

**Development of VR-Simulator Software
for Manufacturing Systems as a
Decision Making and Simulation Tool**

By

Tariq Saad Mujber, Master of Science (MSc)

**This thesis is submitted to Dublin City University as the fulfilment of
the requirement for the award of the degree of**

Doctor of Philosophy

Supervisors:

Professor M.S.J. Hashmi, Ph.D. D.Sc.

Dr. Tamas, Szecsi, Ph.D.

**School of Mechanical and Manufacturing Engineering
Dublin City University**

DCU

2005

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Doctor of Philosophy, is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed: _____



ID No: _____

51165902

Tariq Saad Mujber

Date: _____

31-08-2005

Acknowledgment

First, I am ever grateful to Allah for all the blessings He bestowed on me in my entire life.

My most profound gratitude and thanks go to my supervisors Prof. Saleem Hashmi and Dr. Tamas Szecsi for their guidance, enthusiastic supervision, encouragement and help throughout the duration of the research.

Thanks are extended to the Mechanical and Manufacturing Engineering faculty members and technical staff of Dublin City University for their assistance throughout my study. I am equally indebted to Dr. John Geraghty for his time and valuable advice and support.

Special thanks to Dr. Amr Arisha who has given me valuable support and advice to accomplish this project. I would like to thank also my college Mr. Shafi Khanian for the brotherly discussions on academic and personal levels.

I wish to express my appreciation and thanks for the fruitful experience I have enjoyed at the Educational Services Department, DCU.

I am grateful to my country Libya for having offered me the scholarship for pursuing the postgraduate study at Dublin City University. Special thanks to all my friends in Ireland for being the surrogate family and their caring friendship during my study.

At last but never the least, I am forever indebted to my wonderful parents, wife and son for their understanding, endless patience and encouragement when it was most needed. I would like to thank also my brothers, sister, my brother in law, my sister in law and the family of my wife for their love and family support that sustained me throughout my life at home and abroad.

ABSTRACT

By: Tariq Saad Mujber, MSc

Modern manufacturing is characterised by high levels of automation and integration, complex interactions among system elements, and high capital costs. Simulation and Virtual Reality technologies hold tremendous promise for reducing costs, improving quality, process data management, enhancing control over operations and shortening the time-to-market for manufactured goods. Unfortunately, these technologies still remains largely underutilised by industry today due to the requirement for a high standard of skills in programming and modelling methodologies.

Visualisation has become a critical component of simulation technology in manufacturing applications. It provides the simulation practitioners with an environment to discuss and get a better understanding of the simulation model's behaviour. Graphical presentation and animation can be a significant tool to communicate the outcome of simulation models for the non-technical audience. Decision makers often do not have the technical knowledge to understand the statistical results of a simulation model. But when the outcome can be expressed using animation, a better level of understanding becomes possible.

This thesis presents a VR-Simulator software developed entirely by the author to overcome some of the limitations of simulation packages to allow users (who are not specialists in simulation and virtual reality techniques, or have no programming skills) to develop simulation and virtual models of manufacturing systems automatically without any need for excessive training on modelling techniques or programming. The users can interact with the generated models using voice commands and virtual reality devices (e.g. HMD). The VR-Simulator can be used as an operational decision-support tool to enable decision makers to model and analyse manufacturing systems.

Table of Contents

Table Of Contents	I
List Of Figures.....	V
List Of Tables.....	XI
Glossary Of Acronyms.....	XII
Introduction	1
1.1 INTRODUCTION	1
1.2 PROBLEM DEFINITION	2
1.3 OBJECTIVES OF RESEARCH.....	3
1.4 STRUCTURE OF THESIS.....	4
Chapter 2.....	6
Literature Review	6
2.1 INTRODUCTION	6
2.2 MANUFACTURING SYSTEMS DESIGN AND CLASSIFICATION	7
2.2.1 Function In Manufacturing	9
2.3 PROCESS MODELLING.....	10
2.3.1 Modelling Approaches.....	12
2.3.1.1 Flow Diagram	13
2.3.1.2 IDEF.....	13
2.3.2 Commercial Process Modelling Tools.....	15
2.4 SIMULATION.....	17
2.4.1 Applications of Simulation in Manufacturing.....	18
2.4.1.1 System Design	20
2.4.1.2 System Management.....	20
2.4.2 Simulation Tools.....	22
2.4.2.1 Simulation Languages	22
2.4.2.2 Simulation Software Packages	23
2.4.2.3 Limitations of the Current Available Simulation Packages	27
2.4.2.4 Critical Evaluation of Simulation Application in Manufacturing	28
2.4.2.5 Future of Simulation in Manufacturing.....	29
2.5 VIRTUAL REALITY	29
2.5.1 Components of Virtual Reality.....	30
2.5.1.1 Hardware	31
2.5.1.2 Software	35
2.5.2 Types of VR Systems.....	37
2.5.2.1 Non-Immersive VR Systems	37
2.5.2.2 Semi-Immersive VR Systems	38
2.5.2.3 Fully Immersive VR Systems	38

2.5.3	Virtual Reality Tools	39
2.5.4	Virtual Reality in Manufacturing	40
2.5.4.1	Design.....	41
2.5.4.2	Operations Management.....	42
2.5.4.3	Manufacturing Processes	45
2.5.5	Future of Virtual Reality in Manufacturing.....	47
2.6	INTEGRATION OF SIMULATION WITH VR IN MANUFACTURING	48
2.7	CONCLUSIONS	50

Chapter 3..... 51

Pilot Model: Integration of Simulation with VR 51

3.1	INTRODUCTION	51
3.2	DEVELOPING THE SIMULATION MODEL OF THE FACTORY	52
3.2.1	Project Definition.....	52
3.2.1.1	Setting the Objectives	54
3.2.1.2	Determining the Level of Details	54
3.2.1.3	Data Collection	54
3.2.2	Model Building	60
3.2.2.1	Structure of the Model.....	61
3.2.2.2	Coding	64
3.2.3	Model Testing	67
3.2.3.1	Model Verification	67
3.2.3.2	Model Validation	68
3.2.4	Experimental Design.....	70
3.2.4.1	Model Run	70
3.2.4.2	Output Analysis.....	72
3.2.4.3	More Runs	74
3.2.5	Completion Phase	75
3.2.5.1	Implementations.....	75
3.3	DEVELOPING THE VIRTUAL MODEL OF THE FACTORY.....	76
3.3.1	Environment Planning.....	78
3.3.2	VR Model Building.....	78
3.3.3	Create Objects	79
3.3.4	Adding Attributes to Objects	80
3.3.5	Adding Properties and Behaviours to Objects	81
3.3.6	Testing the Virtual Model	84
3.3.7	Model Optimisation	84
3.4	LINKING THE SIMULATION MODEL WITH THE VIRTUAL MODEL OF THE FACTORY	85
3.5	INTERACTION WITH THE DEVELOPED MODELS OF THE FACTORY	88
3.5.1	Navigation the Virtual Model of the Factory	92
3.6	CONCLUSIONS	92

Chapter 4..... 93

Design And Development Of The VR-Simulator 93

4.1	INTRODUCTION	93
4.2	THE VR-REALITY SOFTWARE ARCHITECTURE	95
4.3	TOOLS USED TO DESIGN AND DEVELOP THE VR-SIMULATOR	96
4.3.1	Visual Basic	96
4.3.2	Witness	97
4.3.3	Superscape VRT.....	97
4.3.4	Microsoft Excel.....	97
4.4	VR-SIMULATOR SOFTWARE DESIGN AND DEVELOPMENT.....	98
4.4.1	VR-Simulator Architectural Issues	98
4.4.2	Design the Components of the VR-Simulator Software.....	100
4.4.3	The VR-Simulator Graphic User Interface.....	103
4.4.4	Process Modelling Tool Design	109
4.4.4.1	Process Modelling Tool Presentation	111
4.4.4.2	Factory Layout	128
4.4.5	Simulation Modelling Tool Design	130
4.4.5.1	Formulating WCL Commands	131
4.4.5.2	Generating Simulation Models	139
4.4.6	Virtual Modelling Tool Design	153
4.4.6.1	Design and Implementation of Virtual Environment Template	153
4.4.6.2	Generating Virtual Models.....	160
4.4.7	Models Running Tool	165
4.4.7.1	Simulating the Activities within the Virtual Model	169
4.4.7.2	Visualising the Activities of a Manufacturing System.....	171
4.4.7.3	Navigating the Virtual Model.....	172
4.4.7.4	Interacting with the Models of the VR-Simulator	176
4.4.8	Output Analysing Tools	183
4.5	CONCLUSION	186

Chapter 5.....188

VR-Simulator Software Testing And Evaluations188

5.1	INTRODUCTION	188
5.2	VR-SIMULATOR SOFTWARE TESTING.....	189
5.3	VR-SIMULATOR SOFTWARE EVALUATIONS.....	191
5.3.1	Real Case Study of a Job Shop Model.....	191
5.3.1.1	Defining the Job Shop Model	193
5.3.1.2	Constructing the Job Shop Model Layout	200
5.3.1.3	Developing a Simulation Model of the Job Shop Model	201
5.3.1.4	Developing a Virtual Model of the Job Shop Model	202
5.3.1.5	Loading The Process Plan of the Job Shop Model	204
5.3.1.6	Results and Discussions	204
5.3.1.7	Verification and Validation the Job Shop Model	207

5.3.2	Hypothetical Example of Flow Shop Model of a Manufacturing System	207
5.3.2.1	Flow Shop Model Presentation.....	208
5.3.2.2	Model Assumptions.....	209
5.3.2.3	Defining the Job Shop Model	213
5.3.2.4	Constructing the Job Shop Model Layout	218
5.3.2.5	Developing a Simulation Model of the Job Shop Model	219
5.3.2.6	Developing a Virtual Model of the Job Shop Model	221
5.3.2.7	Loading the Process Plan of the Job Shop Model	222
5.3.2.8	Results and Discussions	222
5.4	MANUFACTURING SYSTEMS	225
5.5	THE VR-SIMULATOR LIMITATIONS AND BENEFITS	226
5.5.1	The VR-Simulator Software General Features	226
5.5.2	The Limiting Factors of the VR-Simulator Software	228
5.5.3	The Scope of The VR-Simulator Application.....	230
5.6	CONCLUSIONS	231
Chapter 6.....		232
Conclusions and Future Research Work.....		232
6.1	THESIS SUMMERY	232
6.2	CONCLUSIONS	233
6.3	RECOMMENDATIONS FOR FUTURE RESEARCH WORK.....	234
Publications Of The Author		237
References		238
APPENDICES		
APPENDIX A.....		248

List of Figures

FIGURE 2.1: TYPES OF PRODUCTION	8
FIGURE 2.2: APPLICATIONS OF SIMULATION IN MANUFACTURING.....	21
FIGURE 2.3: MAIN COMPONENTS OF A VR HARDWARE SYSTEM	31
FIGURE 2.4: THE INTEGRATION OF VARIOUS ELEMENTS OF GENERIC VR SYSTEM.....	33
FIGURE 2.5: VR SYSTEM SOFTWARE.....	36
FIGURE 2.6: FULLY IMMERSIVE VR ENVIRONMENT FOR FACTORY DESIGN.....	44
FIGURE 2.7: VIRTUAL FACTORY CREATED BY WITNESS VR.....	44
FIGURE 2.8: VIRTUAL MACHINE TOOL.....	45
FIGURE 3.1: THE SIMULATION MODELLING PROCESS OF THE FACTORY	53
FIGURE 3.2: FACTORY LAYOUT DATA SHEET	56
FIGURE 3.3: SPANCRETE DATA SHEET	57
FIGURE 3.4: MATERIAL HANDLING1 DATA SHEET	58
FIGURE 3.5: MATERIAL HANDLING2 DATA SHEET.....	59
FIGURE 3.6: LABOURERS DATA SHEET	59
FIGURE 3.7: COLOURS CODE DATA SHEET.....	60
FIGURE 3.8: SNAPSHOT OF GUI OF WITNESS SOFTWARE.....	61
FIGURE 3.9: PROCESSES OF THE FACTORY	62
FIGURE 3.10: SCREEN EDITOR OF WITNESS	63
FIGURE 3.11 SNAPSHOT OF THE SIMULATION MODEL.....	64
FIGURE 3.12: DETAILING BED1	65
FIGURE 3.13: DETAILING THE SPANCRETE MACHINE	66
FIGURE 3.14: DUMMY MACHINES, PARTS AND VARIABLES USED TO MONITOR THE ACTIVITIES OF THE SIMULATION MODEL	71
FIGURE 3.15: SIMULATING THE ACTIVITIES OF THE FACTORY	72
FIGURE 3.16 UTILISATION OF THE MACHINES AND THE LABOURERS	73
FIGURE 3.17: CONCRETE WITHIN THE SPANCRETE MACHINE	73
FIGURE 3.18: MONITORING THE ACTIVITIES OF THE OPERATORS.....	74
FIGURE 3.19: DATA INFORMATION OF THE FACTORY.....	74
FIGURE 3.20: STEPS OF DESIGNING AND CREATING THE VIRTUAL MODEL OF THE FACTORY	77

FIGURE 3.21: WORLD EDITOR OF SUPERSCAPE VRT	79
FIGURE 3.22: SHAPE EDITOR.....	80
FIGURE 3.23: DYNAMIC ATTRIBUTE WINDOW	81
FIGURE 3.24: OBJECT PROPERTIES WINDOW.....	82
FIGURE 3.25: SCL EDITOR OF SUPERSCAPE VRT	82
FIGURE 3.26: SNAPSHOT OF THE SIMULATION MODEL OF THE FACTORY	84
FIGURE 3.27: LINKING THE SIMULATION MODEL WITH THE VIRTUAL MODEL OF THE FACTORY.....	86
FIGURE 3.28: ALGORITHM FOR EXCHANGING THE DATA OF THE SPANCRETE MACHINE IN THE SIMULATION MODEL WITH THE SPANCRETE MACHINE IN THE VIRTUAL MODEL	88
FIGURE 3.29: FLOWCHART ILLUSTRATING HOW THE USER CAN INTERACT WITH THE MODELS OF THE FACTORY.....	89
FIGURE 3.30: SCHEMATIC DIAGRAM OF THE USER INTERACTION WITH THE MODELS.....	90
FIGURE 3.31: INTERACTION DIALOG BOX.....	91
FIGURE 4.1: THE VR-SIMULATOR SOFTWARE ARCHITECTURE	95
FIGURE 4.2: VR-SIMULATOR STRUCTURAL DESIGN	99
FIGURE 4.3: MAIN COMPONENTS OF THE VR-SIMULATOR SOFTWARE	101
FIGURE 4.4: DIAGRAM SHOWS HOW THE COMPONENTS OF THE VR-SIMULATOR COMMUNICATE	102
FIGURE 4.5: GRAPHICAL USER INTERFACE STRUCTURAL DESIGN	103
FIGURE 4.6: GUI OF VR-SIMULATOR SOFTWARE	104
FIGURE 4.7: EXCEL FILE CREATED AUTOMATICALLY	105
FIGURE 4.8: OPEN WINDOW TO LOAD A DATABASE OF A MANUFACTURING SYSTEM	106
FIGURE 4.9: SAVE DATABASE WINDOW.....	107
FIGURE 4.10: MAIN ELEMENTS OF A MANUFACTURING SYSTEM	110
FIGURE 4.11: PART DETAILED DIALOG PAGE	112
FIGURE 4.12: MACHINE DETAILED DIALOG PAGE	113
FIGURE 4.13: LABOURERS DETAIL DIALOG PAGE.....	115
FIGURE 4.14: AGVS DETAIL DIALOG PAGE	117
FIGURE 4.15: TRANSPORTERS DETAIL DIALOG PAGE	118
FIGURE 4.16: CONVEYOR DETAIL DIALOG BOX.....	119

FIGURE 4.17: SHIFT DETAIL DIALOG PAGE	120
FIGURE 4.18: MACHINE TOOLS DETAIL DIALOG PAGE	121
FIGURE 4.19: PROCESSES LIST DETAIL DIALOG PAGE	122
FIGURE 4.20: BUFFERS DETAIL DIALOG PAGE	123
FIGURE 4.21: PROCESS FLOW DETAIL DIALOG PAGE.....	124
FIGURE 4.22: ORDERS DETAIL DIALOG PAGE	126
FIGURE 4.23: RELATIONSHIPS BETWEEN THE ELEMENTS OF THE PMT	127
FIGURE 4.24: FACTORY LAYOUT OF THE VR-SIMULATOR SOFTWARE.....	128
FIGURE 4.25: TRANSPORTERS DETAIL DIALOG	129
FIGURE 4.26: LINKING THE FACTORY LAYOUT TOOL WITH THE MAIN COMPONENTS OF THE VR-SIMULATOR SOFTWARE	130
FIGURE 4.27: FORMULATING WCL COMMANDS	131
FIGURE 4.28: DESIGNER ELEMENTS OF WITNESS	132
FIGURE 4.29: DRAGGING AND DROPPING THE MACHINE RESOURCE IN WITNESS.....	132
FIGURE 4.30: MACHINE DETAIL DIALOG	133
FIGURE 4.31: SAVING THE MODEL AS LIBRARY FILE.....	134
FIGURE 4.32: SNAPSHOT OF THE MODEL LIBRARY FILE OF THE MACHINE	134
FIGURE 4.33: LOADING A DATABASE OF A MANUFACTURING SYSTEM.....	140
FIGURE 4.34: GENERATE AUTOMATIC SIMULATION MODEL COMMAND	140
FIGURE 4.35: ALGORITHM TO GENERATE SIMULATION MODEL AUTOMATICALLY	141
FIGURE 4.36: GENERATING SHIFTS AUTOMATICALLY IN WITNESS	142
FIGURE 4.37: GENERATING PARTS AUTOMATICALLY IN WITNESS.....	143
FIGURE 4.38: GENERATING MACHINES AUTOMATICALLY IN WITNESS.....	144
FIGURE 4.39: GENERATING CONVEYORS AUTOMATICALLY IN WITNESS	146
FIGURE 4.40: GENERATING AGVS AUTOMATICALLY IN WITNESS.....	147
FIGURE 4.41: GENERATING TRANSPORTERS AUTOMATICALLY IN WITNESS	148
FIGURE 4.42: GENERATING LABOURERS AUTOMATICALLY IN WITNESS.....	149
FIGURE 4.43: GENERATING MACHINE TOOLS AUTOMATICALLY IN WITNESS	151
FIGURE 4.44: GENERATING BUFFERS AUTOMATICALLY IN WITNESS.....	152
FIGURE 4.45: STEPS OF DESIGN AND IMPLEMENTATION OF THE VIRTUAL ENVIRONMENT TEMPLATE.....	153
FIGURE 4.46: SNAPSHOT OF THE VIRTUAL FACTORY	155
FIGURE 4.47: PROCEDURE FOR CREATING THE RESOURCES	156

FIGURE 4.48: VIRTUAL DRILLING MACHINE	157
FIGURE 4.49: PROPERTIES OF THE DRILLING MACHINE.....	158
FIGURE 4.50: UPLOADING THE CREATED RESOURCES TO THE WAREHOUSE LIBRARY OF SUPERSCAPE	160
FIGURE 4.51: GENERATE AUTOMATIC VIRTUAL MODEL COMMAND	160
FIGURE 4.52: ALGORITHM TO GENERATE VIRTUAL MODELS AUTOMATICALLY.....	161
FIGURE 4.53: GENERATING MACHINES AUTOMATICALLY IN THE VIRTUAL MODEL	162
FIGURE 4.54: SCL USED TO GENERATE THE LAYOUT OF THE VIRTUAL MODEL	164
FIGURE 4.55: MODELS RUN TOOL OPTIONS	165
FIGURE 4.56: TRANSFERRING THE PROCESS PLAN FROM THE EXCEL DATA SHEETS TO THE SIMULATION MODEL IN WITNESS	166
FIGURE 4.57: ASSIGNING A ROUTE TO A PARTS IN WITNESS	166
FIGURE 4.58: BATCH RUN DIALOG BOX	167
FIGURE 4.59: RUN EXECUTION ALGORITHM	168
FIGURE 4.60: LINKING THE SIMULATION MODEL WITH THE VIRTUAL MODEL.....	170
FIGURE 4.61: EXCHANGING DATA OF THE DRILLING MACHINE IN THE SIMULATION MODEL WITH THE DRILLING MACHINE IN THE VIRTUAL MODEL.	171
FIGURE 4.62: CARTESIAN COORDINATE	172
FIGURE 4.63: CYLINDRICAL COORDINATE	173
FIGURE 4.64: VIRTUAL AGV	174
FIGURE 4.65: AN AGV DELIVERING PARTS TO THE SAW MACHINE	174
FIGURE 4.66: NAVIGATION BAR.....	175
FIGURE 4.67: FLOWCHART ILLUSTRATING HOW THE USER CAN INTERACT WITH THE MODELS OF A MANUFACTURING SYSTEM.....	176
FIGURE 4.68: INTERACTION DIALOG BOX	177
FIGURE 4.69: SPEECH RECOGNITION PROCESS FLOW.....	179
FIGURE 4.70: SNAPSHOT OF THE VOICE TRAINING SOFTWARE.....	179
FIGURE 4.71: VOICE COMMANDS INTERACTIONS.....	180
FIGURE 4.72: INTERFACE CIRCUIT FOR THE EXTERNAL SWITCHES	182
FIGURE 4.73: OUTPUT ANALYSING TOOLS.....	183
FIGURE 4.74: MACHINES REPORTS.....	185
FIGURE 4.75: PRODUCTS REPORTS	185
FIGURE 4.76: LABOURERS REPORTS.....	185

FIGURE 4.77: CONVEYORS REPORTS	185
FIGURE 4.78: AGVS REPORTS	185
FIGURE 4.79: TRANSPORTERS REPORTS	185
FIGURE 4.80: PROCESSES REPORTS	186
FIGURE 4.81: BUFFER REPORTS	186
FIGURE 5.1: VERIFICATION PROCESS OF THE VR-SIMULATOR SOFTWARE	189
FIGURE 5.2: PROCESS PLAN SAMPLE OF THE JOB SHOP MODEL	193
FIGURE 5.3: DEFINING THE SHIFTS OF THE JOB SHOP MODEL	194
FIGURE 5.4: DEFINING THE PRODUCTS OF THE JOB SHOP MODEL	195
FIGURE 5.5: DEFINING THE LABOURERS OF THE JOB SHOP MODEL.....	195
FIGURE 5.6: DEFINING THE OVENS.....	196
FIGURE 5.7: DEFINING THE PROCESSES LIST OF THE JOB SHOP MODEL	197
FIGURE 5.8: DEFINING THE TRANSPORTERS OF THE JOB SHOP MODEL	198
FIGURE 5.9: DEFINING THE WAREHOUSE	198
FIGURE 5.10: DEFINING THE PROCESS FLOW OF THE JOB SHOP MODEL.....	199
FIGURE 5.11: DEFINING THE PROCESS FLOW OF THE JOB SHOP MODEL.....	199
FIGURE 5.12: CONSTRUCTING THE LAYOUT OF THE JOB SHOP MODEL	200
FIGURE 5.13: SPECIFYING TRANSPORTERS NAMES	201
FIGURE 5.14: SIMULATION MODEL OF THE SHOP FLOOR GENERATED AUTOMATICALLY	202
FIGURE 5.15: VIRTUAL MODEL OF THE JOB SHOP MODEL	203
FIGURE 5.16: GANTT CHART DISPLAYING THE RUNNING TIME OF THE PROCESSES VS TIME FOR THE JOB SHOP MODEL	205
FIGURE 5.17: UTILISATION OF OVEN2.....	206
FIGURE 5.18: UTILISATION OF OPERATOR_1.....	206
FIGURE 5.19: REPORT OF THE PROCESSES OF OVEN3	207
FIGURE 5.20: PRODUCT DESIGN	208
FIGURE 5.21: FLOW SHOP MODEL REPRESENTATION.....	208
FIGURE 5.22: MATERIAL HANDLING SYSTEMS OF THE FLOW SHOP MODEL	210
FIGURE 5.23: DEFINING THE MACHINE TOOLS OF THE FLOW SHOP MODEL.....	213
FIGURE 5.24: DEFINING THE MACHINES OF THE FLOW SHOP MODEL	214
FIGURE 5.25: DEFINING THE CONVEYORS OF THE FLOW SHOP MODEL.....	215

FIGURE 5.26: DEFINING THE AGVS OF THE FLOW SHOP MODEL	215
FIGURE 5.27: DEFINING THE PROCESSES OF THE FLOW SHOP MODEL.....	216
FIGURE 5.28: DEFINING THE BUFFERS OF THE FLOW SHOP MODEL.....	217
FIGURE 5.29: DEFINING THE PROCESS FLOW OF THE FLOW SHOP MODEL.....	217
FIGURE 5.30: DEFINING THE ORDERS OF THE FLOW SHOP MODEL	218
FIGURE 5.31: CONSTRUCTING THE LAYOUT OF THE FLOW SHOP MODEL.....	219
FIGURE 5.32: SIMULATION MODEL OF THE FLOW SHOP MODEL GENERATED AUTOMATICALLY	220
FIGURE 5.33: VIRTUAL MODEL OF THE FLOW SHOP GENERATED AUTOMATICALLY	221
FIGURE 5.34: GANTT CHART DISPLAYING THE RUNNING TIME OF THE MACHINES VS TIME FOR THE FLOW SHOP MODEL	223
FIGURE 5.35: UTILISATION OF THE SAW MACHINE	223
FIGURE 5.36: REPORTS ABOUT THE SAW MACHINE.....	224
FIGURE 5.37: REPORTS ABOUT THE SHEETS (PRODUCTS)	224

List of Tables

TABLE 2.1: TYPES OF VR SYSTEMS	37
TABLE 2.2: MANUFACTURING DESIGN APPLICATIONS	42
TABLE 2.3: VR APPLICATIONS IN OPERATIONS MANAGEMENT	43
TABLE 2.4: BENEFITS OF APPLYING VIRTUAL REALITY IN MANUFACTURING APPLICATIONS	46
TABLE 3.1: VOICE COMMANDS AND THEIR FUNCTIONS WITHIN THE FACTORY	91
TABLE 4.1: VOICE COMMANDS AND THEIR FUNCTIONS	180
TABLE 5.1: PROCESSES NAMES AND THEIR DESCRIPTIONS	192
TABLE 5.2: LABOURERS REQUIRED TO HANDLE EACH PROCESS.	192
TABLE 5.3: DESCRIPTION OF THE PROCESSES OF THE WORK SHOP	209
TABLE 5.4: MODELLING ASSUMPTIONS FOR THE MACHINES	211
TABLE 5.5: MODELLING ASSUMPTIONS FOR THE CONVEYORS	212
TABLE 5.6: MODELLING ASSUMPTIONS FOR THE AGVs	212
TABLE 5.7: MODELLING ASSUMPTIONS FOR THE BUFFERS	212

Glossary of Acronyms

Acronym	Definition
AGV	Automated Guided Vehicle
CAD	Computer-Aided Design/Drafting
GUI	Graphic User Interface
HMD	Head Mounted Display
MHS	Material Handling Systems
MRT	Models Running Tool
OAT	Output Analysing Tools
PMT	Process Modelling Tool
SCL	Superscape Control Language
SMT	Simulation Modelling Tool
VB	Visual Basic
VE	Virtual Environment
VM	Virtual Manufacturing
VMT	Virtual Modelling Tool
VR	Virtual Reality
WCL	Witness Command Language
WIP	Work In Progress (Process)
2D	2 Dimensional Presentation
3D	3 Dimensional Presentation

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

In today's global market, manufacturing industries are learning that constant improvement is a prerequisite for continued existence. Global markets for goods and services and the worldwide availability of technology and capital are the main characteristics of today's industrial environment. These markets are highly competitive. Product innovation, quality and cost are the major elements of this competition. To remain competitive industries must implement state-of-the-art technology to support the decision-making process. Industries may rely on these technologies to obtain benefits such as the improvement of process data management, better control over operations and improvement in both the planning and decision-making functions [1].

In general, numerous modelling methodologies have been developed to support industries in gaining a better understanding of their system behaviours and processes that will enable them to make the kind of system designs and operational decisions to ensure market competitiveness. Examples of such methodologies are Process Modelling, Simulation Modelling, and Virtual Reality technology.

The *Process Modelling Methodologies* have been developed to support the description of functionalities and operational behaviour (dynamics) of manufacturing systems. These methods identify both operational activities, and all relevant information such as product, orders, process plans and resources. These methodologies provide models that could assist in developing a better understanding of a company's processes.

The *Simulation Modelling Methodologies* have been developed to address manufacturing system design and operation problems. Simulation Modelling is a technique that evaluates a given set of decisions by providing the user with performance measures.

The *Virtual Reality Methodologies* have been used to support the simulation tools to give a better understanding of the results and the dynamic behaviour of the models of manufacturing systems. It provides the simulation practitioners with a graphical presentation and animation of the manufacturing system to discuss and communicate the outcome of the simulation models, where the user can interact with the virtual model using VR devices.

The aim of this project is to develop new software that integrates process modelling, simulation modelling and virtual reality technology to address manufacturing systems.

1.2 PROBLEM DEFINITION

Simulation provides an alternative to the conventional techniques of designing and modelling manufacturing systems. Simulation has proven to be an excellent strategic tool for high level planning as well as a day-to-day tactical tool to study, model, analyse, design and improve manufacturing systems. Simulation modelling and Virtual Reality (VR) have been widely used as successful tools to address manufacturing systems activities such as product design and modelling, process planning, new product testing, shop floor controls, training, maintenance, and inspection.

Simulation modelling has been applied successfully for investigating current and anticipated planning strategies used to configure/reconfigure manufacturing systems in order to improve system performance. Such popularity of simulation has resulted in a large number of simulation packages being available in the software market. However, those packages have many limitations to fulfill the user requirements. Some of the limitations of simulation package are listed below:

- Modelling and programming skills are required.
- Time consuming to develop simulation models.
- No real time interaction.
- Virtual models of many simulation packages do not support virtual reality devices e.g. Head Mounted Display (HMD).

The main aim of this research is to develop new software that overcomes some of the above mentioned limitations of the presently available simulation packages.

1.3 OBJECTIVES OF THE RESEARCH

The main objective of this research is to develop a software that supports the design and analysis of manufacturing systems by allowing personnel without sufficient skills and knowledge of simulation to automatically develop a robust simulation model and create a virtual model of a manufacturing system to address its behaviour. The software should provide the following features:

1. To be easy to use by non-technical personnel.
2. To provide a Process Modelling (PM) tool.
3. To allow non-expert users to develop simulation models of manufacturing systems, and generate a virtual environment of these models dynamically.
4. To provide framework to simulate the activities of the simulation model in a 3D presentation within the virtual model in real time.
5. To allow the user to interact with the virtual model and the simulation model of a manufacturing system in real time using input devices (e.g. keyboard, joystick, mouse, voice commands).
6. To support Virtual Reality devices (e.g. Head Mounted Display).

1.4 STRUCTURE OF THESIS

This thesis is divided into six chapters. A description of each chapter is described below:

The first chapter presents a brief introduction on the technologies that have been used to carry out this research; subsequently the problem statement and the objectives of the research have been defined.

Chapter 2 “Literature Review”: This chapter is divided into three main sections:

- The first section provides an overview on manufacturing system design and classifications.
- The second section presents the technologies used in this research with an up-to-date review of the use of these technologies in manufacturing applications.
- The final section of this chapter focuses on the current trends of integrating simulation with VR technology.

Chapter 3 “Integration of Simulation with VR”: This chapter is devoted to the development and integration of a simulation model with a virtual model of a real manufacturing system.

Chapter 4 “Design and Development of the VR-Simulator Software”: This chapter describes the software development phases, which are presented in three sections:

- The first section briefly introduces the overall architecture of the developed software.
- The second section reviews the software tools that have been selected to carry out this research.
- The last section of this chapter provides the reader with an overall view of how the software was designed. It explores the different stages involved in designing and developing the software.

Chapter 5 “VR-Software Testing and Evaluations”: This chapter is divided into three sections:

- Software Testing: This section presents the various tests conducted on the developed software after the design phase to ensure that the software functioned properly and generated acceptable results.
- Software Evaluation: This section highlights the limitations and advantages of the developed software by using the software to model a real case study of a Job Shop manufacturing system, and a hypothetical case study of a Flow Shop manufacturing system.

Chapter 6 “Conclusions and Future Work”: This chapter presents the main results achieved during this research. It also outlines the recommendations for the future work based on the present study.

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

As outlined in Chapter one, the objective of this research is to develop new software that integrates three methodologies to address manufacturing systems design, planning and control. These methodologies include Process Modelling, Simulation, and Virtual Reality. This chapter provides the reader with a clear explanation about what is involved in this process, and also presents the relevant work that has been done by applying these methodologies to manufacturing applications. The chapter is divided into three main sections as follows:

- **Process Modelling**

This section defines Process Modelling and describes the objectives and the applications of Process Modelling in general. It also explains how Process Modelling could support simulation studies since it is relevant to the aim of this research. Furthermore a number of existing Process Modelling methodologies and their applications are presented. Finally, a review of some of the existing Process Modelling tools is presented in this section.

- **Simulation**

This section provides some background information on simulation modelling, its objectives and applications within manufacturing. Furthermore, a number of existing and widely used simulation packages are briefly reviewed.

▪ **Virtual Reality**

This section presents an overview of virtual reality technology, its components and types of virtual reality systems, as well as its different applications in manufacturing.

The last section of this chapter presents some of the work that has been done by integrating simulation and virtual reality methodologies in manufacturing applications.

2.2 MANUFACTURING SYSTEMS DESIGN AND CLASSIFICATION

Manufacturing is the process of making a finished product from raw material using different types of machines. Almost everything in the home, at the workplace, or in moving between the two is the result of manufacturing. Examples include home appliances, computers and furniture, automobiles, air-planes and ships. The most common classification of manufacturing systems can be categorised into three types [2]:

1. Job Shop production.
2. Batch (Flow) production.
3. Mass production.

This classification is normally associated with discrete-product manufacture, but it can also serve for plants used in the process industries. The three types of production are related to the production volume as shown in Figure 2.1.

▪ **Job Shops:**

The distinguishing feature of Job Shop production is low volume. The manufacturing lot sizes are small. Job Shop production is commonly used to meet specific customer orders, and there is a great variety in the type of work that a plant must do. Therefore, the production equipment must be flexible and general purpose to allow this variety of work. Also, the skill level of Job Shop workers must be relatively high so that they can perform a range of different work assignments. Examples of products

manufactured in the Job Shop include space vehicles, aircraft, machine tools and equipment, and prototypes of future products.

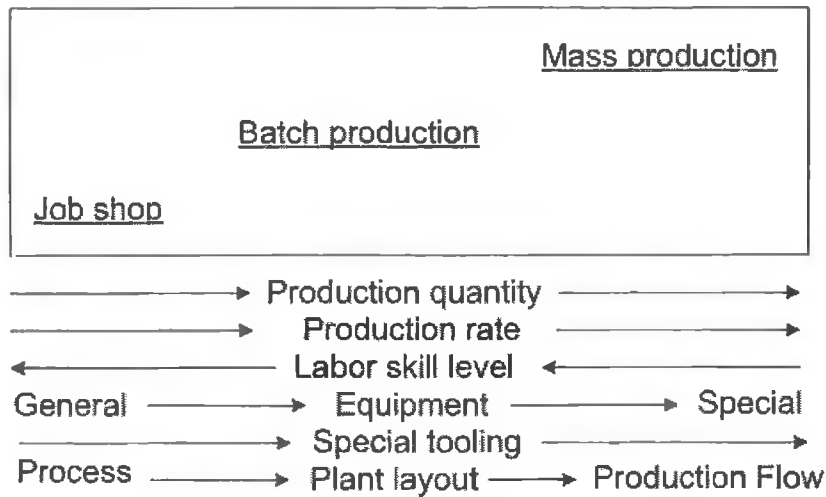


Figure 2.1: Types of Production [2]

▪ **Batch (Flow) production:**

This type of production involves the manufacture of medium-sized lots of the same item or product. The lots may be produced only once, or they may be produced at regular intervals. The purpose of Flow production is often to satisfy continuous demand for an item. However, the plant is usually capable of a production rate that exceeds the demand rate. The manufacturing equipment used in batch production is general-purpose but designed for higher rates of production. Examples of items made in batch-type shops include industrial equipment, furniture, textbooks, and component parts for many assembled consumer products [2].

▪ **Mass production:**

This is the continuous specialised manufacture of identical products. Mass production is characterised by very high production rates, equipment that is completely dedicated to the manufacture of a particular product, and very high demand rates for the product [2].

2.2.1 Functions in Manufacturing

Manufacturing systems can be divided into five interrelated functions, which are [3]:

1. Product design.
2. Process planning.
3. Production operations.
4. Material flow/facility layout.
5. Production planning/control.

- **Product design:** is responsible for taking the inputs from marketing regarding customer demands and constructing the description of products that can be profitably manufactured to satisfy these demands.
- **Process planning:** entails the specification of the sequence of operation required for converting the raw material into parts and parts into products.
- **Production operations:** are generally of either a fabrication or assembly nature.
 - **Fabrication** refers to either the removal of material from the raw stock or a change in its form for the purpose of obtaining a more useful form.
 - **Assembly** refers to the combination of separate parts of raw stock to produce a more valuable combined unit.
- **Material handling & factory layout:** is concerned with the technique used to transport parts, tolling, and scrap throughout the facility. Facility layout is concerned with the physical placing of production processes (machines) within the facility [3].
- **Production planning, scheduling and control:** constitutes an important component of the total manufacturing system. Production planning is responsible for combining information on market demands, production capacities, and current inventory levels to determine planned production levels by product family for the medium to long term.

Manufacturing systems are interrelated sets of: Tasks, materials, resources, products, plans and events [3].

- **Plans:** include process plans (routings) and production plans.
- **Resources:** include worker, handling systems, and machines.
- **Events:** are points in time when resources start and complete tasks.
- **Materials:** are routed through resources, thereby being transformed into products.
- **Status information:** on resources and material/products indicates the state of the manufacturing system at any point in time.

The following sections discuss the three methodologies (Process Modelling, Simulation and Virtual Reality) to address manufacturing systems.

2.3 PROCESS MODELLING

It is widely recognised that shop-floor activities in manufacturing enterprises of any size are growing ever more complex due to emerging factors such as [4]:

- Increased off-line activities that coexist with regular production (e.g. Prototyping, special orders for non-standard products, maintenance of production lines, etc).
- Increased variability of products, components and process phases.
- Increased process complexity, due to factors such as compliance with quality, security standards and environmental directives.
- Advances in process technologies, requiring the introduction of new processes. Therefore resource availability is a major criterion to avoid any losses in production.
- Increased autonomy and responsibility of operators.

These factors call on manufacturing enterprises to implement new methodologies to gain a better understanding of their shop-floor activities. A complete and correct understanding of the processes on the shop floor could extend the scope of operations management from production to the full spectrum of the manufacturing systems

activities. This extended scope would help to improve the reliability of decision-making by considering the interactions between processes of different kinds.

Organising complex systems and collecting data relating to operations and processes from such systems can be very difficult and time-consuming which may be overcome by the use of systematic methods and tools as suggested by the National Research Council (NRC) in the USA [5]:

“... tools for describing process as are critical for the design of individual products, the design and operation of factories, and the development of modelling and simulation technology. Formal descriptions are necessary if processes are to be presented in sufficient detail and with enough specificity to be adequately complete and unambiguous, such formalisms would allow designers to describe factory processes (involving both machines and people), design activities, decision processes, among others.”

In general Process Modelling methodologies have been developed to collect and evaluate knowledge on processes in production, material flow, business, production development, logistics and production procedures. They are used to gain an understanding of the static and dynamic behaviours of systems. The main objectives of Process Modelling are to [6]:

- *Facilitate human understanding and communication.* This requires a group to be able to share a common representational format.
- *Automated execution support.* This requires a computational base for controlling behaviour within an automated environment. Examples of automated environments include online monitoring, scheduling and simulation applications. To control and support these applications a PM tool is required to integrate with these applications, thereby providing and retrieving information to/from these applications. In this way the PM tool acts as an online tool that could be used to manage manufacturing operations.

- *Support process improvement.* This requires a base for defining and analysing processes.
- *Automatic process guidance.* To provide automated guidance, a PM tool should provide a framework or a mechanism that allows one to collect process related information, and using this framework one could add more data or complexity to the process.

The “40-20-40” rule as suggested by Harrell and Tumay [7] states that in developing a simulation model, 40% of an analyst’s time should be spent on data collection and requirements gathering, 20% on model translation (i.e. coding and programming), and finally 40% on model validation and verification. Nethe and Stahlmann [8] suggest that the development of a high-level Process Model of an actual production system prior to the development of the simulation model could greatly help in the collection of relevant information on the operations of the system (i.e. data collection) and therefore reduce the effort and time consumed to develop a simulation model. Therefore Process Modelling could be used as a knowledge acquisition method for simulation studies.

2.3.1 Modelling Approaches

Efficient modelling is the key to ensure the success of any simulation project, and it is one of the critical tasks in simulation model building. The goal here is to develop structures for applying computer technology (e.g. simulation) to manufacturing systems and to use computer-based methods to understand how best to improve manufacturing productivity and follow the production flow (items and information) in the system. As problems become larger, the preparatory phase of analysing the problem becomes larger as well. Manufacturing systems need special requirements in modelling due to [9]:

- Complexity.
- Large amount of data.
- Rapid changes in product configurations due to demand.
- Manual intervention in the system.

These factors make the systems analysis phase more difficult but they reinforce the need for a tool to enable systems modelling. GIGO – garbage in, garbage out – not only applies to data, it applies to the logic of the software and the implementation of the system [9]. A number of modelling methods have been developed to model manufacturing systems. Some tools such as flow charts and block diagrams concern physical process flow, while others such as IDEF, and data flow diagrams relate to process information flow.

2.3.1.1 Flow Diagram

Flow diagrams are one of the most common graphical methods to represent various activities and relationships within manufacturing systems. Although the use of flow diagrams is a simple method used in modelling, it is still widely used in most applications. Flow diagrams are limited in representing complex manufacturing systems because they do not provide useful information.

2.3.1.2 IDEF

One of the most effective tools in modelling complex industrial systems is the Integrated Computer Aided Manufacturing (ICAM) DEFinition (or IDEF). It has been presented relatively quickly through the United States Air Force's (USAF) ICAM programme, which started in 1977 [10]. The USAF was using contractors in the United States and Europe and required a common means of specifying systems and communicating results across the programme. This led to the development of IDEF [11]. The ICAM program developed three well-documented modelling methodologies around the IDEF approach to system study:

- A functional model of a manufacturing system and environment called IDEF0.
- An information model of the system and environment called IDEF1.
- A dynamics model to describe time-varying system behaviour called IDEF2.

The aim of the IDEF was to provide an integrated suite of tools for the purpose of modelling activities within an organisation. The IDEF method was developed to support better understanding and analysing of systems. This method involves functional, informational and dynamic modelling methods. This modelling approach helps people involved in improving manufacturing productivity to understand different aspects of a system such as:

- The activities and their relationships within a system.
- The informational requirements of a system.
- The behaviour of functions and information interacting over time.

The IDEF suite of methods consists of the following:

- **IDEF0 (Function Modelling Methods)**: used for the structured representation of the function, activities, or processes within a system. [12]
- **IDEF1 (Information Modelling Method)**: used for representation of the structure and semantics of information within a system. [12]
- **IDEF1X (Data modelling Methods)**: used for semantic data modelling. It is a data modelling technique that is used by many branches of the United States Federal Government. [12]
- **IDEF3 (Process Flow and Object State Description Capture Method)**: used for collecting and documenting processes to show how a system, process or organisation works. [13]
- **IDEF4 (Object-Oriented Design Method)**: used to assist in the correct application of object-Oriented technology. [14]
- **IDEF5 (Ontology Description Capture Method)**: Designed as a method for fact collection and knowledge acquisition. [15]

A number of researchers have shown that methods from the IDEF approach could be used to support simulation. For instance Jeong [16] used both IDEF0 and IDEF3 to develop an optimised simulation-based scheduling system (OSBSS), while Perera and Liyanage [17] used IDEF0 and IDEF1X to address the rapid collection of input

information for the simulation of manufacturing systems. Also other researchers such as Rensburg & Zwemstra [18] and Al-Ahmari & Ridgway [19] have demonstrated the use of IDEF0, IDEF1X and IDEF3 to support simulation for manufacturing and system design. Furthermore it has been suggested that the use of IDEF techniques in simulation modelling enhanced the quality of simulation models and helped to reduce the time needed to generate simulation models [18]. IDEF0 was originally used to apply structured methods to better understand how to improve manufacturing productivity but it has a number of disadvantages in terms of learning time involved, cumbersomeness, ambiguity of function specification and, perhaps most significantly, its static nature. Such a hierarchy of functions does not explicitly represent the conditions or sequences of processing. Consequently, a great deal of manual effort and interpretation may be required to identify the appropriate functions which should process a specific input and to verify their consistency [20].

2.3.2 Commercial Process Modelling Tools

There are many commercial packages available that can be used to facilitate process modelling and reengineering. What follows is a brief review of a number of packages used in this field:

- **ARIS TOOLSET:** Provides tools for remodelling, analysis and navigation of business processes. The ARIS Toolset and its add-on components enable the enterprise-wide and global definition and design of business processes as well as their analysis and optimisation. In the era of e-business, with this tool decisions can be made quickly about the management of e-business processes. The ARIS Toolset provides realistic simulations of resource utilisation, activity-based cost calculations (e.g. for make-or-buy decisions), as well as web-based communication of modelled and optimised company processes [21].
- **Workflow Modeller:** A process-modelling tool based on the IDEF0 standard process modelling method [22]. WorkFlow Modeller is the authoring tool in Meta Software's

suite of tools for performance improvement. It provides a robust, consistent method for building models of business processes, and, through its integrated link with the MetaSoft Works simulation application. Meta Software's WorkFlow Modeler helps business/management analysts and IT professionals capture the intricacies of business processes and present them in a consistent graphical form for key stakeholders in the organisation to examine and understand. WorkFlow Modeler includes facilities for capturing dynamic behavior properties, such as work volume, staffing, durations, and routing logic in addition to the basic process design [23].

- **OPTIMA**: An integrated tool for creating process maps, modelling behaviour and also performing simulation [24]. OPTIMA is a Process improvement software from AdvanEdge Technologie. It is an easy-to-use Windows application that features process modeling, simulation and reporting capabilities. The program, designed for the front end of reengineering projects, helps users quickly create and edit presentation-quality process maps. OPTIMA is one of the few products that directly supports a process-mapping notation. Although users can create any type of flowchart, the product is specifically designed to automate the creation of the Rummler-Brache deployment flowcharts, with the aim of making this kind of process easy to create [25].

- **ProSim**: This process modelling tool is based on the IDEF3 standard process modelling method. [26]. ProSim is KBSI's simulation design tool based on the IDEF3 process description capture method, introduced previously. In this method the focus is on the abstraction and capture of knowledge about a given real-world system. The tool, in a similar fashion to the capture method, focuses primarily on what fundamentally occurs in a system, the dynamic patterns among elements that repeatedly occur, as opposed to what happens at particular time instances in a system. [27].

- **PMAPPER**: PMAPPER is a tool used for process modelling and analysis. Using this tool one can develop process maps. This involves the description of the process elements and their relationships. PMAPPER is used to model a process to various degrees of decomposition, detail and formality [28].

There are a considerable number of process modelling packages available in the market these days, however they all have some pitfalls including:

- The cost of the software licenses.
- Complex interface in most packages.
- Training is required due to complexity of coding or model building.
- Compatibility with other simulation and/or VR packages is not fully available.

The main goal of the proposed software package is to provide a low-cost tool for process modelling. It has to consider the pitfalls in the commercial packages by developing a user friendly interface for non-expert users. The database, simulation model and the virtual reality model should not require the user to have technical programming/coding knowledge. The challenge of getting the software ready to use is to establish comprehensive routines that allow the user to enter the required data about the manufacturing process(es) in spreadsheets as well as through interface screens using popup menus with easy options to select from. Chapter 4 provides more details about the development of the Process Modelling Tool of the VR-Simulator software.

2.4 SIMULATION

Manufacturing systems, processes, and data are growing ever more complex. Product design, manufacturing engineering, and production management decisions often involve the consideration of many interdependent variables, probably too many for the human mind to cope with at one time. These decisions often have a long-term impact on the success or failure of the manufacturing organisation. It is extremely risky to make these major decisions based on “gut instinct” alone. Simulation provides a capability to rapidly conduct experiments to predict and evaluate the results of alternative manufacturing decisions. It has often been said that you do not really understand your industrial processes and systems until you try to simulate them [29].

Simulation is a modelling technique that is gaining popularity in tackling problems faced by manufacturing and service industries. A number of researchers including Shannon, Pegden, and Sadowski [30] define simulation as the process of designing a model of a real system and conducting experiments with this model for the purpose of understanding the behaviour of the system and/or to evaluate various strategies for the operation of the system. This technique can then assist the user in decision-making, thereby reducing the time and costs associated with experiments on a real system. This also minimises risks associated with experiments on real systems.

Simulation can be used as an evaluation tool, providing for the collection and evaluation of quantitative system performance measures, such as resource utilisation and cycle time. It can also be used as an analysis tool in experiments examining alternative designs and operating strategies on system performance. However simulation as an experimental tool does not solve a problem or optimise a design. It supports evaluating a solution and providing understanding of problematic areas rather than generating a solution. An optimum solution can only be obtained through experimentation by running and comparing the results of alternative solutions [31].

Simulation involves the generation of an artificial history of the system, and the observation of that artificial history to draw inferences concerning the operating characteristics of the real system that is represented. Simulation is indispensable in describing and analysing the behaviours and problems of real world systems, ask “what if” questions about the real system, and aid in the design of such systems. A vast body of research efforts has reviewed simulation in manufacturing (e.g. Banks [32], Banks *et al.* [33], Law *et al.* [34], Pritsker [35], Shannon [36], and Kochhar [37]).

2.4.1 Applications of simulation in manufacturing

In a global economy, successful manufacturers are constantly changing the way they do business in order to stay competitive. Companies are asking such questions as [38]:

- When should the next piece of equipment be purchased?
- How many people will be needed next month to fill orders?
- How will the new plant operate five years from now?
- How can work-in-process inventory and cycle time be reduced while increasing throughput?

Simulation provides a method for finding answers to these and other questions about the behaviours of a manufacturing system. Savolainen et al. [39] indicate that simulation models are really formal descriptions of real systems that are built for two main reasons:

1. To understand conditions as they exist in the system today.
2. To achieve a better system design through performing what-if analysis.

There have been many trends in manufacturing methods with names such as Just-In-Time (JIT), flexible work cells, and kanban. Types of manufacturing systems as defined by Harrell and Tummy [40] include, but are not limited to:

1. Job Shop.
2. Cellular manufacturing.
3. Flexible manufacturing systems.
4. Batch Flow Shop.
5. Line flow systems (production and assembly lines, transfer lines).

Simulation is a powerful, versatile, and flexible tool that is being used in different ways through a variety of manufacturing and service industries, including the following fields [41]:

- System design.
- System management.
- Training and education.
- Communication and sales.
- Public relations.

Harrel & Tumay [41] have classified the main use of simulation in manufacturing into two main categories, which include system design and system management.

2.3.1.1 System Design

When designing or re-designing a system, a simulation model can be developed to enable experiments on the model rather than the actual system. Performing experiments on a simulation model presents a number of advantages including [31]:

- Greater flexibility to make changes.
- Reduced costs vs. experimenting with actual systems.
- No interruptions as associated with experimentation on actual systems.

Simulation models allow validation of a system design before implementation. Verification is supported by making as many changes as possible in both the concept and design stages of systems development. Simulation can model the detailed operations of a system, e.g. flow of parts through machines or workstations and resource allocations. Therefore, it provides an overall understanding of the system dynamics and hence assists the development of an optimised system design over the long term [31].

2.4.1.2 System Management

System management is about managing the operations of a system and involves controlling the flow of resources and materials in the system. Simulation models can be used to develop experiments with alternative courses of action to provide management with data that can assist in more informed decision making for scheduling and utilisation of the system resources. In system management, simulation assists in making the following decisions [41]:

- Production/customer scheduling.
- Resource scheduling.

- Maintenance scheduling.
- Work prioritising.
- Flow management.
- Delay/Inventory management.
- Quality management.

In general simulation modelling can be used to support production planning and control. However, frequent or continuous use of a simulation model requires automated data collection of the input tasks (i.e., schedule inputs, part routings, and shop-floor status). Arisha [42] has classified the use of simulation in manufacturing applications into three main groups as shown in Figure 2.2.

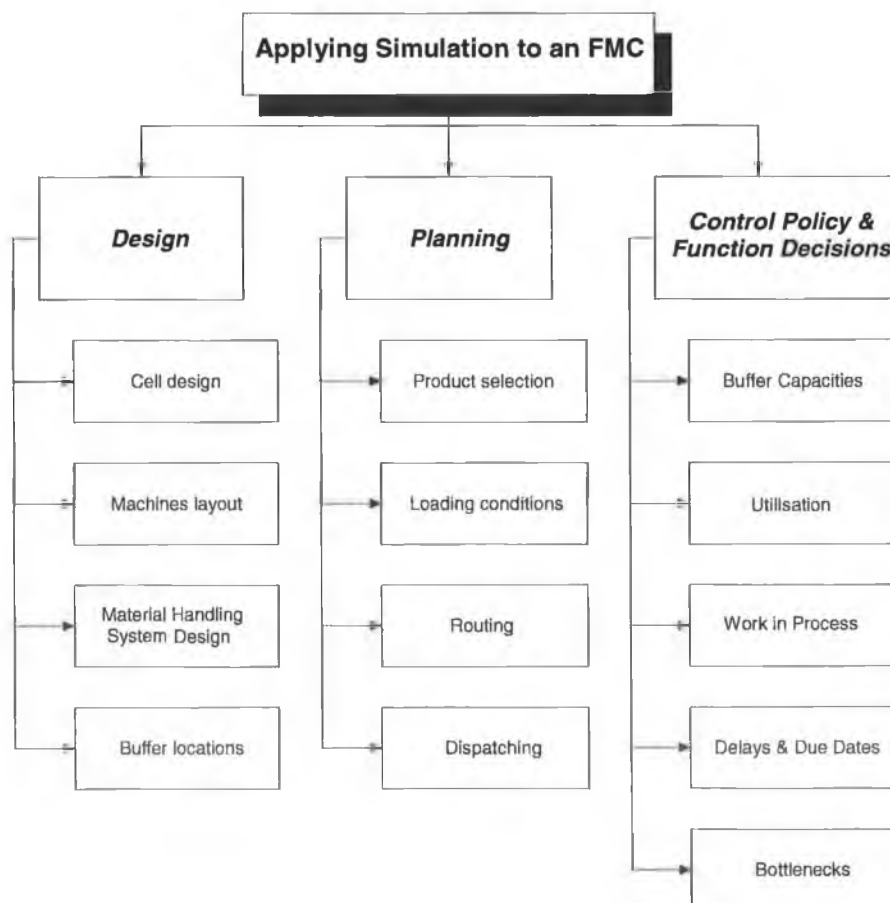


Figure 2.2: Main Applications of Simulation in Manufacturing [42]

2.4.2 Simulation Tools

The growing use of simulation for the analysis of manufacturing systems has resulted in a rise in the number of both general purpose and application oriented simulation software packages. Choosing amongst the vast array of available packages has the potential to overwhelm newcomers to the field. In fact, a survey of hundreds of corporate software development projects indicates that more than 60% of software projects are considered unsuccessful due to wrong software selection decisions and implementation [43]. The simulation software selection decision is often costly and time consuming (careful selection can take as long as a year [44]). However, it is essential that an appropriate simulation package is selected as it can have a significant impact on the ultimate validity of the model and on the timeliness with which the simulation project is completed [45]. Arisha [46] has developed a comprehensive guide that helps the vendor to select simulation software for manufacturing systems. Banks [47] reviewed 53 simulation software packages. Among these packages, 42 packages were suggested to be useful for manufacturing purposes. In general simulation software packages available in the market can be classified as Simulation Languages and Simulators.

