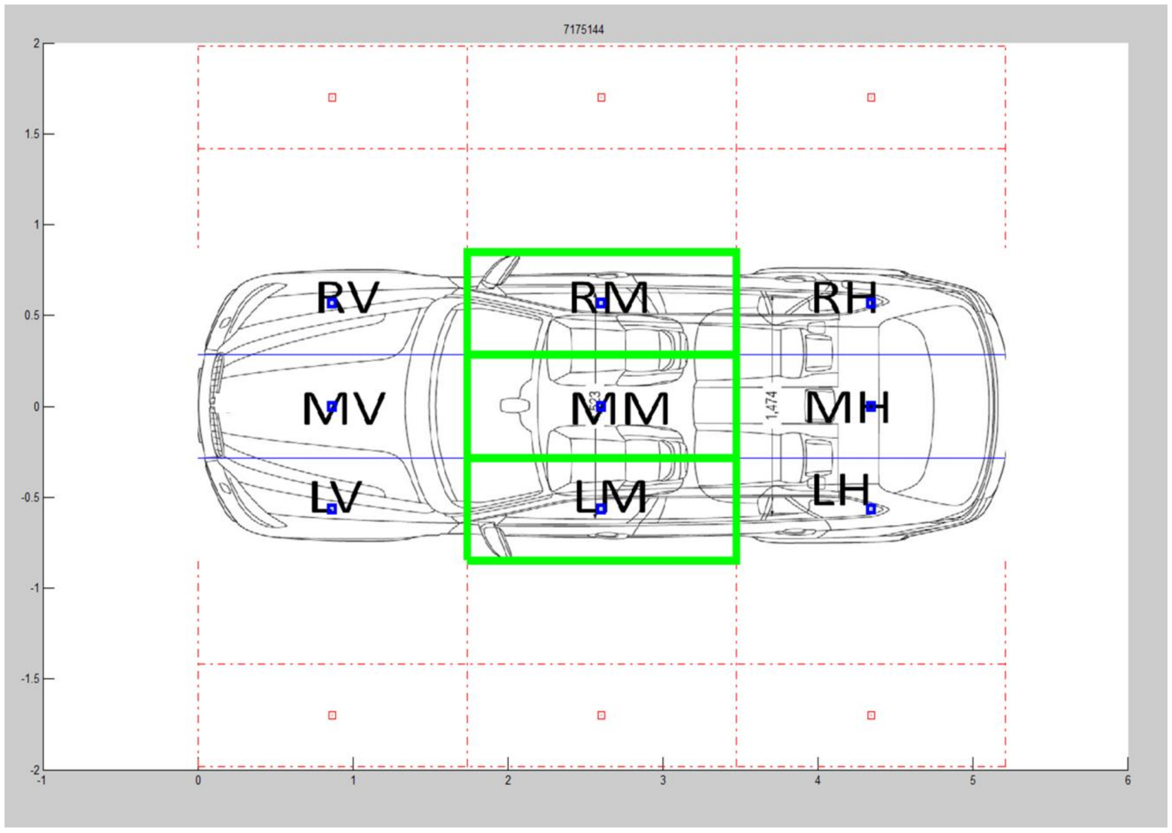


Figure 0.8: PWIT output for part number 7175144

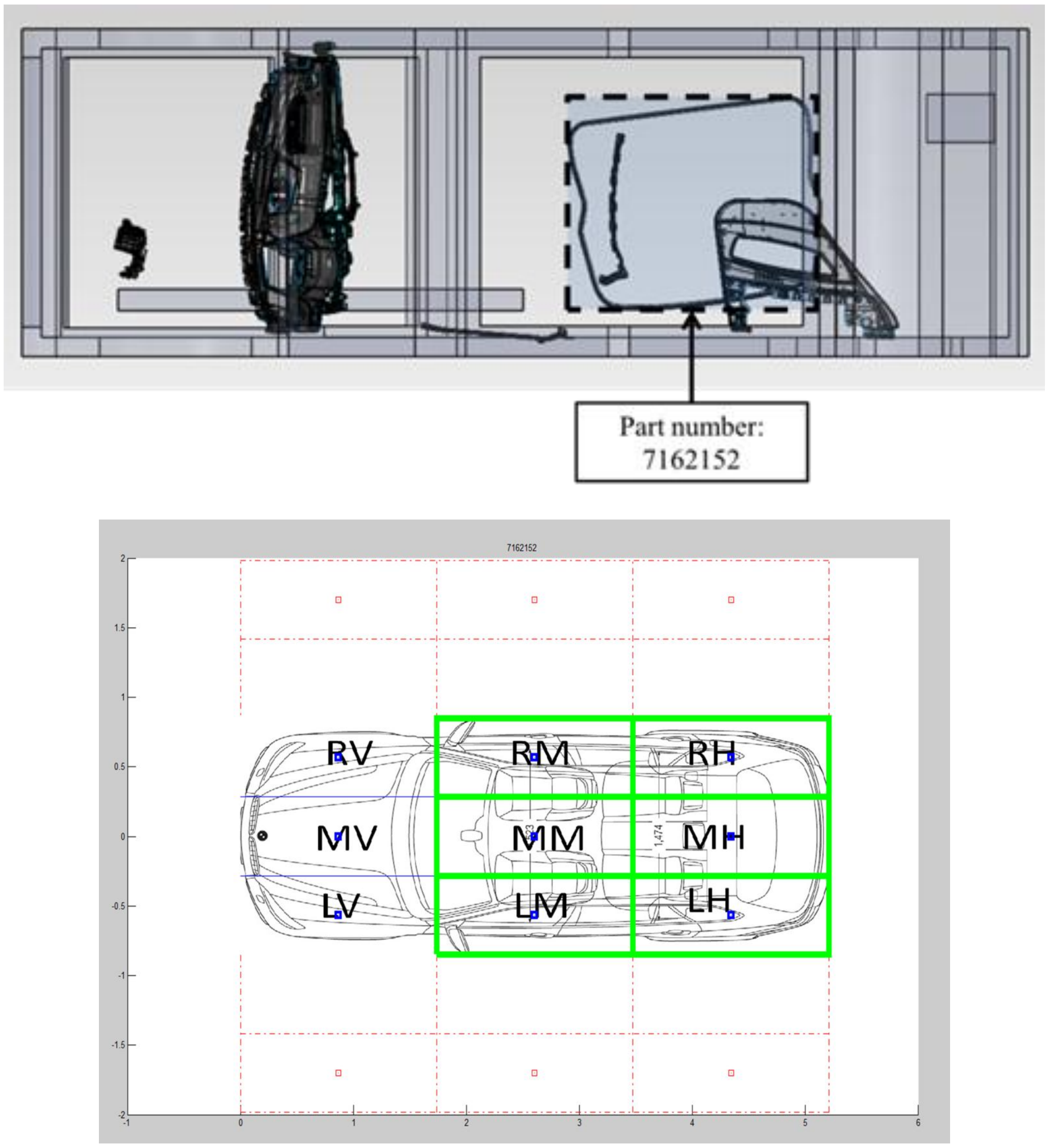
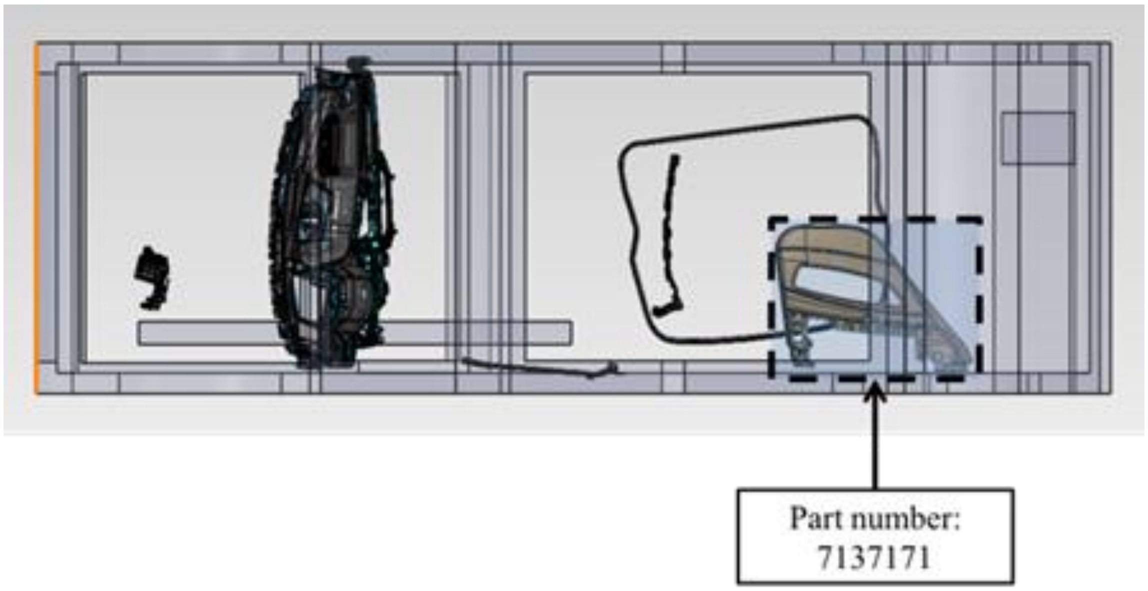


Figure 0.9: PWIT output for part number 7162152



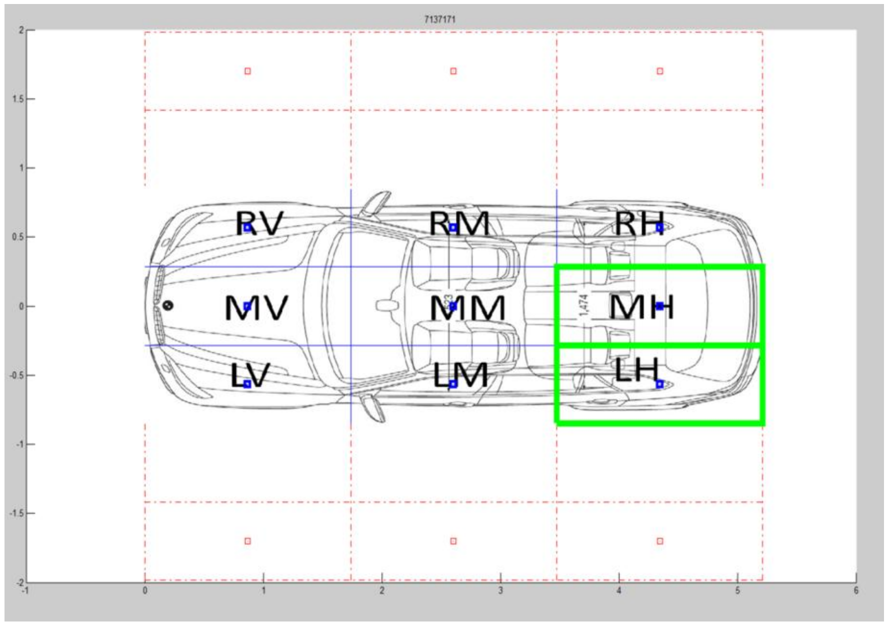


Figure 0.10: PWIT output for part number 7137171

In all the test cases that were conducted, PWIT recognized the product workspaces accurately and consistently. A single product workspace was recognized in the case of part number 7137001. Two product workspaces were identified in the case of part number 7271357, 7220381, 9193778 and 7137171. Part number 7175144 was a case which had three product workspaces. Parts 9201699 and 9237769 were nearly overlapping geometries and this was reflected in the similarity of their product workspaces as recognized by PWIT.