2.4.2.1 Simulation Languages

Simulation languages are general-purpose software, but may have some specific features for certain applications. Models are developed by writing programs using the language-modelling construct. Simulation languages have the advantage of being very flexible permitting the user to model any system regardless of its complexity. However one of the possible disadvantages is the need for programming expertise combined with extensive coding and debugging, as illustrated by Law & Kelton [48].

Simulation-oriented languages form the oldest set of tools dedicated to assist the development of computer simulated models. Simulation languages have been around since the 60's. The first languages have changed in many ways over the years, and in

some cases were fully integrated with modeling and simulation development environments. Among these first languages to emerge were Geoffrey Gordon's GPSS and Simscript. GPSS has two general-purpose commercial versions, Wolverine's GPSS/H [49] and Minuteman's GPSS/PC [50]. Both of them offer MS Windows based integrated development environments (IDEs). Like GPSS, Simscript II.5 [51] is a general purpose discrete simulation language with graphic capabilities. It was created by Rand Corp. and is currently developed by CACI. It is available on UNIX systems, MS Windows, and IBM OS/2.

2.4.2.2 Simulation Software Packages

Simulation software packages allow users to model and analyse manufacturing and material handling systems. Their advantage is that they require less programming effort than simulator languages. But they have less modelling flexibility. Also, a simulator's standard features limit their modelling capabilities to specific systems. Examples of Simulators are WITNESS, ProModel, Flexsim ED, Arena and Extend.

The remainder of this section provides a review of several widely used manufacturing simulation packages.

2.4.2.2.1 WITNESS

WITNESS is a Windows application from the Lanner Group. The WITNESS simulation package was developed specifically to model material flow in manufacturing systems. It is a hybrid of a simulator and a general simulation language. The interface provides the user with a set of menu options, which are used to develop a generic model of a manufacturing system. The key features of the WITNESS approach are [52]:

- Powerful building block design.
- Powerful range of logic and control options.
- Comprehensive statistical input and reports.

- Openness: Can be linked to other software such as CAD, BPR and Spreadsheets easily.
- Optional fully integrated 3D views.
- Optional fully integrated Model Optimisation.

The WITNESS simulation package is an 'OLE Automation Server'. Therefore, it can be controlled by an 'OLE Automation controller' mechanism, such as Visual Basic. Through this mechanism it is possible to control, edit or construct an entire model.

Once the standard model is created within Witness, a "fast build" facility allows the user to automatically view the model in a virtual world using standard images from the WITNESS VR library. Alternatively, the user can create a more accurate representation of his environment e.g. show the details of a piece of equipment specific to a company's requirement.

2.4.2.2.2 ARENA

ARENA [53] is a simulator produced by System Modelling Corporation. Arena is based on the simulation language SIMAN, and is a user interface to develop SIMAN code. The basic building block of the SIMAN language is Blocks and Elements. Blocks and Elements templates contain all the flexibility of the SIMAN language. Simulation models are developed by placing blocks in the model window and providing data for these blocks and also identifying flows of entities through blocks. It combines the modelling power and flexibility of the SIMAN simulation language, with the benefits of a windows application environment resulting in ease of use.

Arena offers Application Solution templates. Templates are groups of modules that have been designed to capture the entities, processes and modelling construct of specific applications. The use of templates allows the user to build up a library of commonly used modules. Application Solution templates are being developed in different fields such as: Business Process Reengineering, call centres and high-speed manufacturing.

Arena operates on both the Microsoft Windows XP and the Windows NT operating systems. It is fully compatible with other Windows software including the Microsoft Office Suite. This provides the user with the ability to integrate the data collected from other sources such as EXCEL and ACCESS into the simulation model through the use of ActiveX. It also integrates well with Visual Basics. Arena has a tool that provides the user with the ability to create and view 3D animations of Arena models. The user can use Arena 3DPlayer to create a realistic 3D animation layout to produce a simulation history for a particular simulation run. In addition the animation of Arena simulation objects, the user will be able to enhance the layout of a system with a variety of static images to add to the realism of the animation setting.

2.4.2.2.3 ProModel

ProModel [54] is a simulation package produced by ProModel Corporation. This simulation package has programming capabilities similar to WITNESS. ProModel uses Script as its simulation programming language. This simulation package integrates with other applications through OLE, providing flexibility to customise an interface either on the front or back end of ProModel. One can create an interface by using Visual Basic for Applications, or C/C++. Moreover, ProModel includes the Run Time Interface (RTI) and SimRunner packages, which are used for the design of experiments and optimisation of the software respectively.

2.4.2.2.4 SIMUL8

This Simulation Package has been developed by SIMUL8 Corporation [55]. Some of the features of this simulation package are as follows:

- The current version of SIMUL8 has implemented an internal language called “visual logic”. This language enables the user to express:
 - More sophisticated product routings.
 - Conditional work sequencing.

- Resource management control.
 - Conveyor.
 - Transport operating rules.
-
- SIMUL8 integrates with Microsoft products and interfaces directly to Excel and Visual Basic.
 - SIMUL8 is used to simulate various applications including production processes, assembly operations, high volume transaction processes, distribution systems and business process re-engineering applications.
 - Once the model is built in the 2D environment, SIMUL8 provides the user with a tool to view the model in a 3D presentation automatically. The user can use images from the SIMUL8's 3D library or 3D images imported from other graphic software.

2.4.2.2.5 EXTEND

EXTEND [56] is a dynamic, iconic simulation environment developed by Imagine That, Inc. Extend permits simulation of discrete event, continuous, and combined discrete event/continuous processes and systems. It provides the modeller with a library and built-in tools to extend the library. Furthermore, it allows the user to construct components and build customised user interfaces. It is based on the C programming language and allows the modeller to add customised functionalities to the package. Extend also supports integration with Microsoft products.

Models are built by dragging and dropping blocks onto the model worksheet. The necessary input data for blocks can be entered either directly into the block dialogue or obtained from files. Animation is automatically a part of every Extend model. A default animation is displayed when "Show Animation" is turned on. Animation features can be added to a model in the form of different animation pictures that represent different types of items, displaying values, levels, colour changes, or even sounds in response to

simulation events. In addition, custom animation can be added to display pictures and text, level indicators, and pixel maps [57].

2.4.2.2.6 Flexsim

Flexsim Software Products [58] has been in the simulation software and consulting business since 1993. Taking ten years of experience with simulation and using the latest advances in software technology. Flexsim is the only Object-Oriented simulation product in the world to incorporate a C++ IDE and compiler into a graphical modeling environment. C++ can be used directly in defining model logic and compiled right in the Flexsim application. Flexsim is a software modelling application that can be used to simulate and visualise any business process whether the process is manufacturing, logistics or administration. With Flexsim, all the simulation models are built and run natively in 3D to present the activities of the simulated process.

2.4.2.3 Limitations of the Current Available Simulation Packages

The previous section reviewed the most popular simulation packages available on the market today that can be used to model and analyse manufacturing systems. While these packages have been proven to be powerful tools, several shortcomings have been identified based on the literature review conducted:

- Expensive in time and money.
- Require high levels of expertise in fields such as: probability, statistics, design of experiments, modelling and computer programming.
- Require skills to use and maintain simulation models.
- Do not allow real time interactions with the models in real time.
- Do not support Virtual Reality devices (e.g. HMD).
- Virtual environments created using these packages are 3D images of the resources and not virtual objects.

To highlight some of the above points further, Shannon [59] suggests that to use simulation correctly and intelligently requires 720 hours of instructions and 1,440 hours of application experience to gain the basic tools. These difficulties have resulted in a slow up-take of simulation by industries in general, and especially for small and medium sized enterprises.

2.4.2.4 Critical Evaluation of Simulation Application in Manufacturing

Simulation modelling is a highly flexible technique. It is used to address a number of engineering problems, at the design, planning, and operation levels. In these areas, it provides information necessary for decision-making. Some of its benefits are listed below [60]:

- Obtain a better understanding of the system by developing a model of the system of interest and observing the system's operation in detail over long periods of time.
- Experiment with hypotheses for the feasibility of the system.
- Reduce time to investigate a long event, and extend time to observe a complex system in detail.
- Test the effects of changes of system inputs without disturbing the real operation.
- Identify and solve problems of certain phenomena.

Law and Kelton [61] and Banks et al. [62] discuss numerous benefits for simulation, perhaps the most important of which is that after the model is validated and verified, proposed changes can be investigated in the simulation model rather than in the real system. Despite the benefits mentioned above, simulation also has some disadvantages that should be kept in mind, which include [31]:

- Model building requires special training.
- Simulation results may be difficult to interpret.
- Simulation modeling and analysis can be time consuming and expensive.

2.4.2.5 Future of Simulation in Manufacturing

Ultimately manufacturing simulation will have a major impact on the way products are manufactured. Due to the high costs of acquisition, integration, maintenance, limited interoperability, functionality, and performance - simulation technology is not for everyone yet. The following areas need to be improved to encourage and enhance the use of simulation in manufacturing applications [31]:

- Ease-of-Use.
- Reusability.
- Scalability.
- Interoperability.
- Productivity.
- Promotion/Support of Collaborative Modeling.
- Supporting Component-Based Modeling.
- Integrating Simulation Software with.
 - Database Tools.
 - Supply Chain Software.
 - Browsers and other very widely used tools.

2.5 VIRTUAL REALITY

Virtual Reality (VR) is not a new concept. The origins of VR can be traced as far back as at least “The Ultimate Display” [63], a seminal paper published by Ivan Sutherland in 1965 introduced the key concepts of immersion in a simulated world, and of complete sensory input and output. The following challenge was set: “The screen is a window through which one sees a virtual world. The challenge is to make that world look real, act real, sound real, and feel real [63].”

Virtual Reality allows a user to step through the computer screen into a 3-Dimensional (3D) world. The user can look at, move around, and interact with these worlds as if they

were real. The primary concept behind VR is that of illusion. It focuses on the manifestation of the fantasy world of the mind in computer graphics. High technology is used to convince users that they are in another reality, experiencing some event that doesn't physically exist in the world in front of them. It is also a new media for information and knowledge acquisition, and representing concepts of ideas in ways not previously possible [64]. Virtual reality has been explored for several years. One of its initial pioneers was Jaron Lanier who developed many of its basic concepts [65]. Due to the decreasing cost and increasing power of computers, virtual reality is being implemented by many industries such as aviation [66], medicine [67] and manufacturing [68] to address key areas like training, design, and testing.

Visualisation has become a critical component of simulation technology in manufacturing applications. It provides the simulation practitioners with an environment to discuss and get a better understanding of the simulation model's behavior. Graphical presentation and animation can be a significant tool to communicate the outcome of simulation models for the non-technical audience. Therefore, a Virtual Reality could be a very significant tool to support simulation activities in design, planning, validation and verification and to achieve credibility for simulation models in manufacturing applications.

2.5.1 Components of Virtual Reality

A VR system consists of two main components [69]:

- Hardware.
- Software.

Both of these components play an important role in the successful implementation of a VR set-up and in the degree of realism achieved. A brief description of these two components is presented in the following subsections:

2.5.1.1 Hardware

The hardware in a VR system consists of three main components, which are: *Main Processor*, *Input Devices*, and *Output Devices*. The main processor is also called the Reality Engine because it produces the sensations of reality. The user can interact with a virtual world created by the computer using various types of input devices such as gloves (for gestures), voice commands and traditional keyboard input. The virtual world in turn responds to the users actions by using appropriate output devices such as, visual display, sound response, and tactile feedback system. The following diagram describes the three sub-components of the hardware that comprise the VR system (Figure 2.3).

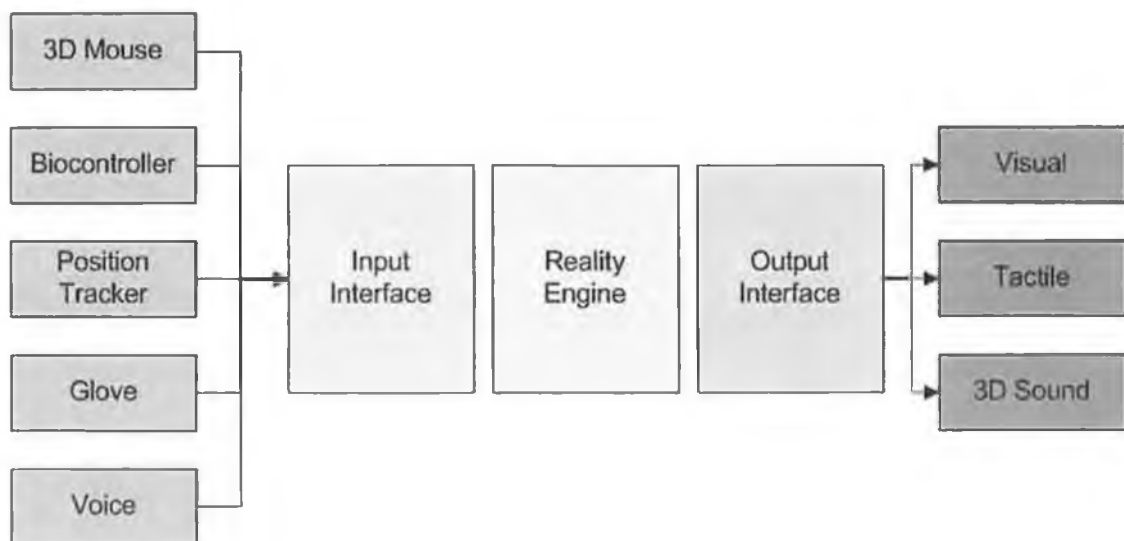


Figure 2.3: Main Components of a VR Hardware System

▪ Input devices

The principal objective of a Virtual Environment (VE) is to allow a more realistic interaction with a graphical image. Input devices, therefore, play an important role in allowing the user to interact with the virtual world. Traditional 2D devices like the mouse can still be used (for picking type operations) in a Virtual Environment (VE).

However it is the new generation of 3D devices, which provide the real tools to reach out into the 3D virtual world. Input devices that can be used in a virtual environment can be divided into the following categories.

- **Position Trackers:** These devices are used in tracking position and orientation. Figure 2.4 shows the tracking of the user's head position, and updating the virtual world scene based on this position and orientation criteria. Another use of the tracker is to track the user's hand position in space so that interaction with objects in the 3D world is possible. Tracking sensors based on mechanical, ultrasonic, magnetic and optical systems are available.
- **Digitizers:** These devices have been adapted from the mouse/trackball technology to provide a more advanced form of data input. Included in this category is the six Degrees Of Freedom (6 DOF) mouse and force ball. The 6 DOF mouse functions like a normal mouse on the desktop with additional freedom. Force ball uses mechanical strain developed due to the forces and torque the user applies in each of the possible three directions.
- **Gloves:** These consists of a wire cloth glove that is worn over the hand like a normal glove (see Figure 2.4). Fiber-optical, electrical or resistive sensors are used to measure the position of the joints of the fingers. The user can, therefore, use gestures to communicate with the VE. The Glove is typically used along with a tracking device that measures the position and orientation of the glove in 3D space.
- **Biocontrollers:** This process controls indirect activity such as, muscle movements and electrical signals produced as a consequence of muscle movement. As an example, a dermal electrode placed near the eye to detect muscle activity could be used to navigate through the virtual worlds by simple eye movements.

- **Voice Input:** Voice input provides a more convenient way for the user to interact with the VE by freeing his/her bodily actions in the VE. Such a facility is very useful in a VR environment because it does not require any additional hardware such as the gloves or biocontrollers to be physically attached to the user (see Figure 2.4).

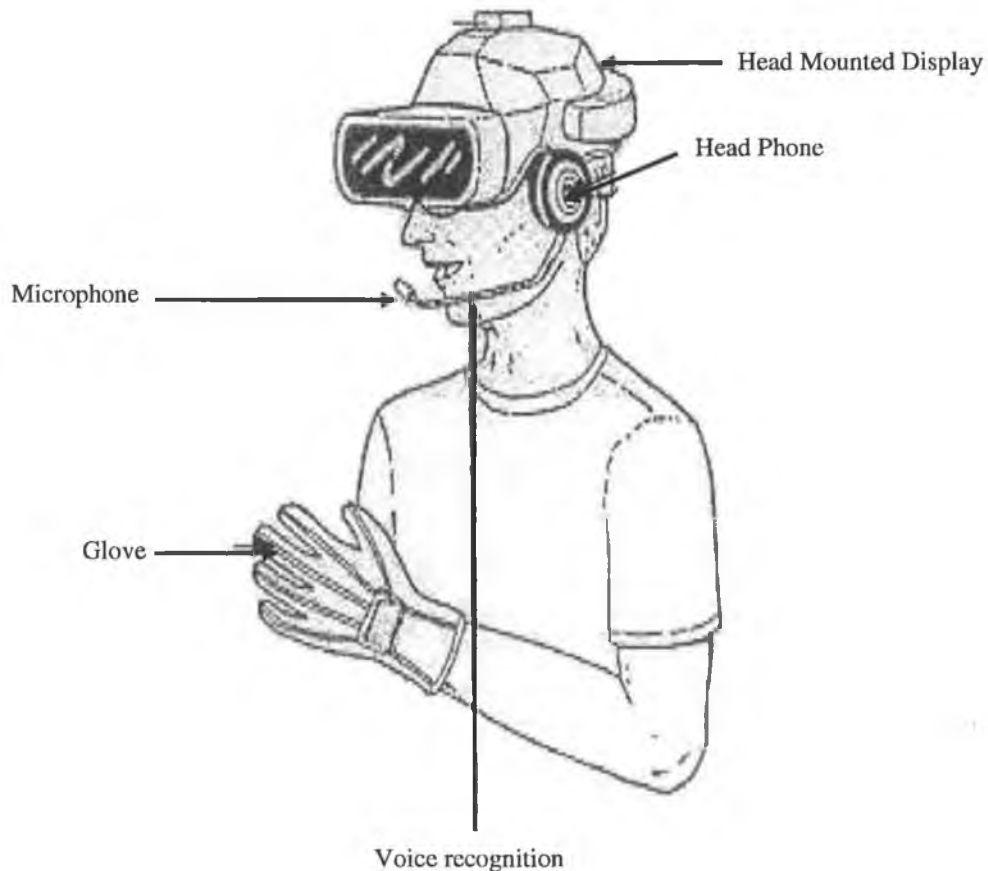


Figure 2.4: The Integration of the Various Elements of Generic VR System [70]

- **Reality engine**

This is the main processor that performs the task of creating the virtual environment and handling the interactions with the user. The engine works as an interface between the input devices and the output devices. It provides the computing power to run various

aspects of a virtual world simulation. The first task of the reality engine is usually the display of the virtual world (may be in Stereo mode) and it utilises a majority of the CPU time. This display processor is often the bottleneck in the performance of a VR system. Also the degree of realism achieved depends on the speed with which images are updated when the user interacts with the virtual world. A measure of the speed of a reality engine is the number of shaded polygons it can render per second. The second task of the reality engine is to interface with different input and output devices and to provide feedback to the user while minimizing lag-time as much as possible. Depending on the architecture of the VR system this secondary task can be delegated to other processors thereby allowing the main processor to handle the visual rendering alone.

▪ **Output devices**

Output devices are used to provide the user feedback about his/her actions in the VE. The way in which the user can perceive the virtual world is limited to the five primary senses, sight, sound, touch, smell and taste. Only the first three have been incorporated in commercial output devices.

- **Visual:** Two types of devices are available for visual feedback. The first is the head mounted display. This uses two Liquid Crystal Display (LCD) screens to show independent views (one for each eye). The human brain puts these two images together to create 3D view of the virtual world. The second and much cheaper way is to use a stereo image display monitor and LCD shutter glasses. In this system two images (as seen by each eye) of the virtual scene are shown alternately at a very high rate on the monitor. An infrared transmitter co-ordinates this display rate to the frequency with which each of the glasses is blacked out. The user perceives 3D images.
- **Sound:** After sight, sound is the next most important sensory channel for virtual experiences. It has the advantage of being a channel of communication that can be processed in parallel with visual information. The most apparent

use is to provide auditory feedback to the user about his actions in the virtual world. However, 3D sound, in which the different sounds would appear to come from separate locations, can be used to provide a more realistic VR experience.

- **Haptic:** This type of feedback could either be touch or force. For the sense of touch feedback, various systems including using electrical signals (on the finger strips) are being currently used to simulate different type of textures. Another approach is to provide resistance as the user tries to manipulate objects in the virtual world.

2.5.1.2 Software

Most VR software systems consist of a simulation manager and a virtual database [69]. The simulation manager executes the main event loop of the VR program and maintains the virtual world database (see Figure 2.5). The main event loop consists of a cyclic process that the input devices queried, the user actions are processed and finally some feedback is provided to the user using the output devices.

In addition the simulation manager maintains the current state of the virtual world in a virtual world database. This database contains information about all the objects in the virtual world and their attributes. Examples of attributes include color, texture and physical properties of the objects in the virtual world and their attributes.

The state of the virtual world can be modified as a result of the user issuing a command or interacting with the objects in the virtual environment in some manner. The event loops consist of:

- Input processor.
- Application software.
- Output processor.

▪ **Input Processor**

The input processor is the software that obtains data from the devices used in the VR system. It provides the simulation manager with the current position and orientation of all input devices with a minimal lag time. The simulation manager can then act upon this information and pass it to the application software so that appropriate action can be taken.

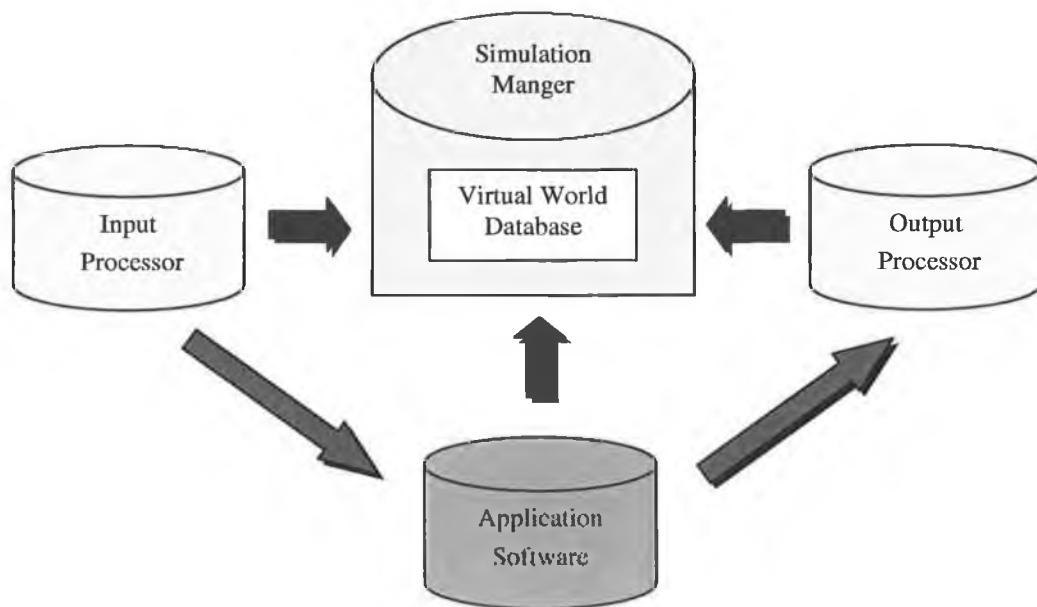


Figure 2.5: VR system software

▪ **Application Software**

This is the software that runs within the VE to perform a particular task. For example, in a virtual CAD system, the solid modeller will continuously check the geometry being created by the user for interactions. There are a number of Virtual Reality packages available on the market today as well be discussed in the coming sections.

▪ **Output Processor**

The output processor displays the results of the user’s interaction with the virtual world primarily through a visual display. Depending on the other hardware available on the system, other feedback mechanisms like sound and force feedback can also be used.

2.5.2 Types of VR Systems

Although it is difficult to categorise all VR systems, most configurations fall into three main categories [71] and each category can be ranked by the sense of immersion, or degree of presence it provides. These categories include non-immersive (Desktop) systems, semi-immersive projection systems and fully immersive systems as shown in Table 2.1.

Table 2.1: Types of VR systems

VR System	Non-Immersive	Semi-Immersive	Fully-Immersive
Input Devices	Mice, keyboards, joysticks and trackballs.	Joystick, space balls and data gloves.	Gloves and voice commands.
Output Devices	Standard high resolution monitor	Large screen monitor, large screen projector system, and multiple television projection systems	Head Mounted Display (HMD), CAVE
Resolution	High	High	Low – Medium
Sense of Immersion	Non – Low	Medium – High	High
Interaction	Low	Medium	High
Price	Lowest cost VR system	Expensive	Very expensive

2.5.2.1 Non-Immersive VR Systems

Non-immersive systems, as the name suggests, are the least immersive implementation of VR techniques. Using the desktop system, the virtual environment is viewed through a portal or window by utilizing a standard high-resolution monitor. Interaction with the virtual environment can occur by conventional means such as keyboards, mice and

trackballs or may be enhanced by using 3D interaction devices such as a Space Balls or Data Gloves.

2.5.2.2 Semi-Immersive VR Systems

Semi-immersive systems are a relatively new implementation of VR technology and borrow considerably from technologies developed in the flight simulation field. A semi-immersive system will comprise of a relatively high performance graphics computing system. These systems increase the feeling of immersion or presence experienced by the user. This may have a considerable benefit in educational applications as it allows simultaneous experience of the VE, which is not available with head-mounted immersive systems.

2.4.2.3 Fully Immersive VR Systems

Fully Immersive VR is a system in which the user is surrounded by a virtual environment. The user can hear, visualise and interact with the artificial environment. In order for a person to participate in a fully immersive VR system, a considerable amount of hardware is necessary. Immersive VR systems can be defined as: "A computer system used to create an artificial world in which the user has the impression of being in that world and with the ability to navigate through the world and manipulate objects in it [72]." The most direct experience of virtual environments is provided by fully immersive VR systems. These systems are probably the most widely known VR implementation where the user either wears an HMD or uses some form of head-coupled display such as a Binocular Omni-Orientation Monitor (BOOM). [73] CAVEs are the most expensive VR systems that can provide the user with a full impression of being within the virtual world. The following are descriptions of the HMD and the CAVE systems.

- *Head Mounted Displays (HMDs):* An HMD uses small monitors placed in front of each eye, which can provide stereo, bi-ocular or monocular images. Stereo images are

provided in a similar way to shutter glasses, in that a slightly different image is presented to each eye. The major difference is that the two screens are placed very close (50-70mm) to the eye, although the image, which the wearer focuses on, will be much further away because of the HMD optical system. Bi-ocular images can be provided by displaying identical images on each screen and monocular images by using only one display screen.

- **CAVE:** A quite different approach to 3D output is the CAVE Automatic Virtual environment. The CAVE is a room size, projection-based VR system that projects the scene onto three walls and a floor. The graphics are rear projected onto the walls and there is an overhead projector pointing into a mirror that reflects the scene onto the floor. The projected scene is animated and controlled using an SGI Onyx Infinite Reality (IR) workstation. A viewer moving inside the CAVE wears a stereo graphics' crystal eyes liquid crystal shutter glass and a 6-DOF tracking device that calculates the new stereoscopic projection each time the user moves inside the CAVE. The stereographic system provides liquid crystal viewing lenses mounted in an eyeglass frame, whose polarities are switched on command. These commands provide the system with infrared signals that are synchronised with the rendering update rate of the image generation system. These infrared emitters are mounted at various locations, which can be rearranged if needed to ensure that the receiver on the eyeglass frame receives the command signals regardless of their location and orientation. The user can also use a hand wand to help him interact with the virtual environment.

2.5.3 Virtual Reality Tools

The fact that virtual reality software is intrinsically difficult to design and implement emphasizes the importance of user interface tools, such as toolkits, frameworks, user interface management systems, or graphical user interface builders [74]. Current systems supporting virtual reality software construction are subdivided into two categories: toolkits and authoring systems. Toolkits are programming libraries that provide a set of functions supporting the creation of a virtual reality application.

Authoring systems are complete programs with graphical interfaces for creating worlds without resorting to detailed programming. These usually include some sort of scripting language in which to describe complex actions (e.g., VRML, which is becoming a de-facto standard for describing virtual worlds). While simpler to use, current authoring systems do not offer all the functionalities of toolkits.

A typical VR toolkit provides supports for high-speed rendering (mostly through the use of some form of level-of-detail modeling), low-level interfacing with a variety of input devices (at the minimum supporting high frequency sensor reading for input and a variety of graphics display formats for output). Tools of this kind range from low-cost solution (e.g., Rend386, Superscape VRT, OpenGL Optimizer) to high-end professional packages (e.g., Division dVS and dVise, Sense8 World Tool Kit WTK). In addition to generic support systems, a variety of software tools exist to solve specific problems. Examples are domain-specific toolkits for supporting distributed application [75,76], libraries for implementing high-speed collision detection [77], and tools for supporting physical simulation with haptic feedback [78, 79].

Moreover, software packages have been developed for virtual applications in manufacturing (e.g. DELMIA). The DELMIA package [80] provides authoring applications that can be used to develop and create virtual manufacturing environment to address process planning, cost estimation, factory layout, ergonomics, robotics, machining, inspection, factory simulation and production management.

2.5.4 Virtual Reality in Manufacturing

Manufacturing industries are the most important contributors to prosperity in industrialised countries. However, it is becoming increasingly difficult to meet customers' demands and to compete. The advances in virtual reality technology in the last decade have provided the impetus for applying VR to different engineering applications such as product design, modelling, shop floor controls, process simulation, manufacturing planning, training, testing and verification. VR holds great potential in

manufacturing applications to solve problems before being employed in practical manufacturing thereby preventing costly mistakes. Virtual reality not only provides an environment for visualisation in the three-dimensional environment but also to interact with the objects to improve decision making from both qualitative and quantitative perspectives [81]. This section discusses the use of virtual reality in manufacturing applications, which include design, prototyping, machining, assembly, inspection, planning, training and simulation. Virtual Reality applications in manufacturing have been classified into three groups; design, operations management, and manufacturing processes. A brief description of every group and its relevant subgroups is provided in the coming subsections.

2.5.4.1 Design

Virtual Reality may play a very significant role in designing a new product. VR technology has been applied into two different applications in design; design and prototyping as shown in Table 2.2. The benefits of applying VR in design are shown in Table 2.4.

VR provides a virtual environment for the designers in the conceptual design stage of designing a new product. Once designers are satisfied with their design, then the design could be detailed to make the necessary modifications. In the product development process, prototyping is an essential step. Prototypes represent important features of a product, which are to be investigated, evaluated, and improved. Virtual prototyping could be used before building the physical prototype to prove design alternatives, to do engineering analysis, manufacturing planning, support management decisions, and to get feedback on a new product from prospective customers. The virtual environment for prototyping should include [82]:

- *Functionality*: the virtual prototype should be clearly defined and realistically simulated to address product functionality and dynamic behaviour.

- *Human Interaction*: the human functions involved must be realistically simulated, or the human must be included in the simulation.
- *Environment*: an off-line computer simulation of the functions can be carried out, or a combination of computer offline and real time simulation can be carried out.

Table 2.2: Manufacturing Design Applications

Application	Definition	Example
Product Design	Virtual design is the use of VR technology to provide the designer with a virtual environment to evaluate the design, compare alternate designs, effectively interact with the product model and conduct ergonomic studies using full human body tracking.	A virtual workshop for mechanical design was developed at Massachusetts Institute of Technology. The goal of the project was to develop a simulated workshop for designers to do conceptual design work while having to take into account manufacturing processes. The simulated workshop consists of a band saw, a drill press, a milling machine, a radial arm saw and a table saw. [83]
Prototyping	Virtual prototyping means the process of using virtual prototypes instead of or in combination with physical prototypes, for innovating, testing and evaluating specific characteristics of a candidate design.	University of Illinois, Chicago, and Purdue University have designed and implemented a prototype of a virtual reality based computer aided design system. The focus of this work is to allow a simplified method of designing complex mechanical parts through the use of virtual reality techniques [84].

2.5.4.2 Operations Management

Operations management has been classified into three categories; planning, simulation and training. Table 2.3 shows the applications of VR technology on the operations management categories. The benefits of applying VR technology to these categories are shown in Table 2.4. Due to the necessity for smarter factory planning; Virtual Reality is a useful method to improve understanding of the plans and to support interdisciplinary discussions. Figure 2.6 shows a fully immersive VR environment, which has been used as a tool for future factory design. This environment has been developed to provide a

visual, three-dimensional space in which to explore the effects of various product mixes, inspection schedules, and worker experience on productivity [85].

Table 2.3: VR Applications in Operations Management

Area	VR applications
Planning	<ul style="list-style-type: none"> ▪ VR can lead to an optimal planning of a manufacturing system by giving a visual environment to all persons involved in the planning process to monitor the factors that lead to inadequate planning results and delay the start of product. ▪ Visual comparison of possible solutions based on human experiences and facts lead to a rapid start of production and robust manufacturing processes [88].
Simulation	<ul style="list-style-type: none"> ▪ VR convinces people who do not believe in the simulation tools or understand its capabilities. [89] ▪ VR helps to verify the model logic and real-world behaviour of the model [90]. ▪ VR is an important factor in the verification process as it provides a visual trace of events as they happen. ▪ VR gives an opportunity to people who have not built the model to verify it. ▪ VR supports the simulation tools to understand the results and the dynamic behaviour of the model. ▪ VR provides virtual environment to the employees, managers and non-technical staff to communicate and understand the statistical outcome of a simulation.
Training	<ul style="list-style-type: none"> ▪ VR offers the best training by allowing each employee to have a full access to the entire facility. ▪ The virtual environment of the modelled system allows the employees to practice existing and new tasks in safe environment, and see how a product takes shape as it moves through the manufacturing system, thus providing a more effective training.

Virtual reality-based training is the world's most advanced method of teaching manufacturing skills and processes to employees. Using cutting-edge VR technology, training takes place in a realistic, simulated version of the actual facility, complete with the actions, sights, and sounds of the plant floor [86]. Some of the commercially available manufacturing simulation products provide some form of visualisation for depicting model output, (e.g. Witness 2003, Simul8, and Flexsim). Figure 2.7 shows a virtual environment created by Witness VR for a factory [87].



Figure 2.6: Fully Immersive VR Environment for Factory Design [85]

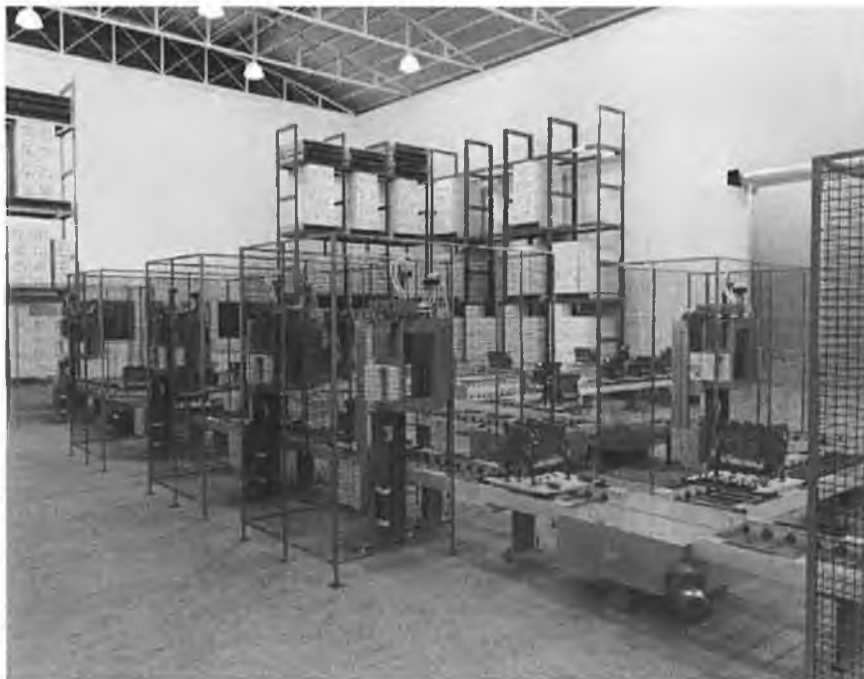


Figure 2.7: Virtual Factory Created by Witness VR [87]

2.5.4.3 Manufacturing Processes

Manufacturing processes has been classified into three different areas; machining, assembly, and inspection. Table 2.4 gives a brief description of the benefits of using VR in manufacturing processes.

- **Machining**

Virtual machining mainly deals with cutting processes such as turning, milling, drilling, and grinding, etc. Figure 2.8 shows an engineer using a Virtual Reality “semi-immersive environment” to simulate the use of a hexapod machine tool [91]. Researchers at the University of Bath have developed an interactive virtual shop floor containing a three-axis numerical control milling machine and a five-axis robot for painting. The user can mount a workpiece on the milling machine, choose a tool and perform direct machining operations, such as axial movements or predefined sequences. [92]



Figure 2.8: Virtual Machine Tool [91]

Table 2.4: Benefits of Applying Virtual Reality in Manufacturing Applications

Area		Benefits
Design	Design	<ul style="list-style-type: none"> ▪ To allow the whole design team to work together in the virtual environments. ▪ To improve visualisation of the product by allowing the user to co-exist in the same environment with the product model. ▪ To improved interaction with design in terms of more intuitive model manipulation and functional experimentation.
	Prototyping	<ul style="list-style-type: none"> ▪ To reduce significantly the amount of hardware prototypes during conception, design, and evaluation of new products. ▪ To provide a virtual environment for innovating, testing and evaluating of specific characteristics of a candidate design.
Operations Management	Planning	<ul style="list-style-type: none"> ▪ To improve the understanding of the plans and to support interdisciplinary discussions. ▪ To allow the users to interact and change the model during runtime. ▪ To enable unskilled users to understand and participate in the planning process. ▪ To support the technological as well as the economical modelling of diverse production planning scenarios.
	Simulation	<ul style="list-style-type: none"> ▪ To convince the use the simulation tools. ▪ To verify and validate a simulation model. ▪ To enable the user to understand the results. ▪ To provide a virtual environment for communication of results. ▪ To achieve the credibility for the simulation. [96]
	Training	<ul style="list-style-type: none"> ▪ To create a virtual environment of an entire manufacturing process for the benefit of trainees. ▪ To provide users with an environment to explore the outcomes of their decisions without risks to themselves or equipment. ▪ To allow the employees to practice existing and new tasks in safe environment.
Manufacturing processes	Machining	<ul style="list-style-type: none"> ▪ To evaluate the feasibility of a part design and the selection of processing equipment. ▪ To allow the user to study the factors affecting the quality, machining time and costs based on modelling and simulation
	Assembly	<ul style="list-style-type: none"> ▪ To reduce design cycle time, re-design efforts, and design prototypes. [93] ▪ To predict the quality of an assembly, product cycle and costs. ▪ To address assembly and disassembly verification.
	Inspection	<ul style="list-style-type: none"> ▪ To model and simulate the inspection process, and the physical and mechanical properties of the inspection equipment. ▪ To provide an environment for studying the inspection methodologies, collision detection, inspection plan, factors affecting the accuracy of the inspection process, etc [95].

▪ **Assembly**

Virtual Assembly (VA) is a key component of virtual manufacturing and is defined as: “The use of computer tools to make or “assist with” assembly-related engineering decisions through analysis, predictive models, visualisation, and presentation of data without realisation of the product or support processes’. In assembly work, Virtual Manufacturing (VM) is mainly used to investigate the assembly processes, the mechanical and physical characteristics of equipment and tooling, interrelation among different parts and factors affecting the quality based on modelling and simulation [93]. Virtual Reality can be used for assembly/disassembly operations.

▪ **Inspection**

Virtual Inspection makes use of VM technology to model and simulate the inspection process, and the physical and mechanical properties of the inspection equipment. This aims at studying the inspection methodologies, collision detection [94], inspection plans, factors affecting the accuracy of the inspection process, etc. [95]

2.5.5 Future of Virtual Reality in Manufacturing

Virtual Reality is simply the next stage in the user interface. The communication aspects of realistic, three dimensional graphical representations of the simulation model are what we would have liked to have at the beginning if we only had the combination of hardware power and graphics tools. However, the use of immersion and interaction with the simulation world is still in its infancy. In the future, we expect to see a design team, immersed entirely in the virtual world, building and testing its manufacturing processes. In this world, problems and improvement possibilities will be identified by a mixture of observation, analysis and human communication. Solutions will be tried interactively, using a mixture of tools including behavioral logic that will be constructed with voice input. [96]

2.6 INTEGRATION OF SIMULATION WITH VR IN MANUFACTURING

Simulations and Virtual Modelling tools have often been used separately to support design and study of manufacturing systems [98]. Dynamic 3D visualisation of construction processes modelled using discrete-event simulation can be of significant help in the verification, validation, and accreditation of simulation analyses [99]. The integration of these technologies is becoming more popular to address manufacturing applications due to the interest of the vendors of simulation software packages (e.g. Witness, Arena, Flexsim, etc.) to introduce visualization tools incorporated with their products.

Since researchers at the University of Maryland first introduced the concept of Virtual Manufacturing [100] in 1995, much research has been done and numerous applications have been developed in this area. For example, Researchers at the University of Illinois at Chicago have applied Virtual Reality techniques to construct a virtual gear factory and used it to analyse complex processes in a manufacturing environment [101].

Tools to assist production planning, visualisation and simulation are gradually finding acceptance by manufacturing industries. Some auto companies (e.g. BMW, General Motors, and Ford) have utilised visualisation and simulation software to verify a factory layout design and to plan processes [96,97]. The Boeing Company greatly reduced the cost, time, rework and labor hours of the T-45 stabiliser production using some 3D simulation tools [104]. Many other industrial companies and academic institutes also have their own successful experiences of applying the virtual manufacturing technique to assembly process design [105], sheet metal forming process scheduling [106], electronics assembly system design [107] and training [108].

George Chryssolouris, et. al. [109] has discussed the use of a virtual experimentation environment as a planning and training tool for machining processes. This approach involves the virtual performance of machining processes within a Virtual Machine Shop environment. The features of this environment enable the user to set up a process,

operate a machine tool, edit and execute an NC part program in an immersive and interactive way. Additional functions are provided in order to support process verification in terms of geometrical, technical and economic characteristics. A test case is carried out in order to demonstrate the simulation functions of the Virtual Machine Shop and the capability of the system to deal with machining, planning and training problems.

Schaefer, et. al. [110] has developed a software called Mod!Fact. It provides specific objects for factory planning and designing to animate and simulate manufacturing systems in VR. Anthony P. Waller [87] has developed a simulation model for Ford Motor Company integrated with a virtual environment of their company to visualise a facility more accurately overcoming obstacles for understanding their activities and discussion. Mujber, et. al. [111] has presented a new hybrid modelling approach for generating a simulation model and virtual environment from a process plan of an existing manufacturing system. The proposed approach integrates simulation and virtual reality in order to render a dynamic shop floor model automatically using the basic process configurations. This is accomplished using an input database. The model interface is user-friendly so that the new users of simulation and administration can use the model easily once the input database requirement is fulfilled. Moreover, the model allows the user to trace the performance criteria of any process on the shop floor level at any instant as well as to experiment any production scenario. The virtual environment is used to verify through visualisation the simulation model and also to provide a better understanding of the activities of the shop floor.

Vineet R. Kamat and Julio C. Martinez [112] have presented VITASCOPE. a general-purpose, user-extensible 3D animation system for visualising simulated processes in smooth, continuous, 3D virtual worlds. It is not tied to any particular simulation language or application nor does it depend on any specific CAD modeling tool. VITASCOPE's open and loosely coupled architecture makes it possible to be used in animating processes modeled in any general or special-purpose simulation tool capable of writing formatted text output during a simulation run. It provides an API to the

visualisation engine that others can use to seamlessly extend the animation language. These unmatched features make VITASCOPE a perfect tool for animating processes in any domain during all phases (verification, validation, and communication) of a simulation study.

2.7 CONCLUSIONS

This chapter has discussed the three technologies (Simulation, Process Modelling, and Virtual Reality) that are used nowadays to study manufacturing system. A review of the literature of using these technologies in manufacturing application has been presented in this chapter. The outcome of the review is that these technologies are proven to be successful tools to address manufacturing systems design, planning and control. The literature review failed to identify a documented attempt to integrate the above mentioned tools for design and analysis of manufacturing systems. The limitations of the commercially available simulation packages in terms of their VR capabilities include:

- Requirement for modelling and programming skills.
- Absence of User-Friendly PMT.
- Absence of real time interaction capabilities.
- Lack of support for VR devices (e.g. Head Mounted Display).

Based on what has been reviewed, a new approach has been suggested to design and implement a software that integrates process modelling, simulation and virtual reality to allow non-expert personnel to design and analyse manufacturing systems and address their behaviours. Also, the new approach aims to overcome some of the limitations of commercial simulation packages.

CHAPTER 3

INTEGRATION OF SIMULATION WITH VR (PILOT MODEL)

3.1 INTRODUCTION

Simulation modelling and Virtual Reality (VR) have been widely used as successful tools to address manufacturing systems activities such as product design and modelling, process planning, new product testing, shop floor controls, training, maintenance and inspection. Based on the literature review that has been conducted and presented in Chapter two, there are a number of simulation software packages available on the market that can be used to design and model manufacturing systems. These simulation packages have some limitations, e.g. the 3D models that can be created or generated using these packages do not allow real time interaction within the virtual model. Moreover, the virtual models do not support Virtual Reality devices, e.g. Head Mounted Display or Gloves. To overcome these limitations, a new method is introduced by the author to integrate a simulation model developed using a commercial simulation package called Witness with virtual model designed and created using a commercial VR package named Superscape VRT.

The main aim of this chapter is to show the complexity involved in developing simulation models and virtual models of manufacturing systems by presenting a new approach of development and integration of a simulation model with a virtual model of a new factory which is looking to supply materials to the construction industry. The interest of this factory is to design, manufacture, and install hollowcore prestressed concrete slabs, which can be used to provide versatile and economic design solutions for a variety of applications - from buildings or parking ramps to other impressive and demanding structures. The factory is using Spancrete

technology, a complete production and material handling system, which results in more efficient manufacturing and a high quality hollowcore unit.

The chapter is divided into three main sections: the first section describes the development phases of the simulation model. The second section presents the steps that have been taken to design and implement the virtual model of the factory. The last section introduces a new method of integrating simulation model with the virtual model of the factory.

3.2 DEVELOPING THE SIMULATION MODEL OF THE FACTORY

Manufacturing systems, processes and data are growing ever more complex. Product design, manufacturing engineering and production management decisions often involve the consideration of many interdependent variables, probably too many for the human mind to cope with at one time. These decisions often have a long-term impact on the success or failure of the manufacturing organisation. It is extremely risky to make these major decisions based on “gut instinct” alone. Simulation provides a capability to rapidly conduct experiments to predict and evaluate the results of alternative manufacturing decisions. It has often been said that you do not really understand your industrial processes and systems until you try to simulate them [113]. Figure 3.1 shows the steps that have been followed to develop the simulation model of the factory. Similar figures and their interpretations can be found in other sources, such as Pegden et al. [114], Law and Kelton [115], Banks et al. [116]. The simulation modelling process of the factory has been achieved in five phases, which include: project definition phase, model building phase, model testing phase, experimental design phase, and completion phase. Description of these phases is provided in the remainder of this section.

3.2.1 Project Definition

Developing a simulation model that represents the characteristics of reality for a manufacturing system is a very difficult task to achieve, it is time consuming, and expensive. Therefore, it is very important to first define a problem, formulate an

objective and then build a model that is 100% designed to solve the problem. The definition phase is divided into three stages (setting the objectives, determining the level of details and data collection).

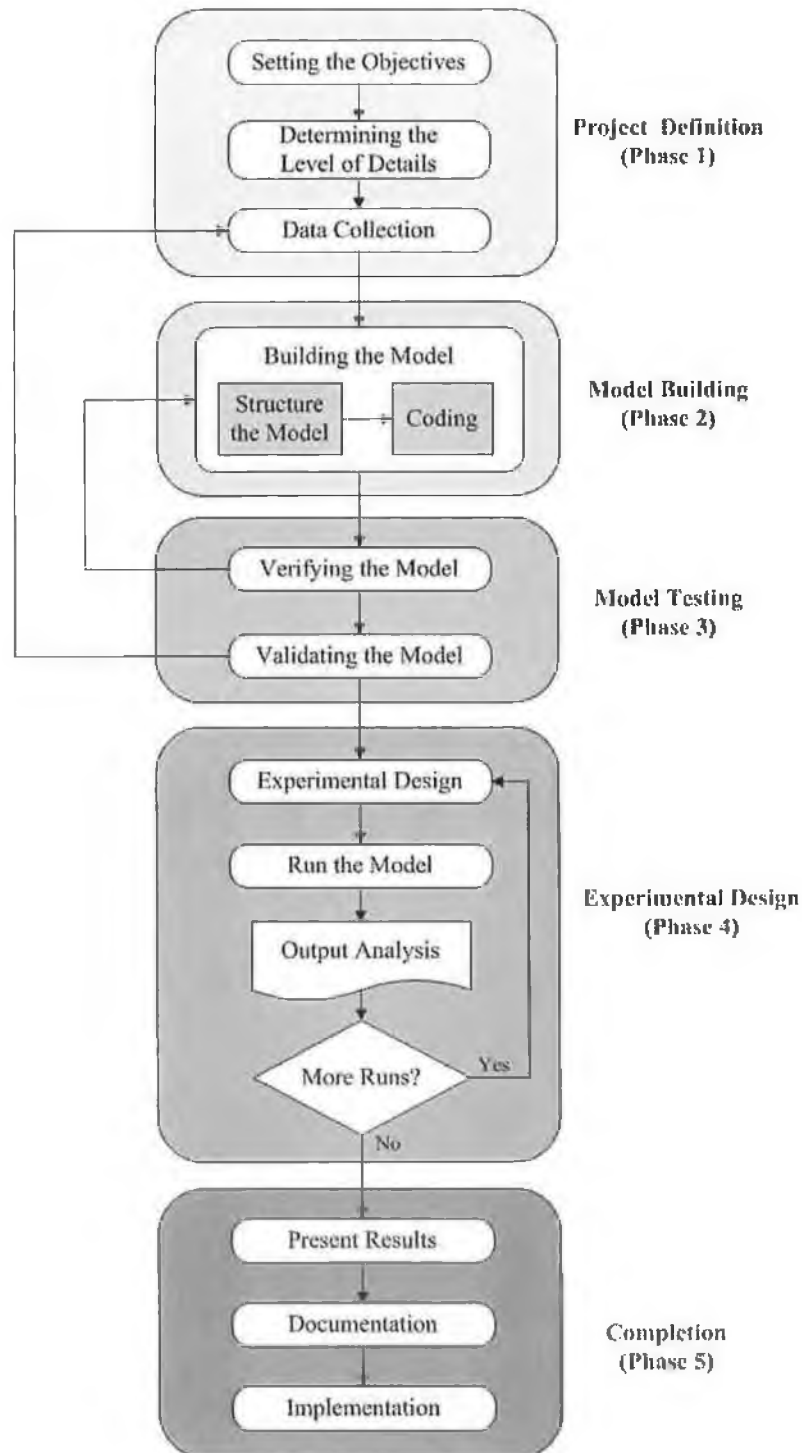


Figure 3.1: The Simulation Modelling Process of the Factory

3.2.1.1 Setting the Objectives

The first step of developing the simulation model of the factory was setting the objectives for developing the simulation model. These objectives have been set based on meetings and discussions with the managers of the factory. The main objective for the simulation model is to answer the following questions:

- What is the output capacity of each line (3 beds) in terms of m^2/day ?
- What is the current demand on the 3 beds in terms of m^3 or tonnes/day?
- How many labourers are required?
- How many AGVs are required?
- What is the plant throughput?
- What is the operators' utilisation?
- What is the utilisation of the machines?
- What is the best schedule that increases the throughput and reduces the cost?

3.2.1.2 Determining the Level of Details

The level of detail is determined once the objectives of the simulation model have been set. More consideration has been taken to determining the level of detail that should be included within the model as having more detail than required to meet the objectives will lead to increasing the time and cost of developing the simulation model.

3.2.1.3 Data Collection

Data collection is one of the most important steps in developing the simulation model. A model is able to represent the real system only if data is good enough. In general, data collection depends on the nature, objectives and the scope of the study. It also relies on availability of financial resources, time and manpower. Data collection is a crucial component of building a quality simulation model. Therefore, efforts have been exerted to accurately compile the required data to develop the simulation model of the factory as the simulation model means nothing if the collected data is incorrect.

The data for this model was collected by consulting with system experts who have practical experience in this respect. Before the data of the factory was collected, an AutoCAD drawing¹ of the factory was provided by the designer of the factory to assist in understanding and constructing the layout of the model of the factory.

Six Microsoft Excel data sheets (see Figure 3.2, through 3.7) have been created based on the objectives of the model to collect the data of the factory. The obtained information was used to construct and develop the simulation model. The sheets include the factory layout data sheet, Spancrete data sheet, two material handlings data sheet, labour data sheet and colour codes data sheet. A list of questions was prepared to be answered by the system experts, for later use to fill the Excel sheets. Here are some of the questions that were used to collect some data about the factory:

- How fast does a forklift travel km/hr (full and empty)?
- How long does it take for a batcher to fill the forklift?
- What is the capacity of the Spancrete machine in m³ to hold concrete?
- How long does it take to install strand on a bed?
- How long does it take to stress strand on a bed?
- How long does it take after concrete pour before slab is ready to be cut?
- What is the concrete batcher output/minute (m³/tonne)?

The factory layout data sheet (see Figure 3.2) with the AutoCAD drawing of the factory was used to design the layout of the factory. The data that was required to complete the factory layout data sheet included:

- Distance of the three doors from the concrete batching plant that feeds the forklift with concrete.
- Distance from the store to different location points in the factory in meters.
- Length of each bed in meters.

¹ The AutoCAD drawing could not be provided in this thesis due to the confidentiality of the developed model which represents the planning phase for a real factory.

	A	B	C	D	E	F
1	Distance of Doors from Batching Plant					
2	Door 1 (closest offices)	?	m			
3	Door 2 (middle)	?	m			
4	Door 3 (closest store)	?	m			
5						
6	Distance for materials removal to store					
7	Center of bed to factory end	?	m			
8	end of bed to door 3	?	m			
9	Door 3 to Store	?	m			
10						
11	Spancrete bed length	?	m			
12						

Figure 3.2: Factory Layout Data Sheet

As shown in Figure 3.3, the data that was required by the Spancrete data sheet includes the following:

- Speed of the Spancrete machine while extruding the concrete in m/min.
- Concrete extrusion rate from the Spancrete machine in m^3/m .
- Length of part of each product in meters.
- Production ratio of each product against the others.
- Number of labourers required to drive and handle the Spancrete machine while extruding the concrete.
- Speed of the Spancrete machine while returning to the initial location and how many labourers are required to drive the machine.
- Concrete capacity of the Spancrete machine in m^3 .
- Working hours of the Spancrete machine.
- Time required for cleaning the beds after the concrete is removed.
- Time required for cleaning the Spancrete machine after pouring the three beds, and how many labourers are required for this job.

There are some other machines in the factory that do different processes, namely: stranding machine, saw machine, crane machine, forklift and tractor. The stranding machine is used to strand the cables on the beds; the saw machine is used to cut the concrete into slabs as finished products; the crane machine is used to remove the slabs from the beds; the forklift is used to feed the Spancrete machine with the required concrete; and the tractor is used to carry the slabs to the store.

	A	B	C	D	E	F	G
1		Speed of machine extruding concrete (m/min)	Concrete Extrusion Rate (m3/min)	Length of Part (m)	Production Ratio	Labour	
2		?	?	?	100%	2	
3	Product A				0%	?	
4	Product B				0%	?	
5	Product C				0%	?	
6					TOTAL	100%	
7							
8					Labour		
9	Speed of machine returning	?	m/min	?			
10	Spancrete Concrete Capacity	?	m3				
11							
12	Shift Mon - Fri						
13	Start	?					
14	Finish	?					
15							
16							
17	Other Statistics						
18	Time to clean machine after pouring 3 beds	?	min	?			
19	Time to clean beds after concrete removal	?	min	?			

Figure 3.3: Spancrete Data Sheet

Two Excel data sheets named “Material handling1” and “Material handling2” were created to collect information regarding the above mentioned machines. The Material handling1 data sheet (Figure 3.4) was used to collect data regarding the saw machine, crane machine, stranding and tensioning machine. The following data was collected for these machines:

- Time required for cutting the dried concrete.
- Speed of the saw machine on the bed.
- Number of labourers required to operate the saw.
- Working hours of the saw machine.
- Time required for removing the cut parts by the crane.
- Number of labourers required to operate the crane.
- Speed of the crane machine on the bed.
- Working hours of the crane machine.
- Time required for stranding the cables on the bed.
- Speed of the Stranding machine on the bed.
- Number of labourers required to operate the Stranding machine.
- Time required for tensioning the cables on the bed.
- Number of labourers required for tensioning the cables.
- Working hours of the stranding and tensioning machine.

	A	B	C	D	E	F	G	H	I	J	K	L
1	Saw Machine			Labour								
2	Time per cut	?	min	?								
3	Speed on bed	?	m/min	?								
4												
5	Shift Men - Fri											
6	Start	?										
7	Finish	?										
8												
9												
10	Crane Machine			Labour								
11	Time to remove part	?	min	?								
12	Speed on bed	?	m/min	?								
13												
14	Shift Men - Fri											
15	Start	?										
16	Finish	?										
17												
18												
19	Stranding Machine			Labour								
20	Time to strand bed	?	min	?								
21	Speed on bed	?	m/min	?								
22												
23	Tensioning Machine			Labour								
24	Time to tension bed	?	min	?								
25	Speed on bed	?	m/min	?								
26												
27	Shift Men - Fri											
28	Start	?										
29	Finish	?										
30												

Figure 3.4: Material Handling1 Data Sheet

The Material handling2 data sheet (see Figure 3.5) is associated with the data of the forklift and the tractor. The following data sought in this sheet:

- Speed of the forklift with load of concrete in km/hr.
- Speed of the Spancrete forklift without load of concrete in km/hr.
- Concrete capacity of the Spancrete forklift in m³.
- Number of labourers required to operate the forklift.
- Working hours of the forklift.
- Time to fill a bucket and load it to the forklift.
- Time to unload the bucket from the forklift.
- Speed of the tractor loaded with slabs in km/hr.
- Speed of the tractor unloaded in km/hr.
- Capacity of the tractor in pieces.
- Number of labourers required to operate the tractor.
- Working hours of the tractor.
- Time to hook up a trailer in the factory.
- Time to unhook trailer in the store.

	A	B	C	D	E	F	G	H	I	J	K	
1	Spancrete Forklifts						Tractor (Cast Concrete removal)					
2	Speed with Load	?	km/hr	?	m/min		Speed with Load	?	km/hr	?	m/min	
3	Speed without Load	?	km/hr	?	m/min		Speed without Load	?	km/hr	?	m/min	
4												
5	Concrete capacity (for spancrete)	?	m ³				Concrete capacity (trailer)	?	piece per circuit			
6												
7	Labour Requirements						Labour Requirements					
8	Extruding	?	Op/truck				Extruding	?	Op/truck			
9												
10	Shift Mon - Fri						Shift Mon - Fri					
11	Start	?					Start	?				
12	Finish	?					Finish	?				
13												
14	Other Statistics						Other Statistics					
15	Time to Fill Bucket at Batcher	?	min				Time to hook up trailer in factory	?	min			
16	Time to unload at Spancrete	?	min				Time to unhook trailer in store	?	min			
17												

Figure 3.5: Material Handling2 Data Sheet

Figure 3.6 shows the Excel data sheet of the labourers with the following required data:

- Names of the labourers.
- Working hours of each labourer, including start time, finish time, and break times (e.g. lunch, coffee break).
- Labourers' skills were required to identify types of activities each labourer could handle.

	A	B	C	D	E	F	G	H	I	J	K	L
1							Skill Matrix					
2	Operator	Start Time	Finish Time	Lunch (min)	Break 1 (min)	Break 2 (min)	Spancrete	Saw	Forktruck	Tractor	Gentry	Strand
3	Operator [1]	?	?	?	?	?	Yes/No	Yes/No	Yes/No	Yes/No	Yes/No	Yes/No
4												

Figure 3.6: Labourers Data Sheet

The last data sheet is the colours data sheet (see Figure 3.7). This sheet was used to help the user distinguish the activities within the simulation model while the model is running. Each colour represents a different process, e.g. the white colour indicates when the Spancrete machine pours the concrete on the bed.

	A	B	C	D	E	F	G	H	I	J	K	L
1	Strand Laydown	■										
2												
3	Strand Tensioning	■										
4												
5	Concrete Pounding	■										
6												
7	Concrete Drying	■										
8												
9	Concrete Dry	■										
10												

Figure 3.7: Colours Code Data Sheet

3.2.2 Model Building

This is the most complex phase in developing simulation models as it requires extensive knowledge of modelling techniques and programming skills. The Witness package from the Lanner Group was chosen as the simulation tool to develop the simulation model of the factory.

Witness package is a MS Windows application software from the Lanner Group. The Witness simulation package was developed specifically to model material flow in manufacturing systems. It is a hybrid of a simulator and a general simulation language. The interface of this package provides the user with a set of menu options, which are used to develop a generic model of a manufacturing system.

Figure 3.8 shows a snapshot of the Graphic User Interface (GUI) of Witness software. Building the simulation model of the factory has been accomplished in two stages using the designer element of Witness: “*structure the model*” and “*coding*”.

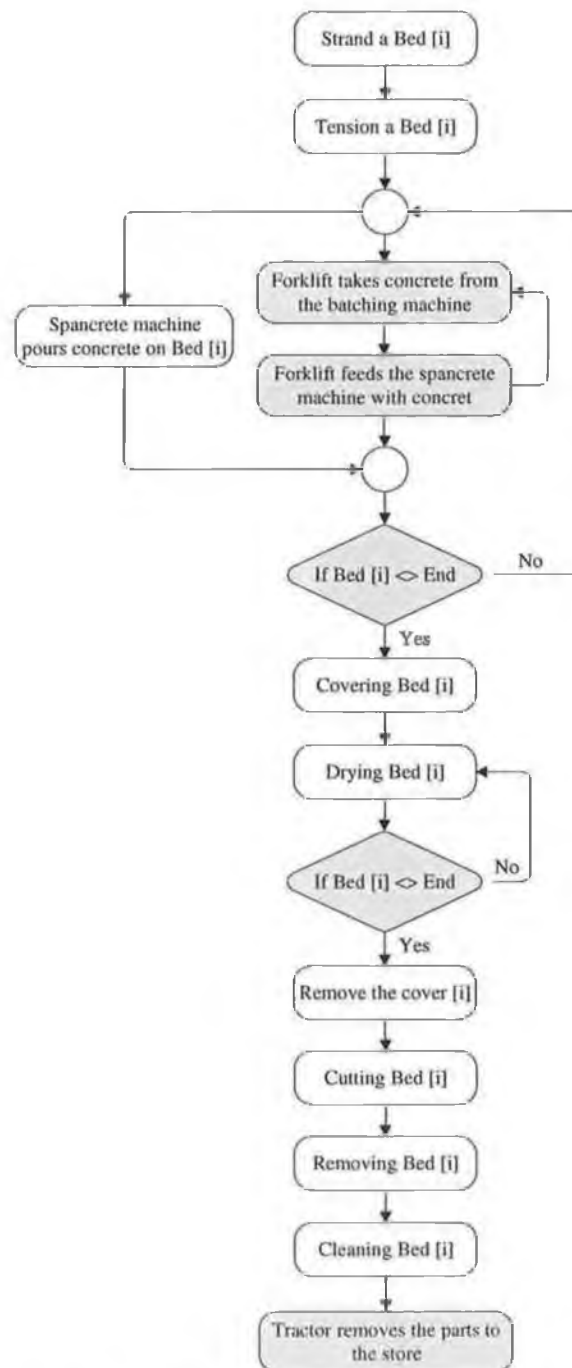


Figure 3.9: Processes of the Factory

As shown in the factory processes flowchart, the first process that needs to be done is to strand cables on the beds, then these cables need to be tensioned. Once the cables are tensioned, the Spancrete machine starts moving and pouring the concrete on the bed. The concrete is fed to the Spancrete machine continuously by the forklift based on how much concrete is left within the batcher of the Spancrete machine. When the Spancrete machine reaches the end of the bed, and the concrete is poured on all the

bed, a special material is used to cover the bed to speed the process of drying the concrete on the bed. Before the saw machine starts cutting the concrete into pieces, the special material needs to be removed. The crane is used to remove the cut pieces from the bed. Once the parts are removed, the labourers start cleaning the beds and at the same time the tractor is used to carry the slabs to the store. The activities that are presented in the flowchart are repeated in the other two beds with a different sequence of the processes based on the schedule of running the factory.

- *Display the simulation elements of the factory:*

The first phase of displaying the simulation elements of the factory was done to construct the layout of the factory based on the AutoCAD drawing and the factory layout data sheet (see Figure 3.2). The Screen Editor of Witness as shown in Figure 3.10 was used to draw the layout of the factory.

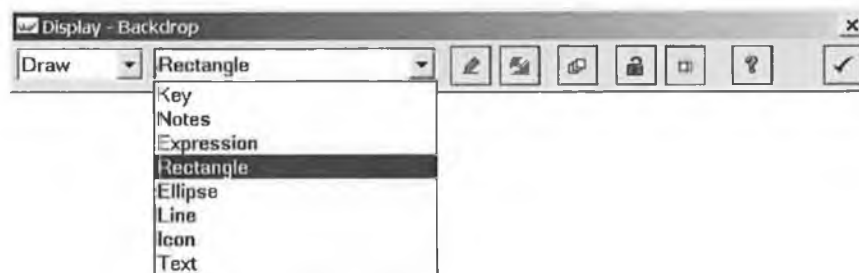


Figure 3.10: Screen Editor of Witness

The elements of the factory were represented by icons and drawings in order to build up a pictorial representation of the factory, while some other elements were hidden to be used for calculations only. Three conveyor objects were used to display the three beds, while the screen editor of Witness was used to draw the beds. Part objects with different colours were used to display the different activities within the beds based on the colour codes data sheet. Some of the activities included stranding and tensioning the cables and pouring the concrete. Hidden machine objects were used to pull parts from the conveyors to represent the cutting and the removing processes. An icon of a tractor was used to display the tractor travelling on a track, which has been displayed using paths to link the location of cut pieces that have been removed from the bed to the store. The forklift was displayed using a part object with an icon of forklift; a

number of conveyors displayed as connected lines are used to display the forklift travelling around the factory. Conveyor and part objects were used instead of a vehicle and a track to represent the forklift because the track in Witness has only a loading point and unloading point that can not be used to accomplish the function of the forklift in the simulation model. The forklift was used to feed the Spancrete machine with concrete at different locations. Figure 3.11 shows a snapshot of the simulation model of the factory.

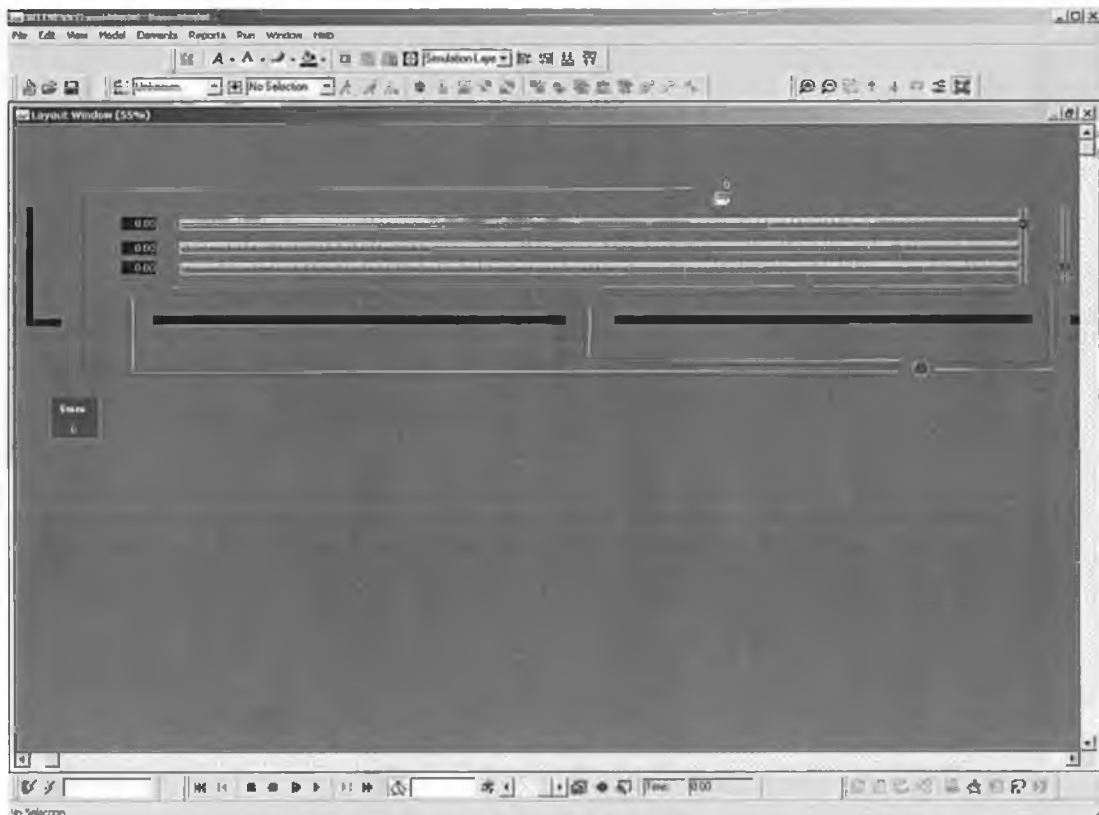


Figure 3.11: Snapshot of the Simulation Model

3.2.2.2 Coding

This phase was achieved in two stages. In the first stage, all the elements of the simulation model of the factory were detailed based on the collected data using the element's detail dialog of Witness. The factory layout sheet was used to detail the beds that have been represented within the model as conveyors. Figure 3.12 shows the detail dialog of the conveyor object which was used to detail Bed1. Conveyors were also used to give the tractor and the forklift paths to travel around the factory.

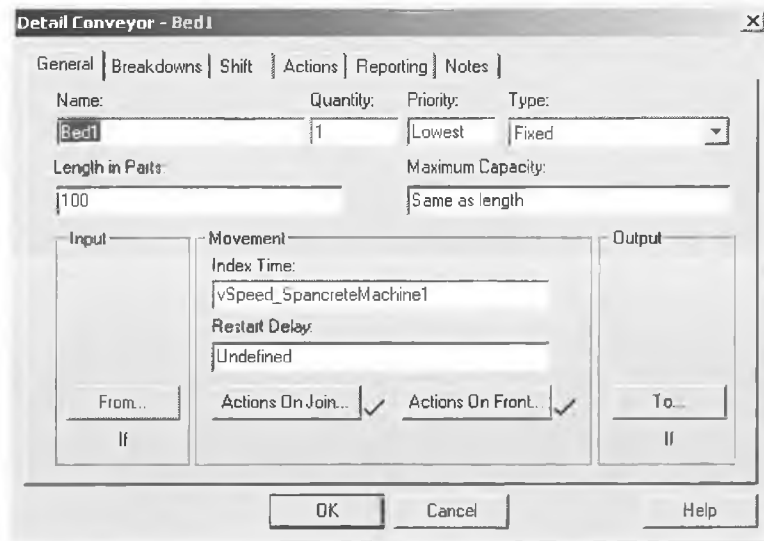


Figure 3.12: Detailing Bed1

As shown in the Figure 3.12, the value 100 was given to the attribute “*Length in Parts*”. This was done to specify the length of bed in meters as each part within the conveyor represents 1 meter. A variable called “*vSpeed_SpancreteMachine1*” was created and assigned to the Index Time to hold the value of the speed of pouring the concrete. This value matches the speed of the Spancrete machine while it is moving along the bed to simulate the process of pouring the concrete. The “*Actions on Join*” and “*Actions on Front*” buttons were used to define all the actions and the logic associated with the Bed1, which was represented by this conveyor.

The Spancrete machine was detailed using the Spancrete Excel data sheet. The Spancrete machine and other machines of the factory were detailed using the machine detail dialog as shown in Figure 3.13. The same variable that has been used to define the Index Time of moving the parts in Bed1 was applied to the cycle time of the machine objects to represent their speed. The value of this variable changes based on whether the Spancrete machine is pouring the concrete or moving back to its initial location. Input and Output rules have been assigned within the editors that can be accessed by clicking on the buttons labelled “*From*” and “*To*”. The number of labourers needed to handle the Spancrete machine was assigned by clicking on the button labelled “*Labour Rule*”. The input actions editor, output actions editor, starts actions editor and finish actions editor can be accessed by clicking on the following buttons “*Actions on Input*”, “*Actions on Output*”, “*Actions on Start*”, and “*Actions*

on Finish” buttons respectively. These editors are used to command the Spancrete machine and other machines to perform different activities within the factory.

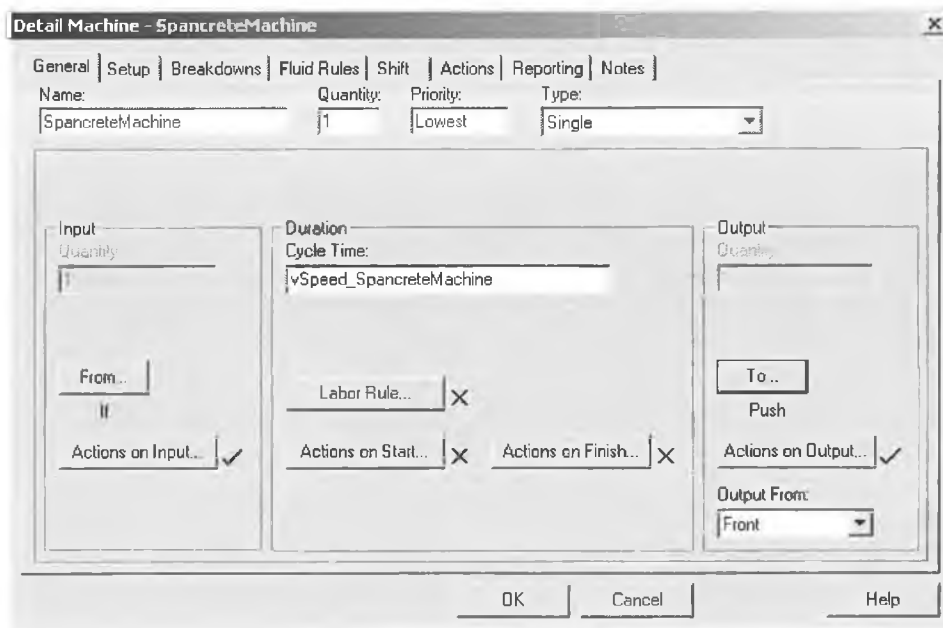


Figure 3.13: Detailing the Spancrete Machine

The data within the material handling1 data sheet and material handling2 data sheet were used to detail the saw machine, crane machine, stranding machine, tensioning machine, forklift, tractor and the processes associated with them; e.g. the speed of the forklift with load and without load, number of labourers needed to drive and handle each machine and some other attributes. Once all the elements of the factory were detailed, the logic of the model was implemented using Witness command language (WCL) by writing different algorithms within the element’s detail dialog box using the actions editors. The logic of the model was applied based on discussions with the system experts who were familiar with the logic of a running factory. Some of the logic that was employed in running the factory includes:

- Once tensioning the cables takes place, all other processes should be stopped and the labourers should leave the shop floor until the tensioning process is over.
- Once the beds are poured, time should be allowed for drying the concrete before cutting the beds.
- The machines can not pass each other.

- The forklift feeds the Spancrete machine when the capacity of the Spancrete machine is less than 2m^3 of concrete.

Modelling the logic of running the factory has been achieved by creating a number of dummy machines, dummy parts, and variables to represent all the elements of the factory. Programme codes were written using WCL within the actions on the input of the machines that pull parts every second to monitor the activities of the factory. The values of the variables were used to feedback data to the elements of the factory to do the right actions based on the assumption that has been made by the code functions.

3.2.3 Model Testing

Model testing is very important to ensure that the developed model is working properly and doing the right job. The model was tested first by verifying the logic and the behaviour of the model, then validating the model. Verification and validation of the model of the factory is not a phase or a step in the life cycle of developing the model, but a continuous activity throughout the developing phases of the model [117]. The following subsections present the verification and the validation processes of the simulation model of the factory.

3.2.3.1 Model Verification

Model verification deals with building the model right to make sure that the model is implemented correctly in the computer and the input parameters and the logical structure of the model is correctly represented. The simulation model of the factory was verified in a number of ways:

1. Viewing the animation and simulation clock simultaneously of Witness while running the model in slow speed. This technique was used to point out any problems, e.g. whether the model was working right based on the assumption.
2. The computerised representation was checked by people other than the author.
3. The model logic was checked for each process.

4. The interact box of Witness was used to monitor the activities of the elements of the simulation model of the factory. It is a special kind of window which displays messages during a simulation run.
5. The statistical graphs, reports and the utilisation of the machines and labourers were used to check the sequence of processes i.e., stranding cables, pouring concrete, cutting the concrete, removing slabs, etc.
6. Algorithms were written using WCL to read the data of all machines and the labourers of the factory to produce trace files, which consist of detailed output representing the step-by-step progress of the simulation model over the simulated time. This allowed error detection. The trace file was used to display the time when the machine starts working, type of the work the machine was doing, names of the labourers handling the machine, etc.
7. Different experiments were tested by having different sequences of the processes (stranding, tensioning, drying, cutting, cleaning, etc.) on each bed to see how the system responds. Then, these results were discussed with the system's experts to see if the logic of running the factory is right.
8. Robinson [90] states that both the model logic and real-world behaviour can be verified by watching the model, so the virtual model of the factory, which is to be discussed later, has been used as a tool to verify the results of the model by monitoring the activities of the virtual model and comparing the statistical results simultaneously to have a visual trace of events as they happened.

3.2.3.2 Model Validation

Once the model was verified, the next step was validation to determine whether the computerised model was an accurate representation of the modelled system. Validation is to determine if the right model has been built and reflects reality. The main goals of the validation process are:

- To produce a model that represents true system behaviour closely enough for the model to be used as a substitute for the actual system for the purpose of experimenting with the system.

- To increase to an acceptable level the credibility of the model so that the model will be used by managers and other decision makers.

However, there are no validation techniques that give 100% certainty of the results of a model. The model was validated by trying to make sure the model performance measures matched the behaviour of the corresponding system. The following techniques used to validate the simulation model of the factory:²

1. *Comparison to other models [118,119,120]* - One of the best ways to validate a simulation model is to compare the results predicted by the model with the performance of the real system and other valid models. So, the first technique that has been used to validate the model was consulting the experts of the modelled system who have practical experience with similar factories to compare their performance.
2. *Extreme condition tests [120,121,122,123]* - This method involves conducting runs to simulate different situations of running the factory to verify that the model performs as intended in such situations. The model was tested to observe its reaction by changing the sequences of the processes on each bed and changing the number of labourers who run the factory.
3. *Parameter variability (Sensitivity Analysis) [118,122,123,124,125]* - Sensitivity analysis can be defined as the systematic investigation of the reaction of the model outputs to drastic changes in model inputs. It involves comparing the effect of change in input parameters, indicated by simulation results, to the expected trends. Sensitivity analysis was performed to see if the model behaved as the system would, and to identify the factors that the model was most sensitive to. Some of the parameters of the factory that have been used to test the sensitivity of the model were the speed of the machines which is represented by changing the cycle time of the machine elements in Witness. Another parameter was changing the number of the labourers who handle the processes of the factory. These parameters were tested to measure the performance of the system in terms of the throughput time and the output, which is calculated by the number

² The detailed results of experiments conducted with this model are subject to a confidentiality agreement with the company.

of slabs located at the store. This information was useful in understanding the relationship between the production and the input parameters.

4. Graphical presentations and reports were also used to validate the system by discussing the results with the system's experts.

As shown in Figure 3.1, the verification and validation processes have been done throughout the life cycle of developing the simulation model of the factory.

3.2.4 Experimental Design

After the model of the factory was verified and validated, it was found that the developed model is a good working model which could be used for experiments to evaluate various aspects of the system. The aspects of the factory, which have been studied, include the following:

- Effects of having different numbers of labourers were investigated.
- Effects of changing the sequences of the processes on each bed, which include stranding and tensioning the cables, poring the concrete, cutting the concrete into slabs, removing the slabs out of the bed, and removing the slabs to the store.
- Different shifts have been applied to the labourers and the machines to determine the outcome for the factory in terms of cost and throughput.
- Different schedules have been investigated to find the best schedule that reduces the cost and increases the throughput.
- Different drying times were investigated.

3.2.4.1 Model Run

Once the model has been validated it can be run to assess the performance of the system. A number of runs were performed with different assumptions to assess the sensitivity of the system's performance to various factors and identify the limiting ones. The activities of the simulation model were monitored every second by a number of dummy machines, parts and variables, which were used for calculations

and to feedback and exchange the data between the machines and the labourers. Figure 3.14 shows some of the machines and the variables that have been created to monitor the simulation model.

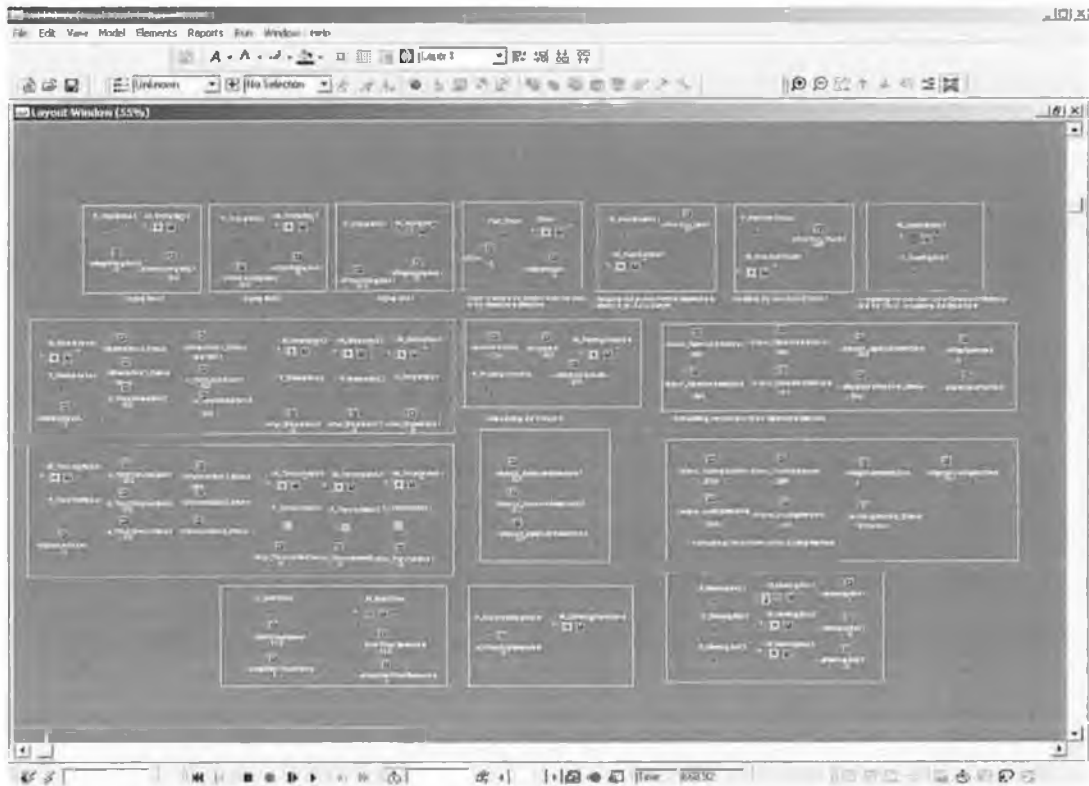


Figure 3.14: Dummy Machines, Parts and Variables Used to Monitor the Activities of the Simulation Model

Witness provides two different modes to run the simulation model. These modes are “*Walk run mode*” and “*Batch run mode*”. To get results with a continuous display of running the factory, the model needs to be run under the “*walk run*” mode, which runs the simulation model in slow mode to display the movement of labourers, the machines and their processes around the factory. Running the model in “*walk run*” mode makes the model running very slow, especially with the complexity of the developed model of the factory. The “*run batch*” mode is used to run the model much faster but without displaying the processes of the factory. This was needed to collect statistics and measure the utilisation of the elements of the factory for a long period of time. Figure 3.15 shows some of the activities of the factory while the simulation model is running. As shown in the figure, the cutting machine is cutting

the first bed into slabs; the tractor is carrying the slabs that have been moved out of the bed to the store that contains 18 slabs.

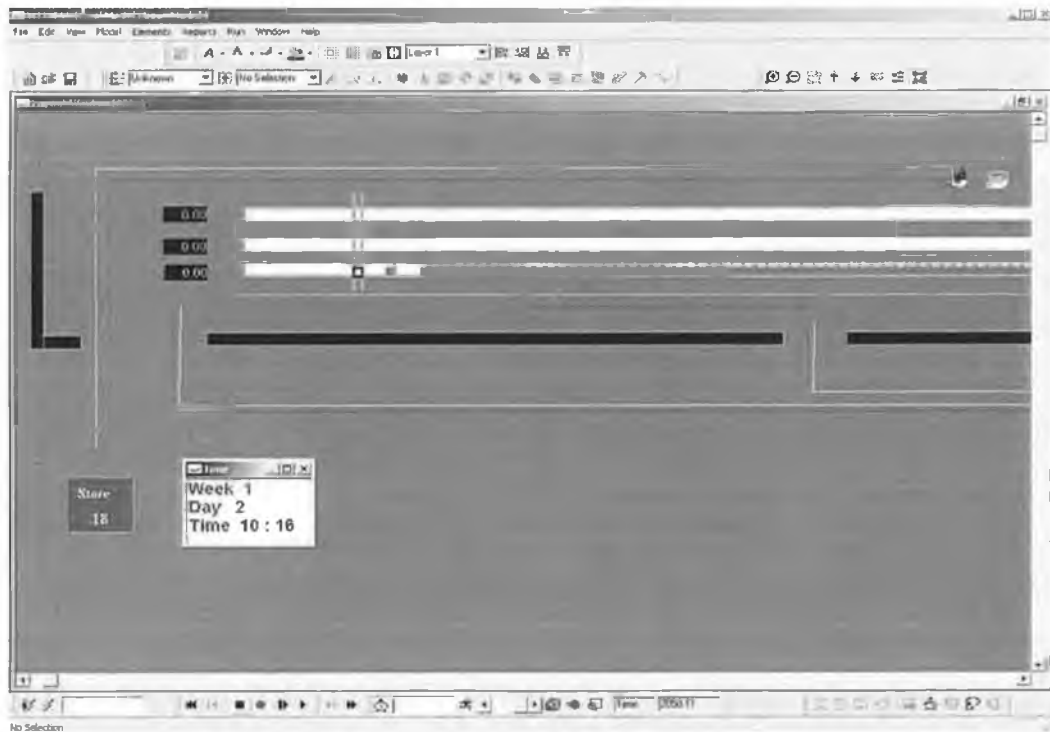


Figure 3.15: Simulating the Activities of the Factory

3.2.4.2 Output Analysis

Output analysis is the analysis of data generated by a simulation. Its purpose is to predict and measure the performance of the activities that are being simulated, or to compare the performance of two or more alternative system designs of the factory. The output measures used were the utilisation statistics of the machines and labourers, also to monitor some parameters while the model is simulating the activities of the factory. The output forms that have been used to determine and measure the performance of the factory are pie charts, reports, data information and timeseries graphs. A description of these output forms is presented below:

- **Pie Charts:** Are used to show the utilisation of the machines and the labourers, which can assist define if more labourers are required or not. If the results show that the utilisation of two labours within a day is 20%, that means one of them can handle the duties of both of them and there is no need for the other. Machine

utilisation is important to schedule the processes of the factory. Figure 3.16 shows a snapshot of the utilisation of the machines and labourers at 11am and eight minutes.

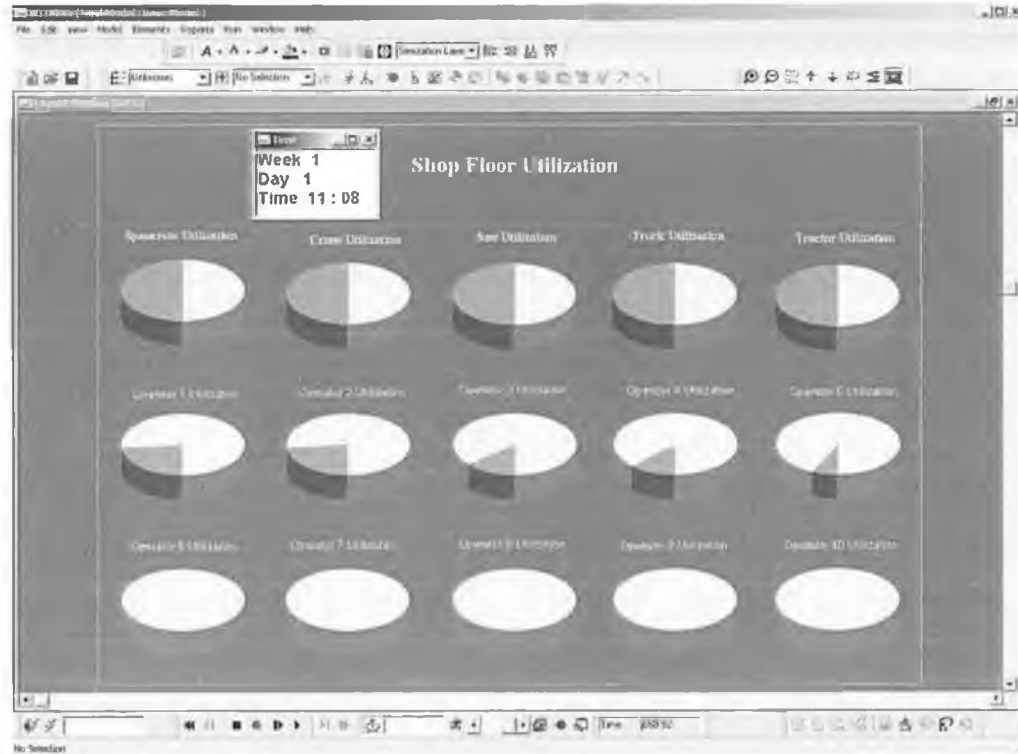


Figure 3.16: Utilisation of the Machines and the Labourers

- **Timeseries:** It is a graphical element that presents the simulation results in the form of a graph that plots values taken from the simulation model against time. Figure 3.17 presents the quantity of the concrete content within the batcher of the Spancrete machine. This was used to validate and verify the results of the simulation model. The vertical Y axis represents the concrete within the Spancrete machine as the full capacity is 5 m³, and the horizontal X axis represents the time of running the simulation model in minutes.



Figure 3.17: Concrete within the Spancrete Machine

The timeseries graph was also used to watch the activities of the labourers against the time. This graph is used to help the decision makers to assign and organize the duty of each labourer as it shows how many labourers are busy or idle while the model is running (see figure 3.18).

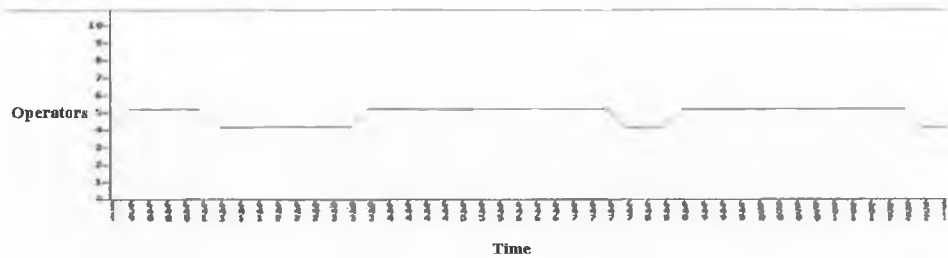


Figure 3.18: Monitoring the Activities of the Operators

Data Information: It is used to provide numerical and text data to present some information about factory. Figure 3.19 shows some of the data information that has been used to display the following:

- Concrete used since the start of the running of the model.
- The current quantity of concrete content within the Spancrete machine.
- Total busy time of the operators in hours and minutes format.
- The text information is used to present the type of job each labour is currently doing.

Operator1	none	Operator6	Driving the Saw (Bed1)	Concrete Used: 216.00
Operator2	none	Operator7	Driving the Crane (Bed1)	
Operator3	none	Operator8	none	Busy time of Operators
Operator4	Cleaning (Bed1)	Operator9	none	
Operator5	Cleaning (Bed1)	Operator10	Driving Trucks	
				Hours: 5
				Minutes: 54

Figure 3.19: Data Information of the Factory

3.2.4.3 More Runs

Based on analysing the results of the runs that have been completed, the analyst of the factory determines if additional runs are needed and what design those additional experiments should be simulated. Different experimentations have been conducted

with the model to see the effects of changing the sequence of the processes on each bed, and the shifts time of the labourers and other elements of the factory.

3.2.5 Completion Phase

The last phase of developing the simulation model is the documentation and reporting. Two types of documentation were used to document the simulation model of the factory:

- Program documentations.
- Progress documentations.

The programme documentation was done to be used as a reference for changing or modifying the simulation model in the future either by the same or different analysts, also to be used as a guide to understand how the programme operates. The programme documentation includes coding and algorithms of how the model of the factory operated.

The progress documentations were made to determine the relationships between the input parameters and the output measures of the factory. This was done to optimize some output measures of the performance of the factory. The progress documentations included the figures and the comments of the output analysis of all the experiments which have been made to address the factory.

The result of the all the analysis was reported in a final report. This was done to enable the model users and the decision makers to review the final formulation, the alternative systems designs that were addressed, the results of the experiments, and the recommended solutions to the problems.

3.2.5.1 Implementation

The success of the implementation phase depends on how well the previous phases have been performed. The system's experts have been involved in every step of

developing the simulation model to ensure a successful implementation of the model. Based on the discussion with the system's experts of the factory, the model will be used as a tool to implement the real model of the factory.

3.3 DEVELOPING THE VIRTUAL MODEL OF THE FACTORY

Visualisation has become a critical component of simulation technology in manufacturing applications. It provides the simulation practitioners with an environment to discuss and get a better understanding of the simulation model's behavior. Graphical presentation and animation can be significant tools to communicate the outcome of simulation models to a non-technical audience. Decision makers often do not have the technical knowledge to understand the statistical results of a simulation model. But when the outcome can be expressed using animation, a better level of understanding becomes possible [126].

As mentioned earlier in this chapter, most of the simulation packages available on the market today have their own visualisation tool, which is used to represent the processes of the simulation model in a 3D presentation. The virtual models that can be created or generated using these packages have some limitations, which include:

- Absence of real time interaction with the model.
- Lack of support for VR devices (e.g. Head Mounted Display).

To overcome these limitations, a new method was introduced to develop a virtual model using a VR commercial package, and then integrate the developed virtual model with the developed simulation model. The following flowchart (Figure 3.20) shows the steps that have been taken to develop the virtual model of the factory using Superscape VRT.

Superscape VRT [127] was chosen as the virtual reality software package to design and create the virtual model of the factory. Superscape VRT is a complete 3D authoring studio for personal computers that lets users create interactive 3D worlds that can be published on the Internet using Superscape's Viscap, or displayed on the

standalone Visualiser platform. The package consists of an integrated suite of editors (world editor, shape editor, sound editor and image editor), which are used to create worlds. The browsers of Superscape, which include Visualiser and Viscap, are used to view the worlds. Textures and sounds can be added to the objects in the virtual worlds to make them more realistic, and different lighting setups can also be introduced. Using Superscape Control Language (SCL), a control language based on the popular C language, behaviour to objects in the world can be assigned, and complex actions can also be performed. The following section presents the steps that have been carried out to develop the virtual model of the factory.

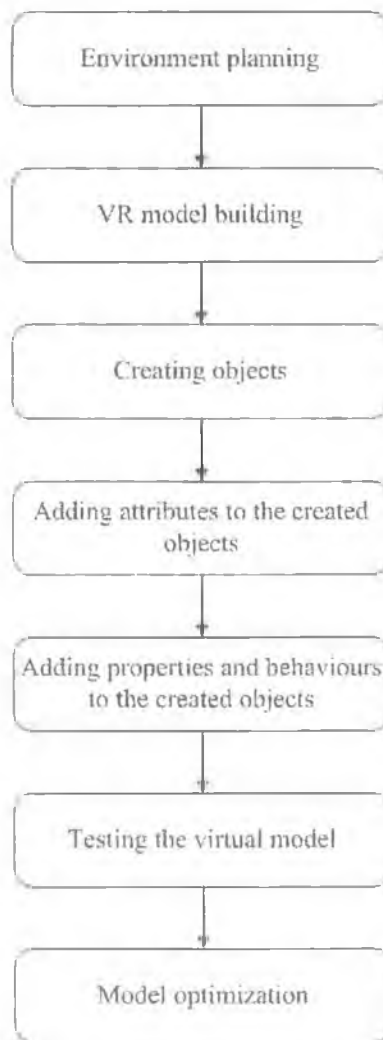


Figure 3.20: Steps of Designing and Creating the Virtual Model of the Factory

3.3.1 Environment Planning

At this stage, all the objects that are going to be included in the virtual environment and the way the user can interact with them have been decided. A list was created that included all the objects which should be included within the virtual world to check if there were any missing objects that may cause a significant change to the sorting later. The list included the following objects:

- Three beds.
- Spancrete machine.
- Batching machine.
- Forklift.
- Operators.
- Tractor.
- Saw machine.
- Crane.

Visualizer and Viscap are the two editors of Superscape VRT that can be used to display the virtual worlds. Visualizer is used for standalone applications to view large complex worlds with many levels of interactions, while Viscap is used for 3D web pages where the created virtual worlds need to be as small as possible to minimise their download time. As the interest of the virtual model of the factory was not to be published on the net, adding details as much as required to model would not affect viewing the virtual model.

3.3.2 VR Model Building

The world editor (see Figure 3.21) of Superscape VRT was used to develop the virtual world of the factory by defining spaces that each object in the world would occupy. A series of groups were created that represented the objects that were to be built, which include the three beds and the other machines of the factory. A name was given to each object of the factory to distinguish them when the shapes were applied. Some objects were added from the virtual clip art libraries of Superscape

(e.g. operators). This stage is considered as a sketching phase using building blocks where the size and the position of the objects do not have to be exact.

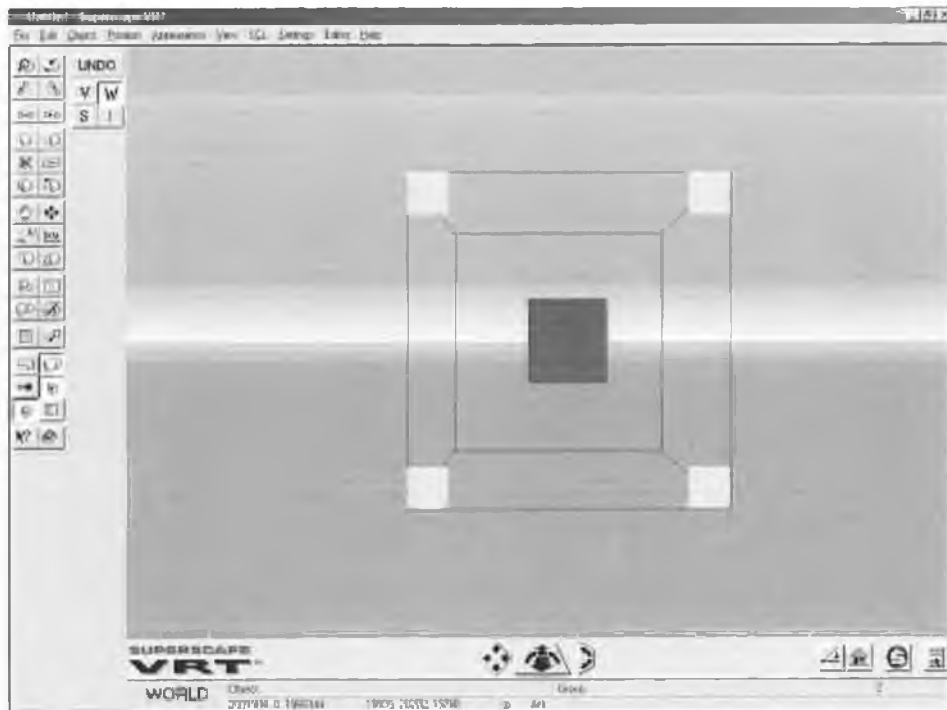


Figure 3.21: World Editor of Superscape VRT

3.3.3 Creating Objects

Once the virtual factory has been created from a number of groups, the shapes of the objects can be constructed using the Shape Editor of Superscape (see Figure 3.22). The size of each shape's bounding cube was set according to the size of the object that would utilise it using the "Shape Size" from the Shape Editor menu.

The X, Y, Z axes of the cube, as can be seen in Figure 3.22 are defined by gray arrowed lines and identifying letters. The bottom left hand corner of the cube is the origin, which has the X, Y, Z coordinates of 0, 0, 0. Initially, the front face is toward the viewpoint through which the points can be seen and lines that show the shape's bounding cube. Relative points or geometric points can be used to create the shape. Geometric points were used where possible to create the shapes of the world. They are generally easier to position and are processed more quickly than relative points.

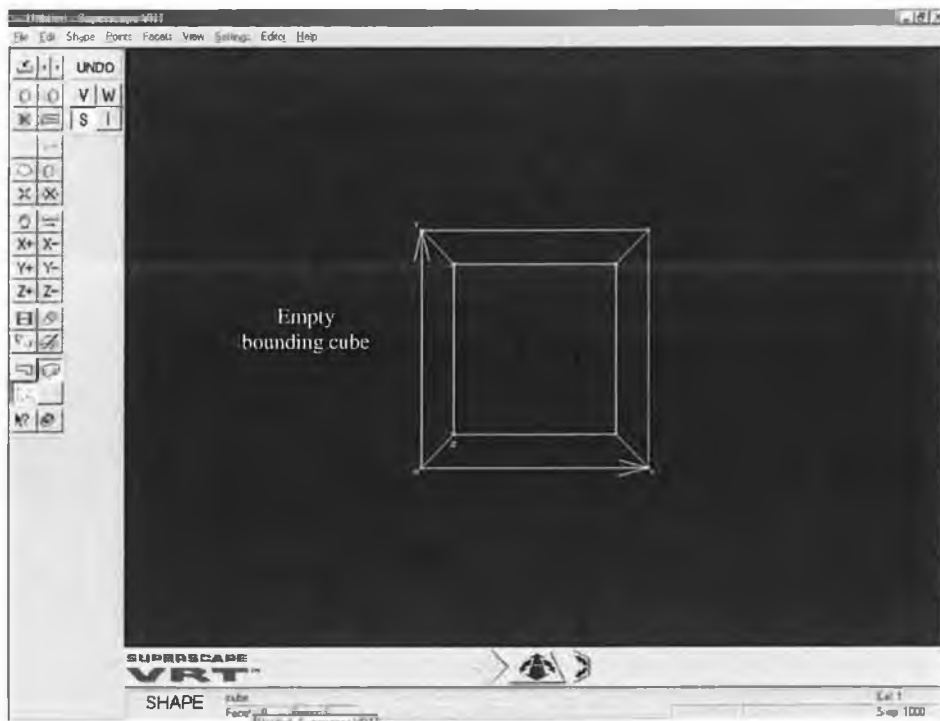


Figure 3.22: Shape Editor

After the shapes of the elements of the factory were created individually, the World Editor was used to apply these shapes to the groups of the factory. Some machines within the factory, such as the Spancrete machine, consist of more than one part, which included the body of the machine, the movable part that is used to pour the concrete and the batcher. A shape was created for each part, and then these shapes were applied to the group object of the Spancrete machine in the virtual world to construct the final shape of the machine.

3.3.4 Adding Attributes to Objects

This was performed after all the objects were created and their shapes are added. Some attributes were applied to the objects within the factory such as size, position, textures, dynamics, colour, and sound. For example, a dynamic attribute was given to machines (Spancrete machine, forklift, tractor, etc) to make them movable. This was done using the dynamics window (see Figure 3.23), where the information regarding the movement, velocity, and response were neglected because the movement of these machines depends on the movement of the machines in the simulation model in Witness.

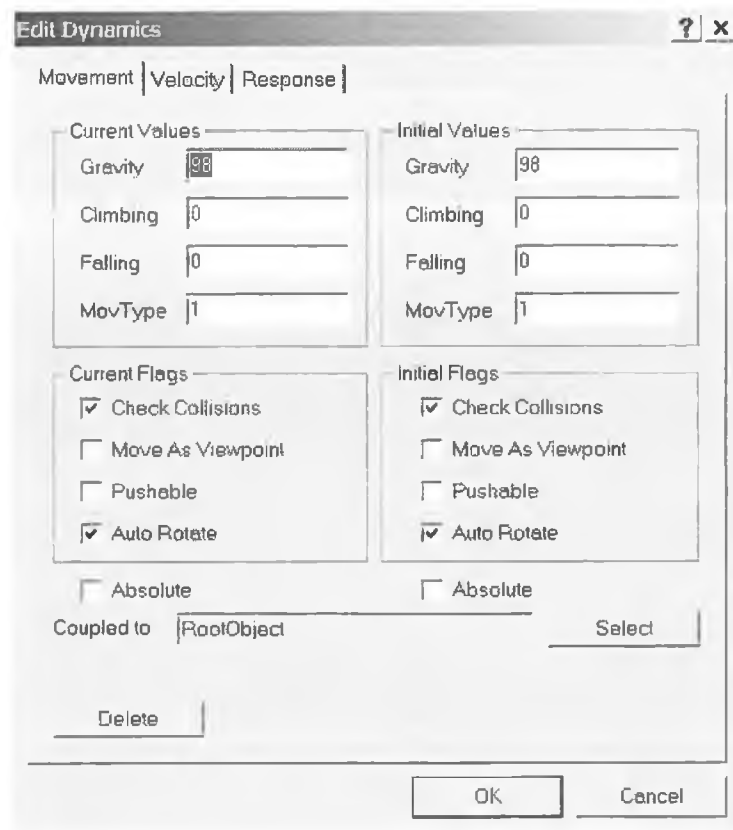


Figure 3.23: Dynamic Attribute Window

Size and position attributes were given to the elements of the factory to construct the layout of the factory based on the layout data sheet that was provided. A colour attribute and different colours were given to elements of the factory for enhancing the look. A texture attribute was applied to the wall of the factory and finally a sound attribute was given to the machines to indicate their working condition.

3.3.5 Adding Properties and Behaviours to Objects

The object properties window as showing in Figure 3.24 was used to assign different types of properties to the machines of the factory. An integer type property was used to hold the values of its positions; A Boolean type property was applied to the machines to hold its working condition; and a string type property was assigned to the machines to hold the names of the processes they perform.