Several test cases were also created in order to test for the robustness of the PWIT. The goal here was to test the performance of the PWIT under circumstances that may cause the tool to fail. The following table summarizes the four test cases that were used.

Table 0.2: List of test cases to check the robustness of PWIT

1.	A part placed outside of the final product.
2.	A part placed on the inside border of the final product.
3.	A part placed on the outside border of the final product.
4.	A part placed on the border of two product workspaces.

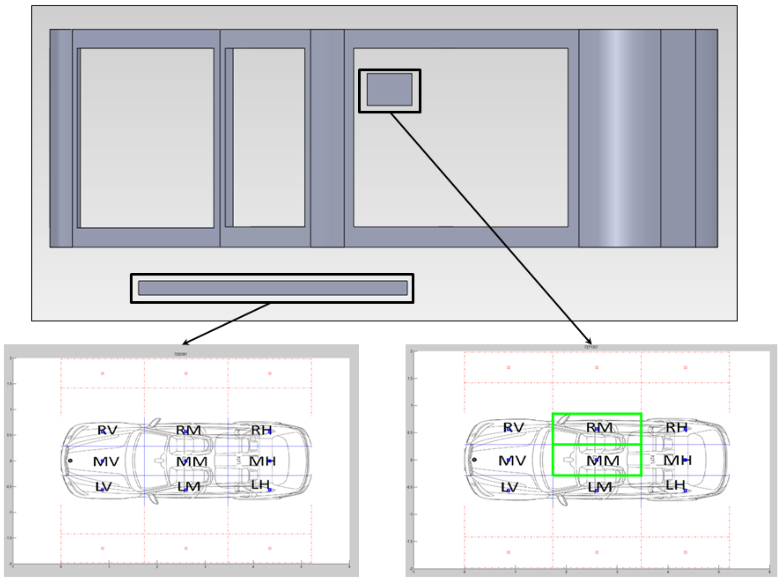


Figure 0.11: A part placed outside the final product

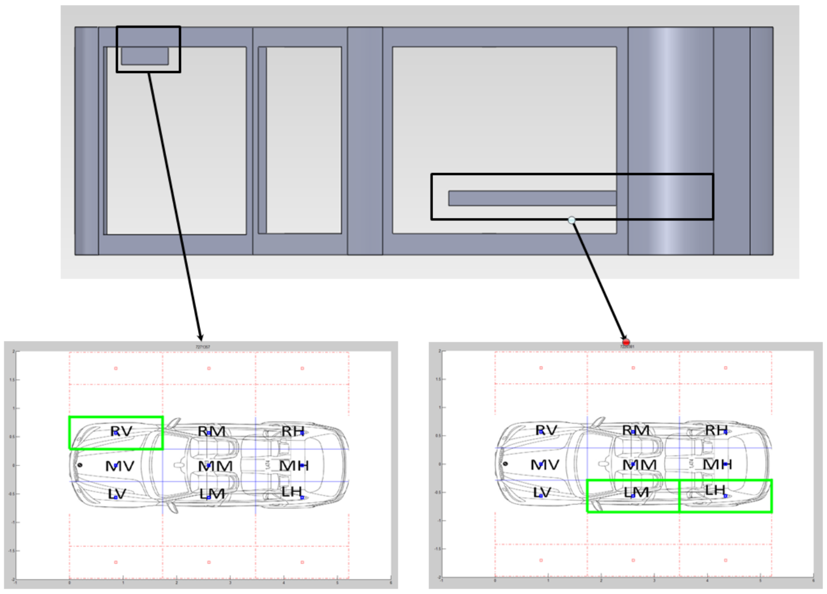


Figure 0.12: A part placed on the inside border of the final product

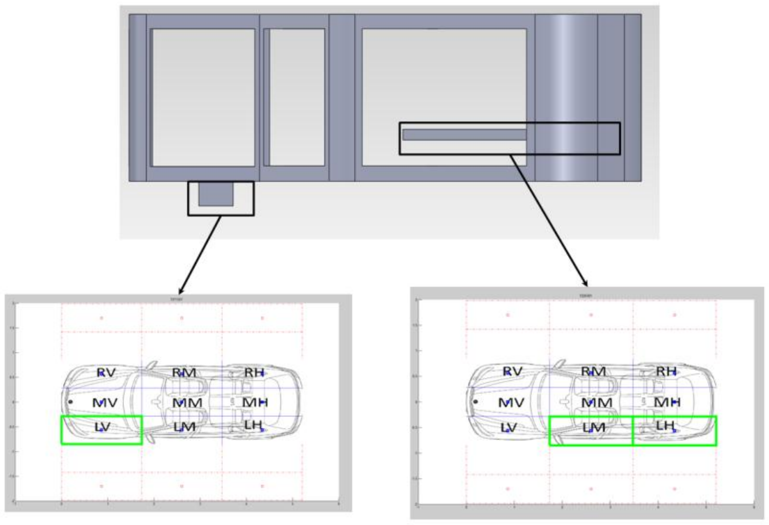


Figure 0.13: A part on the outer border of the final product

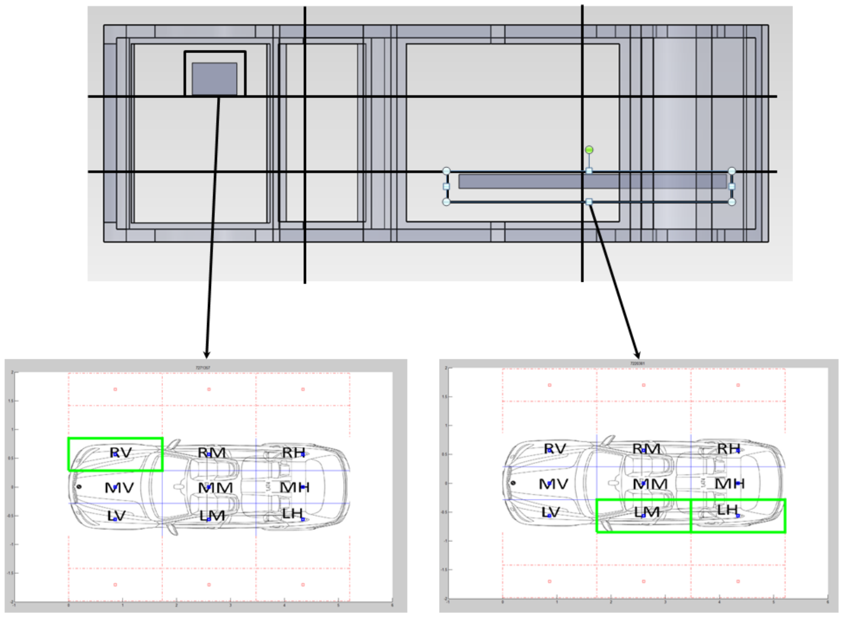


Figure 0.14: Parts on the border of two product workspaces

In each of the four test cases presented above, the PWIT generated expected results. In the first case, where a part was placed outside the final product, no product workspace was recognized for it. In the second case, when a part was placed along the inner border of the final product, the tool computed the product workspace accurately. In the third case, where a part was placed along the outer border of the final product, the PWIT recognized the most relevant product workspace. This test case is similar to the case of the side view mirror on a car. The final test case, where both parts were on the border of product workspace grids, PWIT computed the part's product workspace as that within which the part is contained.

1.13 Testing and validation of MTM estimate generator

The validity of the mappings created between MTM tables and work instruction verbs (from [2]) was evaluated by analyzing seventy work instructions from process sheets (other than the 248 that were used to create the mappings) and checking if the MTM tables present on these process sheets matched the MTM tables presented to the user by the MTM estimate generator. The following table shows the result from this analysis. The work instructions shown have only the verb(s) and the object type. This is the only information that is required to validate the mappings; the object descriptions (part names) themselves are not required.

The first column of the following table shows the verb and corresponding object type of every work instruction. The second column shows the MTM table for each work instruction as found on the process sheet. The third column shows whether or not the