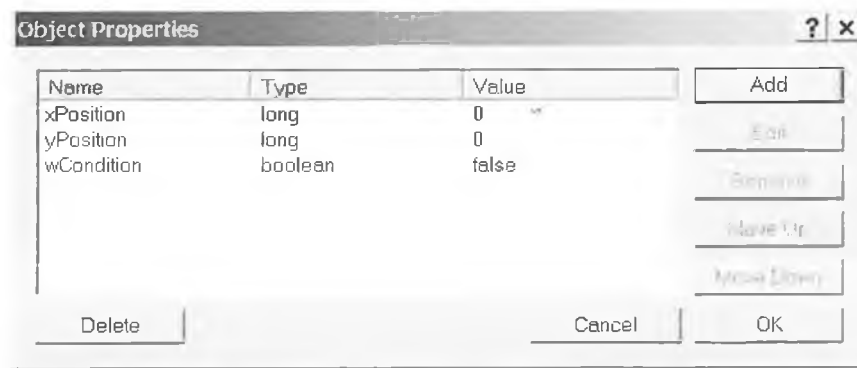


Figure 3.24: Object Properties Window

Behaviours were assigned to the machines by Superscape Control Language (SCL) to simulate the processes of the virtual factory based on the supplied data from the simulation model. Adding behaviours to the objects is the most complex phase of developing the virtual model of the factory as it requires good programming skills. The SCL code was added to the objects using the internal editor of Superscape as Figure 3.25 shows. The SCL code was used to command the objects within the virtual model to perform certain actions once a condition was executed.

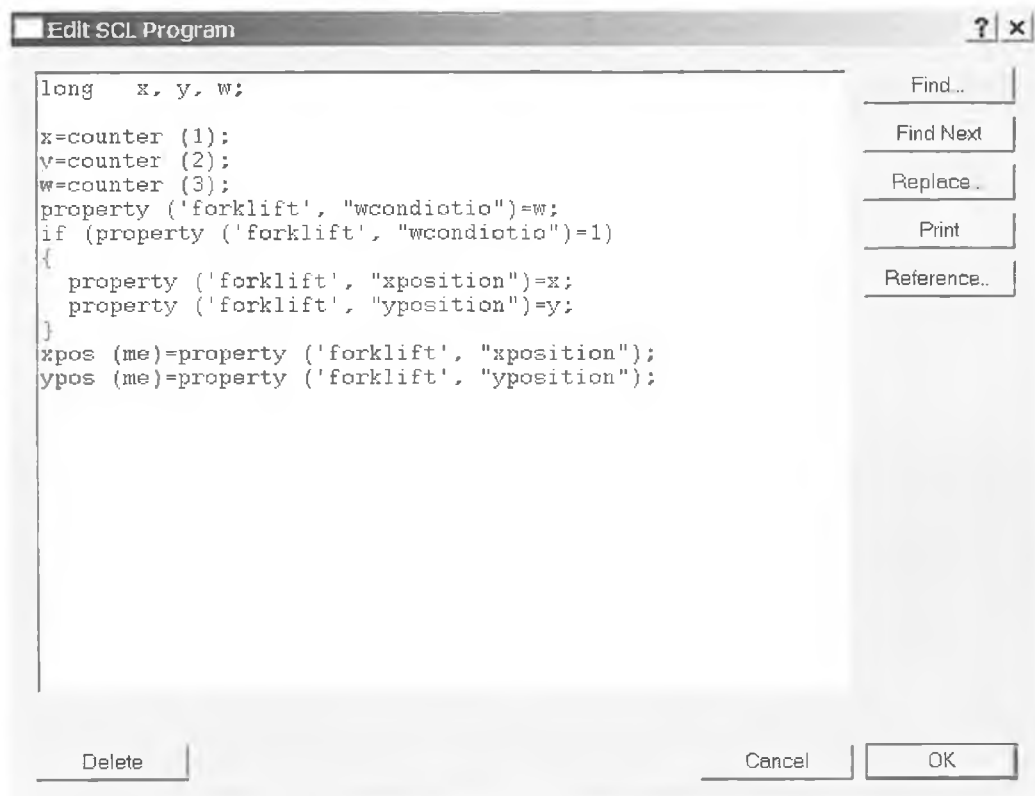


Figure 3.25: SCL Editor of Superscape VRT

Here is an example of assigning properties and behaviours to the forklift:

Two Integer properties called “*xPosition*” and “*yPosition*” were assigned to the forklift to hold the X position and Y position of the forklift within the factory. Also, a Boolean property called “*wCondition*” was given to the forklift to indicate its working condition, with “1” indicates working and “0” indicates idle. The following is an explanation for the code provided in Figure 3.25:

“*Long x, y, w;*” *statement* to define three variables to exchange data.

“*x = counter (1)*” *statement* to read the value of the current y position of the forklift from the simulation model through a variable called “*counter (1)*” and assign it to a variable called “*x*”.

“*y = counter (2)*” *statement* to read the value of the current y position of the forklift from the simulation model through a variable called “*counter (2)*” and assign it to a variable called “*y*”.

“*z = counter (3)*” *statement* to read the value of the current working condition of the forklift from the simulation model through a variable called “*counter (3)*” and assign it to a variable called “*z*”.

```
if (property ('forklift', "wcondiotio")=1)
{
    property ('forklift', "xposition")=x;
    property ('forklift', "yposition")=y;
}
```

This If condition tests if the working condition of the forklift equals “1”, the value of x is assigned to the *xposition* variable of the forklift, and the value of y to the *yposition* variable of the forklift.

```
xpos (me)=property ('forklift', "xposition");
ypos (me)=property ('forklift', "yposition");
```

The last two statements are used to command the forklift to move based on the values of the *xposition* and *yposition* of the forklift.

3.3.6 Testing the Virtual Model

The virtual model was tested using Superscape Control Language to make sure that all the elements within the factory were working correctly to match the behaviour of the elements of the simulation model. Figure 3.26 shows a snapshot of the virtual factory.



Figure 3.26: Snapshot of the Virtual Model of the Factory

3.3.7 Model Optimisation

The final step in building the virtual factory was to optimise it so that it would run as fast and smoothly as possible. The speed of the world depends on many items including the number of facets that the VRT has to process, the speed of the target processor, and the platform on which is intended to display the world. Some steps

were taken while developing the virtual environment of the factory to optimise the virtual world. These included:

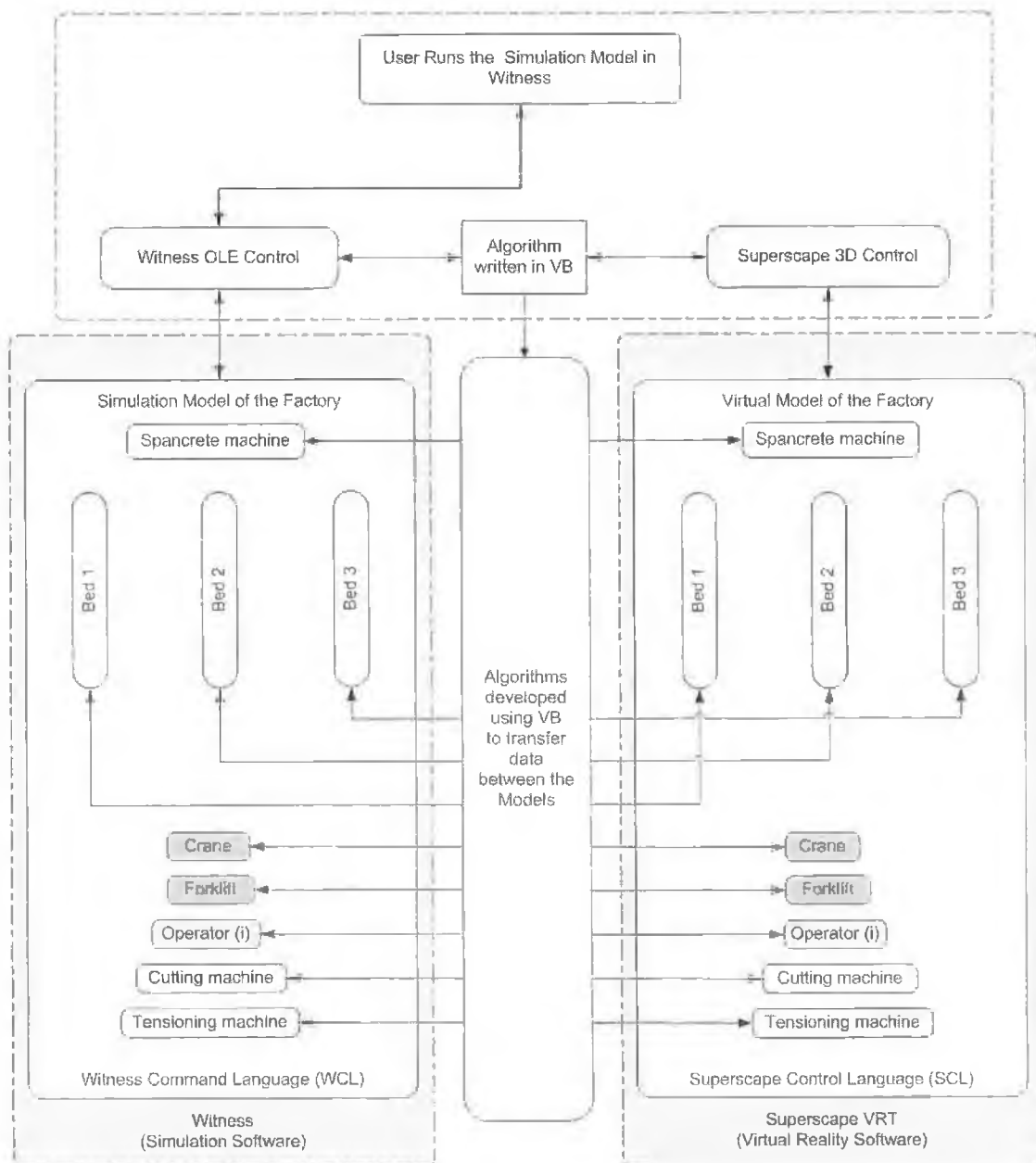
- The objects were grouped together efficiently (in groups of 10 to 15) to make sure that the VRT processed the world in an optimum way.
- The unused shapes, textures and sounds were removed to reduce the file size.
- The timing dialog box in the Visualizer was used to show the elements of the world that take a long time to process and to try to reduce them.
- Adding distance attributes to some objects to increase the speed of the world as the viewpoint moves further away.

3.4 LINKING THE SIMULATION MODEL WITH THE VIRTUAL MODEL OF THE FACTORY

At this stage, both the simulation model and the virtual model of the factory have been developed and tested to be good working models. A new method was introduced to link the simulation model with the virtual model to simulate the activities of the factory in three-dimensional presentation in real time mode. The main aim of linking the models is to overcome the above-mentioned constraints of the simulation packages to provide the user with a virtual environment to monitor the activities of the factory where the user can interact with the model in real time.

The object-based programming language, Visual Basic (VB) was chosen as a tool to link the models of the factory. Algorithms were developed using VB to transfer the data between the simulation model that was developed using Witness and the virtual model that was designed and created using Superscape VRT. Linking the simulation model with the virtual model was accomplished using two mechanisms: Witness OLE Control and Superscape 3D Control. Witness OLE control was used to link the developed simulation model with VB algorithms developed to read the data from the simulation model using Witness Command Language (WCL), then these data were sent to the virtual model using an ActiveX control called Superscape 3D control. Superscape Control Language (SCL) was used to give commands to the machines, labourers and the other elements of the factory to simulate the activities in

real time that matched the processes in the simulation model as the user can interact using voice commands, and navigate the virtual model using Head Mounted Display (HMD) while the models were simulating the activities of the factory. Figure 3.29 shows the algorithms that have been developed to link the simulation model with the virtual model. As shown in the Figure 3.27, VB was used to develop algorithms to link the elements of the factory between the two models. These elements included machines, labourers and the three beds. The algorithms were used for exchanging data between the simulation model and the elements of the virtual model.



3.27: Linking the Simulation Model with the Virtual Model of the Factory

Each element on the simulation model was linked with the same element within the virtual model using an individual algorithm. This algorithm executes every second in VB to check the current status of the element, its properties and behaviour from the simulation model and then another algorithm was used to transfer these data to the virtual model to simulate the activities in 3D presentation in real time. This approach was followed to link all the elements of the factory. The following algorithm (Figure 3.28) shows an example of how the Spancrete machine in the simulation model was linked to the 3D object of the Spancrete machine in the virtual model.

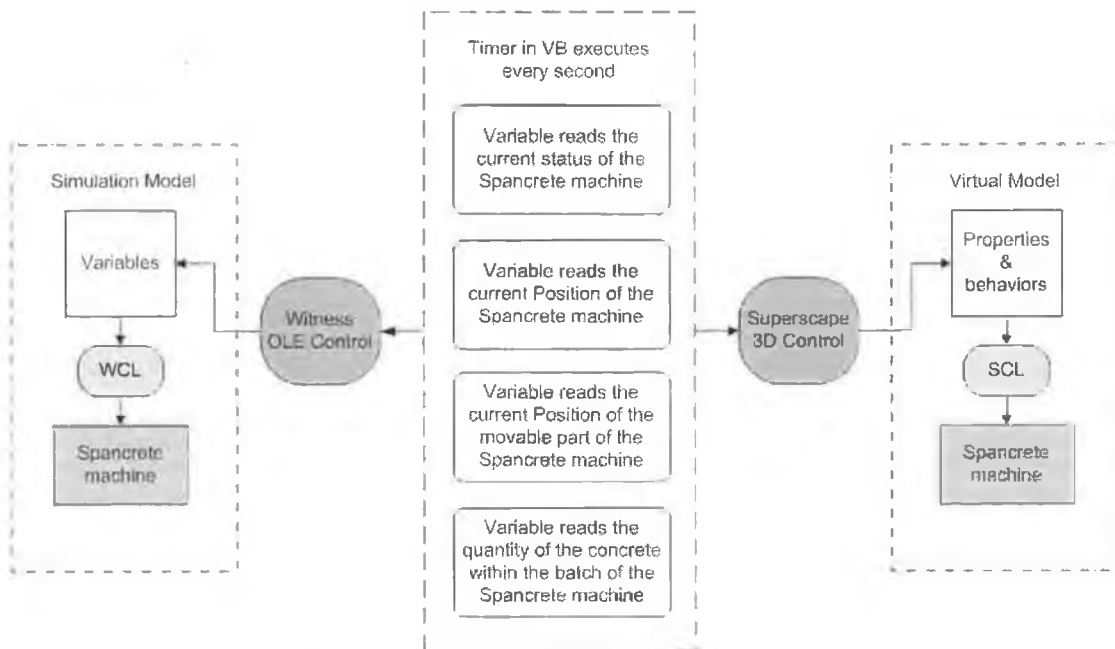


Figure 3.28: Algorithm for Exchanging the Data of the Spancrete Machine in the Simulation model with the Spancrete Machine in the Virtual Model

As shown in Figure 3.28, WCL was used to read the current activities of the Spancrete machine within the simulation model and then convert these activities as data to be saved in variables. Witness OLE control was used to transfer the values of these variables to the algorithm developed using Visual Basic, which reads the parameters of the Spancrete machine in the simulation model. These parameters included:

- Current status of the Spancrete machine.
- Current position of the Spancrete machine.

- Current position of the movable part of the Spancrete machine, which is used to pour the concrete on the beds.
- Quantity of the concrete within the batcher of the Spancrete machine.

The algorithm for transferring the data of the Spancrete machine from the simulation model executed every second to read the above mentioned parameters and save them within variables in VB. Then, the values of these variables were sent to the properties and behaviours of the Spancrete machine through the mechanism “Superscape 3D Control”. Algorithms written by SCL were used to command the Spancrete machine to do the right action that matches its activates in the simulation model. The same approach was repeated to link all the elements of the factory in simulation model with their 3D presentation in the virtual model.

The presented approach of linking the simulation model with the virtual model can be used as an approach to link any model developed using a commercial simulation package with a virtual model developed using a commercial virtual reality package, where both the simulation package and the virtual package support OLE automation.

3.5 INTERACTION WITH THE DEVELOPED MODELS OF THE FACTORY

Most of the simulation packages available on the market today do not allow real time interaction with the model while the model is running. A new method is introduced to allow users to interact with the simulation model of the factory through the virtual model in real time. The users can interact with the virtual model either by mouse or voice commands. Once a command is given by the user, the virtual model mimics the simulation model to simulate the same activity simultaneously in real time. Figure 3.29 shows a simple flowchart that illustrates how the user can interact with the models of the factory.

Figure 3.30 shows a schematic diagram of how the user can interact with the models of the factory in real time. Mouse or voice commands can be used to interact with the machines of the factory. Algorithms written using VB run every second to monitor

the activities of the virtual model. Once the user interacts with one of the machines in the virtual model, a message will be sent to the simulation model to do the right action based on the command that has been given. The interaction with the models of the factory is limited to do stop, run, breakdown and repair the machines. The Supercap 3D control and Witness OLE control are used as an ActiveX controls to link the running simulation model with the virtual model.

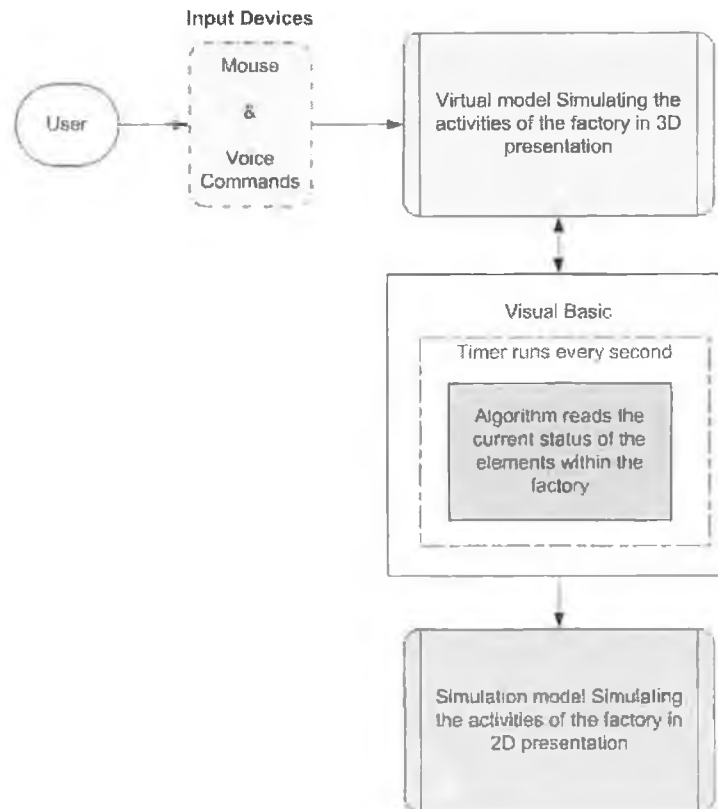


Figure 3.29: Flowchart Illustrating how the User can Interact With the Models of the Factory

As shown in Figure 3.30, algorithms written in SCL are used to translate the user input commands to actions in the virtual model. The conditions of the machines are stored as numerical data within its properties. These properties are watched at all times by codes written using SCL to see if the user has done any interaction, once the user interacts with the models, a message will be sent to the simulation model through the mechanisms Superscape 3D control and Witness OLE control. Witness Command Language translates the received message from the virtual model to actions in the simulation model. Once the user interacts with the virtual model by

clicking on one of the machines, an interaction dialog box appears as shown in Figure 3.31.

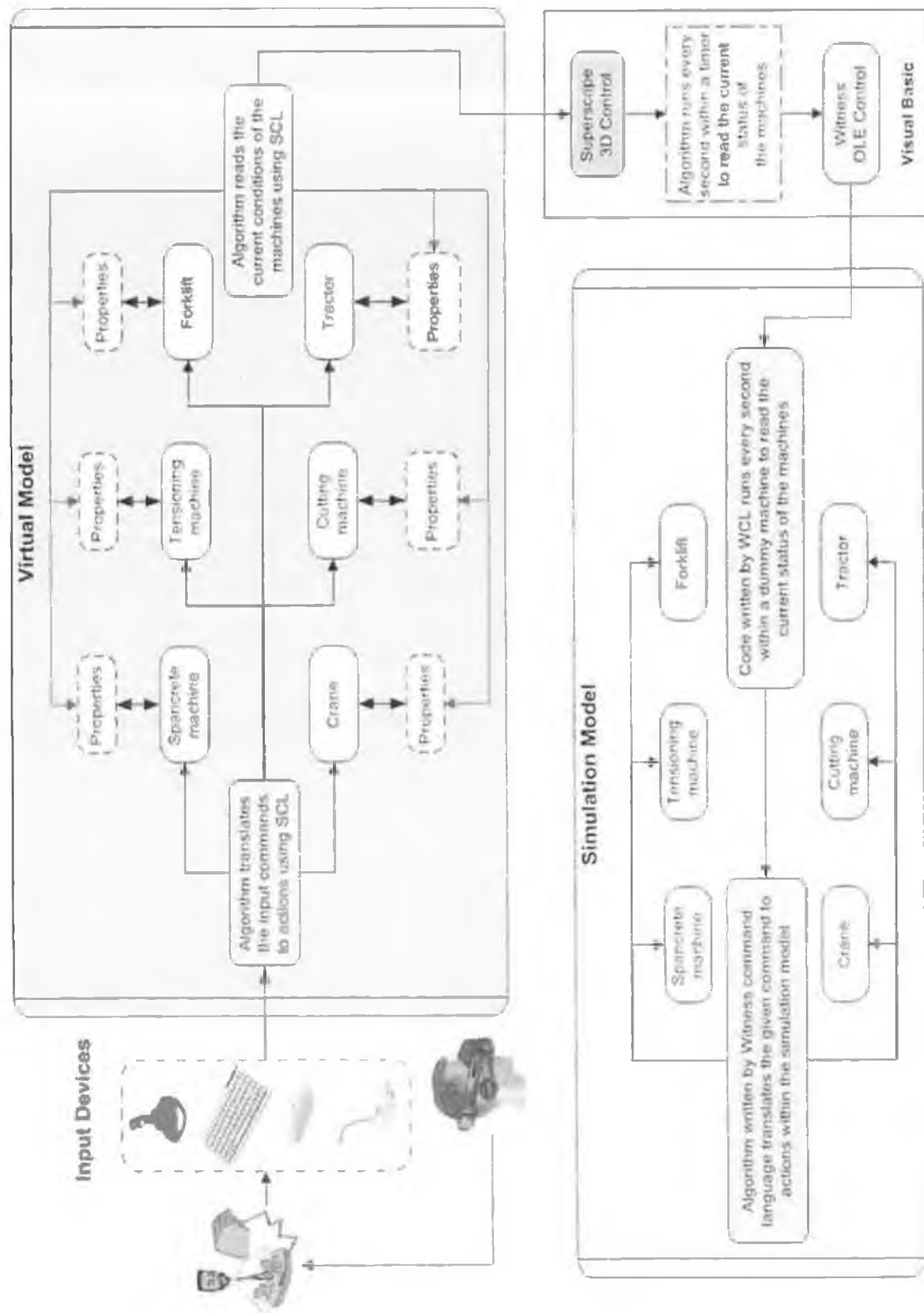


Figure 3.30: Schematic Diagram of the User Interaction with the Models



Figure 3.31: Interaction Dialog Box

As shown in Figure 3.31, the user can interact with any machine to stop it for a period of time and then run it again. Additionally, the user can enforce breakdown and repair of the machines in real time. Real time interaction is a very useful method to do different experiments with the models without the need to change the logic and the code of the models especially for non-expert users.

Voice commands can also be used to interact with the machines of the factory, where the user can stop, run, breakdown and repair the machines. The Table 3.1 presents the commands that can be used to command the machines of the simulation model and the virtual model of the factory. Chapter 4 discusses more details of how the voice commands are translated into actions within the simulation model and the virtual model.

Table 3.1: Voice commands and Their Functions within the Factory

Voice Command	Function
Name of the machine (e.g. Spancrete)	Activate the machine
Stop	Stop the activated machine
Run	Run the activated machine
Breakdown	Breakdown the activated machine
Repair	Repair the activated machine

3.5.1 Navigating the Virtual Model of the Factory

The user can navigate the virtual model using a mouse, keyboard and joystick. A viewpoint was attached to a movable object, which consists of a holding group object that has a dynamics attribute, and a child object that has a rotation attribute. The holding group represents a human body, and the child object represents its head. The user can navigate the virtual factory by changing the position of the holding group object using the stated input devices. The viewpoint is attached to the object to give the user, while navigating the virtual world, the feeling of a walking person “viewing the environment from about 1.6 meters above the ground”.

A Monitor and a Head Mounted Display were used as output devices to view the virtual environment of the factory. The HMD is coupled with a tracker, which enables the user to navigate around the virtual factory by moving his/her head to have a better feeling of being present within the virtual model. The HMD that has been used is hi-Res 900 from Cybermind Interactive Nederland.

3.6 CONCLUSIONS

This chapter has discussed a new approach of integrating a simulation model with a virtual model to address the design and planning of a new factory. The presented approach can overcome some of the limitations of the simulation software packages available on the market today. It allows users to interact with the models of the factory in real time using voice commands and navigate the virtual model using Head Mounted Display.

The presented approach of linking the simulation model with the virtual model can be used as an approach to link any model developed using commercial simulation packages with a virtual model developed using commercial virtual reality packages supporting OLE automation to address manufacturing systems.

CHAPTER 4

DESIGN AND DEVELOPMENT OF THE VR-SIMULATOR SOFTWARE

4.1 INTRODUCTION

A number of Process Modelling methods were reviewed in chapter two. Additionally, reviews of different types of simulation modelling software and Virtual Reality modelling software were presented in the same chapter. After the case study that has been conducted in Chapter three to develop and integrate simulation model with virtual model of a new factory, it was found that, developing a simulation model, using the simulation packages available on the market today, is time consuming and requires special skills in programming and modelling techniques. Additionally developing a virtual environment of a manufacturing system is a very complex task to pursue.

The aim of this chapter is to introduce the VR-Simulator software that has been developed entirely to allow non-expert users to develop simulation models and virtual models of manufacturing systems automatically. As outlined before, the objective of this project is to develop new software to address manufacturing systems design and operation to support planning and decision making. The layout of this chapter takes the following structure:

Section 4.2: presents an overview of the VR-Simulator architecture, which consists of the process modelling tool, simulation modelling tool and the virtual modelling tool. The way the user can interact with the VR-Simulator is also expressed.

Section 4.3: discusses the softwares that have been used to design and develop the VR-Simulator software. These softwares include Visual Basic, Witness simulation package, and Superscape VRT.

Section 4.4: explains the steps that have been taken to design and develop the VR-Simulator. These include the VR-Simulator software architectural issues, as well as the design of:

- Components of the VR-Simulator software.
- VR-Simulator Graphic User Interface.
- Process Modelling Tool (PMT).
- Simulation Modelling Tool (SMT).
- Virtual Modelling Tool (VMT).
- Models Running Tools (MRT).
- Output Analyzing Tools (OAT).

Finally the interaction with models of the VR-Simulator software is also presented in this section.

4.2 THE VR-SIMULATOR ARCHITECTURE

The architecture of the VR-Simulator software is shown in Figure 4.1. It depicts the integration of the three main tools of the software and how the user can interact with them to develop a simulation model and virtual model of a manufacturing system.

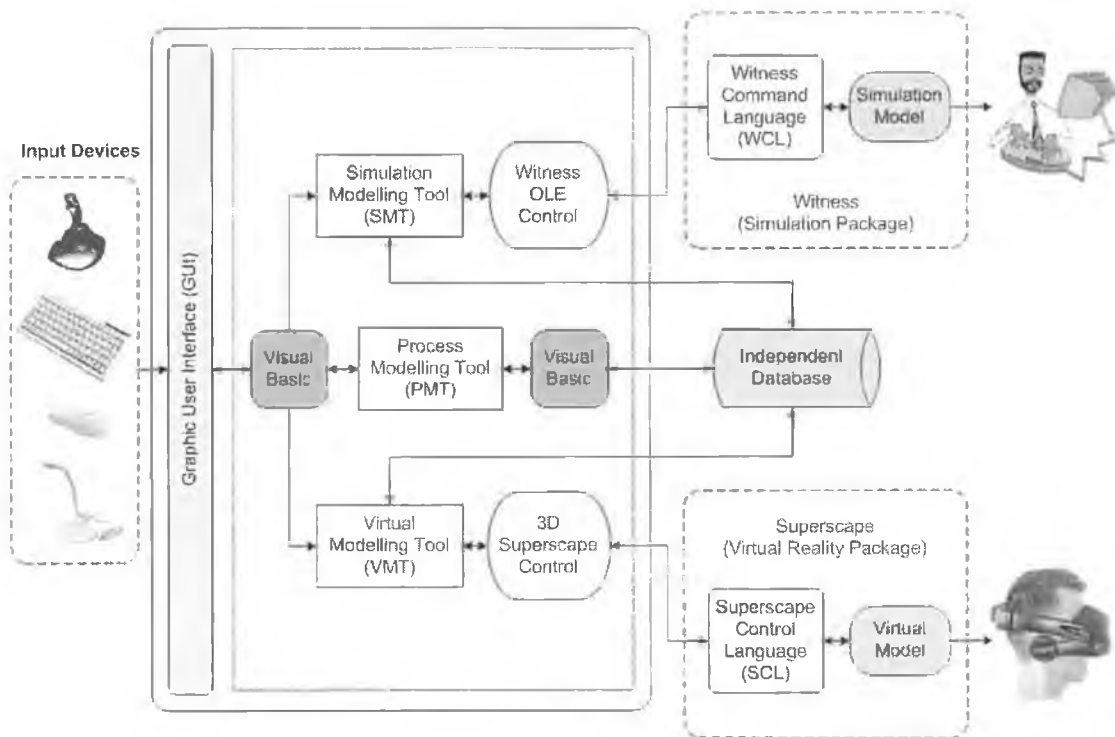


Figure 4.1: The VR-Simulator Software Architecture

As shown in Figure 4.1, the components of the VR-Simulator software include the following tools and devices:

1. **Process Modelling Tool:** The Process Modelling Tool (PMT) is the core of the VR-Simulator, which is used to obtain information regarding a manufacturing system to be stored in an independent database.
2. **Simulation Modelling Tool:** The Simulation Modelling Tool (SMT) is used to build a simulation model of a manufacturing system automatically.
3. **Virtual Modelling Tool:** The Virtual Modelling Tool (VMT) is used to create a virtual model of a manufacturing system automatically.

4. **Independent Database:** The independent database is the media used to store the collected information about a manufacturing system that needs to be addressed.
5. **Input devices:** Input devices can be used to interact with the VR-Simulator software and include mouse, keyboard, joystick and microphone.
6. **Output devices:** A monitor or Head Mounted Display (HMD) can be used as output devices to view the models generated by the VR-Simulator.

The following sections discuss the design and development of the components of the VR-Simulator software.

4.3 TOOLS USED TO DESIGN AND DEVELOP THE VR-SIMULATOR

The main tools that have been used to design and develop the VR-Simulator software are:

- Visual Basic (Object-based programming language).
- Witness (Commercial simulation package).
- Superscape VRT (Commercial Virtual Reality package).
- Microsoft Excel.

4.3.1 Visual Basic

Visual Basic has been used as the main tool to design and develop the VR-Simulator software due to the following considerations:

1. It is a powerful tool providing capability to produce custom libraries and objects that can be loaded at runtime or bound into the distributed application.
2. It is widely used by many researchers.
3. It is very flexible and user-friendly.
4. It is a reasonably simple language to program and test.
5. It supports OLE automation.

4.3.2 Witness

The Witness simulation software was chosen as the simulation tool to be used with the VR-Simulator software to generate simulation models of manufacturing systems. The Witness engine was used as the engine of the VR-Simulator to process the activities of the simulation models. The Witness software has been chosen as a simulation tool to be integrated with the VR-Software for the following reasons:

1. It is a well known commercial simulation package available on the market today.
2. It used as a tool by the most respected and successful organisations and industries across the world.
3. Based on a survey conducted by Vlatka Hlupic [128], the Witness package is the most used simulation software by industrial users and the second most used simulation software by educational users.
4. It has been proven as a successful tool to address real manufacturing systems.
5. It has all the required elements to build a manufacturing system.
6. It supports OLE automation.

4.3.3 Superscape VRT

Superscape VRT was chosen as the Virtual Reality software to implement and create the virtual models of manufacturing systems based on the following considerations:

1. It has its own editors to design and develop a fully interactive virtual model.
2. It has its own programming language called Superscape Control Language (SCL), which can be used to add behaviours to the objects within the virtual model.
3. It supports virtual reality devices (e.g. Head Mounted Display and Gloves).
4. It supports OLE automation.

4.3.4 Microsoft Excel

Microsoft Excel was chosen to store the database of manufacturing systems within data sheets for the following reasons:

1. Easy to use.
2. Data within the Excel sheets can be edited and modified easily.
3. Available on almost every PC.
4. It can be integrated with a great number of softwares.
5. It is used as a tool by most manufacturing companies to store and analyse their data.

The following sections provide more details of how these tools have been employed to design and develop the components of the software.

4.4 VR-SIMULATOR SOFTWARE DESIGN AND DEVELOPMENT

In designing the software, care was taken to meet both the user requirements as well as the architectural considerations. The development of the VR-Simulator incorporated the construction of Process Modelling Tool, Simulation Modelling Tool, and Virtual Modelling Tool. These tools were used in order to generate simulation and virtual models for studying manufacturing systems. Using the VR-Simulator software, one could:

- Describe the behaviour of manufacturing systems.
- Evaluate alternative hypotheses or theories based on the observed behaviour to find the optimum solutions.
- Use models to predict the future behaviour of a system.
- Measure the performance of real manufacturing systems.

4.4.1 VR-Simulator Software Architectural Issues

The Graphic User Interface (GUI) of the VR-Simulator software was developed entirely using an object-based programming language called Visual Basic. The VR-Simulator was designed to work in conjunction with the Witness package and the Superscape VRT software to allow non-expert users to develop simulation and virtual models automatically. The overall structural design of the VR-Simulator software is shown in Figure 4.2, and is described below:

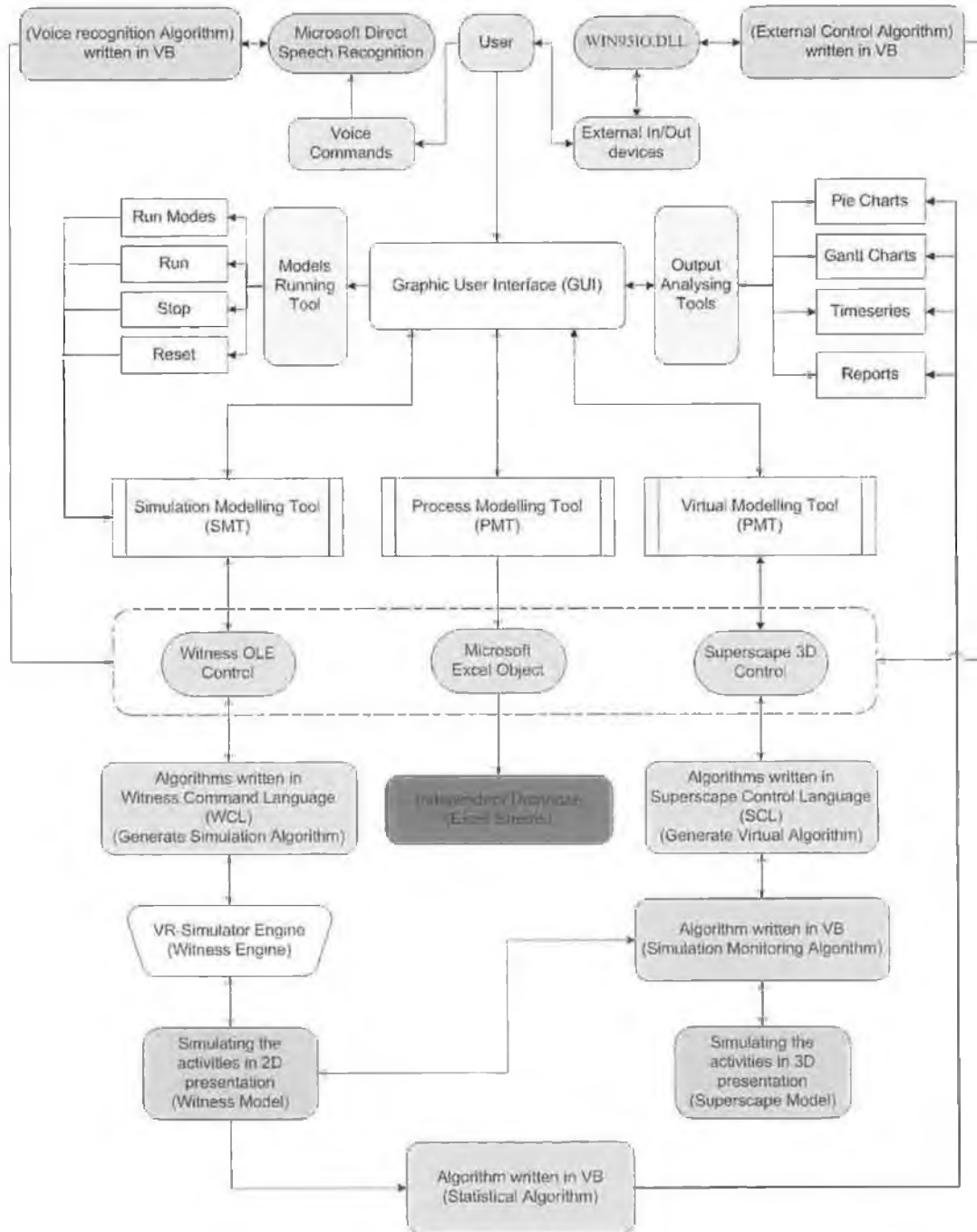
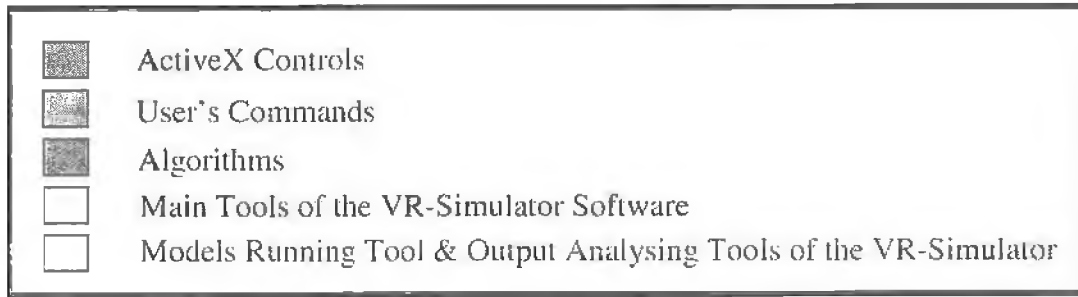


Figure 4.2: VR-Simulator Structural Design

1. The GUI of the VR-Simulator is the interface between the end user and the tools of the VR-Simulator, which includes PMT, SMT, VMT, MRT and OAT.
2. PMT is the tool to be used to gather information regarding the manufacturing system that needs to be addressed.
3. Excel files were used as an independent database to store the data of a manufacturing system.
4. SMT is the tool to be used to automatically generate a simulation model of a manufacturing system in the Witness software.
5. Witness OLE control was used to link the VR-Simulator with the Witness software to exchange data.
6. VMT is the tool to be used to automatically create a virtual model of a manufacturing system in the Superscape environment.
7. Superscape 3D control was used to link the VR-Simulator with the Superscape software to exchange data.
8. The output tools of the VR-Simulator were used to display the statistics of a manufacturing system to measure its performance.
9. Algorithms written in Visual Basic were used to enable the components of the VR-Simulator to communicate with each other to exchange data.
10. The Witness engine acts as the simulation engine for the VR-Simulator to process the data for simulating the activities of a manufacturing system in the simulation model and the virtual model.
11. Algorithms written in VB are used to translate voice commands of the user and the external input devices to actions within the simulation and virtual models.

4.4.2 Design of the Components of the VR-Simulator Software

The VR-Simulator was designed incorporating the architectural and user requirements as described in the previous section. The VR-Simulator consists of different components to allow users to perform various functions to support simulation-modelling and visualising the activities of manufacturing systems. As shown in Figure 4.3, these components are:

1. Graphic User Interface (GUI).
2. Process Modelling Tool (PMT).
3. Simulation Modelling Tool (SMT).
4. Virtual Modelling Tool (VMT).
5. Models Running Tool (MRT).
6. Output Analysing Tool (OAT).

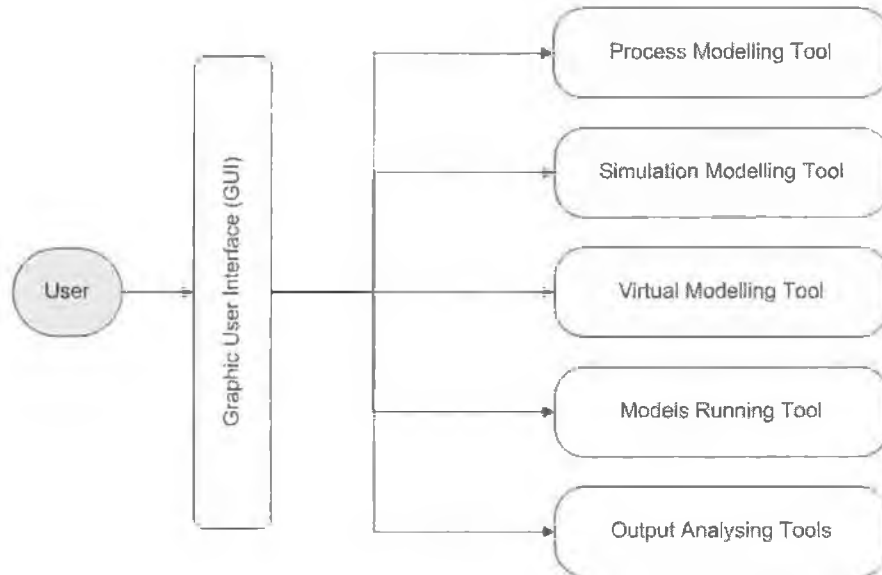


Figure 4.3: Main components of the VR-Simulator Software

The components of the VR-Simulator software communicate to each other through algorithms and functions written using VB. Figure 4.4 presents a diagram that shows how the components of the VR-Simulator software communicate to execute the user commands to develop simulation models and virtual models of manufacturing systems automatically and also to enable the user to do experiments with the generated models and monitor their performance.

As shown in Figure 4.4, the GUI of the VR-Simulator software is the interface between the user and the three main components of the software. These components are the Process Modelling Tool (PMT), the Simulation Modelling Tool (SMT) and the Virtual Modelling Tool (VMT). The user communicates with the PMT through the GUI of the VR-Simulator to define a manufacturing system. The PMT is linked to an Excel file to store the database of a manufacturing system defined by the user

or to load an existing database of a manufacturing system. The SMT uses the information that has been gathered using the PMT to generate a simulation model automatically, while the VMT uses the same information to generate a virtual model automatically. The SMT and VMT communicate to each other to simulate the activities of the simulation model and the virtual model in real time and also to enable the user to interact with the models in real time mode. Witness OLE Control, Superscape 3D control and Microsoft Excel Object are used as mechanisms to link the PMT, SMT and VMT with the GUI of the VR-Simulator software. The link between the GUI and the component of the VR-Simulator has been accomplished using algorithms and functions written in Visual Basic.

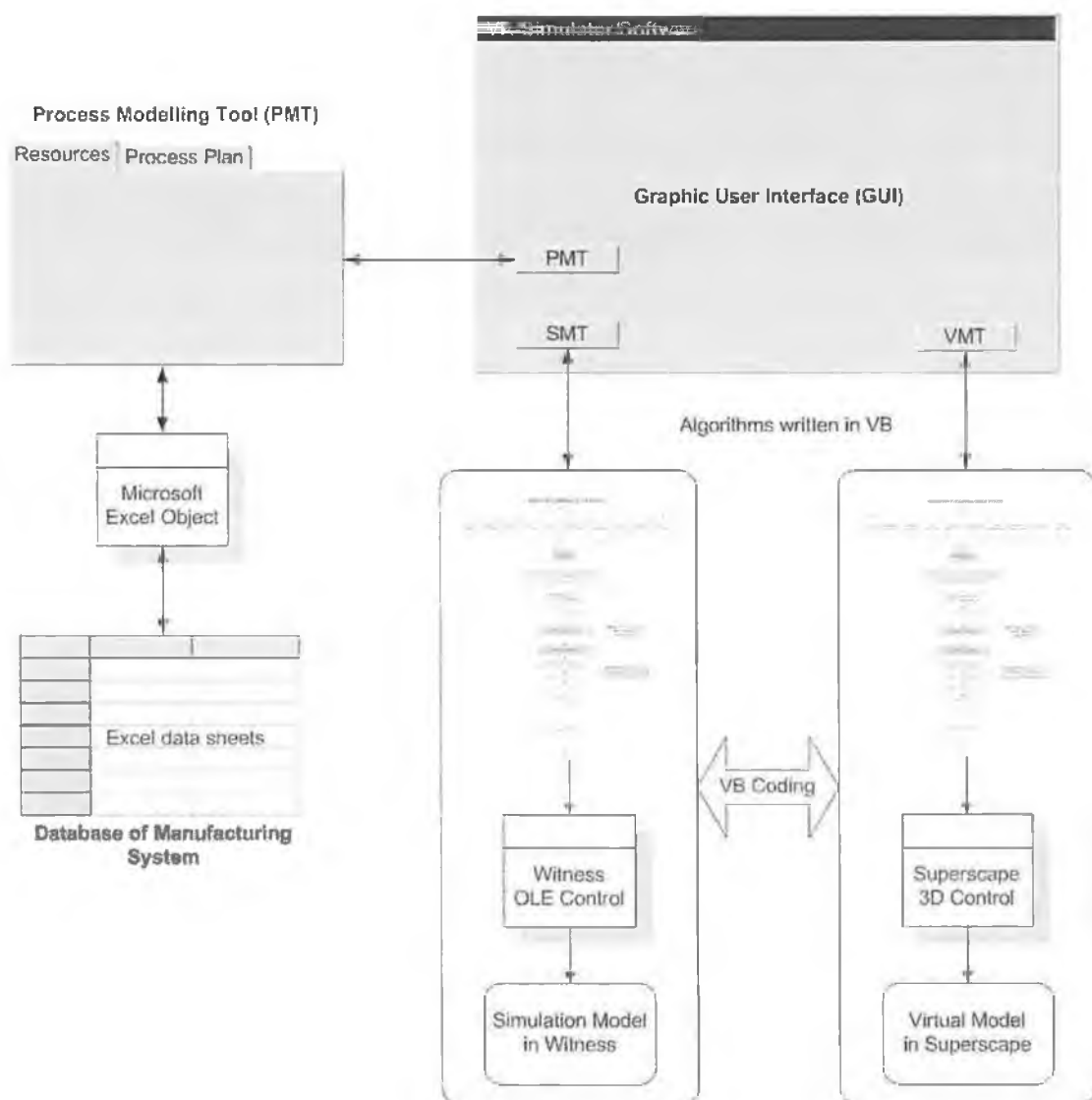


Figure 4.4: Diagram Shows How the Components of the VR-Simulator Communicate

4.4.3 The VR-Simulator Graphic User Interface

This is the main interface, where users can perform modelling activities to generate simulation and virtual models of manufacturing systems. The GUI is designed to be user-friendly so that non-expert personnel can use it without having extensive knowledge about modelling techniques and programming. The GUI operates as an interface between the user and the tools of VR-Simulator to develop simulation and virtual models, perform different experiments and analyse the outputs of the modelled manufacturing systems. As illustrated in Figure 4.2, one can access different functionalities of the VR-Simulator through this interface. The VR-Simulator functionalities are described in more details in the following subsections. Figure 4.5 represents the structural design of the graphic user interface of the VR-Simulator software.

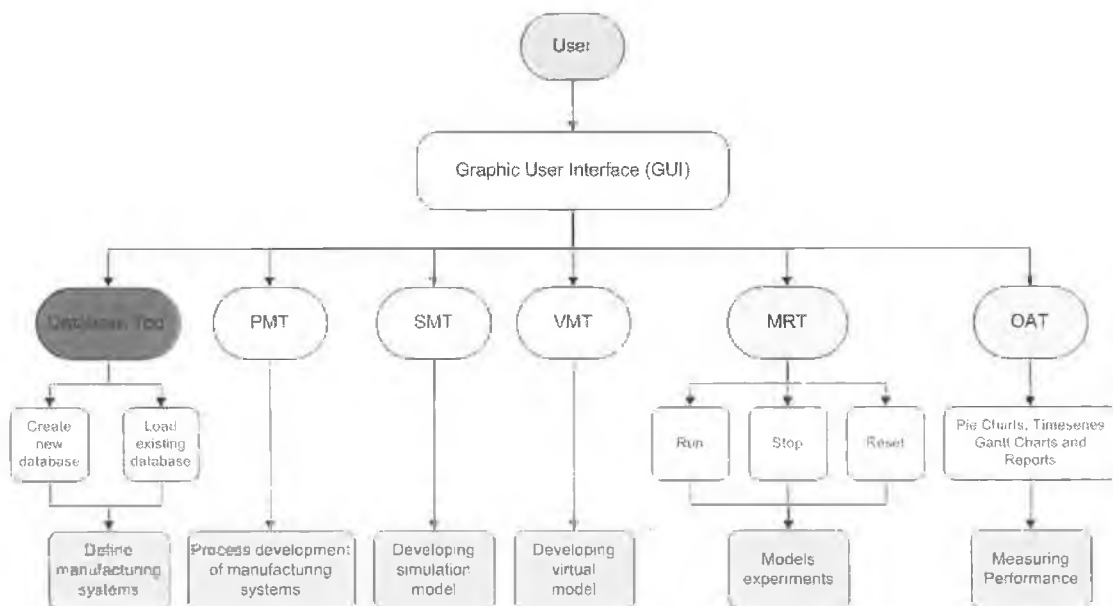


Figure 4.5: Graphic User Interface Structural Design

The GUI provides the user with a set of tools for studying manufacturing systems. These tools comprise a database tool, PMT, SMT, VMT, models running tool, and output analysing tools. The main menu of the GUI of the VR-Simulator software is shown in Figure 4.6.

The main menu of the VR-Simulator consists of seven options, which are File, Process Modelling Tool, Simulation Modelling Tool, Virtual Modelling Tool, Simulate, Output analysing tools, and Help. These options can be accessed either by clicking on the option name from the main menu of the VR-Simulator or by clicking on the icon that represents the option. Icons representing the main options of the tools of the VR-Simulator are used to enhance the interaction between the user and the GUI of the VR-Simulator. The attached CD of this thesis includes the entire code that has been written by the author in VB to develop the VR-Simulator software.




Figure 4.6: GUI of VR-Simulator Software

The following section discusses the options of the VR-Simulator main menu.

1. File:

By selecting “File” from the main menu, or pressing Alt + F, the following options will appear: Create New Database, Load Existing Database, Close Database, Save Database, Save Database As and Exit.

- **Create New Database:** This option can be accessed either from the file options or by clicking on this icon .

The function of this option is to create a new Excel file to hold the database of the manufacturing system that needs to be addressed. By selecting this option, an Excel file will be created automatically containing ten Excel data sheets (see Figure 4.7) to enable the user to define a new manufacturing system to be modelled. The data sheets are used to hold information about the resources and the process plan of a manufacturing system.

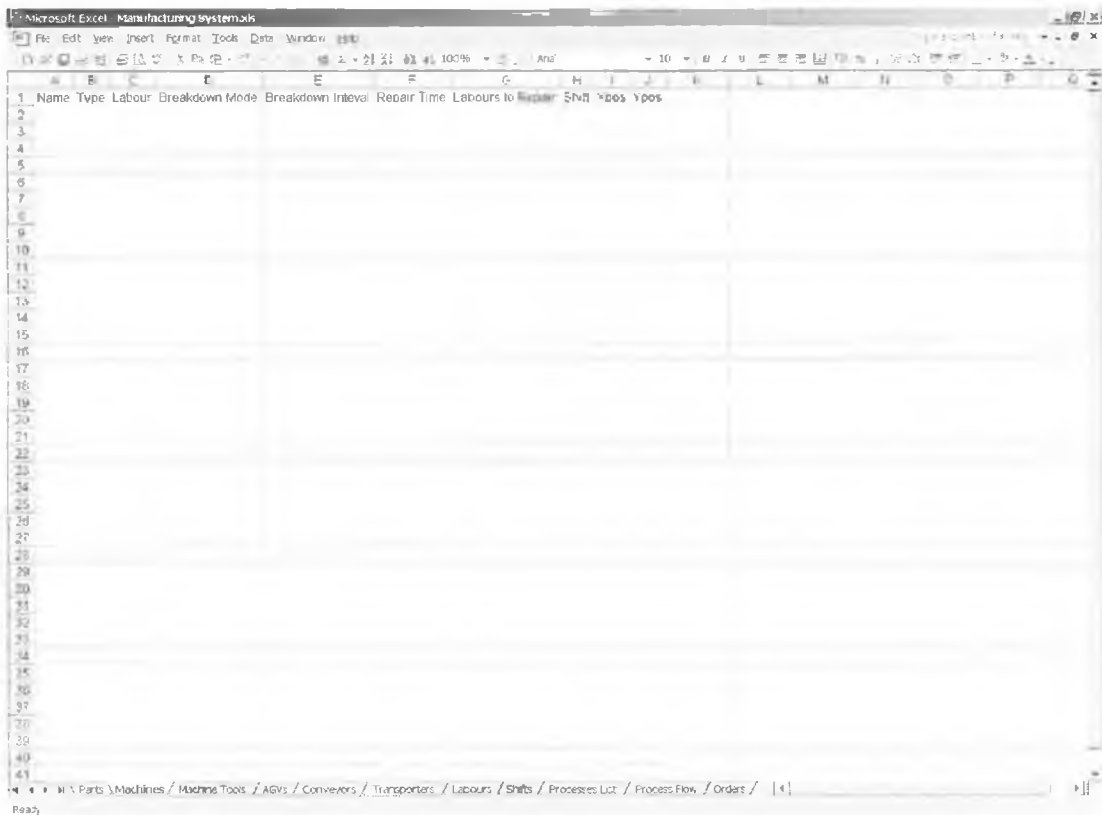



Figure 4.7: Excel File Created Automatically

- **Load an Existing Database:** This option can be accessed either from the file options or by clicking on this icon .

The function of this option is to load an existing database of a manufacturing system either to be modified or to add more data to it. By selecting the option “Load an

Existing Database”, an open window appears allowing the user to specify the name of the database file to be loaded as shown in Figure 4.8.

- **Close Database:** To close an open Excel file enabling the user to open a new database.

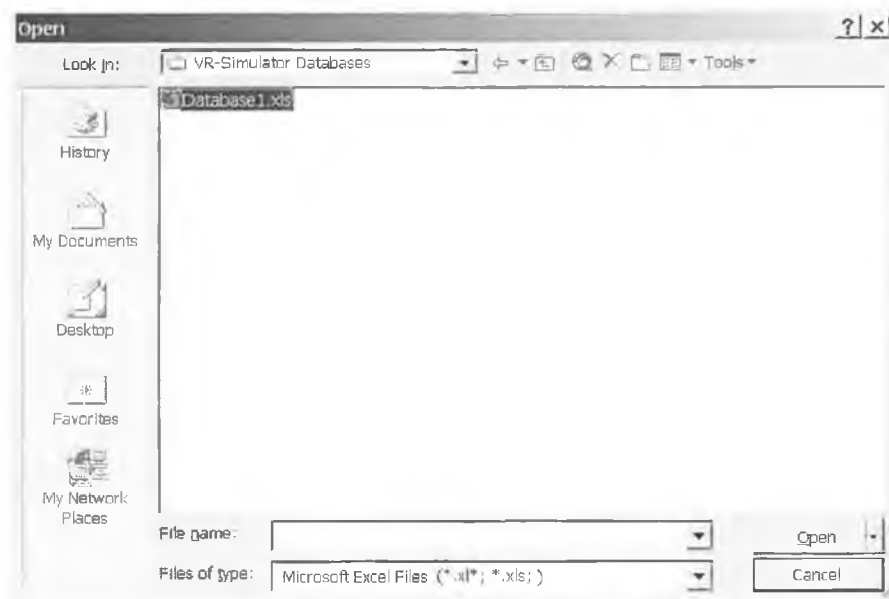


Figure 4.8: Open Window to Load a Database of a Manufacturing System




- **Save Database:** To save the database of the manufacturing system that has been created by the user in a file with the extension “xls” that represents an Excel file. If the file already existed, changes only will be saved, while if no file exists, a Save As window appears to enable the user to save the database under a file name he/she specifies.
- **Save Database As:** To save an existing database file under a new name to allow the user to modify the database for conducting different experiments without changing the original database (see Figure 4.9).
- **Exit:** This option can be accessed from the file menu options or by clicking on this icon , in order to exit the VR-Simulator software.



Figure 4.9: Save Database Window


2. Process Modelling Tool

By clicking on “*Process Modelling Tool*” from the main menu or pressing Alt + P, the following options will appear: Process Development and Factory Layout.

- **Process Development:** This option can be accessed from the Process Modelling Tool options or by clicking on this icon . The function of this option is show the process modelling tool window that needs to be used to define all the resources and the process plan of a manufacturing system. All the information entered by the user using the PMT window is saved within the data sheets of the created database under the file name that has been specified by the user.
- **Factory Layout:** This option can be accessed from the Process Modelling Tool options or by clicking on this icon . The function of this option is to show the window that can be used to define the layout of a manufacturing system. The factory layout window provides the user with a number of icons that represent the machines, material handling devices and storage areas to be assigned to different locations within the factory layout window to construct the layout of the model to be analysed.


3. Simulation Modelling Tool:

By clicking on “*Simulation Modelling Tool*” from the main menu or pressing Alt + S, the following option will appear:

- **Generate Simulation Model:** This option can be accessed from the Simulation Modelling Tool options or by clicking on this icon . The function of this option is to automatically generate a simulation model of a manufacturing system in the Witness software.

4. Virtual Modelling Tool:

By clicking on “*Virtual Modelling Tool*” from the main menu or pressing Alt + V, the following option will appear:

- **Generate Virtual Model:** This option can be accessed from the Virtual Modelling Tool options or by clicking on this icon . The function of this option is to generate a virtual model automatically in the Superscape environment.

5. Simulate:

By clicking on “*Simulate*” from the main menu or pressing Alt + t, the following options will appear: Run, Stop, Reset and Run Modes.


- **Run:** This option can be accessed from the Simulate options or by clicking on “Run” button, which is located at the bottom of the window as shown in Figure 4.6. The function of this option is to run both the simulation model and the virtual model to simulate the activities of the modelled manufacturing system. The “*Run*” option is activated only when a process plan has been loaded to the VR-Simulator.
- **Stop:** This option can be accessed from the Simulate options or by clicking on “Stop” button, which is located at the bottom of the window as shown in Figure 4.6. The function of this option is to halt the current simulation run until the “Run” is given again.

- **Reset:** This option can be accessed from the Simulate options or by clicking on “Reset” button, which is located at the bottom of the window as shown in Figure 4.6. The function of this option is to reset the simulation clock of the VR-Simulator to the start of the run. All the simulation elements are also set to the idle state and all the statistics are cleared.
- **Run Modes:** This option can be accessed from the Simulate options. The function of this option is to provide the user with three modes to run the simulation and virtual models. The modes include Real Time Mode, Speed Mode, and Batch Mode.

6. Outputs Analysing Tools:

By clicking on “*Outputs Analysing Tools*” from the main menu or pressing Alt + O, the following options will appear: Pie charts, Gantt chart, Timeserise and Reports. The output analysing tools are used to display the statistics of the simulation model to measure the performance of the modelled manufacturing system.

7. Help:

By selecting Help from the main menu and clicking Help, pressing Alt+H, or clicking on this icon , the user will be able to access the help file that provides instructions of how the VR-Simulator software operates.

4.4.4 Process Modelling Tool Design

The Process Modelling Tool (PMT) is the core component of the VR-Simulator software, which is used to capture the operational logic and gather information regarding a manufacturing system and then store the gathered data in an independent database. The database is used to provide the simulation modelling tool and the virtual modelling tool with this information to be used to generate a simulation model and a virtual model of a manufacturing system. The PMT is kept separate from the other tools to ensure the following benefits:

1. The analysis of the concepts and properties that are typical of a certain application can be postponed and carried out with an in-depth survey that can benefit from having a basic model already formalised.
2. The addition of new applications does not affect the basic process model and hence it does not require modification to the Process Modelling Tool data structures and functions.

Every manufacturing system comprises products and the facilities which are used to produce them, such as machines, operators, handling devices, tools and so on. Consequently, simulation models of manufacturing systems will have common features. The difference between them reflects the different ways in which the various facilities are combined to form a particular system. The PMT collects information from the user to define the resources and the process plan of the manufacturing system that needs to be studied. Figure 4.10 shows the main elements of manufacturing systems that can be defined by the PMT.

Resources			Process Plan
Parts	Shifts	Buffers	Process Flows
Machines	Machine Tools	Labourers	
AGVs	Trasnporters	Conveyors	
			Orders

Figure 4.10: Main elements of a manufacturing system

The resources that can be defined by the PMT include parts, machines, AGVs, conveyors, transporters, labourers, machine tools, buffers, and shifts. These resources are used to represent a manufacturing system, where the logic of the system can be applied by the process plan that consists of the process flows and the orders.

4.4.4.1 Process Modelling Tool Presentation

The Process Model Tool provides a graphic interface that allows non-expert users to define the resources and the shop floor activities of a real or hypothetical manufacturing system to be analysed by the VR-Simulator software. This section aims to demonstrate how to use the PMT interface to define a manufacturing system.

The GUI of the PMT provides several dialogs on a single form interface to identify the resources and the process plan of a manufacturing system, where each resource has its own detail dialog page that can be used to specify its attributes. The process flows and the orders that represent the process plan of a manufacturing system can be defined using this interface to make it easier for the user than having more than one interface. The detail dialog of each element of the PMT can be accessed by clicking on the tab name of the element or the icon that represents it.

The reminder of this section describes how the detail dialog pages of the PMT interface can be used to define the resources and the process plans of manufacturing systems.

▪ Part:

A part is any component, raw material or semi finished product involved in a shop floor process. It is the result of a purchasing or processing activity. Input and output materials of every operation are considered as parts. This resource represents the product that needs to be manufactured or processed by the other resources. Figure 4.11 shows the detail dialog page that needs to be used to identify the Parts of a manufacturing system.

The user can define a new part by writing its name in the text box, which is located underneath the label “*Name*”. By clicking the button “*Add*”, the name of the new part will be added in the parts list as shown in Figure 4.11. The “*Add*” button is activated only when the name of a part is specified by the user. The “*Delete*” button is used to delete any selected part within the list as well as its attributes. If more than

one part is defined, the buttons “*Move Up*” and “*Move Down*” can be used to arrange the parts names within the list. Attributes can be added to be associated with the parts to represent their characteristics, where each part can have more than one attribute to distinguish them once they are under processing. The attributes can be used to represent weight, length, size, etc.

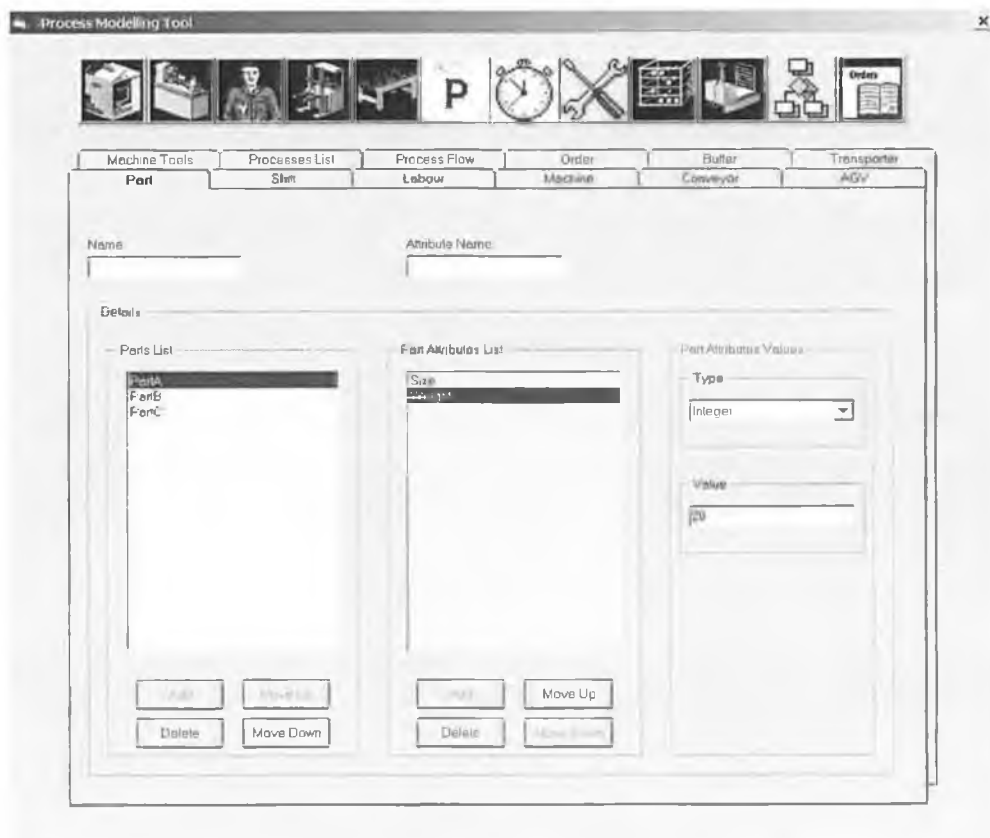


Figure 4.11: Part Detailed Dialog Page

There are three types of attributes that can be assigned to the parts. These include integer, real, or string, which can be introduced based on the function of the attribute. The types can be selected from the types list, and the value of each type can be written in the text box of the value. A function was written in VB to verify if the user has assigned the right value to the defined attribute type. If the entered value does not match the attribute type, a message appears to ask the user to re-enter the correct value.

▪ Machines

Machines are powerful elements that are used to take parts from somewhere, process them and send them to the next destination. The detail dialog page for detailing the machines is shown in Figure 4.12. The user can specify the machines to be included for the manufacturing system that needs to be studied using the machine detail dialog page. “Add”, “Delete”, “Move Up”, and “Move Down” buttons are used to introduce the machines as explained previously in defining the parts. There are four attributes that can be applied to the machines. These include type, labour, breakdown, and shift.

Type: Machine types can be assigned to any newly introduced machine, and selected from the machine types list, which include drilling machine, saw machine, polishing machine, packaging machine and generic machine.

Labour: Up to 10 labourers can be assigned to handle each machine either for operating, setups, repairing, or any other process associated with the machine. The number of labourers can be selected from the labourers’ names list.



Figure 4.12: Machine Detailed Dialog Page

Breakdown: The breakdown option represents the condition of the machine once it is out of service. The breakdown option can be applied to the machines by identifying the following parameters: breakdown mode, breakdown interval, repair time, labourers names, and quantity. Here is a description of these parameters:

- ***Breakdown mode:*** There are three breakdown modes that can be assigned to the machines. The breakdown modes are “*Available Time*”, “*Busy Time*”, and “*Operation*”. The available time mode keeps track of the amount of time that the machine has spent in any state except off-shift. The machine will breakdown when the specified interval of this time has elapsed. This mode should be used only if the interest is to model machines that need maintenance whether they are in use or not. “*Busy time*” mode tracks the amount of time that the machine has spent in a busy state. The machine will breakdown when the specified interval of this time has elapsed. This mode should be used only if the user is looking to model a machine that breaks down while it is running. The last mode is the “*operation*” mode, which is used to breakdown a machine based on the specified number of operations the machine has executed. This mode can be used only to model a machine that breaks down regularly while in use.
- ***Breakdown Interval:*** It is used to determine the interval between successive breakdowns for each of the “breakdown” modes. The “breakdown interval” should be entered as time between failures for the available time and busy time modes, whereas for the operation mode, the breakdown interval should be entered as number of operations.
- ***Repair Time:*** This time should be entered by the user to specify the amount of time needed to repair a machine that has brokedown.
- ***Labourers Names:*** Required to specify the name of the labourer who can handle repair the brokendown machine. A list of the labourers’ names is generated automatically based on the labourers that have been defined by the user. The first labourer name on the list is called “*any*”, that means any labourer can do the repairing. The repair cannot occur until the required labourer is available.

- **Quantity:** It is used to specify the number of labourers needed to repair the machine

Shift: A shift can be applied to the machine to specify its working and non-working periods. The user can select the name of the shift from the shifts list, which is generated automatically based on the shifts that have been defined by the user.

▪ **Labourers**

This resource can handle machines, AGVs, conveyors, and transporters for processing, setups, repairing, cleaning and so on. Figure 4.13 shows the detail dialog page for defining the labourers. The user can define the labourers required to handle a manufacturing system using the detail dialog page. The name of the labourer needs to be provided, then the user can use the “Add”, “Delete”, “Move Up”, “Move Down” buttons to create a list of the labourers names. Three attributes can be assigned to the labourers, which include type, quantity, and shift.

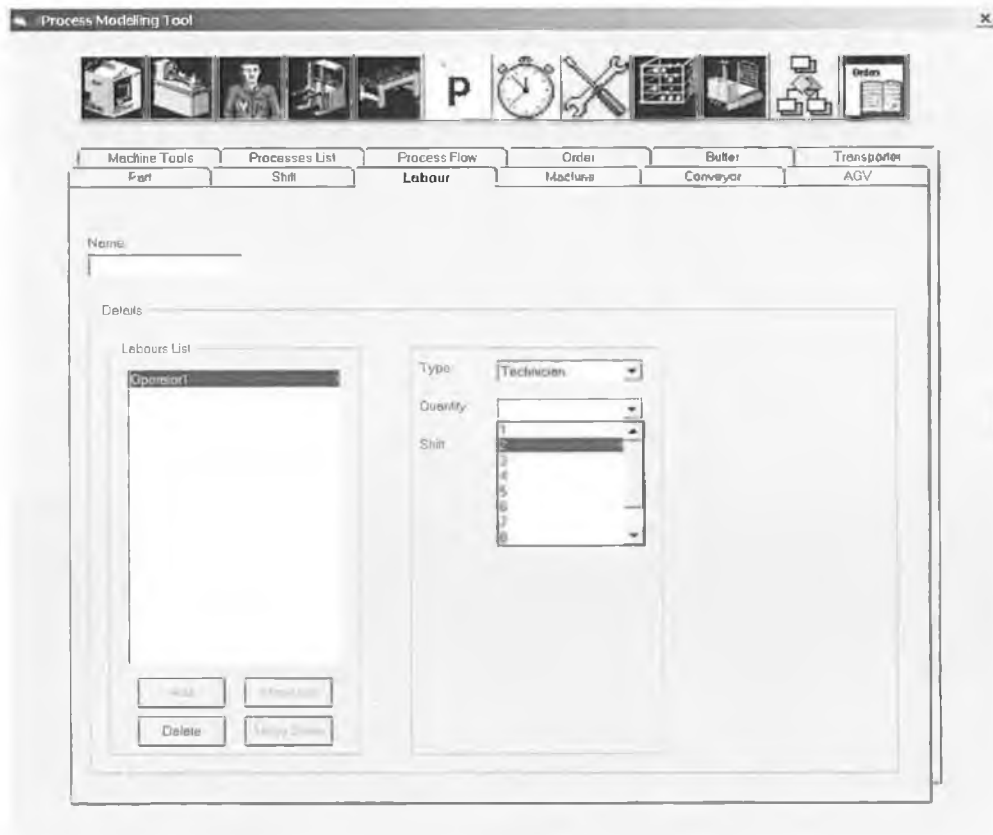


Figure 4.13: Labourers Detail Dialog Page

Type: It is used to define the type of job the labourer can handle. There are three types of labourers that can be applied (technician, driver, labourer).

- ***Technician:*** Defines a labourer that can handle the machines, AGVs, conveyors and the transporters.
- ***Driver:*** Defines a labourer that can drive the AGVs or transporters.
- ***Labourer:*** Defines a labourer that can handle any job type within a manufacturing system.

Quantity: Once a labourer is defined, the quantity can be assigned to identify the number of labourers who are sharing the same name and type.

Shift: A shift can be applied to the labourers to specify its working and non-working periods, which include start working time, finish working time, and break times (e.g. coffee break). The user can select the name of the shift from the shifts list, which is generated automatically based on the shifts that have been specified by the user.

▪ **AGVs**

Automated Guided Vehicles (AGVs) are material handling systems that can be used to deliver parts from one location to another within a manufacturing system. The parts can be raw materials, processed parts, or finished products. Figure 4.14 shows the detail dialog page for defining the AGVs.

A machine instead of a vehicle and a track has been used to model the AGV resource of the PMT because the track in Witness has only a loading point and unloading point that can not be used to accomplish the functions of the AGV in the simulation model. Also, tracks and vehicles do not support breakdown and repair modes which are needed to model the behaviour of the AGVs. Predefined paths were used to model the paths in which each AGV has to follow when delivering parts between the resources of a manufacturing system. A name for the AGV needs to be specified first, then the “Add”, “Delete”, “Move Up”, and “Move Down” buttons are used to create a list of the AGVs for a manufacturing system. The attributes that can be assigned to any defined AGV are capacity in parts, speed, breakdown, and shift.

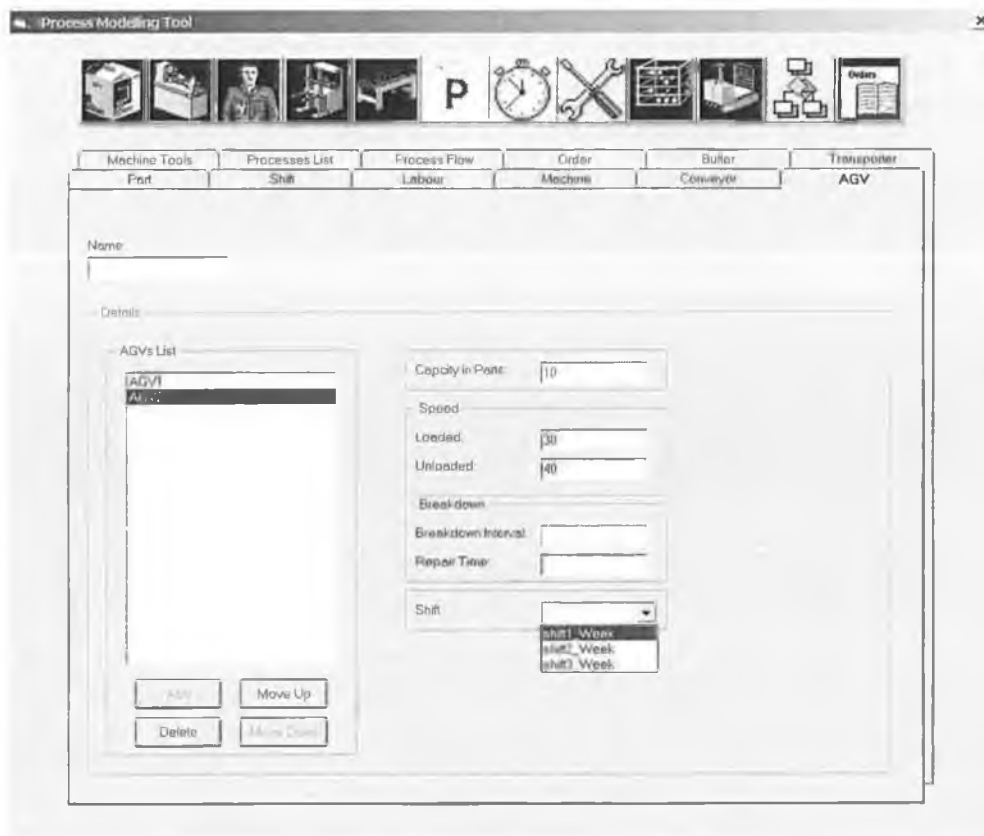


Figure 4.14: AGVs Detail Dialog Page

Capacity in Parts: The value of the capacity in parts needs to be entered to specify the maximum capacity in parts that the AGV can deliver at one time.

Speed: Specifies the speed of the AGVs with or without load.

- ***Loaded:*** To be entered to specify the speed of the AGV when loaded.
- ***Unloaded:*** To be entered to specify the speed of the AGV without load.

Breakdown: Represents the condition of the AGV once it is out of service. The breakdown of the AGVs can be defined by identifying the breakdown interval and repair time.

- ***Breakdown Interval:*** Determines the interval between successive breakdowns.
- ***Repair Time:*** Specifies the time needed to repair the AGV.

Shift: Specifies the AGVs working and non-working periods. The user can select the name of the shift from the shifts list.

▪ Transporters

The transporters can represent any material handling systems (e.g. forklift, trolley, etc). The function of the transporters is to deliver the parts from one location to another within a manufacturing system. Figure 4.15 shows the detail dialog page for defining the transporters. A name for the transporter needs to be specified first by the user, then the “Add”, “Delete”, “Move Up”, and “Move Down” buttons are used to create a list of the transporters for a manufacturing system. The transporters are used to represent the time required to deliver parts from one location to another. Once all the transporters are defined, the time required to deliver the parts between the locations of a manufacturing system needs be specified using the Factory Layout Window which is to be discussed later in the coming sections.



Figure 4.15: Transporters Detail Dialog Page

▪ Conveyors

Conveyors are used to move parts form one fixed point in the manufacturing system to another over time. The detail dialog page as shown in Figure 4.16 is used to define

the conveyors. There are four parameters that can be applied to the conveyors: length in parts, movement index time, breakdown and shift.

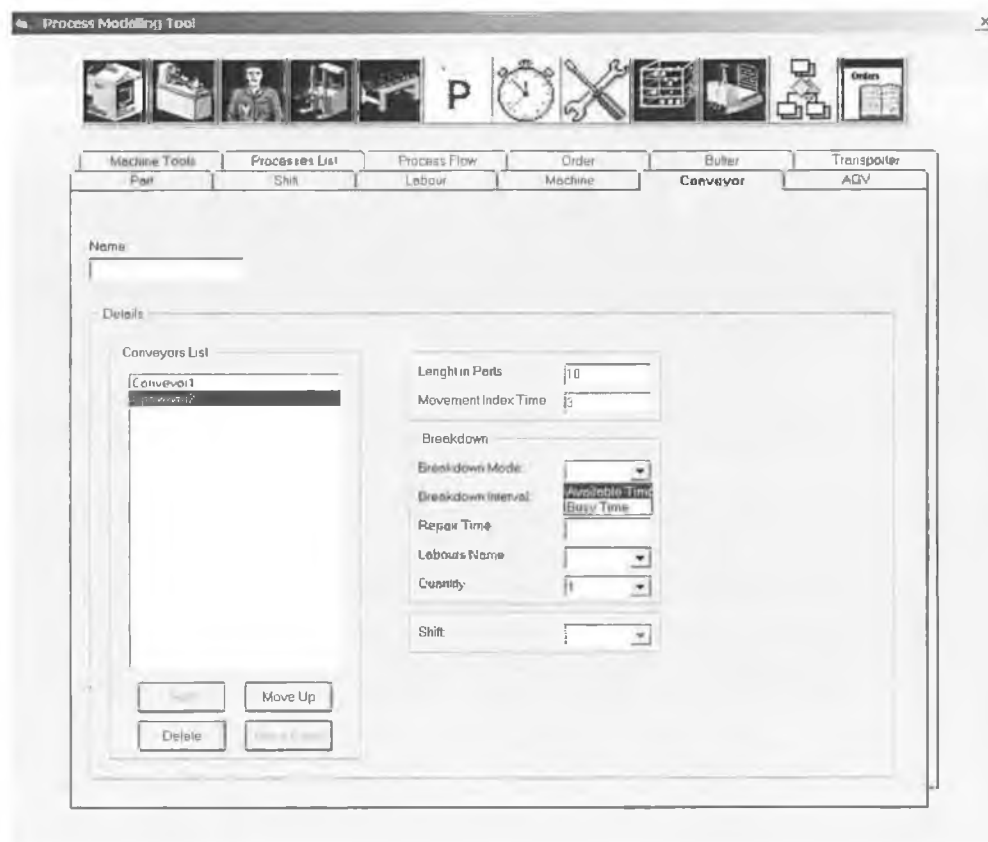


Figure 4.16: Conveyor Detail Dialog Page

Length in Parts: This value needs to be entered by the user to specify the length of the conveyor, measured in parts (assuming that each part only takes up 1 position of the conveyor).

Movement Index Time: It is used to specify the time that takes a part to move through one part length of the conveyor.

Breakdown: The breakdown parameter represents the condition of the conveyor once it is out of service. The breakdown parameter of the conveyors can be applied by identifying the following parameters: breakdown mode, breakdown interval, repair time, labourers names, and quantity. These parameters are similar to those for the machine object (see page 114) except that there are only two breakdown modes for the conveyors, namely “Available Time” and “Busy Time”.

▪ Shift

A shift is a logical element that can be used to create a shift pattern or a series of shift patterns, which are in effect a sequence of working and non-working periods. Shifts can be applied to the following resources: machines, labourers, AGVs and conveyors. Figure 4.17 shows the detail dialog page for defining the shifts. A name for the shift needs to be specified to define the main shift pattern that can be applied to the other resources which include machines, labourers and AGVs. The buttons “Add” and “Delete” are used to define new shifts or delete created shifts. This needs to be done to create the list of the shifts. The buttons “Move Up”, and “Move Down” are used to organise the list of the shifts that can be applied to the resources of a manufacturing system. The main shift includes three patterns, which are Mondays-Fridays, Saturdays, and Sundays. Start time and finish time should be specified for each pattern.

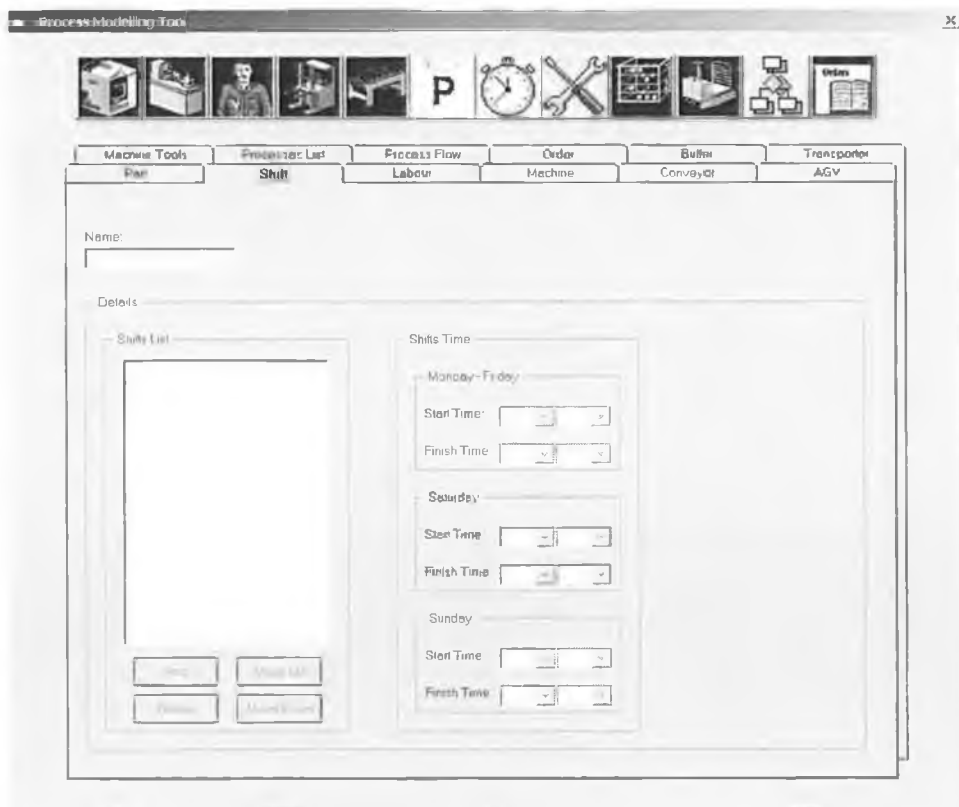


Figure 4.17: Shift Detail Dialog Page

▪ Machine Tools

Machine tools can be defined to be used either by the labourers or the machines to handle the processes of the machines. The detail dialog page as shown in Figure 4.18, can be used to define the machines tools. After specifying the name of the machine tool, the buttons “Add”, “Delete”, “Move Up” and “Move Down” can be used to create a list of the machine tools. The two attributes that can be assigned to the machine tools are type and quantity.

Type: Two types can be applied to the machine tools, which include cutting tools and hand tools.

Quantity: Quantity can be used to specify the number of each tool type available.



Figure 4.18: Machine Tools Detail Dialog Page

▪ Processes List

The processes list is not a resource as it represents the processes of a manufacturing

system that needs to be analyzed by the VR-Simulator. A list contains all the processes names must be defined by the user using the processes list detail dialog page as figure 4.19 shows. The name of the process can be cleaning, maintenance, coating, etc., where each name refers to a different process. The list of the processes names can be created by specifying the name of the process, then the “Add”, “Delete”, “Move Up”, and “Move Down” buttons are used to create and organize the processes list. A different colour needs to be assigned to each process for analysing the processes using the output tools of the VR-Simulator.

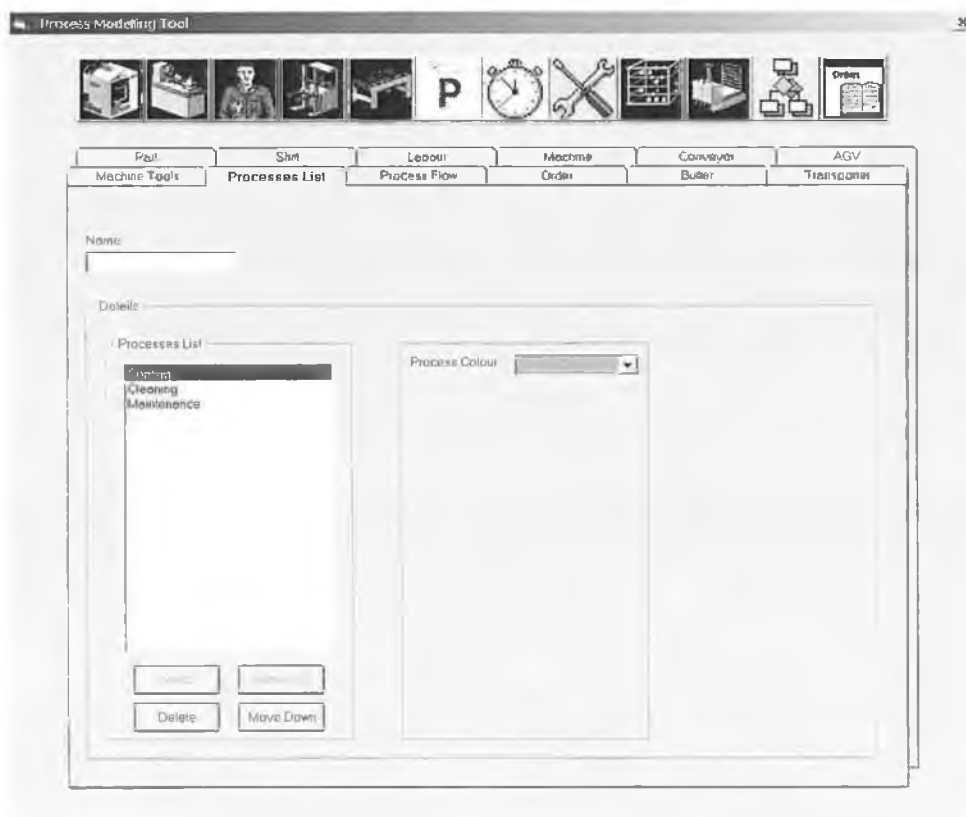


Figure 4.19: Processes List Detail Dialog Page

▪ **Buffers**

A buffer is a resource that stores parts. Buffers cannot pull parts in; another resource must push parts into them. Buffers can use a buffer exit rule to push parts out, or another resource can pull parts out of it. The buffers can be used to represent the following:

- A skip holding parts that are waiting for their next operation.

- A hopper at an assembly station containing product components
- A store or a warehouse.

Figure 4.20 shows the detail dialog page that needs to be used to define the buffers of a manufacturing system.

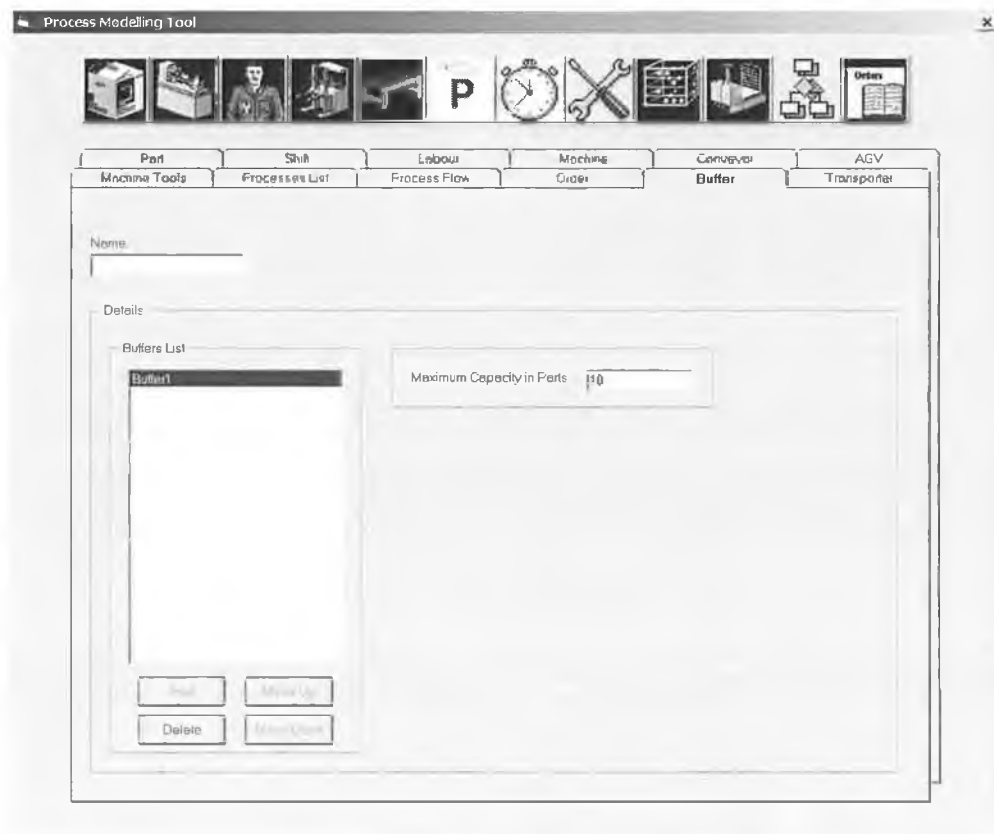


Figure 4.20: Buffers Detail Dialog Page

By specifying the name of the buffer, the buttons “Add”, “Delete”, “Move Up” and “Move Down” are used to create a list of the buffers. The attribute that can be assigned to the buffers is maximum capacity in parts.

Maximum Capacity in Parts: This value specifies the maximum capacity in parts of each buffer

▪ **Process Flow**

As mentioned earlier, the process plan of the PMT comprises the process flow and

orders. The process flow detail dialog page (Figure 4.21) shows how each part of a compiled list is to be processed or manufactured according to a job routing that predetermines the sequence of the processes. Here is a description of how one can use the elements of the process flow detail dialog page.

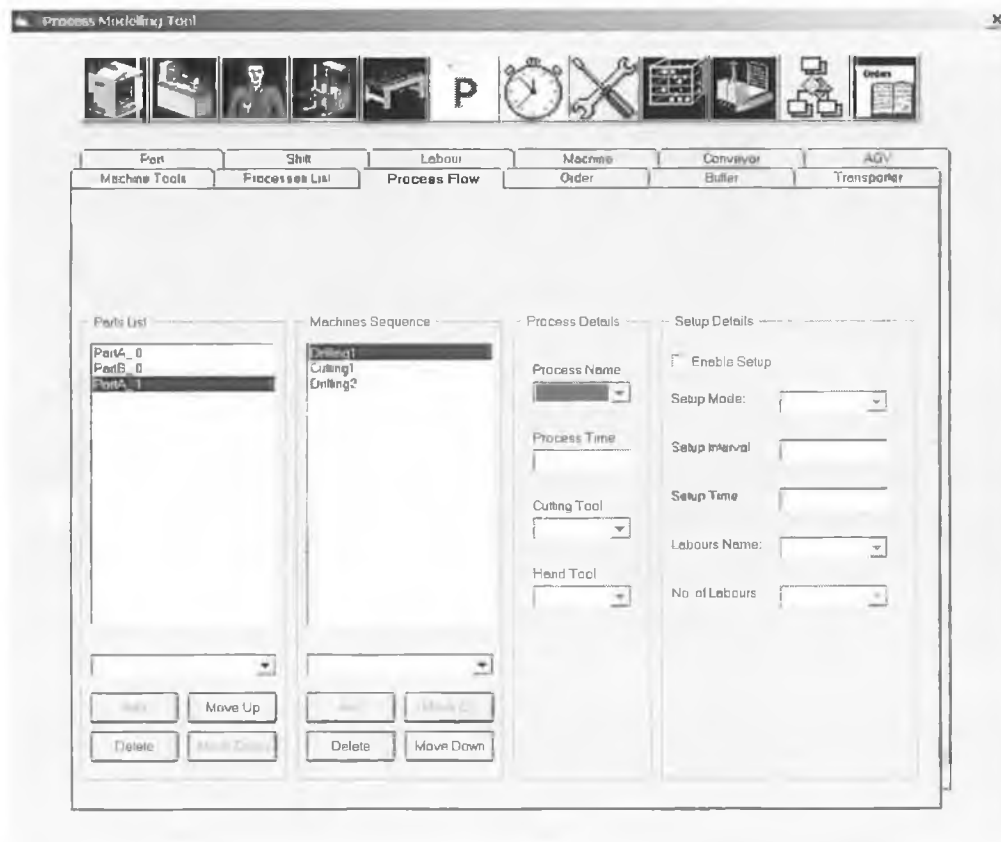


Figure 4.21: Process Flow Detail Dialog Page.

Parts List: A list of parts to be processed or manufactured can be compiled using the buttons “Add”, “Delete”, “Move Up” and “Move Down”.

Machines Sequence: The user needs to create a list for the machines that define the sequence of how each part can be manufactured. The available machines list can be used to select the name of the machine. The “Add” and “Delete” buttons are used to create the list of the machines sequence. The “Move Up” and “Move Down” buttons are used to specify the sequence of the machines that each part has to follow to be manufactured.

Process Details: The user needs to provide details for each machine process by defining the following parameters:

- **Process Name:** The process name can be selected from the process names list (e.g. cleaning, coating, etc.). The process name list is generated automatically based on the processes that were predetermined by the user.
- **Process Time:** To be provided by the user to specify the time needed to process the part.
- **Cutting Tool:** The user can select the appropriate cutting tool from the cutting tools list if a machine requires a tool to process a part. The cutting tools list is generated automatically based on the cutting tools defined by the user.
- **Hand Tool:** To be defined to be used by the labourers. The user can select the hand tool from the hand tools list, which is generated automatically based on the hand tools that have been defined by the user.

Setup Details: The user can apply a setup operation to the process by checking the box “enable setup”. The setup operation can be applied to a process which needs to be setup before processing a part. The setup operation can be used to change the machine tool, cleaning, etc. The following parameters should be provided by the user to apply the setup operation to a process:

- **Setup Mode:** There are two types of modes that can be assigned to the machines, which are: “operations mode” and “part change mode”. The “operations mode” refers to a setup that takes place after the specified number of operations, where an operation is regarded as a full cycle of the machine. The “part change mode” refers to the setup that occurs if the part about to be processed is of a different type to previously process part, that is, after the new part’s input into the machine but before the first machine cycle. This is useful as different parts need different tools.
- **Setup Interval:** The value of the setup interval is required to be entered by the user to specify the time between setups. The setup interval can be specified only for the “operations mode” to define the number of operations (or machine cycles) that need to take place between setups.

- **Setup Time:** To specify the time required to complete the setup.
- **Labourer's Name:** The user can select the labourer name from the labourers' names list to specify the labourer who can handle the setup. Setup can not occur until the required labourer is available.
- **No. of Labourers:** The user can specify the number of labourers required for the setup from the number of labourers list.

▪ Orders

The orders represent the other element of the process plan, which can be used to specify the number of parts that should be manufactured or processed. The orders can be defined by using the orders detail dialog page as shown in Figure 4.22. The “Add” and “Delete” buttons can be used to create a list of the orders. The “Move Up” and “Move Down” buttons are used to sequence the orders list. Each order should include a part name and a lot size.



Figure 4.22: Orders Detail Dialog Page

Part Name: Once an order is defined, the user can select a part name from the parts names list that is generated from the information provided by the user within the process flow detail dialog page.

Lot Size: To specify the number of selected parts to be manufactured.

This section discussed how the user can use the Process Modelling Tool to define the resources and the process plan of a manufacturing system. Figure 4.23 demonstrates the relationships between the elements of the PMT. The orders contain information about the parts, which include the quantity of products that need to be manufactured. Parts are processed by machines (e.g. drilling, saw, polishing, etc.). AGVs, conveyors, transporters are the material handling systems that are used to move the parts between the resources. Labourers are used to handle the machines as well as the material handling devices. Shifts, setup, and breakdowns can be applied to model the interpretations of the resources. The buffers represent the storage areas of a manufacturing system.

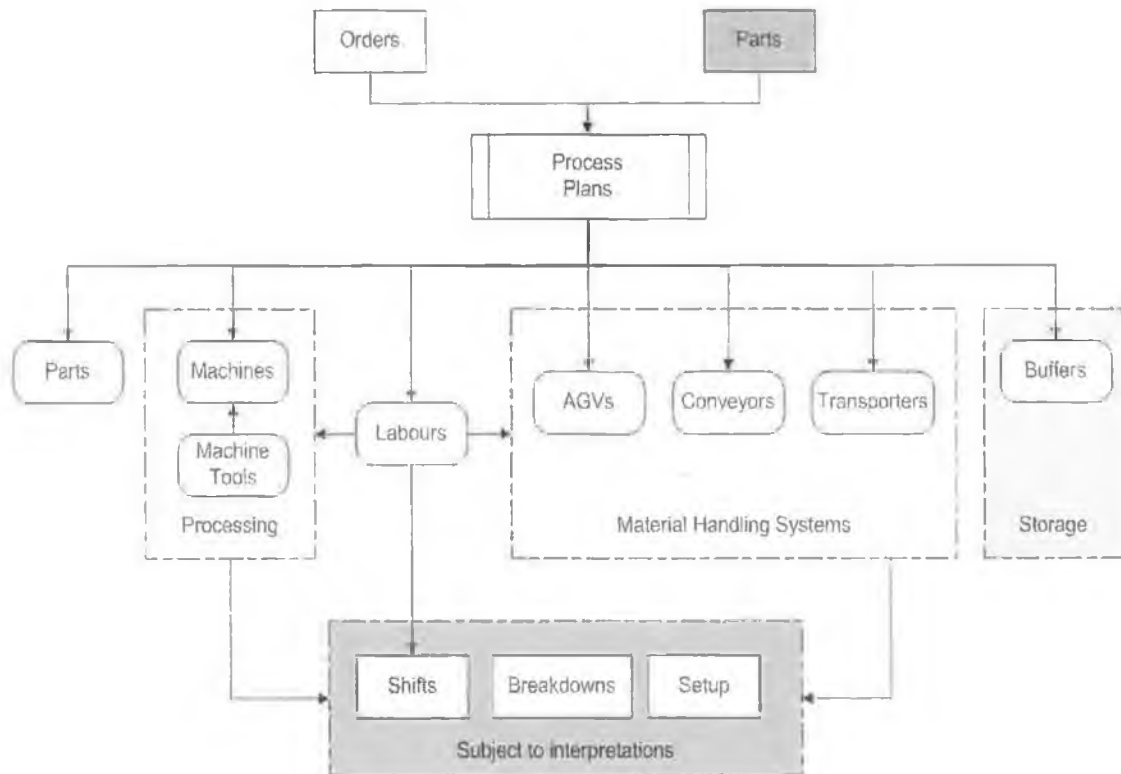



Figure 4.23: Relationships Between the Elements of the PMT

4.4.4.2 Factory Layout

The factory layout window can be accessed from the Process Modelling Tool options of the VR-Simulator software or by clicking on this icon . Once all the resources of a manufacturing system have been defined using the PMT, the factory layout window can be used to construct the layout of the system that needs to be studied. The factory layout of the VR-Simulator is fixed where the user can define up to 13 machines, 25 buffers, a warehouse and 12 material handling devices. Figure 4.24 shows the window of the factory layout that allows the user to construct the layout of a manufacturing system.

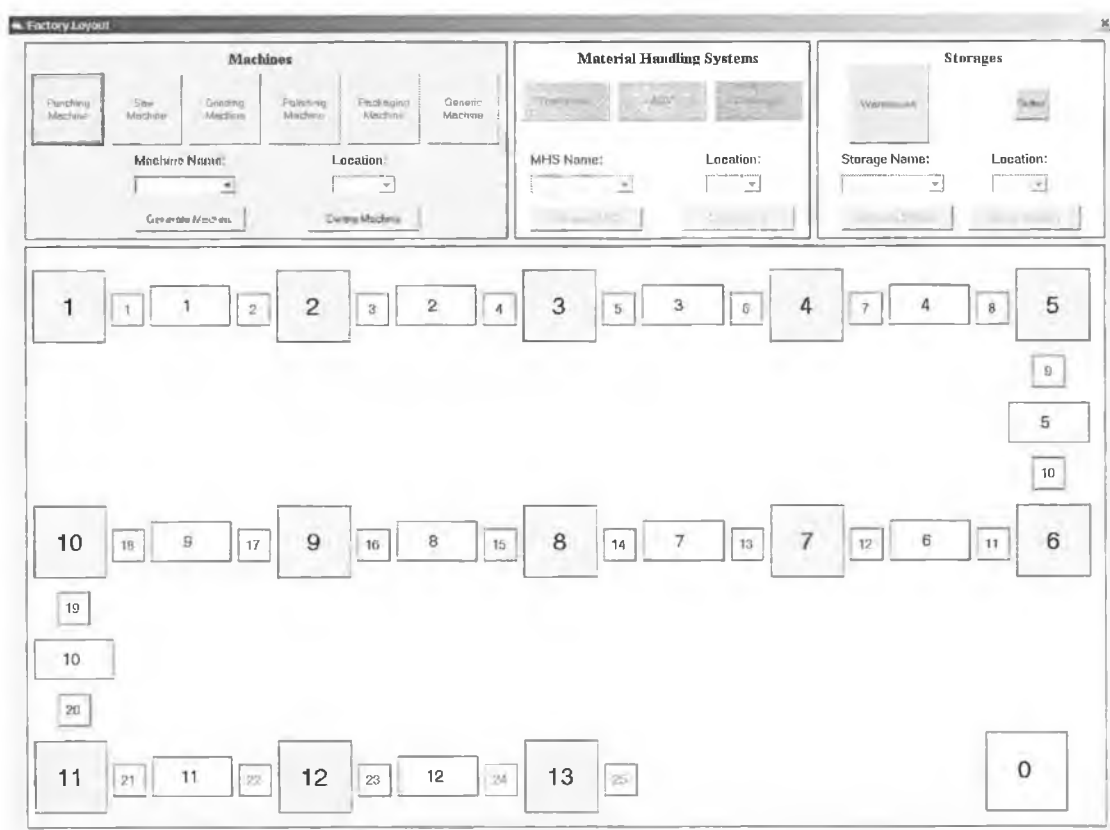


Figure 4.24: Factory Layout of the VR-Simulator Software

As shown on the window of the factory layout, there are three main types of resources that can be used to define the layout of a manufacturing system. These types include machines, material handling devices and storages. The machines resources include a drilling machine, a sawing machine, a polishing machine, a

packaging machine, a grinding machine and a generic machine. The material handling devices include a transporter, an AGV and a conveyor. The storage resources include a buffer and a warehouse.

The user can define the position of each resource to be placed within the factory layout window, first by specifying the name of the resource from the list box which contains all the names of the resources that have been defined by the user and then specifying the location where the resource should be placed. As shown in Figure 4.24, numbers located within boxes with different sizes and colours are used to distinguish between the resources types and their locations to make it easier for the user to construct the layout of a manufacturing system.

As mentioned earlier, the transporters are detailed using the Factory Layout window. Once the user specifies the name of the transporter and the location to be occupied, he has to double click on the transporter icon to identify the required time to deliver the parts between the two predefined locations as shows in Figure 4.25.

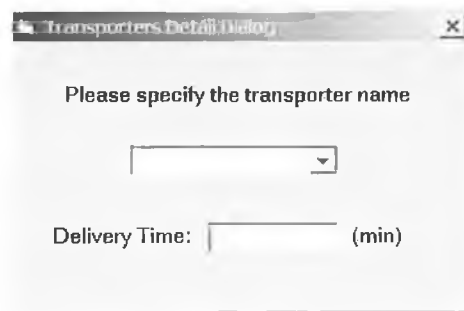


Figure 4.25: Transporters Detail Dialog

To avoid erroneously identifying the same resource in more than one location, functions were written in VB to verify the data entered by the user to make sure that the factory layout of the constructed manufacturing system is error free. For example, the “*Generate Machine*” button is enabled only if the machine name has not been defined before and the location of the machine is not occupied by another machine. The user can remove any resource from the window of the Factory Layout just by specifying the name of the resource and clicking on the “*Delete Resource*” button.

The main function of the Factory Layout window is to identify the x and y positions of each resource which are required to feed the SMT and the VMT with the locations of the resources to generate the layout of the simulation model and the virtual model. Figure 4.26 represents how the Factory Layout tool is linked to the main components of the VR-Simulator to construct the layout of a manufacturing system in the Witness and the Superscape environments.

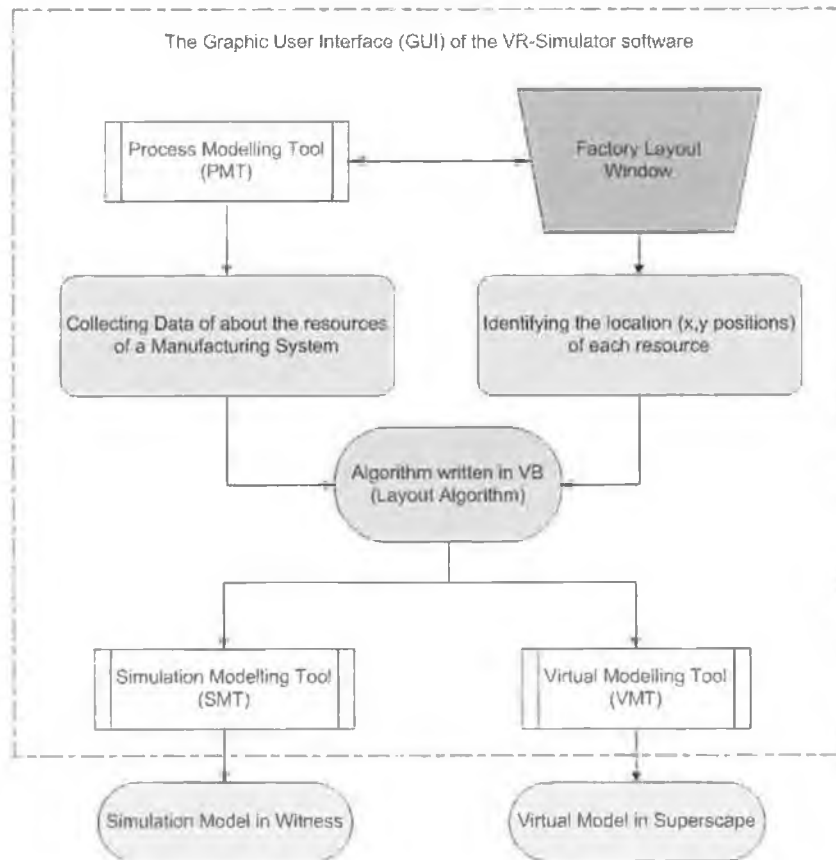


Figure 4.26: Linking the Factory Layout Tool with the Main Components of the VR-Simulator Software

4.4.5 Simulation Modelling Tool Design

The simulation software package that was chosen to be linked with the VR-Simulator software to develop the simulation model in the Witness software package from Lanner. The Simulation Modelling Tool (SMT) allows non-expert users in simulation modelling techniques and programming to automatically develop simulation models of manufacturing systems in Witness software. It also helps expert

users to shrink the time of developing complex simulation models. The SMT uses the information gathered about the resources through the PMT and the Factory Layout window to develop a simulation model automatically. The information gathered is stored within a database in an Excel file with multiple data sheets, where each resource has its own Excel data sheet for storing its relevant data.