MTM estimate generator provided the observed MTM table when input with the work instruction. The last column shows the frequency of occurrence of each work instruction.

Table 0.3: Results from testing and validation of MTM-standard verb mapping

Work Instruction	**MTM Table**	**Does mapping exist?**	**Frequency of Work Instruction occurrence**
Align part	Place	N	4
Apply consumable	Marking and documenting	N	2
Apply consumable	Application of medium	Y	1
Connect part	Laying cables	Y	2
Exchange plant item	Handling containers	Y	3
Get and place part	Place	N	2
Get and place part	Get and Place	Y	11
Get and place tool	Working with screws and bolts	Y	11

Get part	Get and Place	Y	2
Insert part	Sealing work	N	1
Inspect part	Visual Control	N	1
Inspect tool	Visual Control	Y	3
Place part	Place	Y	3
Process time for fastening	Process time	N	5
Process time for handstart	Process time	N	1
Scan part	Marking and documenting	Y	1
Secure part	Advanced Level/ Car body	N	3
Walk to BIW	Body motions	Y	8
Walk to plant item	Body motions	Y	8

For the seventy work instructions that were analyzed, mappings existed in 75% of the cases. It was observed that “Process time” was being used in cases where tools are being used to tighten fasteners or lift assists are operated to raise/lower parts. The need to allow the user to assign process times, where required, is recognized. The absence of mappings could also be attributed to the need for the use of hyponyms to verbs for describing work instructions. For instance, a work instruction used the verb “sit” which is a hyponym of “move”. “Sit” would be mapped to “Advanced level/ Car body” MTM table.

A set of test cases were performed to check if the order of the work instructions affected the results generated by the MTM estimate generator. All permutations (twenty-four) of four work instructions were input to the MTM estimate generator. In all twenty-four instances, the MTM estimate generator generated the same MTM tables. This is behavior meets the requirements of the tool. The following table shows one permutation of the work instructions that was input to the MTM estimate generator. A comprehensive list of the permutations can be found in the Appendix.

Table 0.4: Work instructions used to test the effect of order of work instructions on the performance of the MTM estimate generator

Read plant item
Get tool
Align part
Insert clip

The following table shows a list of the tables that was suggested by the MTM estimate generator. These tables were suggested in all twenty-four permutations of the work instructions, thus implying that the order of the work instructions does not affect the performance of the MTM estimate generator.

Table 0.5: MTM tables suggested by the MTM estimate generator

Working with screws and bolts
Working with clips
Visual control
Read

The information that is extracted from PWIT is used to perform filtering after an MTM table has been selected. The distance ranges generated by PWIT uses the coordinates of points and the distance between points equation (see Figure 0.15) to calculate the distance between a part’s product workspace and the part’s bin. Three distance ranges are used by the MTM tables, D1, D2 and D3. The range of values associated to each of these classifications can be varied.

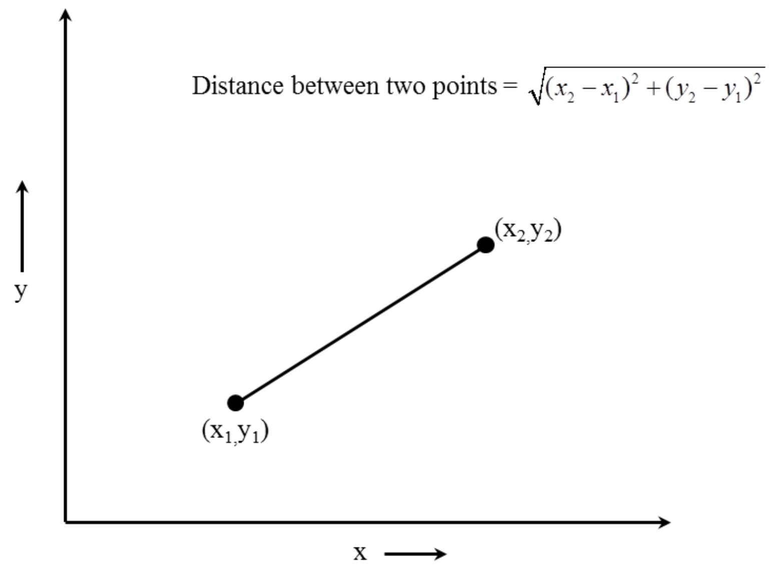

Distance between two points = $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$
(x_2, y_2)
(x_1, y_1)
y
x

Figure 0.15: Distance between two points formula used to get distance between product workspace and product bin

Furthermore, the MTM estimate generator reduces the effort expended by the planner by presenting him/her with a reduced set of MTM tables as opposed to him/her having to parse through twenty two MTM tables. It also reduces effort by performing filtering within the MTM tables based on distance ranges.

- The design process presented in section 1.8 has been used to design and implement the PWIT and the MTM estimate generator.
- The testing and validation of the PWIT showed that the tool computes and records product workspaces consistently.

- The MTM estimate generator provided decision support by presenting a reduced set of MTM tables to the process sheet author. The tool also performed filtering within the MTM tables by applying information extracted from the PWIT.
- Testing and validation of these tools also helped recognize opportunities for improvement of these tools. These potential improvements have been discussed in the following chapter.

CHAPTER FIVE
CONCLUSIONS AND FUTURE WORK

Chapter Objectives:

- Present the tools' contribution towards answering the research questions.
- Discuss the larger impact of the tools and their contribution towards a large scale production management system.
- Identify areas of future work.

This chapter presents an overview of the contribution of the two tools (the PWIT and the MTM estimate generator) towards the opportunities identified in 0. The latter half of this chapter discusses the potential extensions of the work presented in this thesis and the larger impact that the work presented has.

1.14 Review of the tools' performance with regards to the research objectives and questions

The following two sections relate each of the two research objectives to the tools that were developed to answer them.

1.14.1 Research Objective 1

A solution has been presented in this thesis by means of the Product Workspace Identification Tool (Section 1.9). This tool takes the CAD assembly file of the final product (whose assembly process is of interest) as input. It then compares the bounding box coordinates of each part to the product workspace grid. Finally, the tool creates an image of every part's product workspace and stores the product workspace information on an online database. The part-process coupling provided by [2] is extended by creating a link between part geometric information and part number. The advantage of the PWIT is that for every vehicle, irrespective of the option content, the tool will have to be run only one time.

Apart from product workspaces, the PWIT was able to capture other part information (mass properties and distance from product workspace to bin location). This information that the tool was able to capture was found to have application in automating the parsing of MTM tables. An extended and adapted version of the Product Workspace Identification Tool is one of the core constituent members of the tool developed to answer the next research objective.

1.14.2 Research Objective 2

There are twenty-two MTM tables. Assuming that the most appropriate MTM table has been selected, filtering down to select one MTM code requires several pieces of information. Performing MTM time estimation on every work instruction of every process sheet can be a tedious and error prone process.

The MTM estimate generator presented in Section 1.10 uses the part-process coupling to map work instruction action verbs to MTM tables. The part-process coupling provided by Peterson [2] is used to relate the work instruction to the part number of the object that it operates on. The extended part-process coupling provided by the PWIT is used to relate this part number from the work instruction to the part information obtained from CAD data (distance ranges). The MTM estimate generator provides decision support at two levels. First, it selects and presents a narrow range of MTM tables to the process sheet author. The down-selection of MTM tables is driven by the mapping rules that were created between work instruction text and MTM table names. Second, the tool performs filtering within the selected MTM table and reduces the effort required by the process sheet author.

1.15 Contribution of the research towards development of a large scale production management system

The tools that have been developed as a part of the research conducted, and presented in this thesis, add to the contributions of work by Peterson [2] towards building a production management system. This research has contributed towards automation of authoring process sheet information (part documentation and time study estimates). It has also contributed towards providing support to planners while they assign process sheets to stations in the line balancing process.

On a micro-level, a pervasive production management system allows for standardized documentation of the lifecycle of a product. On a macro-level, an internet-based management system will allow for research and development centers of companies to observe and use information on how each plant conducts operations. Such a system will also allow plants to interact with each other, potentially leading to improved plant-performance levels.