4.4.5.1 Formulating WCL Commands

Witness software documentation does not provide the format of WCL commands that can be used to define, detail and display the resources of a simulation model automatically in Witness. The technique that was used to find the syntax of WCL commands was building simple models for each resource and then saving these models as model library files (with *.lst extension). The model library files are used to understand the structure of Witness Command Language (WCL) to construct the syntax of WCL commands that need to be used to define, display and detail the resources automatically. Figure 4.27 illustrates the steps that have been taken to construct the format of WCL commands that can be used to develop a simulation model of a manufacturing system.

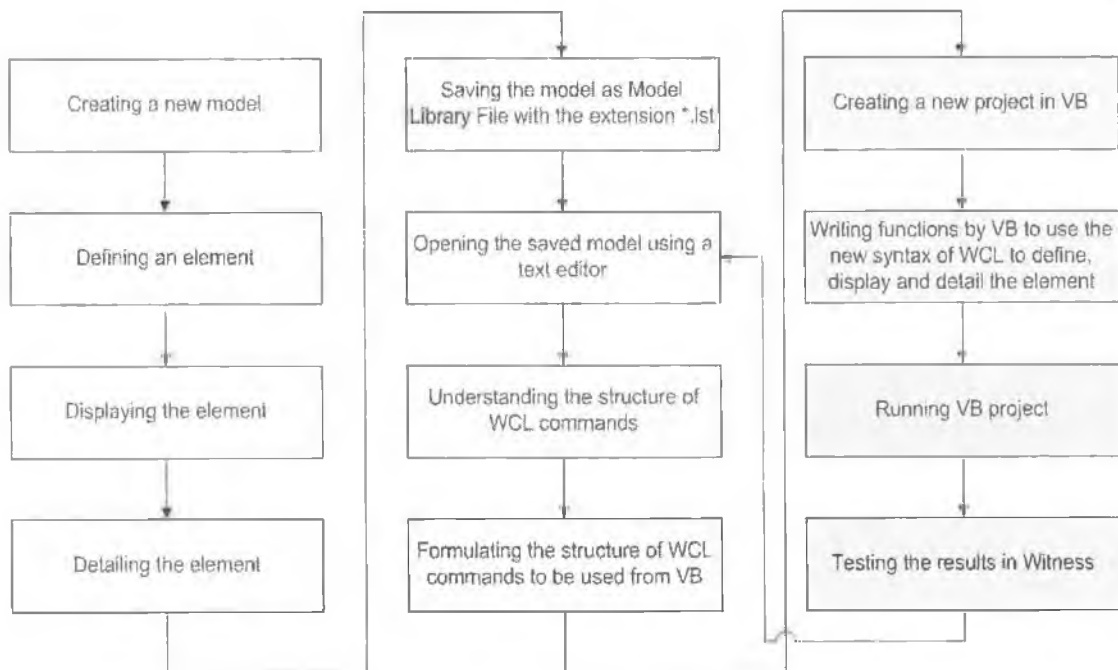


Figure 4.27: Formulating WCL Commands

The steps in Figure 4.27 were followed to formulate the syntaxes of WCL commands to define, display and detail each resource individually from Visual Basic. The resources include part, machine, shift, conveyor, AGV, transporter, machine tool and labourer. The following section describes how the WCL syntaxes were formulated.

The first step of formulating the WCL commands was to create a new model in Witness, then the designer elements window of Witness as shown in Figure 4.28 was used to define and display the resource (e.g. machine).



Figure 4.28: Designer Elements of Witness



Figure 4.29: Dragging and Dropping the Machine Resource in Witness

The machine resource was dragged from the designer element and placed on Witness modelling window as shown in Figure 4.29.

After the machine element was dragged, the machine detail dialog as shown in Figure 4.30 was used to detail the machine. The machine was detailed by giving it a name called “*Drilling*”, and the value “10” for the cycle time. Input and output rules were assigned using the rule editors.

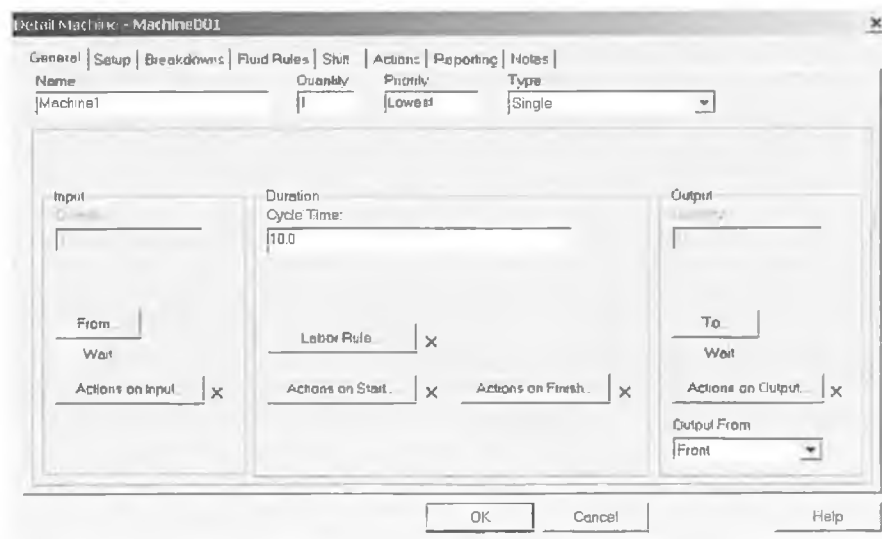


Figure 4.30: Machine Detail Dialog

Once the machine was created and attributes were added to it, the model was saved as library file with the extension of *.lst (see Figure 4.31). The library files of Witness are text files that contain a complete description of all the information in a model. These files are useful for:

- Understating large models, showing the entire model in one document.
- Producing documentation about a model.
- Transferring models between different computers as the *.lst file is a straightforward text file that supports all computer hardware types.
- Composing new Witness models, or amending existing ones without running Witness. That means that Witness model can be built using a word processor or other applications to generate *.lst file.

The text file of the machine model was analysed by consulting the vendor of Witness as well as the programming experience of the author. The library file of the machine model is a text file which contains all model definitions. The file consists of a header, and then a series of WCL commands that have been written as functions to define, display and detail each resource within the simulation model. The header includes some information about the model (e.g. model name, title, author, date and version number of Witness). WCL commands within the machine model are organised in three main functions that represent defining, displaying and detailing the machine.

The three main functions within the machine library file that have been used to define, display and detail the drilling machine are:

Define Function:

```
DEFINE
    MACHINE: Drilling,1,Single,0,0,2,1;
END DEFINE
```

Display Function:

```
DISPLAY
Drilling
    NAME: #0,Name,Group,-
    13,RGB(255,255,128),RGB(0,0,0),2,100,80,400,0,Arial,0,0,0,0,0,3,2,1,34;
    MACHINE ICON: #0,Icon,Member,Status,1,100,100,1,1,0,0;
    LOCK: Off;
    END Drilling
END DISPLAY
```

Detail Function:

```
DETAIL
Drilling
    NAME OF MACHINE: Drilling;
    QUANTITY: 1;
    TYPE: Single;
    PRIORITY: Lowest;
    LABOR:
        Cycle: None;
    END
    DISCRETE LINKS :
```

```

    Fill: None
END
DISCRETE LINKS :
    Empty: None
END
SETUP_DETAIL
    Setup number: 1
    * Mode: Operations;
    * Ops to first: Undefined;
    * Ops between setup: Drilling_SI_Machine;
    * Setup time: Drilling_ST_Machine;
    * Description: Setup Number 1
    * Station number: 1;
DIALOG DISPLAY NUMBER: 0;
LABOR:
    Setup: Drilling_NameLS_Machine#Drilling_NumLS_Machine;
    Pre-empt level: None;
END
    Setup number: 2
    * Mode: Part change;
    * Setup time: Drilling_ST_Machine;
    * Description: Setup Number 2
    * Station number: 1;
DIALOG DISPLAY NUMBER: 1;
LABOR:
    Setup: Drilling_NameLS_Machine#Drilling_NumLS_Machine;
    Pre-empt level: None;
END
END SETUP_DETAIL
CYCLE TIME: Drilling_CT_Machine;
BREAKDOWN_DETAIL
    Breakdown number: 1
    * Mode: Available time;
    * Down interval: 10.0;
    * Repair time: 10.0;
    * Scrap part: No;
    * Description: Breakdown
    * Setup on repair: No;
    * Check at cycle start only: Yes;
DIALOG DISPLAY NUMBER: 0;
LABOR:
    Repair: Labour1#1;
    Pre-empt level: None;
END
END BREAKDOWN_DETAIL
INPUT RULE: IF TIME = StartTime_Drilling (i_Drilling) AND ELEMENT =
MachineName_Drilling (i_Drilling)
    PULL from PartName_Drilling (i_Drilling) out of WORLD
ELSE

```



```

        Wait
    ENDIF;
    OUTPUT RULE: PUSH to SHIP;
    ACTIONS, In
    Add
        Drilling_CT_Machine = CycleTime_Drilling (i_Drilling)
    End Actions
    ACTIONS, Out
    Add
        i_Drilling = i_Drilling + 1
    End Actions
    Output_From: Front;
    REPORTING: Individual;
    SHIFT: Shift1_Week,0,0;
    END Drilling
    END DETAIL

```

The above functions of WCL commands are formulated in a format that can be used from Visual Basic to define, display and detail a machine automatically in Witness software. The following VB functions represent the syntaxes of WCL commands that have been formulated by the author to define, display and detail a machine named "Drilling" automatically in Witness.

VB function is used to define a machine automatically in Witness:

```

Public Sub DefineMachines()
    WitObj.wcl ("define" & vbCrLf & "MACHINE:" + nameMachine(i) +
    ",1,Single,0,0;" & vbCrLf & "enddefine")
End Sub

```

VB function is used to display a machine automatically in Witness:

```

Public Sub DisplayMachines()
    WitObj.wcl ("Display" & vbCrLf & "select" & vbCrLf & nameMachine(i) &
    vbCrLf & "MACHINE ICON: #0,Icon,Member,Status,1," +
    Str(xpos_Machines_DesElements(i)) + "," + Str(ypos_Machines_DesElements(i)) +
    ",1,1,0,0;" & vbCrLf & "NAME: #0,Name,Group,-
    13,RGB(255,255,128),RGB(0,0,0),2," + Str(xpos_Machines_DesElements(i)) + "," +
    Str(ypos_Machines_DesElements(i) - 20) + ",400,0,Arial,0,0,0,0,0,0,3,2,1,34;" &
    vbCrLf & "end" + nameMachine(i) & vbCrLf & "end select" & vbCrLf &
    "enddisplay")
End Sub

```

VB function is used to detail a machine automatically in Witness:

```
Public Sub DetailMachines()
```

```
    WitObj.wcl ("Detail" & vbCrLf & "select" & vbCrLf & nameMachine(i) & vbCrLf & "cycle time: " + varCycleTime_Machine(i) + ";" _
```

```
    & vbCrLf & "BREAKDOWN_DETAIL" & vbCrLf & "Breakdown number: 1" & vbCrLf & "Mode: " + breakdownMode_Machines_DesElements(i) & vbCrLf & "Ops between breakdown:" + breakdownInterval_Machines_DesElements(i) & vbCrLf & "Repair time:" + repairTime_Machines_DesElements(i) + ";" & vbCrLf & "Scrap part:No;" & vbCrLf & "Description: Breakdown" & vbCrLf & "Setup on repair: No;" & vbCrLf & "Check at cycle start only: Yes;" & vbCrLf & "DIALOG DISPLAY NUMBER:0;" & vbCrLf & "LABOR:" & vbCrLf & "Repair: " + nameLaboursRepair_Machines_DesElements(i) + " # " + numLaboursRepair_Machines_DesElements(i) + ";" & vbCrLf & "END" & vbCrLf & "END BREAKDOWN_DETAIL" _
```

```
    & vbCrLf & "SETUP_DETAIL" & vbCrLf & "Setup number: 1" & vbCrLf & "Mode: Operations;" & vbCrLf & "Ops to first: Undefined;" & vbCrLf & "Ops between setup:" + varSetupInterval_Machines(i) + ";" & vbCrLf & "Setup Time:" + varSetupTime_Machines(i) + ";" & vbCrLf & "Description: Setup Number 1" & vbCrLf & "Station number: 1;" & vbCrLf & "LABOR:" & vbCrLf & "Setup: " + varNameLaboursSetup_Machines(i) + " # " + varNumLaboursSetup_Machines(i) + ";" & vbCrLf & "END" _
```

```
    & vbCrLf & "Setup number: 2" & vbCrLf & "Mode: Part change;" & vbCrLf & "Setup Time:" + varSetupTime_Machines(i) + ";" & vbCrLf & "Description: Setup Number 2" & vbCrLf & "Station number: 1;" & vbCrLf & "LABOR:" & vbCrLf & "Setup: " + varNameLaboursSetup_Machines(i) + " # " + varNumLaboursSetup_Machines(i) + ";" & vbCrLf & "END" & vbCrLf & "END SETUP_DETAIL" _
```

```
    & vbCrLf & "INPUT RULE: IF TIME = StartTime_" + nameMachine(i) + "(i_" + nameMachine(i) + ") and element = MachineName_" + nameMachine(i) + "(i_" + nameMachine(i) + ")" & vbCrLf & "PULL from " + "PartName_" + nameMachine(i) + "(i_" + nameMachine(i) + ")" + " out of WORLD" & vbCrLf & "ELSE" & vbCrLf & "Wait" & vbCrLf & "ENDIF;" & vbCrLf & "OUTPUT RULE: PUSH to " + "Ship" + ";" & vbCrLf & "ACTIONS, In" & vbCrLf & "Add" & vbCrLf & nameMachine(i) + "_CT_Machine = CycleTime_" + nameMachine(i) + "(i_" + nameMachine(i) + ")" & vbCrLf & "End Actions" & vbCrLf & "ACTIONS, Out" & vbCrLf & "Add" & vbCrLf & "i_" + nameMachine(i) + "= i_" + nameMachine(i) + "+ 1" & vbCrLf & "End Actions" & vbCrLf & "REPORTING: Individual;" & vbCrLf & "SHIFT:" + shift_Machines_DesElements(i) + ",0,0;" & vbCrLf & "end" + nameMachine(i) & vbCrLf & "end select" & vbCrLf & "enddetail")
```


```
End Sub
```

The approach that was presented above was followed to find the WCL syntaxes of the other resources, which included labourer, part, buffer, conveyor, shift, transporter, AGV and the variables that hold the attributes of each resource.

4.4.5.2 *Generating Simulation Models*

This section presents the approach that was used to generate simulation models of manufacturing systems automatically in the Witness software. Witness, as mentioned earlier, was chosen as the simulation tool to be integrated with the VR-Simulator to develop simulation models. Witness [129] is a Windows application from the Lanner Group. The Witness simulation package was developed specifically to model material flow in manufacturing systems. It is a hybrid of a simulator and a general simulation language. The Witness simulation package is an “OLE Automation Server”. Therefore, it can be controlled by an “OLE Automation Controller”, such as Visual Basic. Through this mechanism it is possible to control, edit or construct an entire model.

Building simulation models in Witness involves three steps: “*define*”, “*display*”, and “*detail*”. In the define phase, all the elements of simulation model should be defined. Witness provides the modelling elements used by manufacturing systems, e.g. machines, parts, labourers, and transportation systems. In the display phase, all the elements need to be presented by icons in order to build up a pictorial representation of the manufacturing system. Finally, the detail phase in which the logic of the simulation model is specified.

The SMT of the VR-Simulator software used the same principle of Witness to generate a simulation model, where all the resources have to be defined first, then icons that represent the resources of a manufacturing system were used to construct the graphic presentation of the simulation model. Finally, attributes and coding were applied to detail each resource. The database of a manufacturing system needs to be loaded first to the VR-Simulator before the user can generate the simulation model. The database can be loaded by selecting the “*Load Database*” option from the “File” menu options or by clicking on its icon. An open window appears to allow the user to specify the name of the Excel file where the database of the manufacturing system is stored as shown in Figure 4.33. Once the database is loaded, the user can select the “*Generate Simulation Model*” from the Simulation Modelling Tool menu options or clicking on this icon  as shown in Figure 4.34.

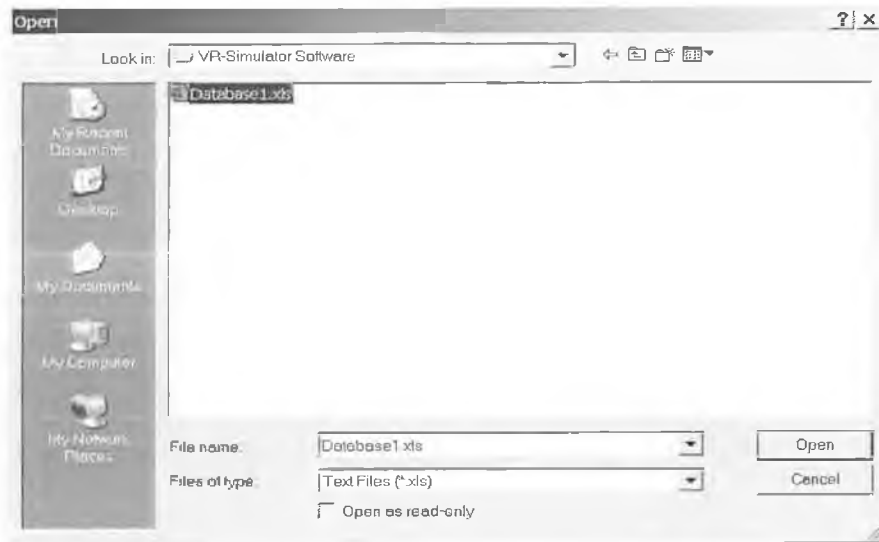


Figure 4.33: Loading a Database of a Manufacturing System

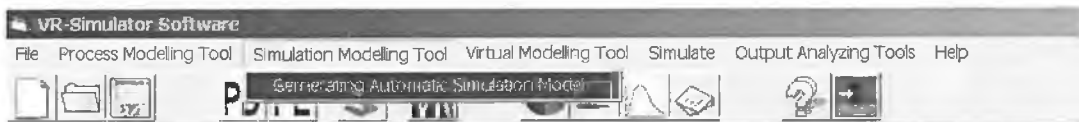


Figure 4.34: Generate Automatic Simulation Model Command

Upon activating the command to generate the simulation model automatically, the algorithm shown in Figure 4.35 is executed. Witness OLE control is the mechanism that has been used to link the VR-Simulator software with Witness software and WCL commands which are used to build and display the simulation model in Witness. The following command is an OLE Automation command, which has been used to link a running version of Witness with the VR-Simulator software:

$$\text{Set WitObj} = \text{GetObject} (, "WITNESS.WCL")$$

Algorithms developed in VB were used to read the Excel sheets database of the resources which include information about machines, parts, conveyors, AGVs, labourers, machine tools, processes, transporters, buffers and shifts, and then this information is transferred into arrays that act as temporary database for the VR-Simulator software. Algorithms written in VB linked with WCL commands are used to develop simulation models automatically in Witness in three phases as follows:

- *Define phase:* In the phase, all the resources of a manufacturing system are defined and included in the simulation model.

- *Display phase:* In this phase, some of the resources are presented by icons in order to build up a pictorial representation of the manufacturing system.
- *Detail phase:* The logic of the simulation model is specified in this phase, where the process plan and the orders have been used to detail the resources. Code is generated automatically to be attached to each resource for monitoring and performing different activities within the simulation model.

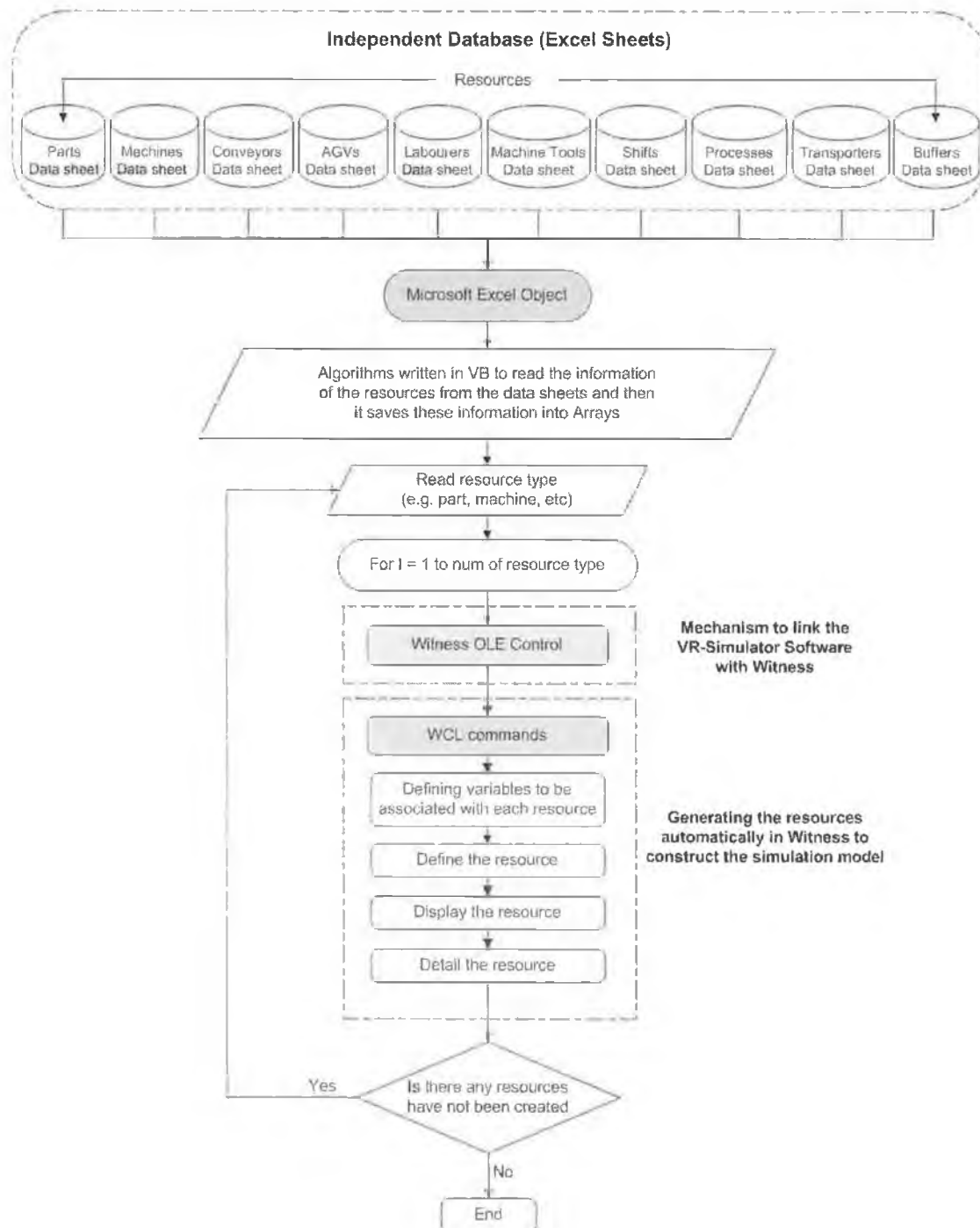


Figure 4.35: Algorithm to Generate Simulation Model Automatically

The parameter of this section presents how the Simulation Modelling Tool is used to generate each resource automatically in Witness.

▪ **Shifts:**

The shifts are the first resources to be generated in Witness as they are used by the other resources. Figure 4.36 depicts the mechanism of the algorithm developed to generate shifts automatically in Witness.

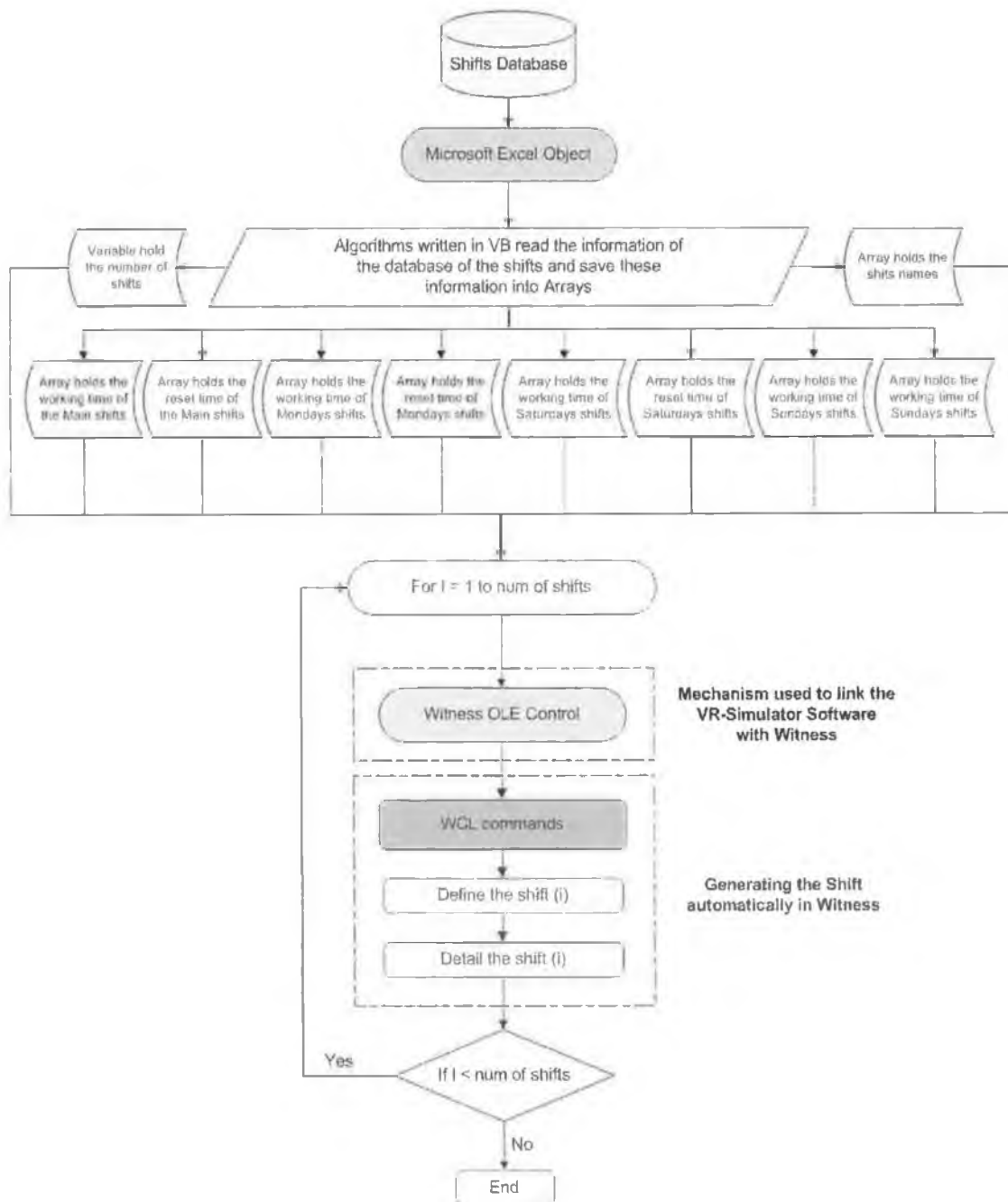


Figure 4.36: Generating Shifts Automatically in Witness

Shifts are generated automatically based on the value of the variable that holds the number of shifts. In the define phase of the shifts, the array that holds the shift's name is used to assign names to each shift. The other arrays that hold details regarding the shifts pattern are used to detail each shift individually.

▪ **Parts:**

Figure 4.37 depicts the algorithm developed to generate parts automatically in Witness.

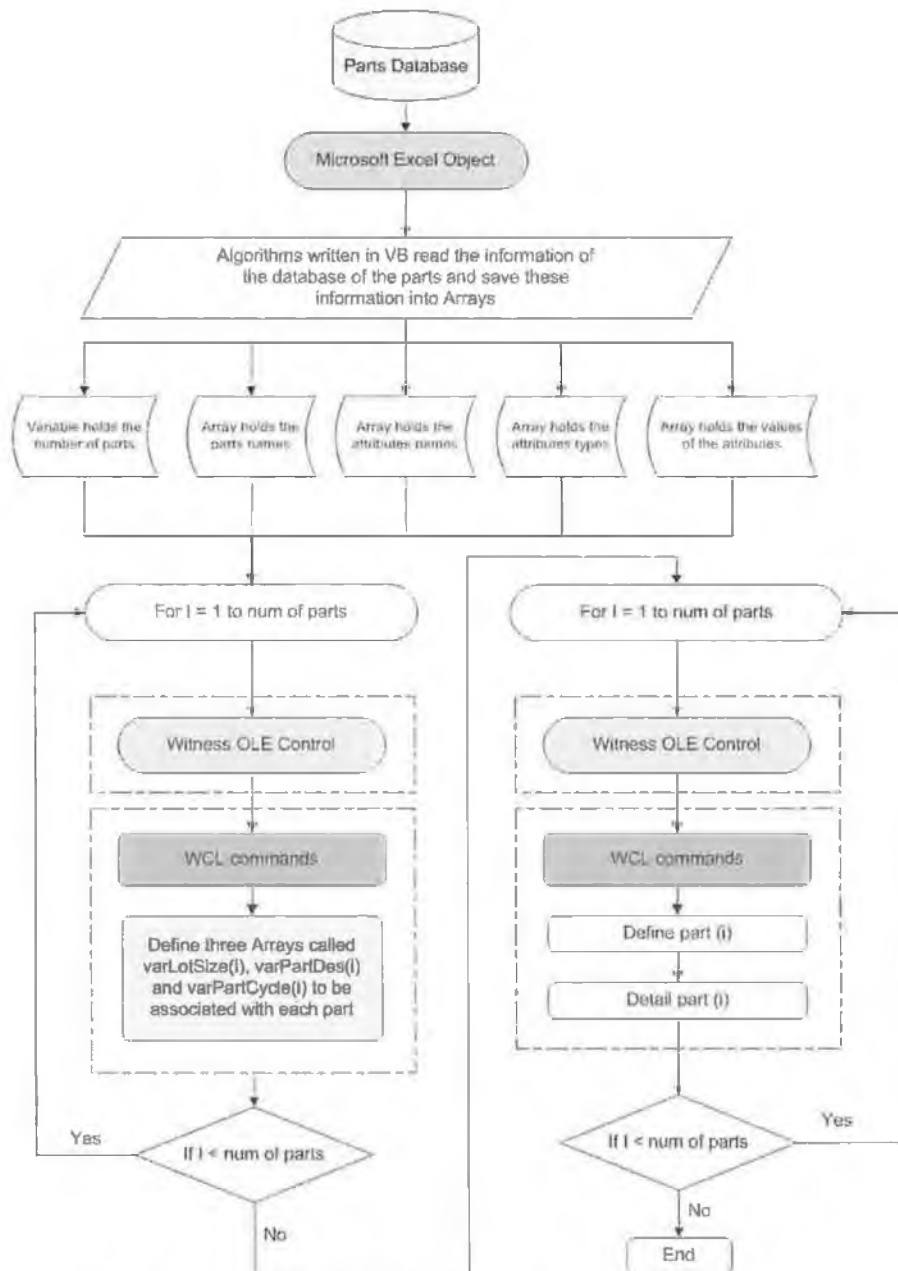


Figure 4.37: Generating Parts Automatically in Witness

Parts are generated automatically based on the value of the variable that holds the number of parts. Three arrays called “*varLotSize(i)*”, “*varPartDes(i)*” and “*varPartCycle(i)*” were defined in Witness to be used to hold the orders of each part. In the define phase, the array that holds the part names were used to assign names for each part. Finally, each part has been detailed by assigning a variable to the “*Lot Size*” using WCL commands.

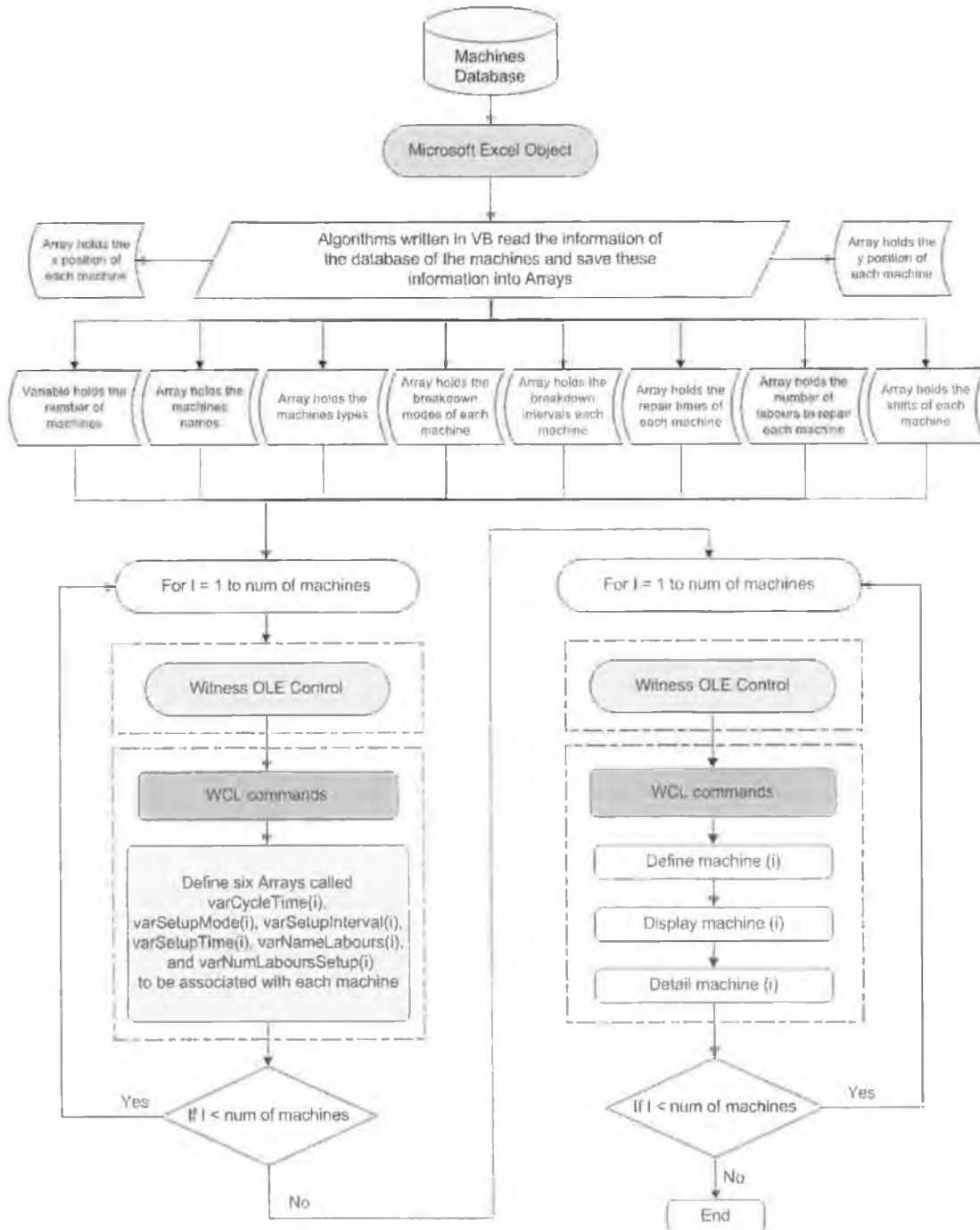


Figure 4.38: Generating Machines Automatically in Witness

▪ **Machines:**

Figure 4.38 represents the algorithm developed to generate machines automatically in Witness. Machines are generated automatically based on the value of the variable that holds the number of machines. Arrays were defined in Witness to hold the values of the attributes for each machine. These attributes included cycle time, setup mode, setup interval, setup time, name of the labourers for the setup, and number of labourers for the setup. In the define phase, the array that holds machine names were used to assign a name for each machine.

Once the machines have been defined, the arrays that hold the values of x, y positions are used to position each machine within the simulation model. Finally, each machine has been detailed by assigning a variable for the cycle time and other variables to hold the breakdown details as well as the shift name for each machine. Also, WCL commands were used to assign the input rules, actions on input, actions on output, and output rules for every generated machine.

▪ **Conveyors:**

Figure 4.39 represents the algorithm developed to generate conveyors automatically in Witness. Conveyors are generated automatically based on the value of the variable that holds the number of conveyors. Arrays were defined in Witness to hold the values of the attributes for each conveyor. These attributes included length in parts, movement index time, breakdown mode, breakdown interval, repair time, name of the labourer for the repair, and number of labourers for the repair.

In the define phase, the array that holds conveyor names were used to assign a name for each conveyor. Once the conveyors have been defined, the arrays that hold the values of the x, y positions are used to position each conveyor within the simulation model. Finally, each conveyor has been detailed by assigning a variable for the length in parts, movement index time and other variables to hold the breakdown details as well as the shift name for each conveyor. Also, WCL commands were used

to assign the input rules, actions on joins, actions on front, and output rules for every generated conveyor.

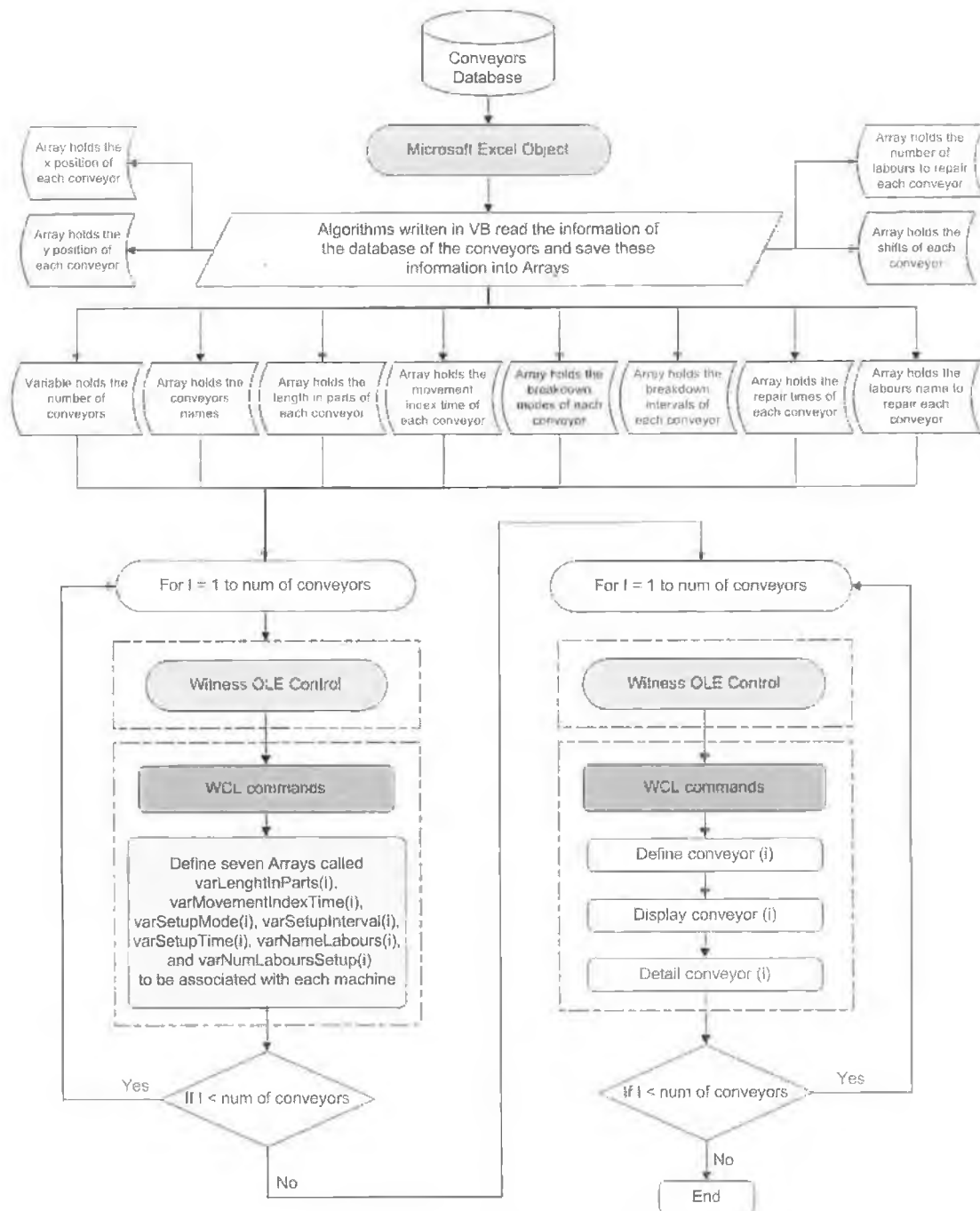


Figure 4.39: Generating Conveyors Automatically in Witness

▪ **AGVs:**

Machine objects instead of the Witness defined vehicle and track objects were used to model AGVs because Witness does not provide a method to breakdown vehicles

or tracks. The behaviours of the AGVs were modelled by assigning variables to be associated with each machine. Figure 4.40 represents the algorithm developed to generate AGVs automatically in Witness.

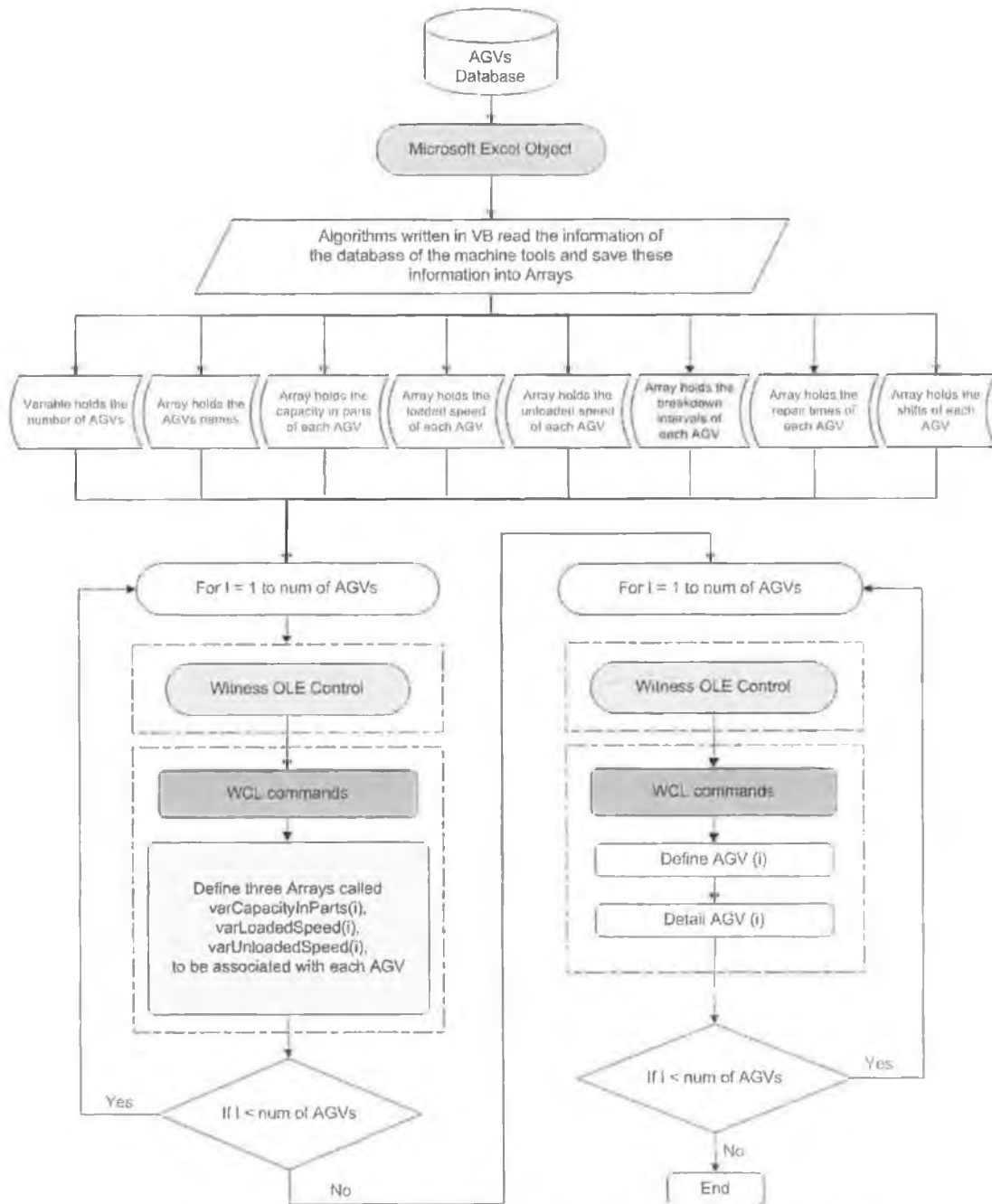


Figure 4.40: Generating AGVs Automatically in Witness

The AGVs are generated automatically based on the value of the variable that holds the number of AGVs. Three array variables were defined to be associated with each AGV. These arrays were “*varCapacityInParts(i)*”, “*varLoadedSpeed(i)*”, and “*varUnloadedSpeed*”. The arrays were defined in Witness to be used to hold the

capacity in parts, loaded speed and the unloaded speed of each AGV. In the define phase, the array that holds the AGVs names were used to assign a name for each AGV. Finally, each AGV has been detailed by assigning the cycle time to it.

▪ **Transporters:**

Transporters were used to model any material handling system without having it displayed within the simulation model.

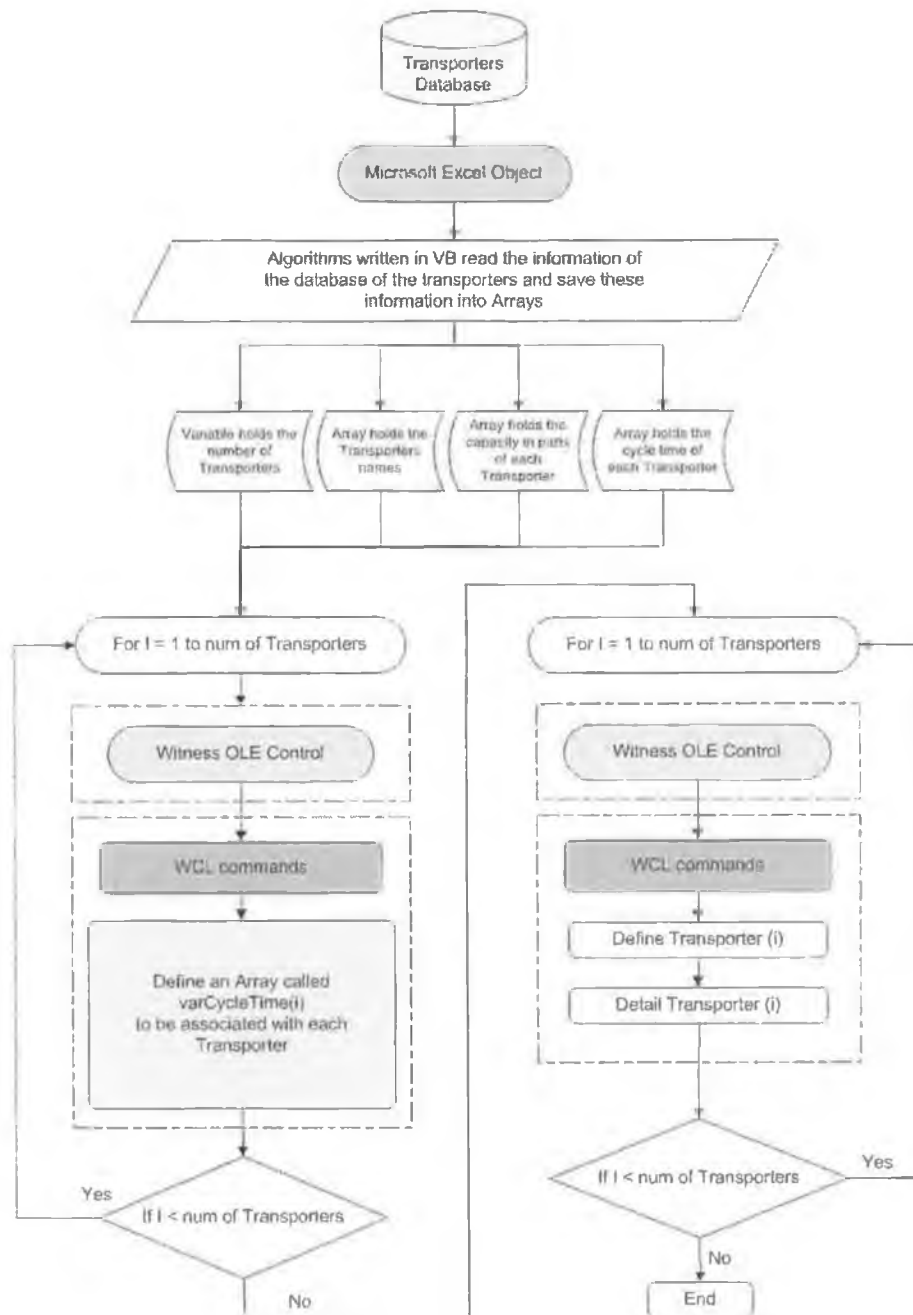


Figure 4.41: Generating Transporters Automatically in Witness

The process of transferring the products between the resources was represented by a time delay. The transporters can be used to model a forklift, a trolley, etc. Figure 4.41 represents the algorithm developed to generate transporters automatically in Witness. The transporters have been defined and displayed in a manner similar to the AGVs. The transporters were modelled by machine objects that represent the required time to deliver the parts between two predefined locations.

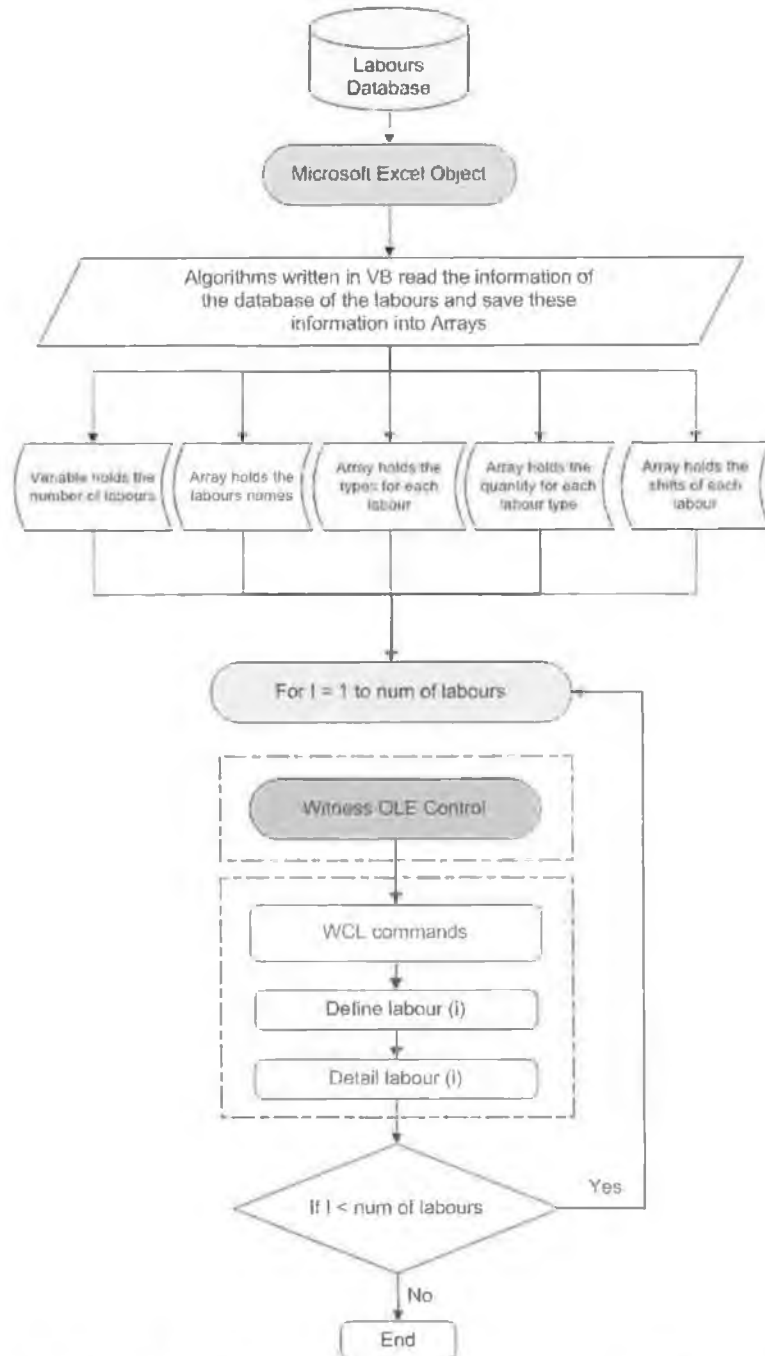


Figure 4.42: Generating Labourers Automatically in Witness

- **Labourers:**

Figure 4.42 depicts the algorithm developed to generate labourers automatically in Witness. The labourers are used to handle the machines for setting up, repairing, driving a material handling system etc. There are three types of labourers that can be generated, which include technician, driver, and labourer.

The labourers are generated automatically based on the value of the variable that holds the number of labourers. Two arrays called “*varLabourType(i)*” and “*varLabourQuantity(i)*” were defined in Witness to be used to hold the types of the labourers and the quantity of each labourer type. In the define phase, the array that holds labourers names was used to assign a name for each labourer. Finally, each labourer has been detailed by assigning a variable to the “*Quantity*” attribute.

- **Machine Tools:**

Figure 4.43 depicts the algorithm developed to generate machine tools automatically in Witness to be used by the machines or the labourers. Witness does not provide an element that can be used to define a machine tool so the labourers element was used to define the machine tools as it has the same modelling characteristics.

The machine tools are generated automatically based on the value of the variable that holds the number of machine tools. Two arrays called “*varMachineToolsType(i)*” and “*varMachineToolsQuantity(i)*” were defined in Witness to be used to hold the types of the machine tools and the quantity of each machine tool type. In the define phase, the array that holds machine tools names was used to assign a name for each machine tool. Finally, each part has been detailed by assigning the “*Quantity*” attribute.