1.16 Future Work

1.16.1 Product Workspace Identification Tool

The current tool identifies the location of a part relative to a nine cell grid on the top view of the car and another nine cell grid on the side view of the car. It is important to note that the granularity of the grid used in the PWIT can be changed according to the product on which is it being used.

The information captured by PWIT can be used to provide some insight on the ergonomic impact of the installation of the part on the worker performing this installation. For instance, if we know that the part is located on the bottom of the vehicle, we can infer that this will potentially cause strain on the worker's back. Evidently, this will need to be coupled with workstation information and work instruction information. Workstation information can provide insight into the type of carrier that is being used to transport the vehicle. If the part is being installed on the bottom of the vehicle and a conventional skid is being used, then it can be stated with more assurance that the worker's back is being strained. If an over-head tilt is being used to raise the vehicle and tilt it sideways, then although the part is located at the bottom of the vehicle, the worker's back is less/not strained. Work instruction information plays an important role to trace the previous steps that the worker has performed. This can be used to analyze the overall ergonomic evaluation of the process sheet. Use of a screw driver can have an ergonomic impact on the wrist of the worker. This information cannot be inferred from knowledge of the product's workspace, but can be inferred from structured work instructions. The following research question summarizes the identified future work for the PWIT:

How can part-process coupling, workstation information and structured work instructions be collectively used to provide decision support during ergonomic evaluations?

1.16.2 MTM table parsing tool

The mapping between the work instructions and the MTM tables can be made more robust by recording and applying usage patterns over time (crowd-sourcing). The filtering performed after the application of the work instruction-MTM table mapping is based on distance ranges and geometric information of the part. This allows for semi-automated parsing of the selected MTM table. Manual input is necessary to perform further filtering to arrive at one MTM estimate. Other sources of information must be identified in order to completely automate the parsing of the MTM table and selection of one MTM time estimate.

Generation of distance ranges is based on manual input of the bin locations for the part. Ideally, a workstation model coupled with input from a logistics system should directly interact with and provide necessary (bin location) information to the extended Product Workspace Identification tool. The following research question summarizes the areas of future work identified for the MTM estimate generator:

How can crowd-sourcing usage patterns be applied in order to make the MTM estimate generator more intelligent? What other sources of information are required to further automate the filtering within an MTM table?

1.16.3 Other sub-systems and modules to form a complete production management system

The advantages of a pervasive production management system have been pointed out previously. There is a need to identify other modular sub-systems that will fit into this production management system, research their role, design and implement them. One such modular sub-system is a workstation model. The need for this information and its fit in the production management system can be seen in the ER diagram in Figure 0.33. The workstation model information can be used to generate workstation layouts, which can then be made available to the planners for editing. The future work identified here can be summarized by the following research question:

What other sub-systems must be designed and developed to ensure that an all pervasive production management system is created? How will these sub-systems fit into the information model presented in this thesis?

REFERENCES

[1] Noh S. D., Park Y. J., Kong S. H., Han Y.-G., Kim G., and Lee K. I., 2004, "Concurrent and collaborative process planning for automotive general assembly," The International Journal of Advanced Manufacturing Technology, 26(5-6), pp. 572–584.

[2] Peterson M. G., 2012, "STANDARDIZATION OF PROCESS SHEET INFORMATION TO SUPPORT AUTOMATED TRANSLATION OF ASSEMBLY INSTRUCTIONS AND PRODUCT," (December).

[3] Torenli A., 2009, "Assembly line design and optimization," Chalmers University of Technology.

[4] Maynard H. B., Stegemerten G. J., and Schwab J. L., 1948, "Methods-Time Measurement."

[5] Boysen N., Fliedner M., and Scholl A., 2007, "A classification of assembly line balancing problems," European Journal of Operational Research, 183(2), pp. 674–693.

[6] Merengo C., Nava F., and Pozzetti A., 1999, "International Journal of Balancing and sequencing manual mixed-model assembly lines," International Journal of Production Research, 37(12), pp. 2835–2860.

[7] Scholl A., and Becker C., 2006, "State-of-the-art exact and heuristic solution procedures for simple assembly line balancing," European Journal of Operational Research, 168(3), pp. 666–693.

[8] Rychtyckyj N., Klampfl E., and Rossi G., 2007, "Application of intelligent methods to automotive assembly planning," 2007 IEEE International Conference on Systems, Man and Cybernetics, pp. 2479–2483.

[9] Bukchin J., Dar-El E. M., and Rubinovitz J., 2002, "Mixed model assembly line design in a make-to-order environment," Computers & Industrial Engineering, 41(4), pp. 405–421.

[10] Vilarinho P. M., and Simaria A. S., 2002, "A two-stage heuristic method for balancing mixed-model assembly lines with parallel workstations," International Journal of Production Research, 40(6), pp. 1405–1420.

[11] Funk J. L., 1995, “Just-in-time manufacturing and logistical complexity : a contingency model,” International Journal of Operations & Production Management, 15(5), pp. 60–71.

[12] Alizon F., Shooter S. B., and Simpson T. W., 2009, “Henry Ford and the Model T: lessons for product platforming and mass customization,” Design Studies, 30(5), pp. 588–605.

[13] Gilmore J. H., and Pine II B. J., 1997, “Four faces of mass customization,” Harvard Business Review, 75(1), pp. 91–102.

[14] Weber J., 2009, Automotive Development Process.

[15] Hyun C. J., Kim Y., and Kim Y. K., 1998, “A genetic algorithm for multiple objective sequencing problems in mixed model assembly lines,” Computers & Operations Research, 25(7-8), pp. 675–690.

[16] Altemeier S., Helmdach M., Koberstein A., and Dangelmaier W., 2010, “Reconfiguration of assembly lines under the influence of high product variety in the automotive industry – a decision support system,” International Journal of Production, 48(21), pp. 37–41.

[17] Klampfl E., Gusikhin O., and Rossi G., 2006, “Optimization of workcell layouts in a mixed-model assembly line environment,” International Journal of Flexible Manufacturing Systems, pp. 277–299.

[18] Boothroyd G., Dewhurst P., and Knight W., 2002, “Product Design for Manufacture and Assembly.”

[19] Miller M., Griese D., Peterson M., Summers J. D., and Mocko G. M., 2012, “INSTALLATION PROCESS STEP INSTRUCTIONS AS AN AUTOMATED ASSEMBLY TIME ESTIMATION TOOL,” Chicago, pp. 1–8.

[20] Liu S., and Young R., 2004, “Utilizing information and knowledge models to support global manufacturing co-ordination decisions,” International Journal of Computer Integrated Manufacturing, 17(6), pp. 479–492.

[21] Howard M., Powell P., and Vidgen R., “Automotive Industry Information Systems : From Mass Production to,” pp. 89–102.

[22] Frohlich M. T., and Westbrook R., 2002, “Demand chain management in manufacturing and services: web-based integration, drivers and performance,” Journal of Operations Management, 20(6), pp. 729–745.

[23] Swanson C. A., and Lankford W. M., 2005, “Just-in-time manufacturing,” (1998).

[24] Taylor P., Sugimori Y., Kusunoki K., Cho F., and Uchikawa S., 1977, “Toyota production system and Kanban system Materialization of just-in-time and respect-for-human system,” International Journal of Production Research, 15(6), pp. 553–564.

[25] Pahl G., Beitz W., Feldhuesen J., and Grote K. H., 2007, Engineering design: a systematic approach, Springer-verlag, London.

[26] Pin-shan P., 1976, “The Entity-Relationship Unified View of Data Model-Toward a,” 1(1), pp. 9–36.

[27] Beynon-Davies P., 2004, “Database Systems.” 3rd Edition. Palgrave, Basingstoke, UK. ISBN 1-4039-1601-2

APPENDIX A – DATABASE SCHEMA

A.1 SQL Code to recreate relational schema for PWIT

This section shows the database schema that can be used to create a database similar to that which has been used for the Product Workspace Identification Tool .

```
-- phpMyAdmin SQL Dump
-- version 2.8.0.4
-- http://www.phpmyadmin.net
--
-- Host: localhost
-- Generation Time: Feb 18, 2013 at 02:31 PM
-- Server version: 4.1.20
-- PHP Version: 4.3.9
--
-- --------------------------------------------------------
--
-- Table structure for table ` Part`
--
CREATE TABLE `Part` (
  `ID` int(77) NOT NULL auto_increment,
  `Object` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `Part_number` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `x_coordinate` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `y_coordinate` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `z_coordinate` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `mass` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `volume` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  PRIMARY KEY  (`ID`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=6 ;
-- --------------------------------------------------------
--
-- Table structure for table `Preposition`
--
CREATE TABLE `Preposition` (
  `ID` int(77) NOT NULL auto_increment,
  `Preposition` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  PRIMARY KEY  (`ID`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=5 ;
-- --------------------------------------------------------
--
```

```
-- Table structure for table `Process_Sheet`
--
CREATE TABLE `Process_Sheet` (
  `ID` int(77) NOT NULL auto_increment,
  `Title` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  PRIMARY KEY  (`ID`)
)  ENGINE=MyISAM  DEFAULT  CHARSET=utf8  COLLATE=utf8_unicode_ci
AUTO_INCREMENT=5 ;
-- --------------------------------------------------------
--
-- Table structure for table ` WorkInstruction`
--
CREATE TABLE ` WorkInstruction` (
  `ID` int(77) NOT NULL auto_increment,
  `Process_Sheet_ID` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `Verb_ID` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `Object1_ID` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `Prep_ID` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `Object2_ID` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  PRIMARY KEY  (`ID`)
)  ENGINE=MyISAM  DEFAULT  CHARSET=utf8  COLLATE=utf8_unicode_ci
AUTO_INCREMENT=14 ;
```

```
-- --------------------------------------------------------

--

-- Table structure for table `Verb`

--

CREATE TABLE `Verb` (

  `ID` int(77) NOT NULL auto_increment,

  `Verb` varchar(77) collate utf8_unicode_ci NOT NULL default '',

  PRIMARY KEY  (`ID`)

)  ENGINE=MyISAM  DEFAULT  CHARSET=utf8  COLLATE=utf8_unicode_ci
AUTO_INCREMENT=8 ;

-- --------------------------------------------------------

--

-- Table structure for table `product_workspace`

--

CREATE TABLE `product_workspace` (

  `ID` int(77) NOT NULL auto_increment,

  `part_number` int(7) NOT NULL default '0',

  `LV` int(10) NOT NULL default '0',

  `LM` int(10) NOT NULL default '0',

  `LH` int(10) NOT NULL default '0',

  `MV` int(10) NOT NULL default '0',
  `MM` int(10) NOT NULL default '0',
  `MH` int(10) NOT NULL default '0',
  `RV` int(10) NOT NULL default '0',
  `RM` int(10) NOT NULL default '0',
  `RH` int(10) NOT NULL default '0',
  PRIMARY KEY  (`ID`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=7 ;

-- --------------------------------------------------------
```