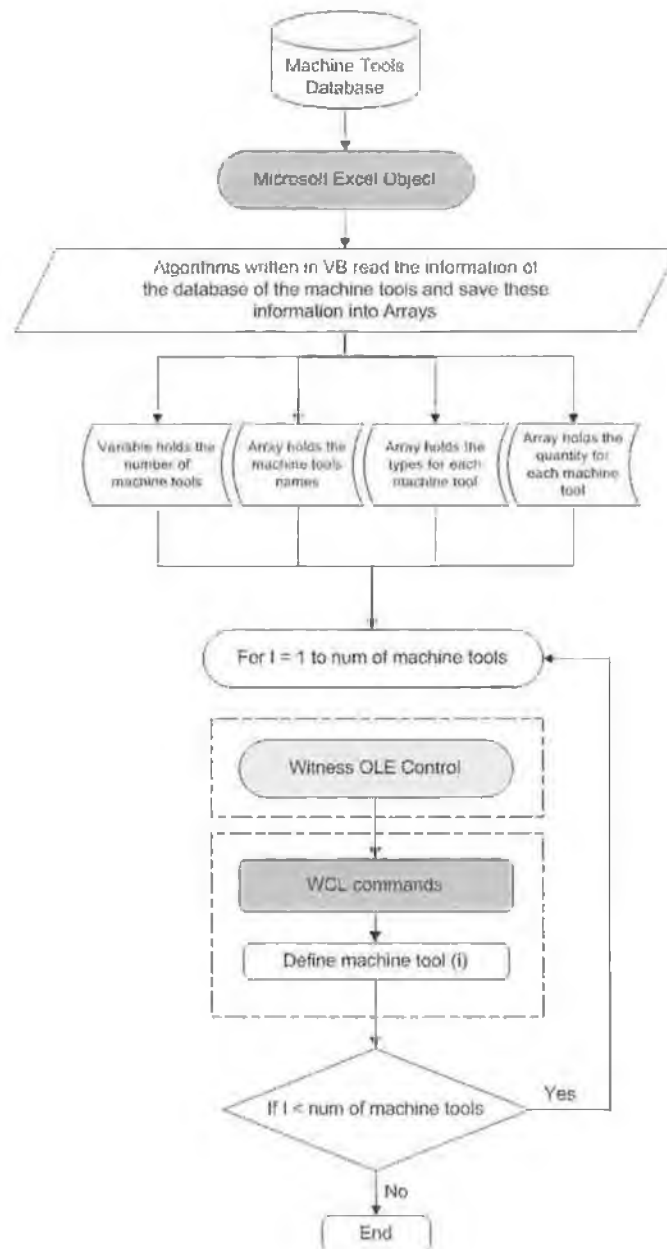


Figure 4.43: Generating Machine Tools Automatically in Witness

▪ **Buffers:**

Buffers can be located in front of or after a machine where the parts have to wait until the next destination is free. Buffers can also be used to model a warehouse or store where parts should be stored. Figure 4.44 depicts the algorithm developed to generate buffers automatically in Witness.

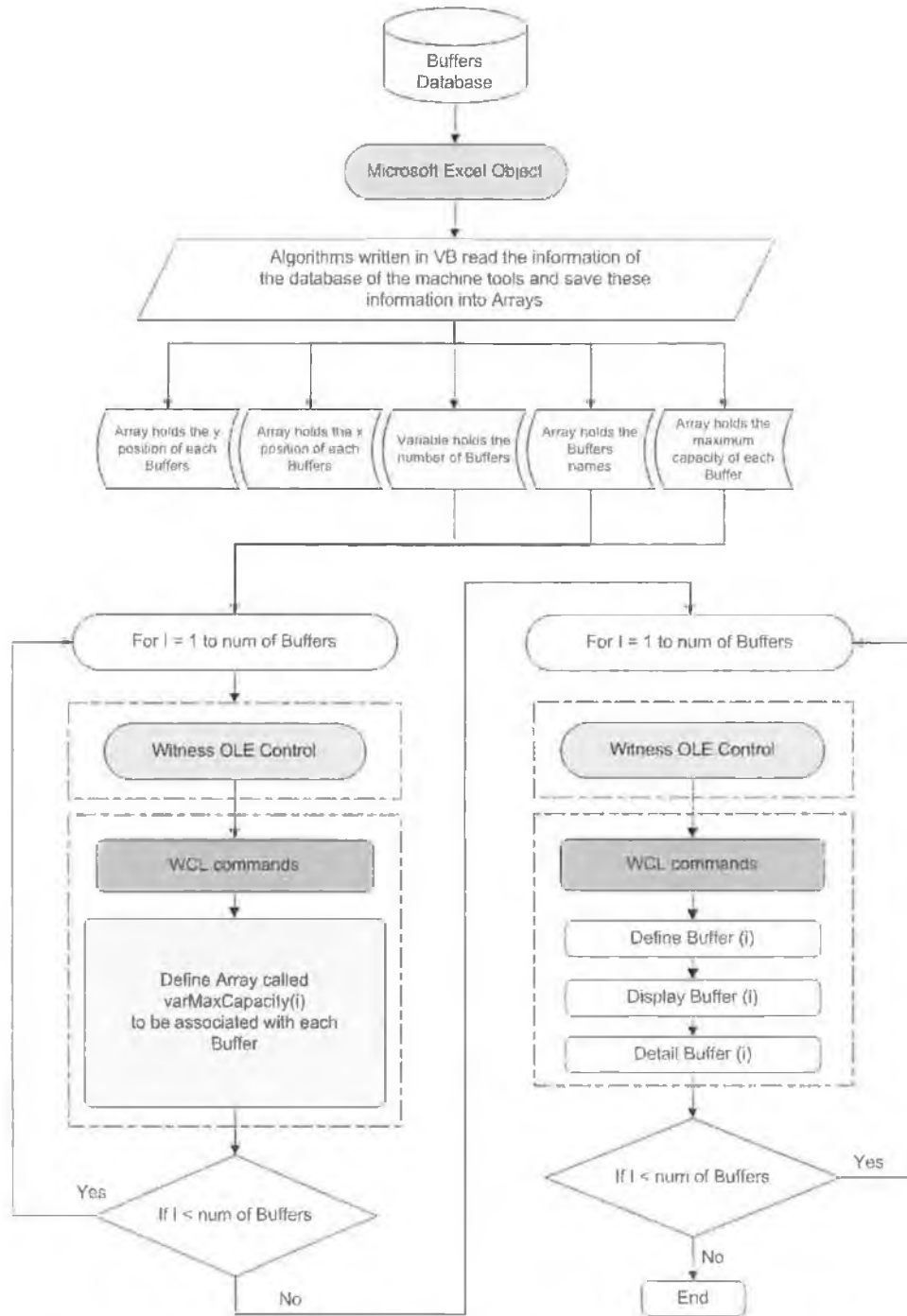


Figure 4.44: Generating Buffers Automatically in Witness

The buffers are generated automatically based on the value of the variable that holds the number of buffers. An array called “*varMaximumCapacity(i)*” was defined in Witness to be used to hold the maximum capacity in parts and the quantity of each buffer. In the define phase, the array that holds buffers names were used to assign a name for each buffer. Then, buffers are displayed based on the x, y positions of each

buffer. Finally, each buffer is detailed by assigning a variable to the “*CapacityInParts*” attribute.

4.4.6 Virtual Modelling Tool design

The Virtual Modelling Tool (VMT) is used to automatically create virtual models of manufacturing systems that represent simulation models in 3D environment. The VMT allows users who are not familiar with 3D modelling and programming to generate a virtual model of a manufacturing system automatically. The virtual reality software that was chosen to be linked with the VR-Simulator software to develop virtual models is called Superscape VRT.

4.4.6.1 Design and Implementation of Virtual Environment Template

The virtual environment template was developed using Superscape VRT to enable the users of the VR-Simulator software to generate virtual models of manufacturing systems automatically.

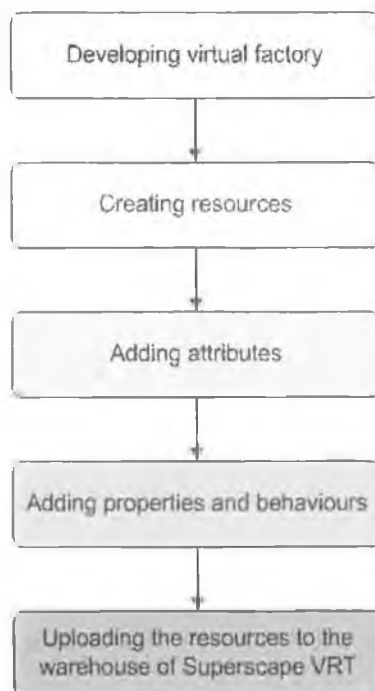


Figure 4.45: Steps of Design and Implementation of the Virtual Environment Template

The virtual worlds of Superscape VRT [127] can be targeted towards either Visualiser for standalone applications that can be distributed on a CD, or Viscap for 3D Web Pages. Complex worlds with many levels of interaction can be achieved using the Visualiser. Viscap worlds need to be as small as possible to minimise their download time. The aim of this research is to develop virtual models that are integrated with the VR-Simulator to work as a standalone application and not to be distributed on the Web so details can be added as much as required. The following flowchart (Figure 4.45) illustrates the steps that have been taken to design and implement the virtual environment to be used as a template to generate the virtual models.

▪ *Developing a virtual factory*

The first step to design the virtual environment template of the VR-Simulator was building a virtual factory that can be used as a virtual environment to represent manufacturing systems. The virtual factory was developed using the world editor (see Figure 3.21) and the shape editor (see Figure 3.22) of Superscape VRT. A group object that represents the virtual factory was created by the world editor of Superscape VRT. The size of this object is set to be 200m by 300m that matches the dimensions of the template of the factory layout.

The shape editor was used to create four walls, a ceiling and a roof for the factory. Once these shapes were created, the world editor was used to apply them to the factory object to create the virtual factory. Textures and colours were applied to the virtual factory to enhance its look to give a better feeling of a real factory. Some objects were added (e.g. doors, windows) to the factory. Figure 4.46 shows a snapshot of the virtual factory.

▪ *Creating resources*

All the resources that represent a manufacturing system, which need to be displayed within the virtual factory, were created individually as objects, where each object represented a different resource. The resources included labourer, drilling machine,

saw machine, grinding machine, polishing machine, packaging machine, generic machine, AGV, buffer, and conveyor.

Each object within the virtual world has a set of standard attributes that define its:

- Shape type: the outline of the object.
- Size: the size of the bounding cube.
- Position: its position in the world.
- Name: this must be unique.

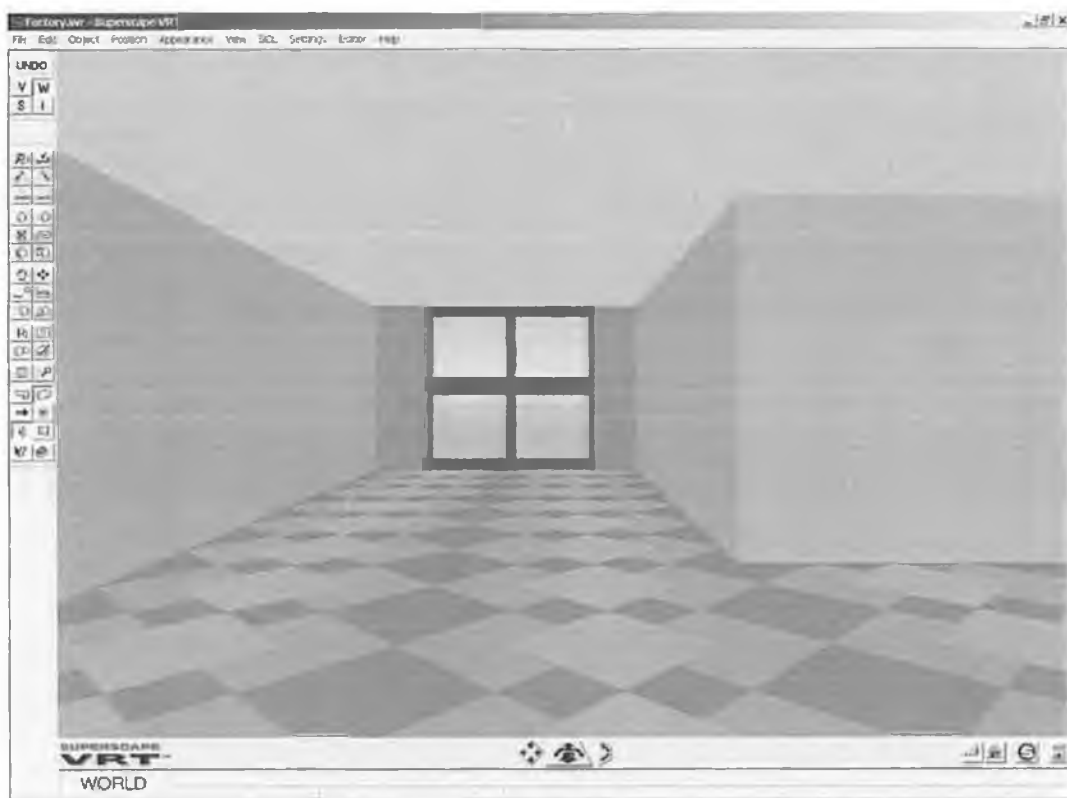


Figure 4.46: Snapshot of the Virtual Factory

The objects that represent the resources were created using the shape editor and the world editor of Superscape VRT. The following flowchart (see Figure 4.47) presents the procedure for creating the resources, which are referred to as objects, using the shape editor and the world editor of Superscape VRT. Each object was created individually to represent a resource of a manufacturing system. An object was created for each resource from the designer menu of Superscape VRT. The type of

all created objects was set to be a Group to hold the shapes and the subgroups of each resource. A name was given to each object based on the resource type. For example, the name “*Saw Machine*” was given to the object that represents the saw machine.

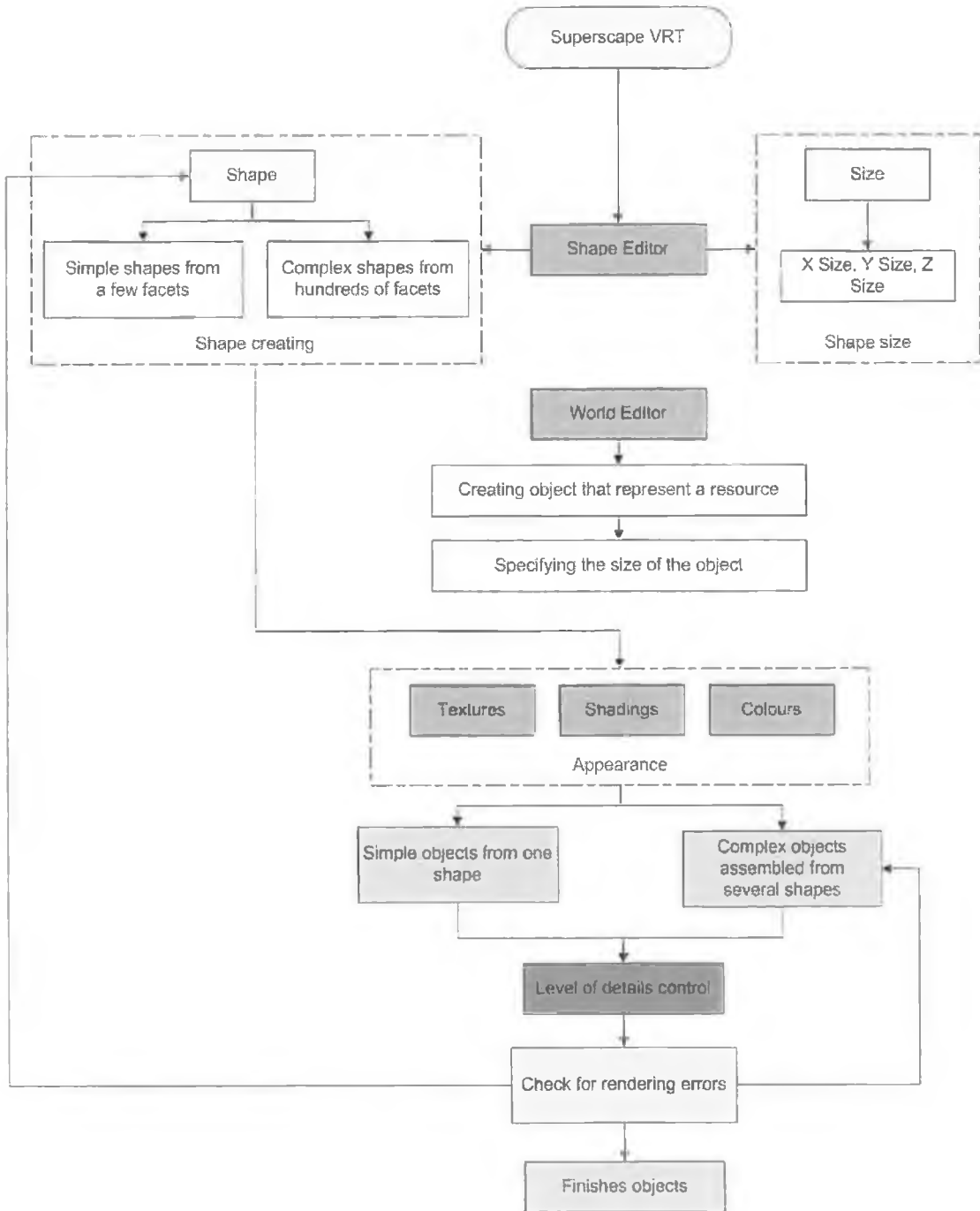


Figure 4.47: Procedure for Creating the Resources

In their simplest form objects have shape, size and appearance. They may be made up of several child objects in which case they will have their own internal layout.

They may also have behaviours and dynamics, which will be covered later in this section. The shapes for each resource were created using the shape editor of Superscape VRT. At this stage the shapes consisted of a number of flat facets. The number of facets used to create the shape has implications for the VE, particularly rendering speed.

The simplest way of rendering a shape is to give each facet a colour. More complex visual effects can be achieved by applying textures to the facets to create visually realistic objects. Applying gradual and smooth shading across several facets can simulate curved surfaces realistically, although the geometry is still made up of flat facets. Earlier monitoring of the number of facets used and the number and resolution of the textures used should help to produce an object that does not require too much processor power to render. However at this stage it is prudent to visually check the finished object looking for redundant facets or textures that are unnecessarily detailed. It is important to check objects for rendering errors. These errors may be due to the way a shape has been constructed (e.g. the order in which the facets were added), or the way the shapes have been assembled to make an object (e.g. forming overlaps).



Figure 4.48: Virtual Drilling Machine

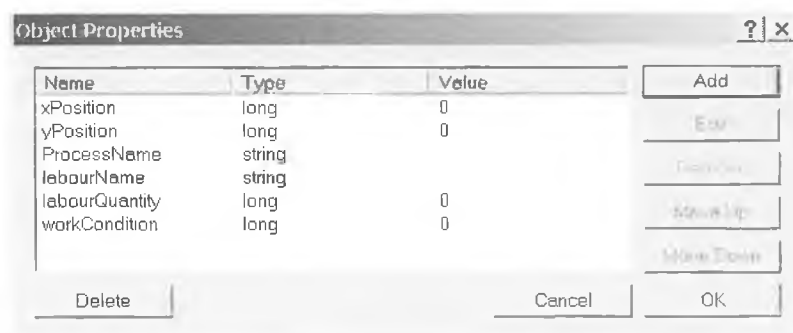
Each virtual resource comprised a number of objects, where each object represented a different part of the resource. For example, the drilling machine consisted of four main objects, which were table, drill chunk, drill tool, spindle. An object was created for each part for the drilling machine. Then the main group of the drilling machine was used to assemble these parts to construct the final shape of the drilling machine resource object as shown in Figure 4.48. All the other virtual resources were created by following the same approach.

▪ *Adding attributes*

Attributes were added to the created resources, these included size, position, textures, dynamics, colour and sound attributes. Dynamic attributes were given to resources to enable them to move within the virtual factory. Size attributes were applied to the resources to change their size to be relevant to the virtual factory. Position attributes were given to the resources to give each resource a location and an orientation to other resources within the factory. A colour attribute and different colours were given to resources to enhance their look; and finally a sound attribute was given to the resources to indicate their working conditions.

▪ *Adding properties and behaviours*

Different types of properties were assigned to the resources (e.g. Integer type properties were used to hold the values of their positions; and the String type property was assigned to hold the name of the processes for each resource). Figure 4.49 shows the properties that have been assigned to the drilling machine.



The screenshot shows a dialog box titled "Object Properties" with a table of properties. The table has three columns: Name, Type, and Value. The properties listed are xPosition, yPosition, ProcessName, labourName, labourQuantity, and workCondition. The values for xPosition, yPosition, labourQuantity, and workCondition are all 0. The value for ProcessName is empty. The value for labourName is empty. There are buttons for Add, Edit, Delete, Cancel, and OK. There are also buttons for Up, Down, and Move Down.

Name	Type	Value
xPosition	long	0
yPosition	long	0
ProcessName	string	
labourName	string	
labourQuantity	long	0
workCondition	long	0

Figure 4.49: Properties of the Drilling Machine

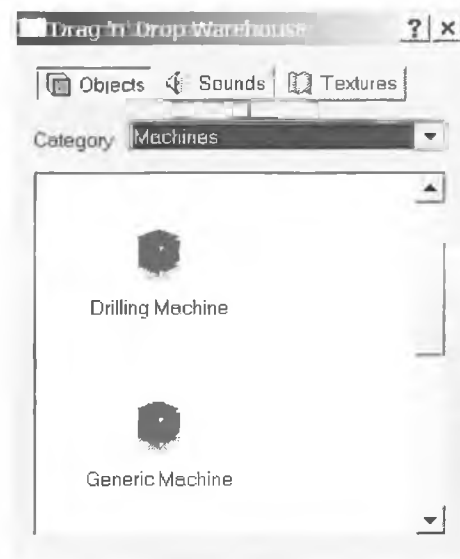
The “*xPosition*” and “*yPosition*” properties were assigned to the drilling machine to hold its position values within the virtual factory. The “*processName*” property was used to hold the name of the processes associated with the drilling machine. The “*labourName*” property was applied to hold the name of the labourer who was handling the drilling machine. The “*labourQuantity*” property was applied to hold the value of the number of labourers needed to handle the drilling machine.

Finally, the “*workCondition*” property was assigned to the drilling machine to hold the value of its working condition, which includes “0”, “1”, “2”, or “3”. Each value represents different activity. The value “0” indicates the machine is idle. The value “1” indicates the drilling machine is processing a part. The value “2” indicates the machine needs to be setup. The value “3” indicates the machine is under maintenance.

▪ ***Uploading the created resources to the warehouse library of Superscape***

Once all the properties were applied to the resources, behaviours were assigned to them using Superscape Control Language. The behaviours are represented by codes written in SCL attached to each resource. The purpose of the code was to continuously monitor the values of properties on each resource and to command the resource to perform different actions based on these values. The behaviours were added to the resources to simulate the activities of the simulation model in 3D scope within the virtual model.

After all the resources were created, and their attributes, properties and behaviours were added, each virtual resource was uploaded to the warehouse of Superscape VRT by saving the resource under the name of the resource type with the extension *.VCA (Visual Clip Art) as shown in Figure 4.50. These resources can be loaded later using the VMT to construct the virtual model of a manufacturing system that needs to be addressed.



4.50: Uploading the Created Resources to the Warehouse Library of Superscape

4.4.6.2 Generating Virtual Models

The VMT can be used to generate a virtual model of a manufacturing system automatically. The virtual model presents the simulation model in a 3D presentation as it helps the users to understand the activities of the manufacturing systems. Superscape VRT comes with an ActiveX control called “Superscape 3D Control. This control was used to link VR-Simulator software with the Superscape environment to display the virtual models within an application written in VB. Superscape control language (SCL) was used to create the virtual models of manufacturing systems.


The database must first be uploaded to the VR-Simulator before the user can generate a virtual model of a manufacturing system. Once the database is loaded, the user can either select the “*Generate Virtual Model*” from the simulation modelling tool menu options as shown in Figure 4.51, or by clicking on this icon .



Figure 4.51: Generate Automatic Virtual Model Command

As the user activates the command to generate the virtual model automatically, the algorithm that is depicted in Figure 4.52 is executed to build the virtual model.

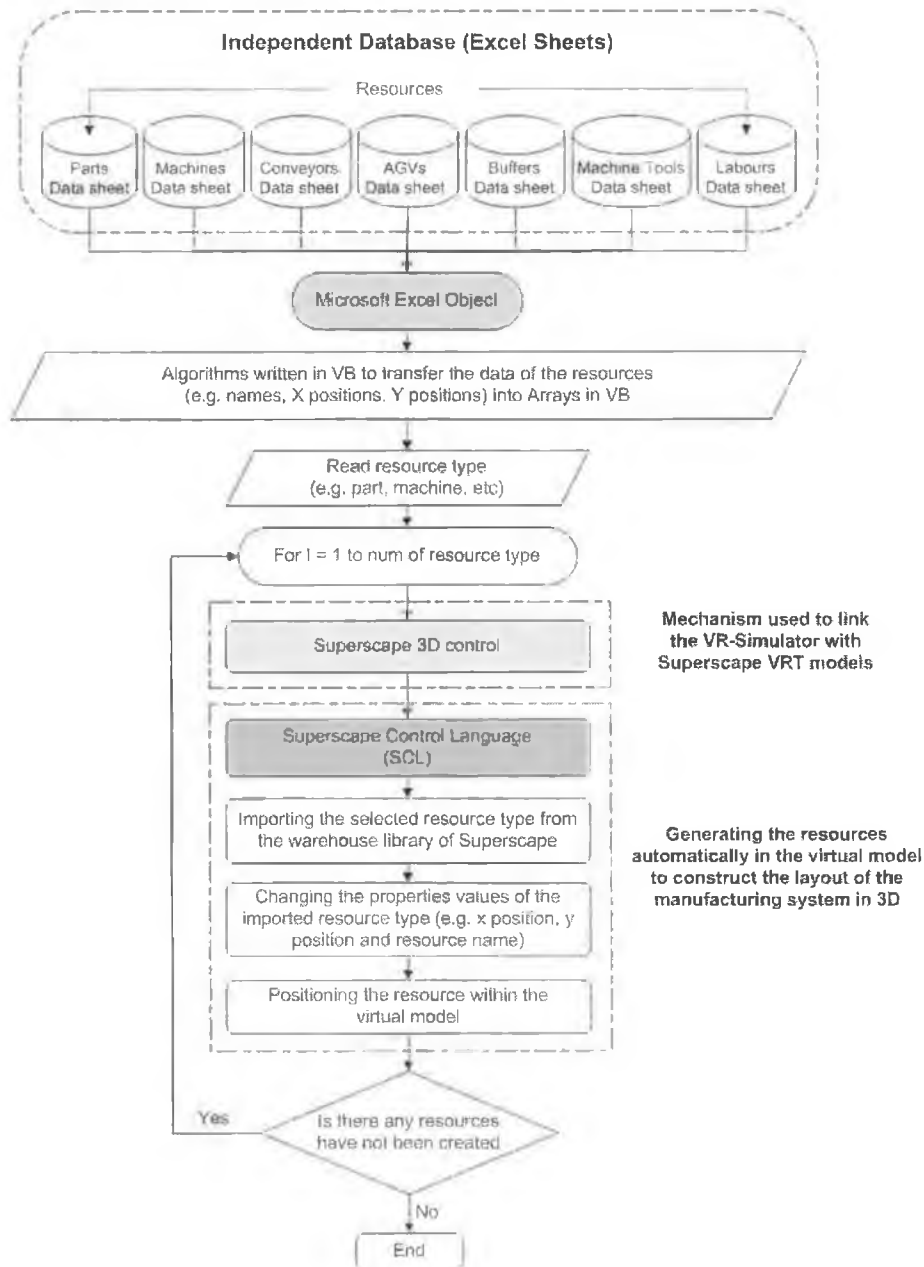


Figure 4.52: Algorithm to Generate Virtual Models Automatically

Algorithms were written in VB to read the independent database to gather information about the resources. This information includes the resource name, and the x, y positions of each resource. This information is fed to the virtual model through the Superscape 3D control mechanism that links the VR-Simulator software with the virtual model. Then, code written in SCL was used to call the resources

from the warehouse of Superscape VRT and position them based on the x, y positions of each resource. The resources that are displayed once the user activates the command to generate virtual model automatically are machines, conveyors and buffers. The labourers and AGVs are displayed only when the virtual model is simulating the activities of a manufacturing system.

Figure 4.53 illustrates the algorithm used to generate the machines automatically in the virtual model.

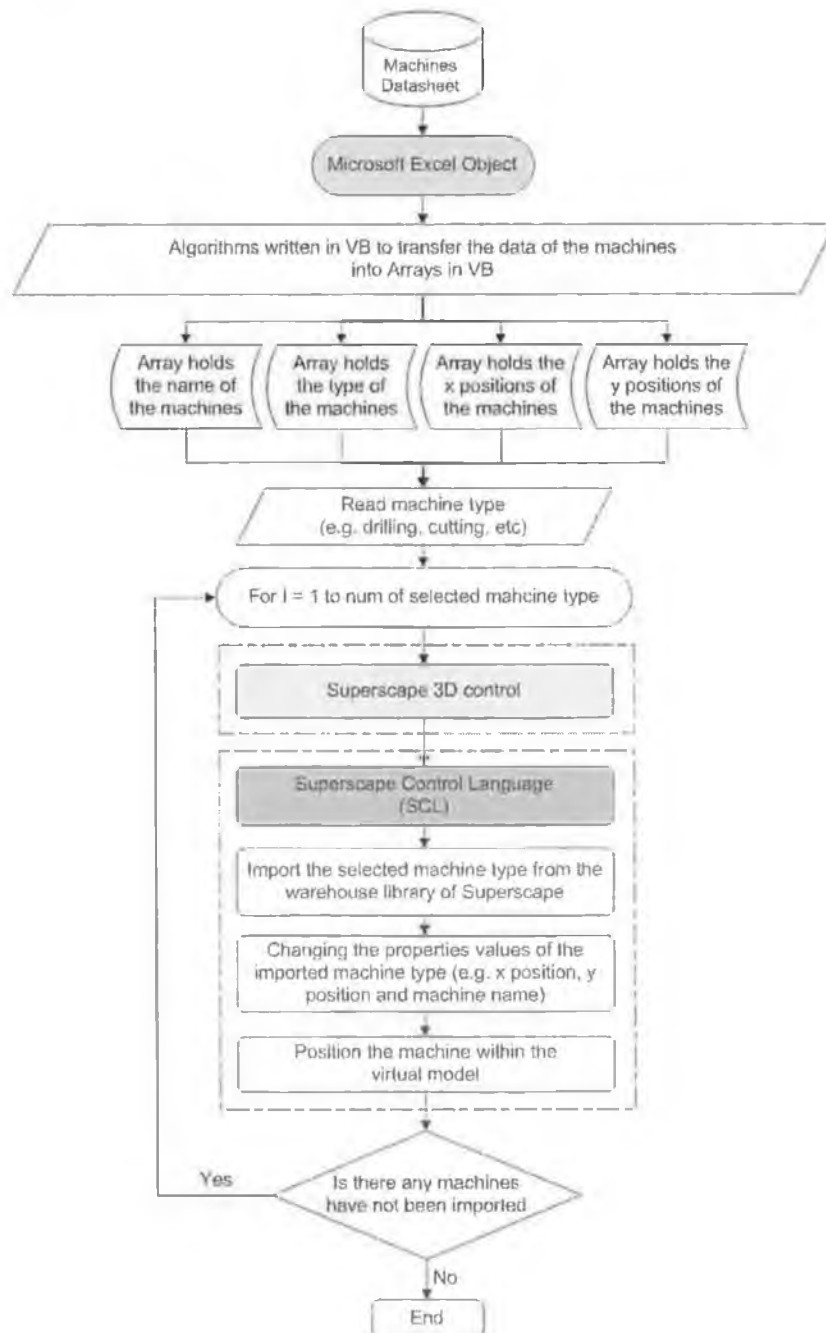


Figure 4.53: Generating Machines Automatically in the Virtual Model

As shown in Figure 4.53, the data required to generate the machines automatically includes machine names, machine types and the x, y positions of each machine. The following VB code was written to communicate with the SCL code through the Superscape 3D control mechanism to generate the machines automatically in the virtual model.

```

For i = 1 To num_Machines
  Select Case type_Machines_DesElements(i)
    Case "Drilling"
      type_Machines_VR(i) = 1
    Case "Saw"
      type_Machines_VR(i) = 2
    Case "Grinding"
      type_Machines_VR(i) = 3
    Case "Polishing"
      type_Machines_VR(i) = 4
    Case "Packaging"
      type_Machines_VR(i) = 5
    Case "Generic"
      type_Machines_VR(i) = 6
  End Select
  If (Superscape3DControl1.GetCounter(1) = 2) Then
    'counter(2) holds the Machine type number
    Superscape3DControl1.SetCounter 2, type_Machines_VR(i)
    'Counter(3) hold the X position of the Machine
    Superscape3DControl1.SetCounter 3, xpos_Machines_DesElements(i)
    'Counter(4) hold the Y position of the Machine
    Superscape3DControl1.SetCounter 4, ypos_Machines_DesElements(i)
    Superscape3DControl1.SetCounter 1, 1
  End If
Next

```

The array named “*type_Machines_VR(i)*” can hold six values where each value represents a different machine type, which include drilling, saw, grinding, polishing, packaging and generic. “*Superscape3DControl1.SetCounter 2, type_Machines_VR(i)*” command is used to transfer the machine type to “*counter(2)*”. “*Superscape3DControl1.SetCounter 3, xpos_Machines_DesElements(i)*” and “*Superscape3DControl1.SetCounter 4, ypos_Machines_DesElements(i)*” commands are used to transfer the values of x position and y position to “*counter(3)*” and “*counter(4)*” respectively. Figure 4.54 shows the SCL code that was used to read the data from VB code to generate the drilling machine and position it within the virtual model.

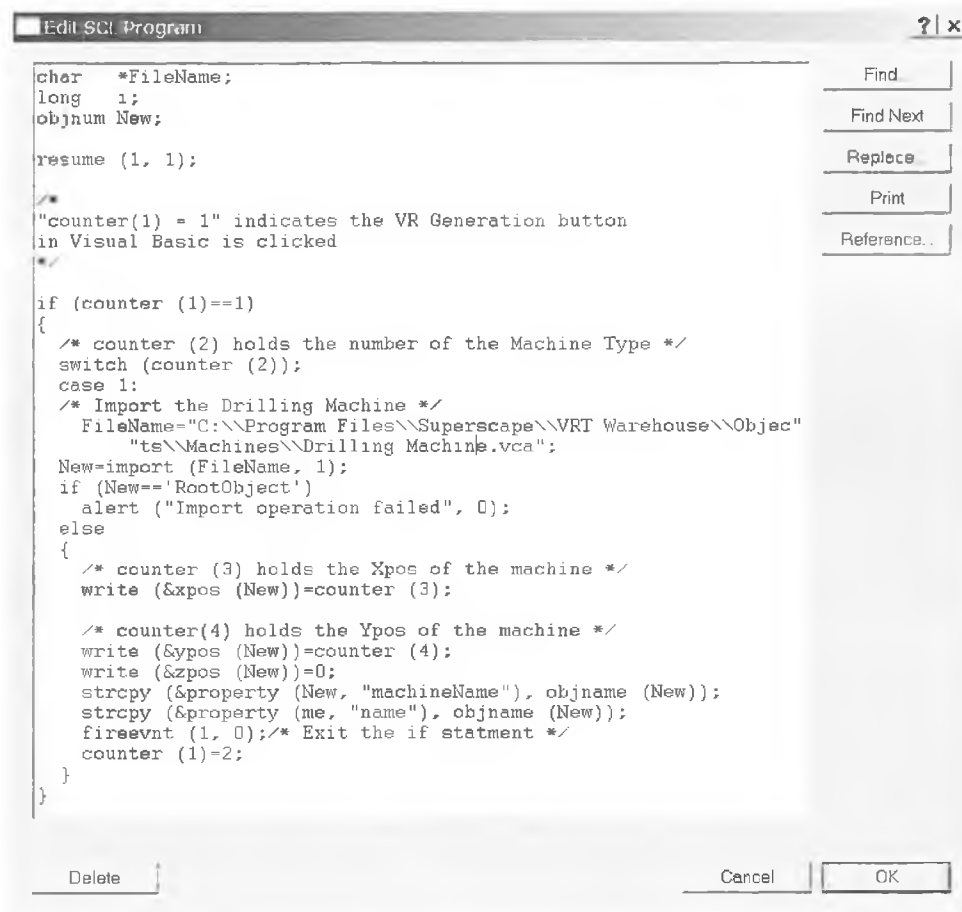


Figure 4.54: SCL Used to Generate the Layout of the Virtual Model

Counter (2) holds the value of the machine type. The SCL command “*switch (counter (2))*” is used to read the machine type that was fed by VB code. “*case 1:*” command represents the value of *counter (2)*, which indicates the machine type is drilling. The drilling machine is imported from the warehouse library using the command “*FileName="C:\\Program Files\\Superscape\\VRT Warehouse\\Objects" \\Machines\\Drilling Machine.vca";*”.

Once the drilling machine is imported, the commands “*write (&xpos (New))=counter (3);*” and “*write (&ypos (New))=counter (4);*” are used to position the drilling machine within the virtual model. Finally, the commands “*strcpy (&property (New, "machineName"), objname (New));*” and “*strcpy (&property (me, "name"), objname (New));*” are used to give the drilling machine a name that has been assigned by the user. All the other virtual objects of the machines, conveyors, buffers are generated automatically using this approach.

4.4.7 Models Running Tool

The Models Running Tool (MRT) is used to load the process plan of a manufacturing system and then to run the simulation model and the virtual model to simulate its activities. The MRT can be accessed by clicking on “*Simulate*” from the main menu of the VR-Simulator software as shown in Figure 4.55.

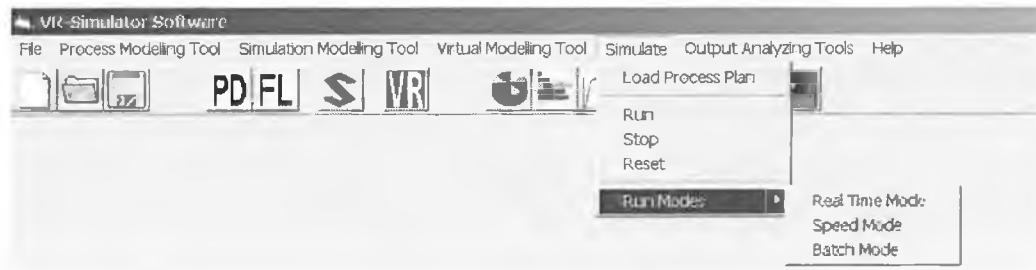


Figure 4.55: Models Run Tool Options

The process plan of a manufacturing system must be loaded before the user can run the models. The process plan can be loaded by clicking on the option “*Load Process Plan*” from the Simulate options. This will display an “Open File” dialog box to enable the user to specify the name of the file where the process plan is saved. The process plan is saved in the same file where the database of the resources is saved. The process plan is stored within two Excel data sheets named “*Process Flow*”, and “*Orders*”.

The process flow data sheet contains information that describes the manufacturing sequence of each part. The orders data sheet includes a predefined order of release for the parts to be manufactured as well as the lot size of each part. Figure 4.56 depicts the algorithm used to transfer the data of the process plan from the Excel data sheets to the simulation model in Witness.

The information that needs to be transferred from the Excel data sheet to the simulation model includes part names, machine names, process names and the cycle time of each process. An algorithm written in VB is used to read the data from the Excel data sheets and then Witness OLE Automation Control is used as a mechanism to transfer these data from VB to the arrays that have been generated in the

simulation model using WCL. The parts names array, machines names array and cycle times array will be associated with each machine. Dummy machines and dummy parts are used to collect the statistics of a manufacturing system.

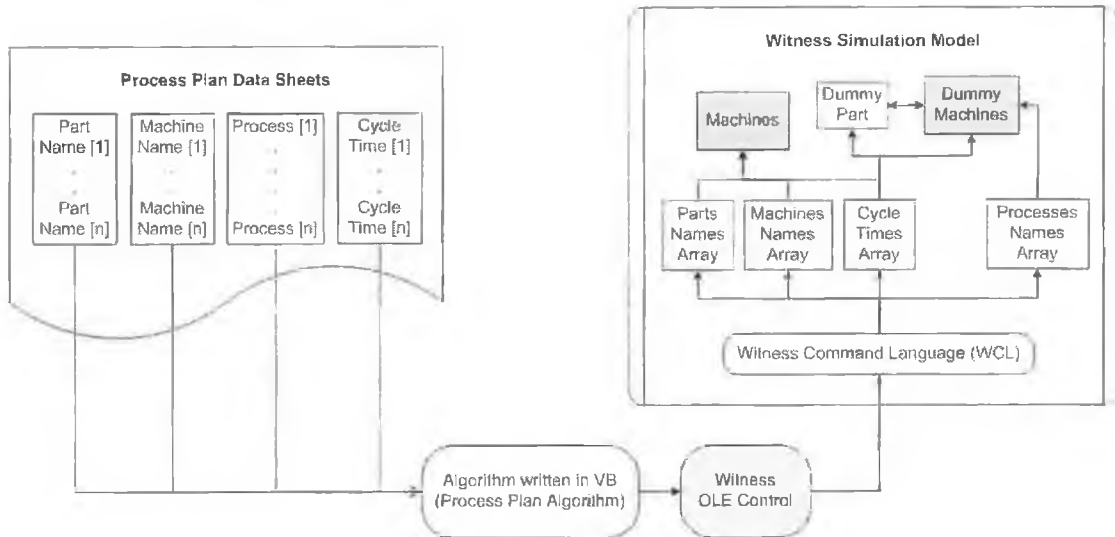


Figure 4.56: Transferring the Process Plan from the Excel Data Sheets to the Simulation Model in Witness

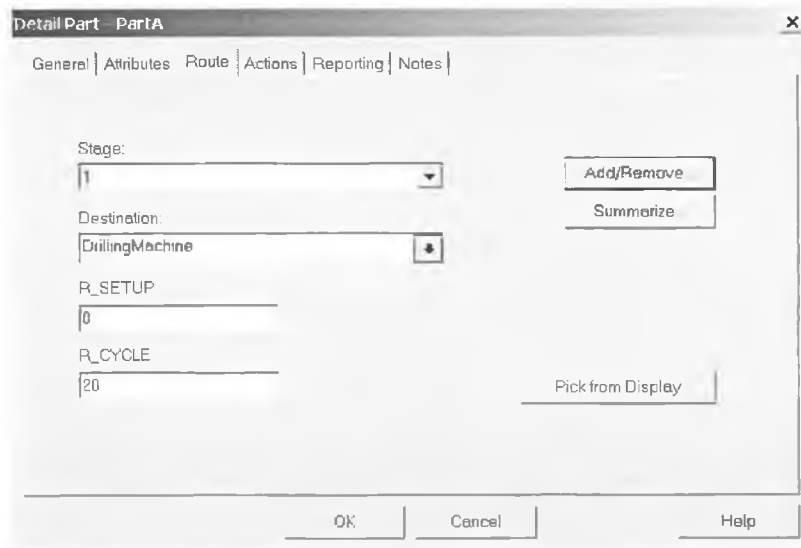


Figure 4.57: Assigning a Route to a Part in Witness

Code written in WCL is used to read the supplied data of the process plan from arrays in the simulation model to detail the parts by assigning a route for each part automatically. The route consists of a number of stages that represent the destinations where the part is to be processed or delivered. Setup time and cycle time are applied

for each stage as shown in Figure 4.57. The output rule of each part is set to “*Push to route*” to enable the part to be pushed to the destinations that have been pre-assigned based on the process plan. Once the process plan is loaded to the VR-Simulator, the user can run the models using the Simulate commands (Run, Stop and Reset) or their icons that are located at the bottom of the main menu of the software. The functions of these commands are as follows:

- **Run:** Runs the simulation model with an animated display in which parts appear to move instantly between the resources on the screen. The virtual model is used to simulate the activities of the manufacturing system in 3D presentation once models are running.
- **Stop:** Halts the current simulation model run as well as the virtual model run.
- **Reset:** Resets the simulation clock to the start of the run. All the simulation resources and the virtual resources are set to the idle state and all the statistics are cleared.
- **Run Modes:** There are three modes to run the simulation model and the virtual model of a manufacturing system. These modes can be accessed from the *Simulate options* which include Real time mode, Speed mode, and Batch mode:
 - **Real time mode:** This mode can be used to simulate the activities of a manufacturing system in real time.
 - **Speed mode:** This mode can be used to allow the user to run the simulation model and the virtual model very quickly.
 - **Batch mode:** This mode is useful to run the simulation more quickly when the user does not require an animated display of the activities. This mode is mainly used to collect statistics for a long run period. Time should be specified at which the run should stop as shown in Figure 4.58.



Figure 4.58: Batch Run Dialog Box

Once the user activates the “*Run command*”, the simulation engine of witness starts processing the data to simulate the activities of the simulation model. The algorithm in Figure 4.59 shows how the activities are simulated.

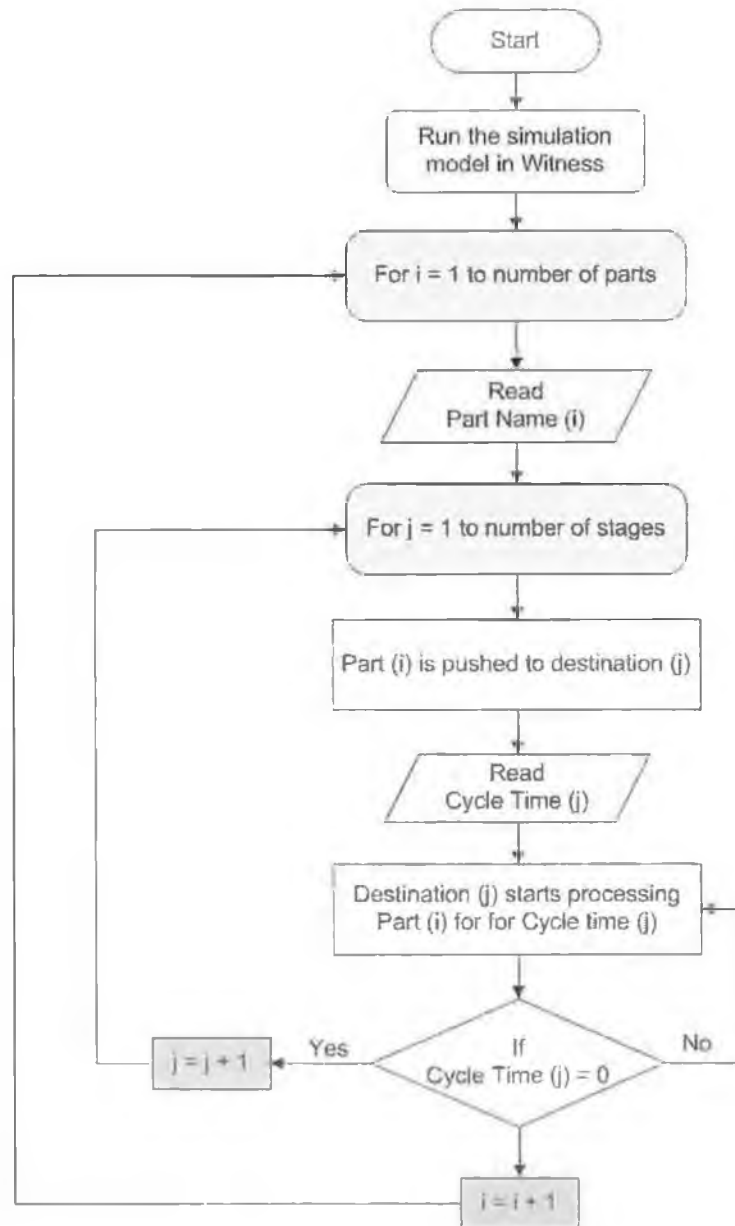


Figure 4.59: Run Execution Algorithm

As mentioned earlier, each part has to follow a number of destinations to be processed or delivered from one location to another. The destinations include machines, conveyors, transporters, AGVs and buffers. The part has to follow a predefined sequence of stages, where the part has to spend specific time at each destination depending on the value of the “*Cycle Time*” attribute. Once the time has

elapsed, the part is pushed to the next destination. The VR-Simulator software does not incorporate a scheduling tool to schedule the priorities of the processes. The simulation engine of Witness is used to select which part should be processed first based on the available destinations and the cycle time of each process.

Another algorithm is also executed when the user activates the run command. This algorithm is used to collect statistics from the simulation model regarding the processes that have been defined by the user. A dummy machine and a dummy part are associated with each process for each machine to be used to collect statistics regarding the process (e.g. coating, cleaning).

4.4.7.1 Simulating the Activates Within the Virtual Model

Once the user activates the run command, the activities on the simulation model will be simulated in a three-dimensional presentation in real time mode within the virtual model. Algorithms developed in VB are used to transfer the data between the simulation model and the virtual model to simulate the activities of the simulation model in a 3D presentation. Linking the simulation model with the virtual model was accomplished using two mechanisms: Witness OLE Control and Superscape 3D Control. Witness OLE control was used to link the developed simulation model with VB algorithms to read the data from the simulation model using WCL, then this data were sent to the virtual model using Superscape 3D control. Figure 4.60 depicts the algorithm that has been developed to link the simulation model with the virtual model.

Code written in SCL is used to command the machines, labourers and the other resources to simulate the activities in real time that matches their activities in the simulation model. The user can interact with the models using voice commands, and navigate the virtual model using Head Mounted Display (HMD) while the models are simulating the activities of a manufacturing system.

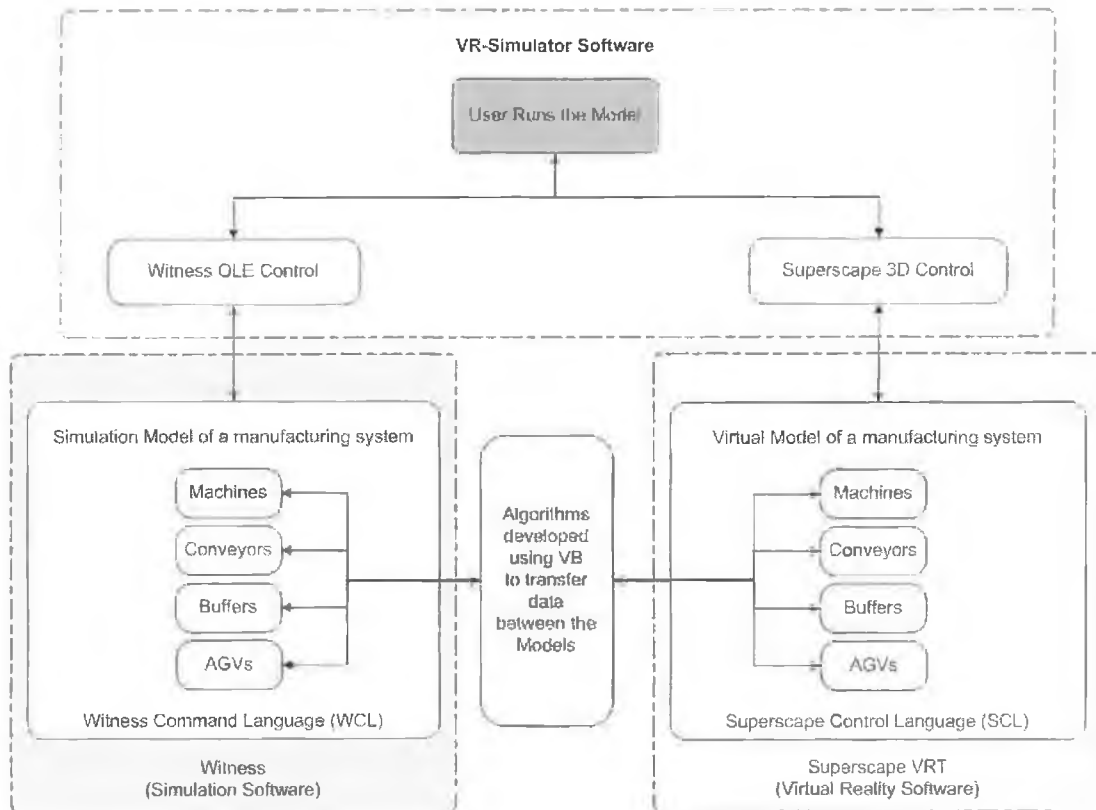


Figure 4.60: Linking the Simulation Model with the Virtual Model

Each resource in the simulation model was linked with the same resource within the virtual model using an individual algorithm. This algorithm executes every micro second in VB to check the current status of the resource, its properties and behaviour from the simulation model and then the algorithm is used to transfer this data to the virtual model to simulate the activities in the 3D presentation in real time. This approach was followed to link all the resources. The algorithm depicted in Figure 4.61 shows an example of how the drilling machine in the simulation model was linked to the 3D object of the drilling machine in the virtual model.

As shown in Figure 4.61, WCL was used to read the current activities of the drilling machine within the simulation model and then convert these activities as data to be saved into variables. Witness OLE control was used as the mechanism to transfer the values of these variables to algorithm developed using VB which reads the parameters of the drilling machine in the simulation model. These parameters include the current status of the drilling machine, setup details, breakdown details, number and names of labourers who are handling the machine.