A.2 SQL code to recreate relational schema for MTM estimate generator

The following SQL code can be used to create a relational schema that will replicate the schema used for the MTM estimate generator.

```
-- Table structure for table `Part`
--

CREATE TABLE `Part` (
  `ID` int(77) NOT NULL auto_increment,
  `Object` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `GUID` varchar(77) collate utf8_unicode_ci NOT NULL default '',
`Object_type_ID` varchar(77) collate utf8_unicode_ci NOT NULL default '',

  `x_coordinate` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `y_coordinate` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `z_coordinate` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `mass` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `volume` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  PRIMARY KEY  (`ID`)
) ENGINE=MyISAM  DEFAULT  CHARSET=utf8  COLLATE=utf8_unicode_ci
AUTO_INCREMENT=6 ;

-- --------------------------------------------------------

--
-- Table structure for table ` Object_Type`
--

CREATE TABLE ` Object_Type` (
  `ID` int(77) NOT NULL auto_increment,
  `Description` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  PRIMARY KEY  (`ID`)
```

```
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=1 ;

-- --------------------------------------------------------

--
-- Table structure for table ` Preposition`
--

CREATE TABLE ` Preposition` (
  `ID` int(77) NOT NULL auto_increment,
  `Preposition` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  PRIMARY KEY  (`ID`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=5 ;

-- --------------------------------------------------------

--
-- Table structure for table `Process_sheet`
--
```

```
CREATE TABLE `Process_sheet ` (
  `ID` int(77) NOT NULL auto_increment,
  `Title` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  PRIMARY KEY  (`ID`)
)   ENGINE=MyISAM   DEFAULT   CHARSET=utf8   COLLATE=utf8_unicode_ci
AUTO_INCREMENT=5 ;

-- --------------------------------------------------------

--
-- Table structure for table ` Process_sheet_has_code`
--

CREATE TABLE `Process_sheet_has_code` (
  `Process_Sheet_id` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `Code_id` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `Table_id` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `TMU` varchar(77) collate utf8_unicode_ci NOT NULL default ''
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

-- --------------------------------------------------------
```

```
--
-- Table structure for table `MTM_Table`
--

CREATE TABLE `MTM_Table` (
  `ID` int(77) NOT NULL auto_increment,
  `Table_name` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  PRIMARY KEY  (`ID`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=23 ;

-- --------------------------------------------------------

--
-- Table structure for table `MTM_WorkInstruction`
--

CREATE TABLE `MTM_WorkInstruction` (
  `ID` int(77) NOT NULL auto_increment,
  `Process_sheet_ID` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `Verb_ID` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `Object1_ID` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `Prep_ID` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `Object2_ID` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  PRIMARY KEY  (`ID`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=14 ;

-- --------------------------------------------------------

--
-- Table structure for table `MTM_code_has_table`
--

CREATE TABLE `MTM_code_has_table` (
  `Code_id` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  `Table_id` varchar(77) collate utf8_unicode_ci NOT NULL default ''
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

-- --------------------------------------------------------

--
-- Table structure for table `MTM_codes`
--
```

```
CREATE TABLE `MTM_codes` (
  `ID` int(77) NOT NULL auto_increment,
  `Code` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  PRIMARY KEY  (`ID`)
) ENGINE=MyISAM  DEFAULT  CHARSET=utf8  COLLATE=utf8_unicode_ci
AUTO_INCREMENT=231 ;

-- --------------------------------------------------------

--
-- Table structure for table `Verb`
--

CREATE TABLE `Verb` (
  `ID` int(77) NOT NULL auto_increment,
  `Verb` varchar(77) collate utf8_unicode_ci NOT NULL default '',
  PRIMARY KEY  (`ID`)
) ENGINE=MyISAM  DEFAULT  CHARSET=utf8  COLLATE=utf8_unicode_ci
AUTO_INCREMENT=8 ;

-- --------------------------------------------------------
```

--

A.3 Comprehensive list of permutations for work instructions used to test MTM estimate generator

Sl. No.	Step 1	Step 2	Step 3	Step 4
1.	Read plant item	Align part	Get tool	Insert clip
2.	Read plant item	Align part	Insert clip	Get tool
3.	Read plant item	Get tool	Align part	Insert clip
4.	Read plant item	Get tool	Insert clip	Align part
5.	Read plant item	Insert clip	Align part	Get tool
6.	Read plant item	Insert clip	Get tool	Align part
7.	Align part	Read plant item	Insert clip	Get tool

8.	Align part	Read plant item	Get tool	Insert clip
9.	Align part	Get tool	Insert clip	Read plant item
10.	Align part	Get tool	Read plant item	Insert clip
11.	Align part	Insert clip	Get tool	Read plant item
12.	Align part	Insert clip	Read plant item	Get tool
13.	Get tool	Read plant item	Align part	Insert clip
14.	Get tool	Read plant item	Insert clip	Align part
15.	Get tool	Align part	Read plant item	Insert clip
16.	Get tool	Align part	Insert clip	Read plant item
17.	Get tool	Insert clip	Read plant item	Align part
18.	Get tool	Insert clip	Align	Read

			part	plant item
19.	Insert clip	Read plant item	Get tool	Align part
20.	Insert clip	Read plant item	Align part	Get tool
21.	Insert clip	Align part	Get tool	Read plant item
22.	Insert clip	Align part	Read plant item	Get tool
23.	Insert clip	Get tool	Align part	Read plant item
24.	Insert clip	Get tool	Read plant item	Align part