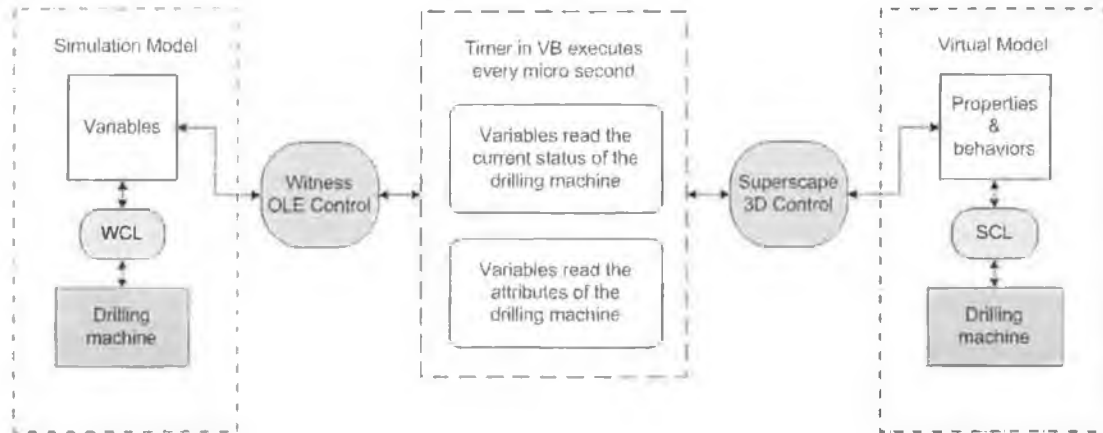


Figure 4.61: Exchanging Data of the Drilling Machine in the Simulation Model with the Drilling Machine in the Virtual Model

The algorithm for transferring the data of the drilling machine from the simulation model to the virtual model executes every micro second to read these parameters and save them within variables in VB. Then, the values of these variables are sent to the properties and behaviours of the drilling machine in the virtual model through the “Superscape 3D Control” mechanism. Algorithms written in SCL were used to command the drilling machine to perform the right action that matches its activities in the simulation model. The same approach was repeated to link all the resources in the simulation model with their 3D objects in the virtual model.

4.4.7.2 Visualizing the Activities of a Manufacturing System

The activities of a manufacturing system were simulated within the virtual model by showing parts placed within the machines, buffers and conveyors. The processes of the machines were represented by a bulb light located on top of each machine. The yellow, green, red and blue lights indicate the idle, busy, breakdown and repair conditions of the machines, respectively. Delivering parts from one location to another was simulated by showing an AGV standing next to the collecting location to load the parts and then the AGV appears next to the delivering location to unload the parts. The movement of the AGV between the two locations was not simulated.

The labourers who are required to handle the machines were simulated within the virtual model by showing a labourer standing next to the machine that needs to be

handled. By clicking on the labourer, a dialog box appears to indicate the number of labourers who are handling the machine at that time. More complex animation can be added to visualise more realistic activities of a manufacturing system. A pilot virtual model was developed using Superscape VRT that simulates an AGV collecting a part from one location and then it delivers the collected parts to the saw machine to be processed. The virtual AGV was created by using the shape and world editors of Superscape. Because it is time consuming to create some parts of the AGV using the shape editor of Superscape, AutoCAD were used to create these parts and then these parts have been imported into the world editor to assemble them to construct the virtual object of the AGV.

The AGV can move forward, backward, pivot in place to the right and to the left by 360 degree. The AGV has a buffer to store the parts while delivering them from one location to another. The arm of the AGV has been constructed as a combination of two kinds of robot arms, which are Cartesian Coordinates and Cylindrical Coordinates. Figure 4.62 and Figure 4.63 shows the Cartesian coordinate and Cylindrical coordinate configurations.

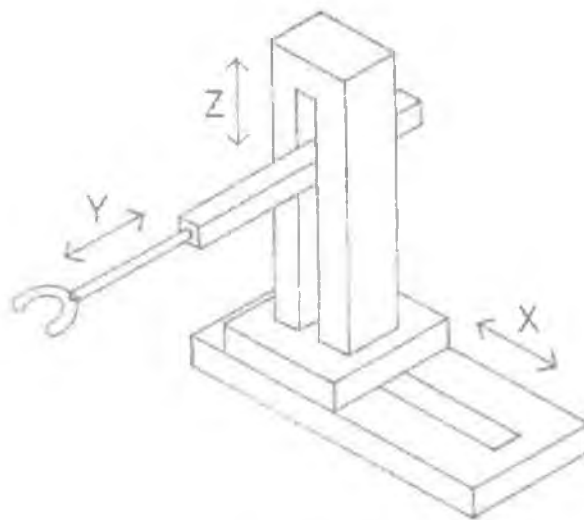


Figure 4.62: Cartesian Coordinate

The Cartesian coordinate configuration consists of a column and an arm. It is sometimes called an x-y-z robot, indicating the axes of motion. The x-axis is lateral motion, the y-axis is longitudinal motion, and the z-axis is vertical motion. Thus, the

arm can move up and down on the z-axis, and slide along its base on the x-axis; and then it can telescope to move to and from the work area on the y-axis.

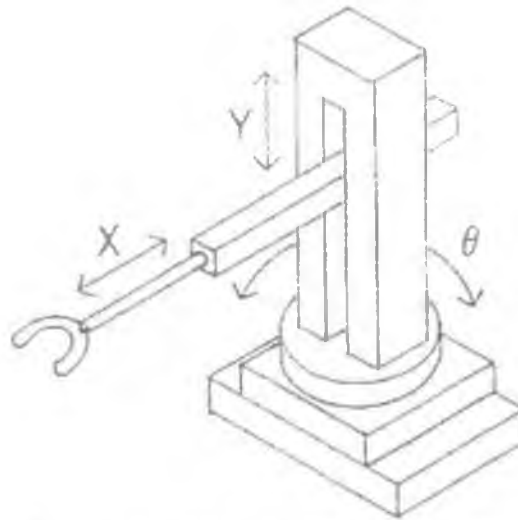


Figure 4.63: Cylindrical Coordinate

The cylindrical coordinate configuration typically has two linear axes: vertical and radial, and the rotations are achieved by a separate axis. The dedicated vertical axis gives smooth, rapid motion in the vertical direction, and the radial axis allows the robot to retract or extend quickly. These motions are usually useful in assembly, material handling, and machine loading/unloading. The gripper of the arm was constructed from two fingers to enable the AGV to grasp the parts during the work cycle.

The integration of Cylindrical coordinate and Cartesian coordinate configurations enable the arm of the AGV to grab the parts from the collecting location and store them in the buffer of the AGV, and vice versa. Figure 4.64 shows the developed virtual AGV.

Full movement and animation have been modelled to show how the AGV collects parts, moves from one location to another, rotates and delivers the parts to a machine to be processed. Figure 4.65 shows a snapshot of an AGV delivering a part to the saw machine to be cut.

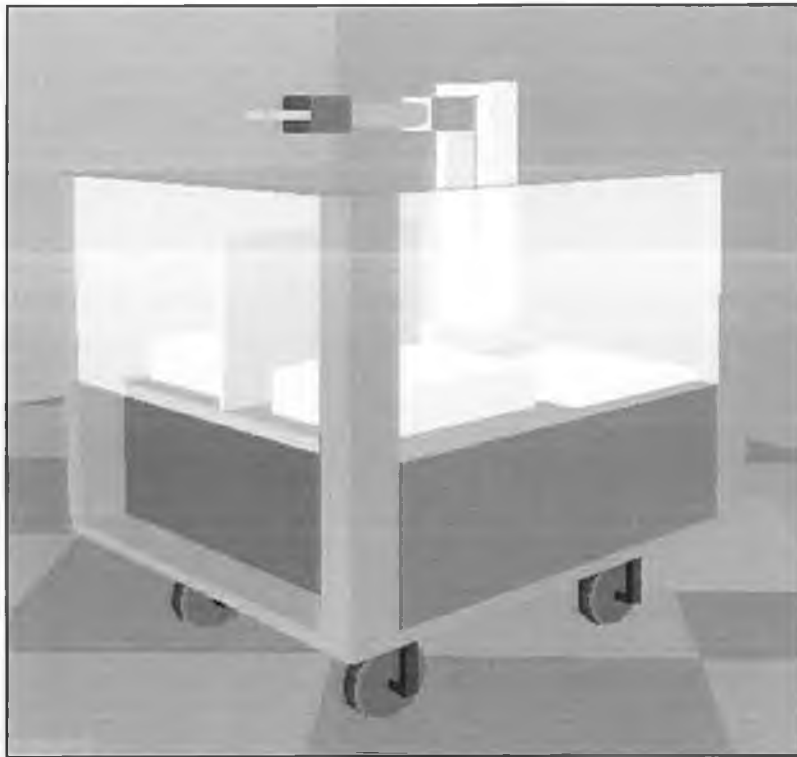


Figure 4.64: Virtual AGV

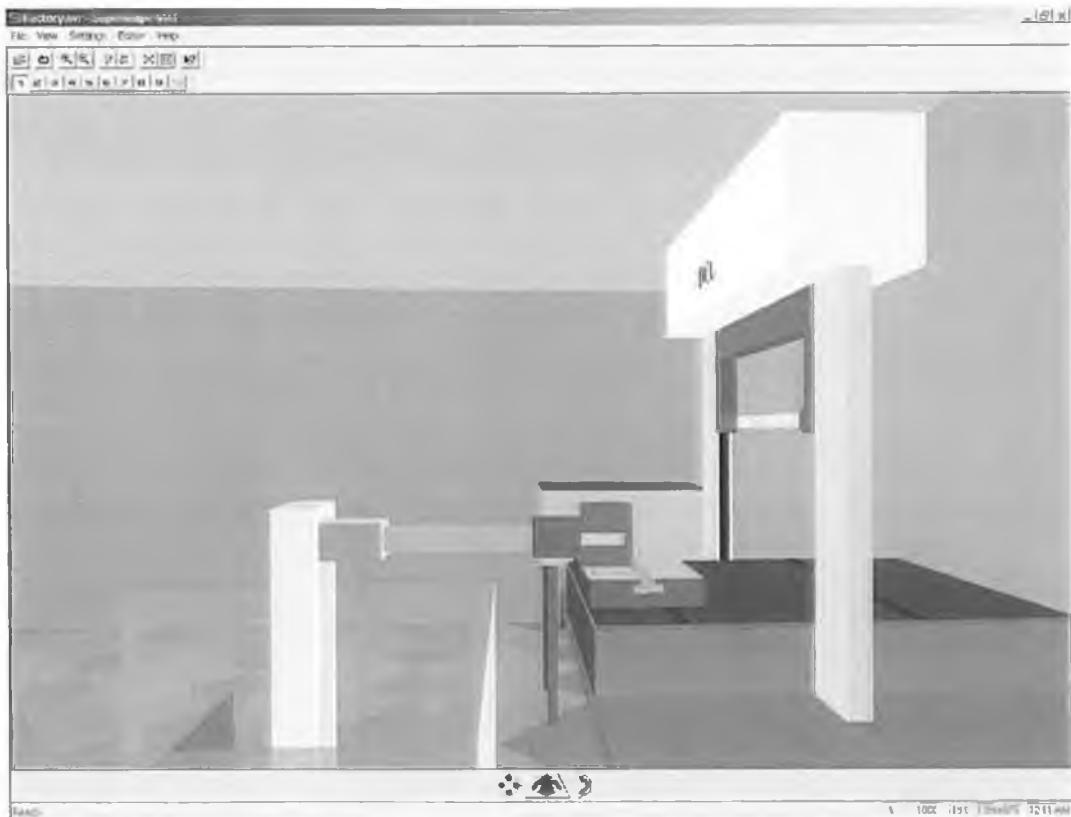


Figure 4.65: An AGV Delivering Parts to the Saw Machine

Once the part is delivered to the saw machine, the saw tool is used to cut the part into two pieces. Then, the arm of the AGV is used to collect the cut part to storage it in its buffer. The same principle can be used to model complex animation to visualise the processes of the other resources.

4.4.6.3 Navigating the Virtual Model

A viewpoint was attached to a movable object, which was made of a holding group object that has a dynamics attribute and a child object that has a rotation attribute. The holding group represents a human body, and the child object represents its head. The user can navigate the virtual world of a manufacturing system by changing the position of the object by using the movement bar (Figure 4.66). The viewpoint was attached to the object so that the user, while navigating the virtual model, has a sense of a walking person “viewing the environment from about 1.6 meters above the ground”. The user can also navigate the virtual model by using a joystick or 3D mouse, which makes the navigation easier.

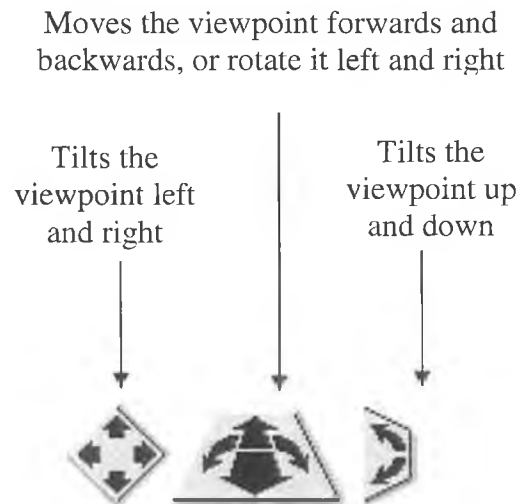


Figure 4.66: Navigation Bar

A Monitor and a Head Mounted Display were used as output devices to view the virtual environment of a manufacturing system. The HMD is coupled with a tracker which enables the user to navigate around the world by moving his/her head to have a better feeling of being present within the virtual model.

4.4.7.4 Interacting with the Models of the VR-Simulator

Most of the simulation packages available on the market today do not allow real time interaction with their simulation models and virtual models when they are running. A new method is introduced by the author to allow users to interact with the simulation model of a manufacturing system through the virtual model in real time. The users can interact with the virtual model by a mouse or voice commands. Once a command is given by the user, the virtual model mimics the simulation model to simulate the same activity simultaneously in real time. Figure 4.67 illustrates the schematic diagram of how the user can interact with the models of the VR-Simulator in real time.

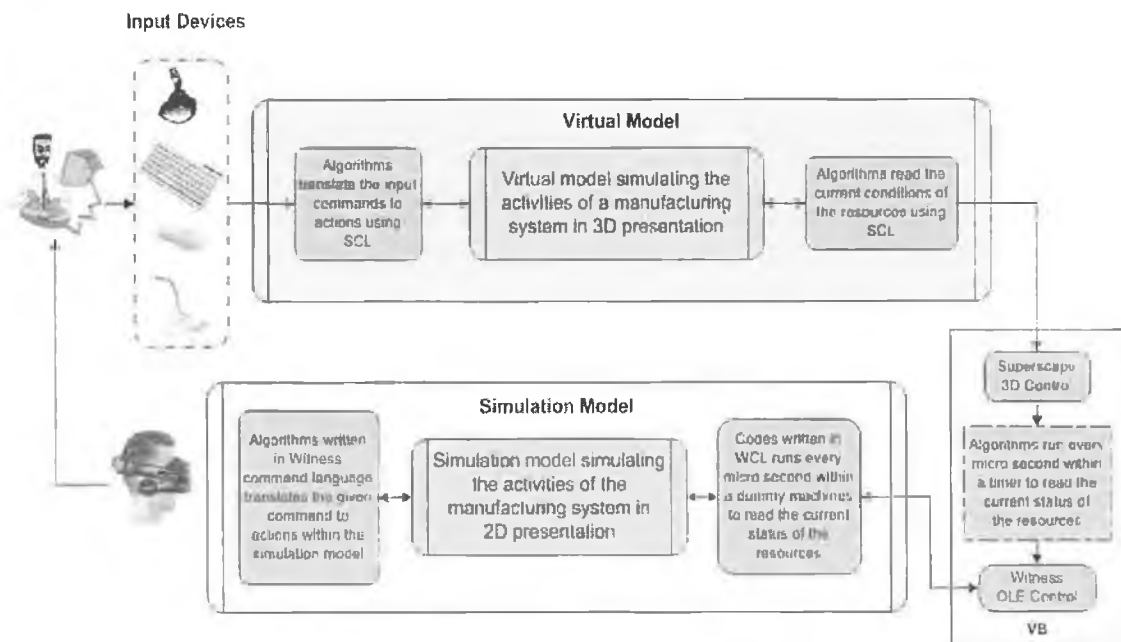


Figure 4.67: Flowchart Illustrating How the User Interacts with the Models of a Manufacturing System

Mouse or voice commands can be used to interact with the machines in the virtual models. Algorithms written in VB run every micro second to monitor the activities of the virtual model. Once the user interacts with one of the machines in the virtual model, a message will be sent to the simulation model to perform the right action based on the command that has been given. The interaction with the models is limited to stop, run, breakdown and repair the machines. Superscape 3D control and Witness OLE control are used as ActiveX controls to link the running simulation

model with the virtual model to transfer the commands that have been given by the user from the virtual model to the simulation model.

As shown in Figure 4.67, algorithms written by SCL are used to translate the user input commands to actions in the virtual model. The conditions of the machines are stored as numerical data within its properties. These properties are watched at all times by code written in SCL to check if the user has interacted with models. Once the user interacts with the models, a message will be sent to the simulation model through the Superscape 3D control and Witness OLE control mechanisms. Witness Command Language translates the received message from the virtual model into actions in the simulation model to simulate the same activity.

Once the user interacts with the virtual model by clicking on one of the machines, an interaction dialog box appears as shown in Figure 4.68.



Figure 4.68: Interaction Dialog Box

As shown in Figure 4.68, the user can interact with any machine to stop it for a period of time and then run it again. Also, the user can enforce breakdown and repair of the machines in real time. Real time interaction is a very useful method to conduct different experiments with the models without the need to change the logic and the code of the models, especially for non-expert users.

▪ *Voice commands*

The VR-Simulator software has an extra feature not available in simulation packages available on the market today. It enables the user to interact with the generated models using voice commands for additional flexibility. One popular use of speech recognition is the user's ability to speak a word or a phrase and have the computer perform a specific task based on what it heard. This method of speech recognition is commonly referred to as "Command and Control." Some popular uses for this type of speech recognition include the following:

- Automatic e-mail handling
- Computer activated security systems
- Computer controlled devices
- Games
- Remote data entry
- Learning systems

Voice input is one of the most natural ways of human communication. Since speech recognition has become quite robust and almost real-time, it was decided to integrate it with the VR-Simulator software. The voice command, as an input method for interaction, can be very useful especially if the user uses VR devices to interact and view the virtual model. For example, if the user is wearing a HMD or wearing gloves, then the user will not be able to see the keyboard and interact with it, so it is going to be very difficult to interact with the virtual model but once the interaction can be made using voice commands then the user can have more freedom and flexibility to interact with the virtual environment.

Several different voice recognition products currently exist on the marketplace, and viable choices are greater in number than they were only a few years ago. Microsoft speech engine, freely downloadable on the internet has been incorporated into the VR-Simulator system. Speech recognition involves capturing and digitising sound waves, converting them into basic language units or phonemes, constructing words

from phonemes, and contextually analysing the words to ensure correct spelling for words. Figure 4.69 illustrates this process.

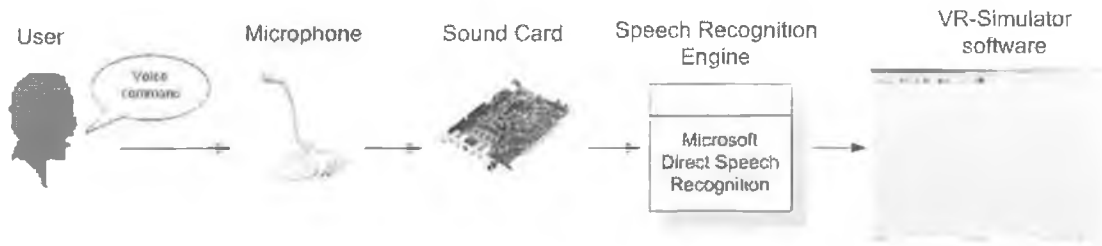


Figure 4.69: Speech Recognition Process Flow

The speech engine needs to be trained to the type of microphone that is to be used as well as the individual speech pattern of the user to increase accuracy of detecting the user commands. Figure 4.70 shows a snapshot of the speech engine training software from Microsoft that needs to be used to train the engine for the user's voice.



Figure 4.70: Snapshot of the Voice Training Software

After the software has been trained to recognise the voice of the user, the user can easily interact with the generated models of the VR-Simulator software. Voice commands can be used to interact with the machines of a manufacturing system, where the user can stop, run, breakdown and repair the machines. Table 4.1 presents the voice commands that can be given to the machines of the generated models.

Table 4.1: Voice Commands and Their Functions

Voice Command	Function
Name of the machine (e.g. Spancrete)	Activate the machine
Stop	Stop the activated machine
Run	Run the activated machine
Breakdown	Breakdown the activated machine
Repair	Repair the activated machine

Figure 4.71 shows the flowchart of how the voice commands were translated into actions within the models. When the user gives a command through a microphone, a program written in visual basic compares this command with predefined words that represent the machine names and the interaction commands. When a match is found, a text file opens and some data are written in it. Then a program written in SCL and linked to an algorithm developed using VB, is used to read the data from that file and then executes the appropriate function in the virtual model and the simulation model of a manufacturing system.

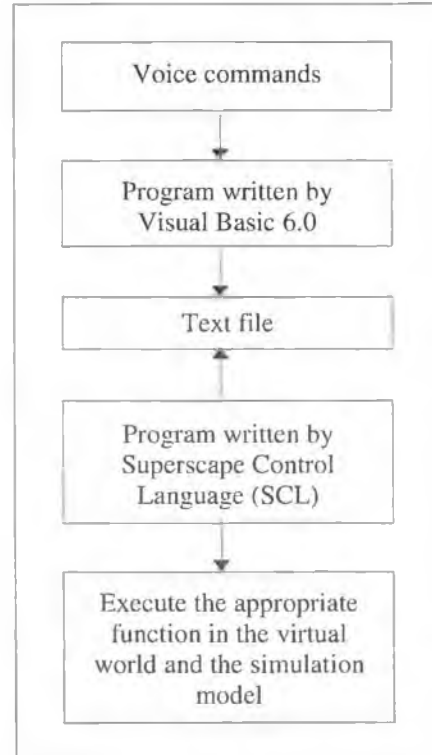


Figure 4.71: Voice Commands Interactions

▪ *Linking the VR-Simulator with external hardware*

A new and evolving area for simulation technology is real-time application. In this context, the simulation model is used to control a real manufacturing system. Whereby, the simulation model is purposely slowed down to run in real time and to execute in parallel to the real system. During execution, the model exchanges messages with the controller (either a computer or a person). These messages allow the model and real system to remain synchronised. They are also used by the model to issue commands to the real system to initiate specific tasks [130].

A simulation-based control system has a number of important advantages over traditional control systems. One very important advantage is that a simulation-based control system can, at any point in time, use its system model to predict the system status at some time in the future [97]. A pilot physical model has been developed and linked to the VR-Simulator software to support real time control of real manufacturing systems. The pilot model consists of four switches which represent the input commands to the models and physical bulbs that represent the condition of the machines. An interface circuit has been developed by the author to act as an interface between the physical switches, bulbs and the parallel port of the PC (Figure 4.72).

The Parallel Port is one of the most commonly used ports for interfacing with external hardware. The parallel port is made up of three different sections, namely the *data lines*, *control lines* and *status lines*. There are eight data lines, and they are the primary means of getting information out of the port. The control lines are the four other outputs. They are meant to provide control signals to the printer (such as form feed or initialise). The status lines are a standard parallel port's only inputs. There are five of them. They were meant to allow the printer to communicate things such as error, paper out and busy to the PC.

The ActiveX control "WIN95IO.DLL" was used to link the parallel port with the VR-Simulator. The five status lines were used to read the data from the parallel port. Four switches have been attached to the status lines; the functions of these switches

are to send commands to the generated models using the VR-Simulator software. An algorithm developed in VB was used to enable the user to control the machines within the simulation model and the virtual model in real time. Two switches were used to breakdown the two machines, while the other two were used to repair them.

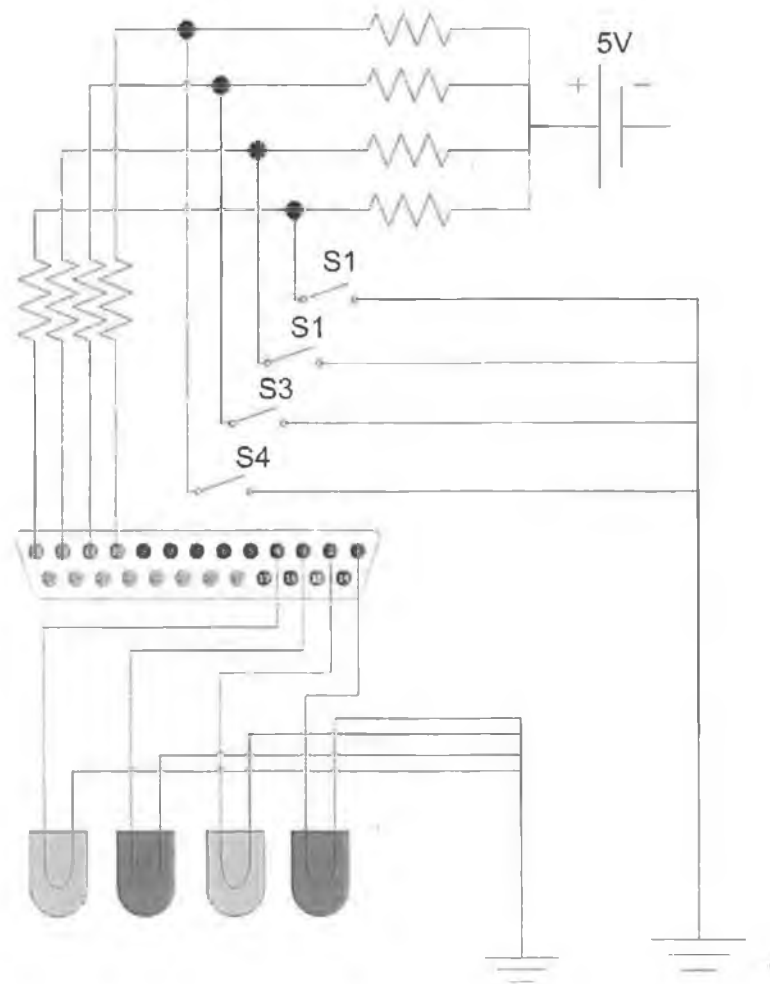


Figure 4.72: Interface Circuit for the External Switches

The command “[variable] = vbInp(889)” is used to read the current status of the status lines. Algorithms written in VB, SCL and WCL were used to translate the received commands to breakdown and repair the machines within the generated models. Algorithms written in VB, SCL and WCL were used to read the activities of the two machines within the simulation and virtual models, which include busy and idle status. Then, the command “vbOut 888,1” was used to send the current status of

the machine to the physical bulb. Each machine within the models was linked to two bulbs. Once the bulb is on, it represents the busy status of the machine and when the bulb is off it represents the idle status of the machine.

4.4.8 Output Analysing Tools

The output analysing tools are used to predict and measure the performance of the activities that are being simulated, or to compare the performance of two or more alternative system designs of a manufacturing system. There are four types of output analysing tools that can be used to display the statistics of a manufacturing system. The output analysing tools of the VR-Simulator are Pie Charts, Timeseries, Gantt Charts and Reports. The output analysing tools can be accessed from the main menu options of the VR-Simulator or by clicking on the icons that represent these tools as shown in Figure 4.73.



Figure 4.73: Output Analysing Tools

- **Pie Charts:** An ActiveX control from Microsoft called “*Microsoft Chart Control*” was chosen to model the Pie chart of the VR-Simulator. It was used to measure the utilisation of the machines and the labourers, which can assist in determining if more labourers are required or not. If the results show that the utilisation of two labourers within a day is 20%. That means one of them can handle the duties of both of them and there is no need for the other. Machines utilisation is important to schedule the processes of a manufacturing system.
- **Timeseries:** The same ActiveX control that was used to display the Pie chartx was used to model the timeseries. The timeseries is a graphical element that presents the simulation results in the form of a graph that plots values taken from the simulation model against time. This tool is useful for determining the trends or cycles that underlay the model, since they provide a history of the specified value

as well as a static mean and standard deviation. A timeseries is analogous to a pen plotter; it plots values over simulated time. The VR-Simulator takes a reading from the simulation model at specified intervals and plots it on a graph, building up a series of values over a period of time. Once the space on the screen allocated to the timeseries has been exhausted, the graph scrolls to allow new plots to be made. The timeseries graph was also used to watch the activities of the labourers against the time. This tool could be used to help decision makers to assign and organise the duty of labourers in a manufacturing system.

- **Gantt chart:** A Gantt chart tool was developed using a number of VB controls (e.g. “*PictureBox*”, “*Shapes*” and “*ScrollBars*”) controlled by functions and algorithms written in VB. The main function of the Gantt chart is to allow the user to monitor the activities of the processes on the machines as they happen against time. The Gantt chart helps the user to plan the tasks that need to be completed. It can also be used as a basic tool for scheduling when these tasks are to be carried out in the future. The Gantt chart is represented by bars with different colours where each colour represents a different process based on the processes that are predefined by the user.
- **Reports:** This tool allows the users to examine the performance of the resources in the model and provide information about their activities at any time the user specifies. Reports can help identify areas where the user may be able to improve the system’s operation. Eight Report dialog boxes have been created to display statistical information about the products, machines, labourers, conveyors, AGVs, transporters, processes and the buffers as shown in Figures 4.74 through 4.81 respectively. The information for these reports can be generated in Excel files which can be used for documentation or other software packages to make the VR-Simulator as an open system to be integrated with other tools.

Algorithms written in VB linked with algorithms written in WCL were used to collect data from the Simulation model in Witness to feed the output analysing tools with the required data to display the statistics.

The screenshot shows a window titled "Reports (Machines)". Inside, there is a section titled "Machines Statistics". It contains four input fields: "Name:" with a dropdown arrow, "Idle (%):", "Busy (%):", and "No. Of Operations:". Each field is followed by a small rectangular box for data entry.

Figure 4.74: Machines Reports

The screenshot shows a window titled "Reports (Products)". Inside, there is a section titled "Products Statistics". It contains five input fields: "Name:" with a dropdown arrow, "No. Entered:", "No. Shipped:", "Throughput time:", and "W.I.P:". Each field is followed by a small rectangular box for data entry.

Figure 4.75: Products Reports

The screenshot shows a window titled "Reports (Labourers)". Inside, there is a section titled "Labourers Statistics". It contains five input fields: "Name:" with a dropdown arrow, "Busy:", "Idle:", "No. of Job Started:", and "No. of Job Ended:". Each field is followed by a small rectangular box for data entry.

Figure 4.76: Labourers Reports

The screenshot shows a window titled "Reports (Conveyors)". Inside, there is a section titled "Conveyors Statistics". It contains five input fields: "Name:" with a dropdown arrow, "Empty (%):", "Busy (%):", "Now On:", and "Total On:". Each field is followed by a small rectangular box for data entry.

Figure 4.77: Conveyors Reports

The screenshot shows a window titled "Reports (AGVs)". Inside, there is a section titled "AGVs Statistics". It contains five input fields: "Name:" with a dropdown arrow, "Idle (%):", "Busy (%):", "Delivered:", and "Now On:". Each field is followed by a small rectangular box for data entry.

Figure 4.78: AGVs Reports

The screenshot shows a window titled "Reports (Transporters)". Inside, there is a section titled "Transporters Statistics". It contains four input fields: "Name:" with a dropdown arrow, "Idle (%):", "Busy (%):", and "Time Travelled:". Each field is followed by a small rectangular box for data entry.

Figure 4.79: Transporters Reports



The screenshot shows a window titled "Reports (Processes)". Inside, there is a section titled "Process Statistics". It contains six input fields: "Machine Name:" with a dropdown arrow, "Process Name:" with a dropdown arrow, "Highest Time:" with a text box, "Best Time:" with a text box, "Average Time:" with a text box, and "No. of Samples:" with a text box.

Figure 4.80: Processes Reports



The screenshot shows a window titled "Reports (Buffers)". Inside, there is a section titled "Buffers Statistics". It contains four input fields: "Name:" with a dropdown arrow, "Total In:" with a text box, "Total Out:" with a text box, and "Now On:" with a text box.

Figure 4.81: Buffers Reports

4.5 CONCLUSIONS

This chapter has discussed the development of the VR-Simulator software to address manufacturing systems. The main advantage of the VR-Simulator software is to allow non-expert personnel to develop a robust simulation model and create a virtual model of manufacturing systems to study its behaviour. This software also reduces the time needed to develop complex simulation models by expert users. The MGT tool overcomes some of the limitations of the simulation packages (as mentioned earlier).

The main functions of the SMT of the VR-Simulator software are:

- To measure the performance of an existing manufacturing system.
- To examine the impact of alternative design & operating strategies on system performance.
- To support decision making.

The VMT of the VR-Simulator software can be used to support the simulation models as follows:

- To enhance confidence in the use of simulation tools.
- To verify and validate the simulation model.

- To provide a virtual environment for better understanding of the processes.
- To allow the users to interact and with the simulation model during runtime.
- To enable unskilled users to understand and participate in the planning process.
- To be used as a training tool for employees.
- To support facility layout planning.

CHAPTER 5

VR-SIMULATOR SOFTWARE TESTING AND EVALUATION

5.1 INTRODUCTION

Having described the conceptual design and development steps of the VR-Simulator software in Chapter four, the objective of this chapter is to evaluate the VR-Simulator software by conducting a real case study and a hypothetical case study of two different manufacturing systems.

It is now important to verify that the general functionalities of the VR-Simulator, such as editing functions, generating results, transferring data and operational and modelling logics work properly. The verification process was an exceedingly time consuming task, which required debugging the VB, WCL and SCL codes that have been used to develop the VR-Simulator software. The verification process of the VR-Simulator is presented to show the credibility and reliability of the developed software and therefore to increase the confidence in the outcome of the VR-Simulator that may be used by managers or other decision makers.

This chapter also evaluates the VR-Simulator by developing models for two different manufacturing system types to determine how well the generated model represents the system under study. The evaluation of the VR-Simulator is performed to examine its accuracy in developing models for different manufacturing systems and therefore identifying the domain of applicability of the VR-Simulator software. In evaluating the VR-Simulator a number of observations about the software capabilities were gained later on.

5.2 VR-SIMULATOR SOFTWARE TESTING

One of the most difficult tasks in developing the VR-Simulator software was the verification process, as it involved performing a number of tests to isolate and remove errors from the programming logic of the software. Tests were carried out throughout the entire software development phase to make sure the software was error free. Verification is actually a process to ensure that the VR-Simulator software operates according to its conceptual design (see Figure 5.1).

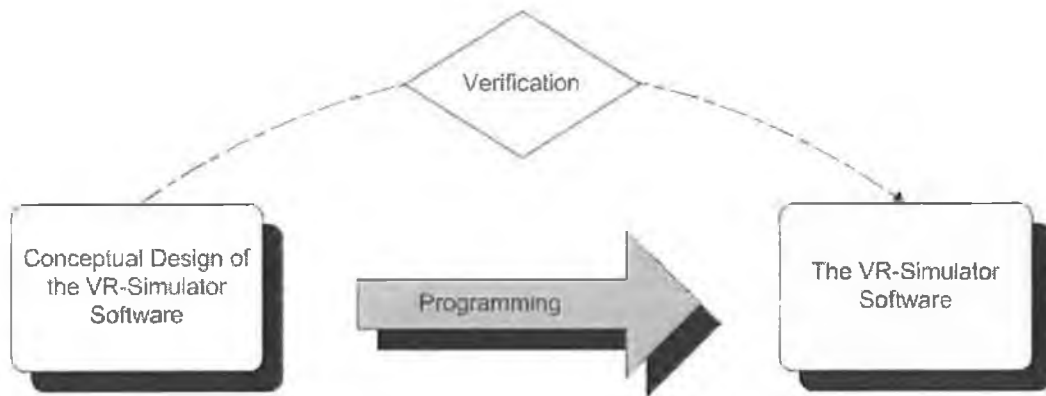


Figure 5.1: Verification Process of the VR-Simulator Software

The Verification of the VR-Simulator was accomplished by testing the developed software in stages as follows:

1. Tests were carried out to check if the main tools of the VR-Simulator were working correctly. These included testing the Process Modelling Tool (PMT), Simulation Modelling Tool (SMT) and the Virtual Modelling Tool (VMT). Each has been tested individually as follows:
 - The PMT is used to model the behaviour and operational logics of the resources of the VR-Simulator. It has been tested to make sure that the functions of the detail dialog pages for each resource were working properly. These functions include defining the resources and the attributes associated with each resource. The validation was done by

presenting the output of each detail dialog page on a different Excel data sheet.

- The SMT is used to generate resources automatically to construct the simulation model and has been tested to verify if the resources were generated correctly within the simulation model in Witness, and if their attributes were properly assigned to match their characteristics as being identified by the user. The Witness environment was used as a tool to verify the resources generated by the SMT.
 - The VMT is used to generate virtual objects that represent the resources of a manufacturing system to create a virtual model automatically which represents a simulation model in 3D presentation. The VMT has been tested to verify if the layout of the virtual model matched the layout of the manufacturing system, and also to make sure that all the resources have been generated automatically within the virtual model with the right attributes and behaviour. SCL was used as a tool to verify the objects generated by the VMT.
 - The data of the independent database were verified by defining an example of a manufacturing system using the PMT and then comparing the generated data in Excel data sheets with the modelled system.
 - Functions written in VB were used to verify data transferring between the three tools of the software (PMT, SMT and VMT).
2. Tests were carried out to check if the developed VR-Simulator provides correct results. This was accomplished by generating small models using the developed software and comparing the results with models developed in the Witness simulation package manually.

3. Tests were carried out to verify if the VR-Simulator could correctly store the information gathered by the PMT for a manufacturing system within an independent database.
4. Tests were carried out to make sure that the user can interact with the resources in real time.
5. Tests were carried out to verify if the simulation model mimics the virtual model in real time to visualise the activities of a manufacturing system as they happen in the simulation model.
6. The above mentioned tests were numerous and carried out throughout the development phase. For this reasons they are not documented here.

5.3 VR-SIMULATOR SOFTWARE EVALUATIONS

The evaluation of the VR-Simulator software is aimed at specifying the scope of the VR-Simulator applications and measuring the accuracy of the developed software in representing the intended applications. For this reason the VR-Simulator capability was examined by developing the following two models:

- A real case study of a job shop model.
- Hypothetical example of a flow shop model.

The following subsections provide descriptions of how these models are developed and analysed using the VR-Simulator software.

5.3.1 Real case study of a Job Shop model

The first developed model using the VR-Simulator software was an existing manufacturing system that does treatment operations. The manufacturing system consists of five machines called Ovens which coat sheets with a special material for heat resistance. The sheets can be coated more than one time. Different processes can

be associated with each Oven as shown in Table 5.1. The manufacturing system does not have any material handling devices. The labourers are used to carry the rolls between the Ovens.

Table 5.1: Processes Names and Their Descriptions

Process Name	Description of the Process
Setup	Setting up the Oven to place a new roll.
Pass	Coating the sheets with special material.
Final Pass	Final coating of the sheets.
Std Changeover	Placing a sheet roll within the Oven to be coated.
Colour Changeover	Changing the liquid of the Oven to coat the sheet roll with a different colour as requested.
Maintenance	Maintaining the Oven.
Cleaning	Cleaning the Oven.

Each process within the job shop model is handled by a number of labourers. Table 5.2 shows the labourers required for each process.

Table 5.2: Labourers Required to Handle Each Process

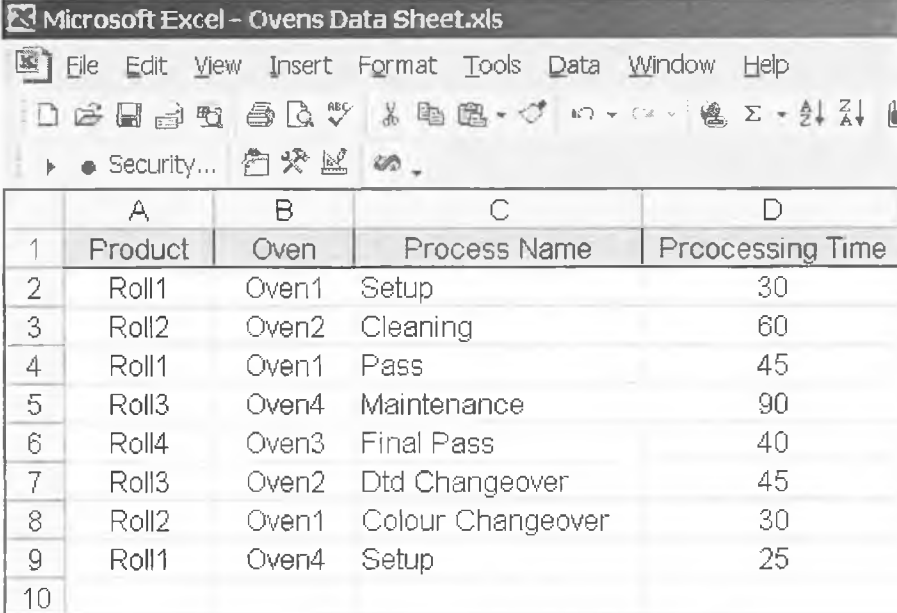
Process Name	Number of labourers
Setup	2
Pass	1
Final Pass	1
Std Changeover	2
Colour Changeover	2
Maintenance	3
Cleaning	3

The data in Table 5.1 and Table 5.2 were gathered by consulting the experts of the manufacturing system.

The main objective for developing a simulation model and a virtual model of the Job Shop was to measure its performance in terms of:

- Utilisation of each Oven.
- Utilisation of each Labourer.
- Monitoring the processes on each Oven.

Due to the confidentiality of the process plan of the manufacturing system, only a sample of the given data is used in this section to show how the VR-Simulator software was used to model and study the Job Shop model. Figure 5.2 shows an Excel data sheet containing information about some of the jobs, such as the name of the product that needs to be coated; the name of the Oven required for processing the product; the name of the process and finally the cycle time the Oven needs to process the product.



	A	B	C	D
1	Product	Oven	Process Name	Processing Time
2	Roll1	Oven1	Setup	30
3	Roll2	Oven2	Cleaning	60
4	Roll1	Oven1	Pass	45
5	Roll3	Oven4	Maintenance	90
6	Roll4	Oven3	Final Pass	40
7	Roll3	Oven2	Dtd Changeover	45
8	Roll2	Oven1	Colour Changeover	30
9	Roll1	Oven4	Setup	25
10				

Figure 5.2: Sample Process Plan of the Job Shop Model

5.3.1.1 Defining the Job Shop Model

Before the resources and the process plan of the Job Shop model were defined using the VR-Simulator software, an Excel file was created to store its database. The Process Modelling Tool (PMT) of the VR-Simulator defines all the resources and the process plan. The resources of the job shop model include shifts, labourers, rolls of sheets, as products, and Ovens. The *Shift Detail Dialog Page* of the PMT was used to

define two shifts, one applied to the labourers and the other to the Ovens. Figure 5.3 shows the two shifts patterns that were defined.

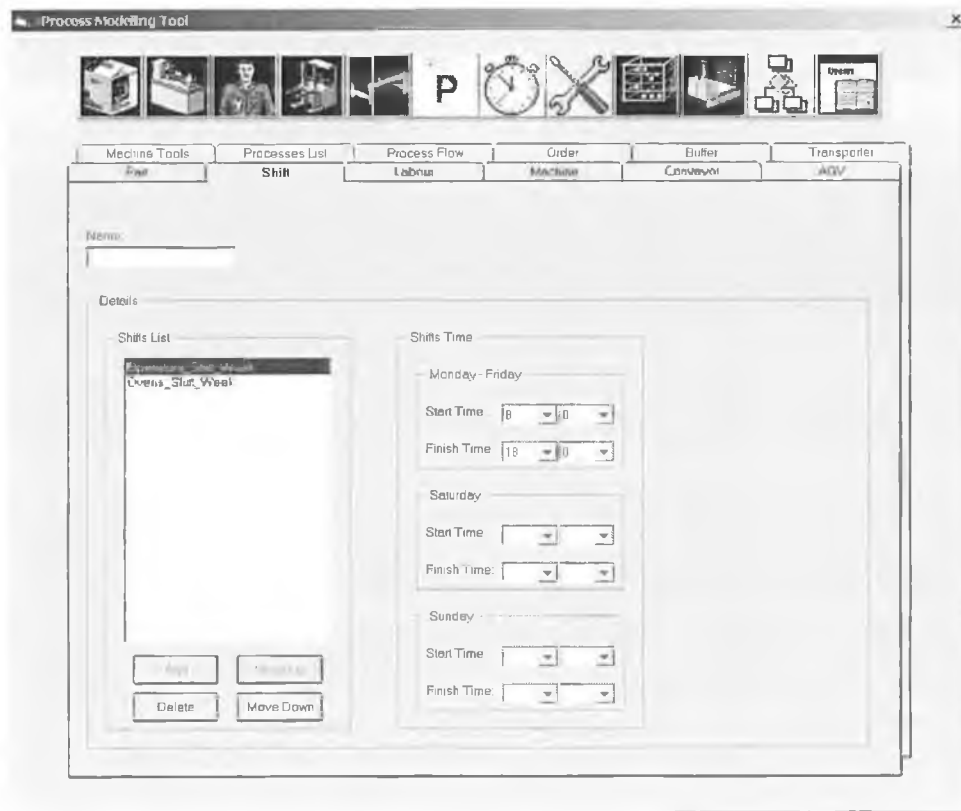


Figure 5.3: Defining the Shifts of the Job Shop Model

The *Part Detail Dialog Page* of the PMT was used to define the sheet rolls “Roll1”, “Roll2”, “Roll3” and “Roll4” that represent the products of the Job Shop model as shown in Figure 5.4.

The *Labour Detail Dialog Page* of the PMT was used to define the labourers who were required to handle the processes of the Ovens. Ten labourers are defined as shown in Figure 5.5. The type of the labourers has been set as “Labourers” who can handle any job type. The quantity was set to ten. Shift called “*ShiftLabourers_Week*”, which is used to identify the work time pattern for each labourer, was applied to the labourers.



Figure 5.4: Defining the Products of the Job Shop Model

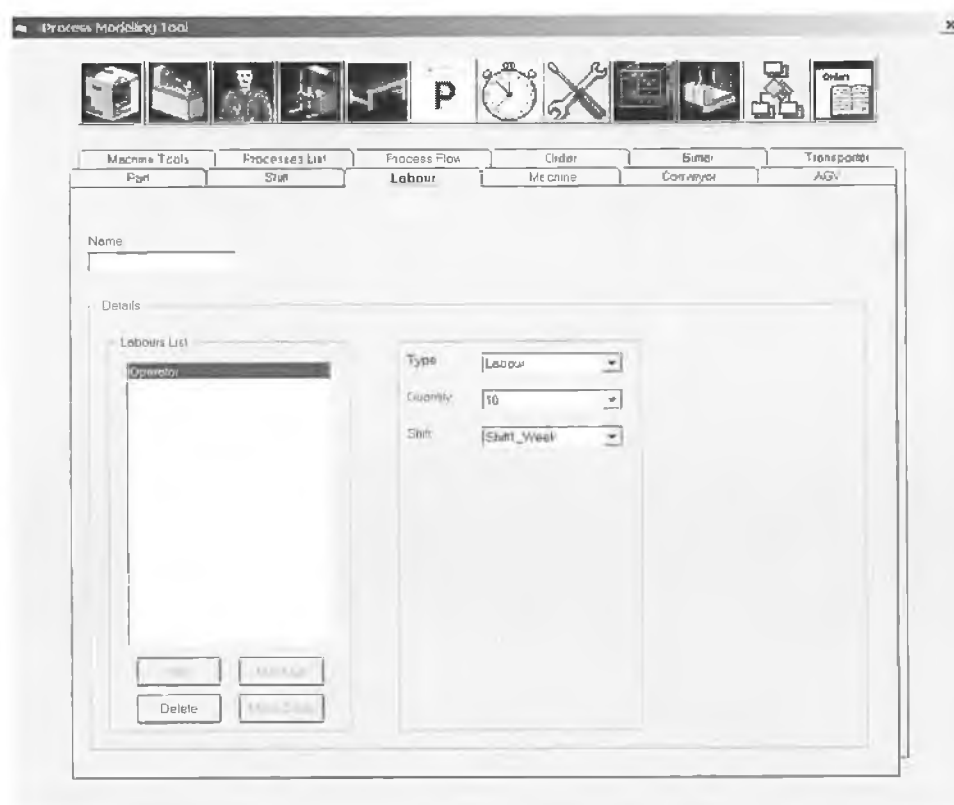


Figure 5.5: Defining the Labourers of the Job Shop Model

The *Machine Detail Dialog Page* of the PMT was used to define the Ovens of the Job Shop model. Five Ovens were defined to process the sheets as shown in Figure 5.6. The machine type “*Generic*” was used to represent the oven as the VR-Simulator software does not include this type of machine. The number of labourers was not specified by this detail dialog page as each process on the Oven requires a different number of labourers. Also, the breakdown details have not been specified since the maintenance is considered as a process. Shift called “*Ovens_Shift_Week*” was applied to all the Ovens.

The *Processes List Detail Dialog Page* was used to create a list of the processes that can be associated with each Oven to be analysed by the VR-Simulator. Figure 5.7 shows the processes list that was created. Seven processes were defined (Setup, Pass, Final Pass, Std Changeover, Colour Changeover, Maintenance and Cleaning). Descriptions of these processes were given in Table 5.1.



Figure 5.6: Defining the Ovens

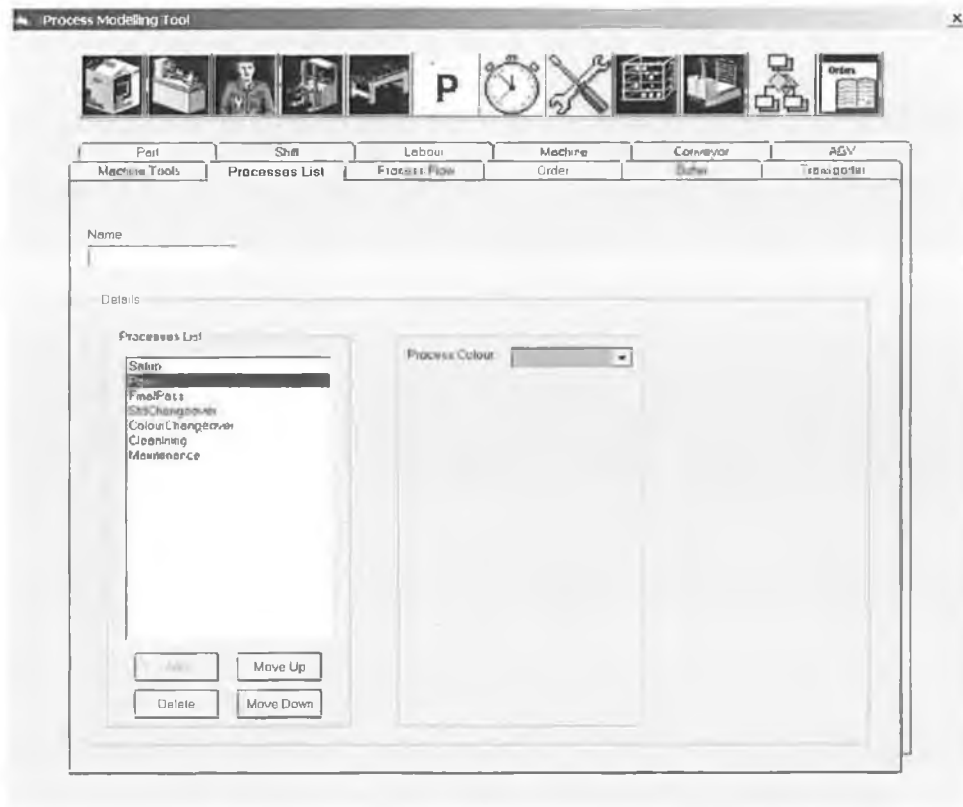


Figure 5.7: Defining the Processes List of the Job Shop Model

The Job Shop model, as mentioned earlier, does not have any material handling systems to move the rolls from one location to another within the shop floor. The labourers are used to carry the rolls between the Ovens. Based on that, carrying the rolls within the shop floor was modelled using the *Transporter Detail Dialog Page* as shown in Figure 5.8. Four transporters have been defined to specify the required time to deliver the rolls between the five Ovens.

The transporters are represented by a time interval that indicates carrying the rolls from one Oven to another within the shop floor. The time interval of the transporters was specified using the *Factory Layout Window* which is discussed in the following section.

The *Buffer Detail Dialog Page* (Figure 5.9) was used to define a warehouse with a capacity set at 500 parts, which is the maximum capacity of the buffer.

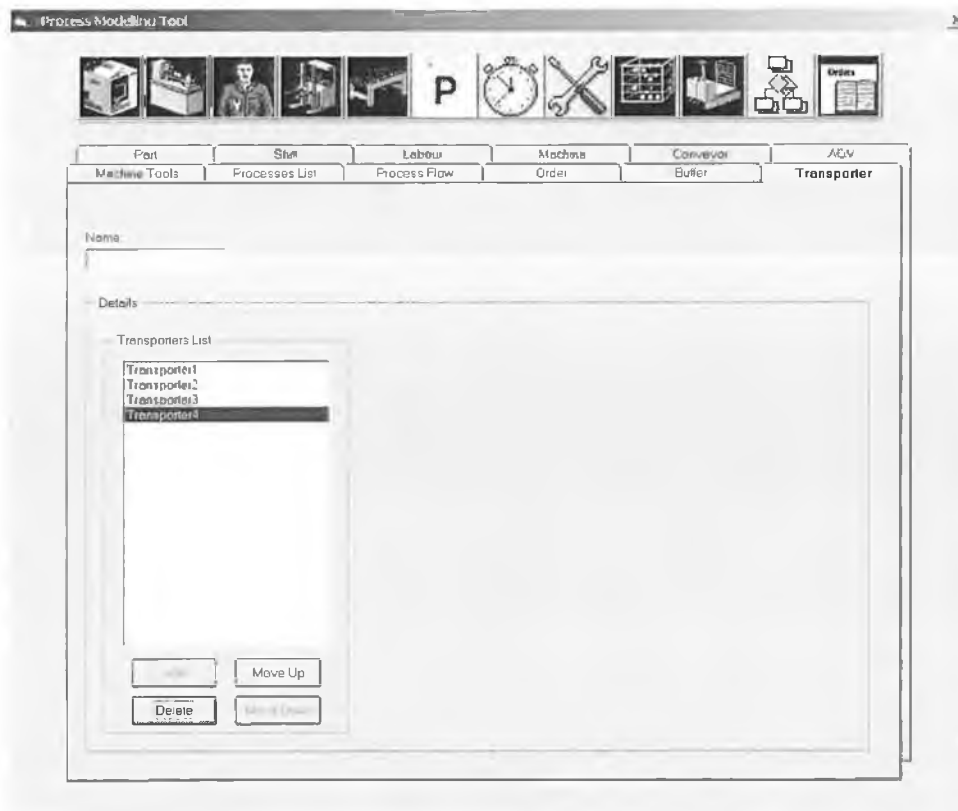


Figure 5.8: Defining the Transporters of the Job Shop Model



Figure 5.9: Defining the Warehouse

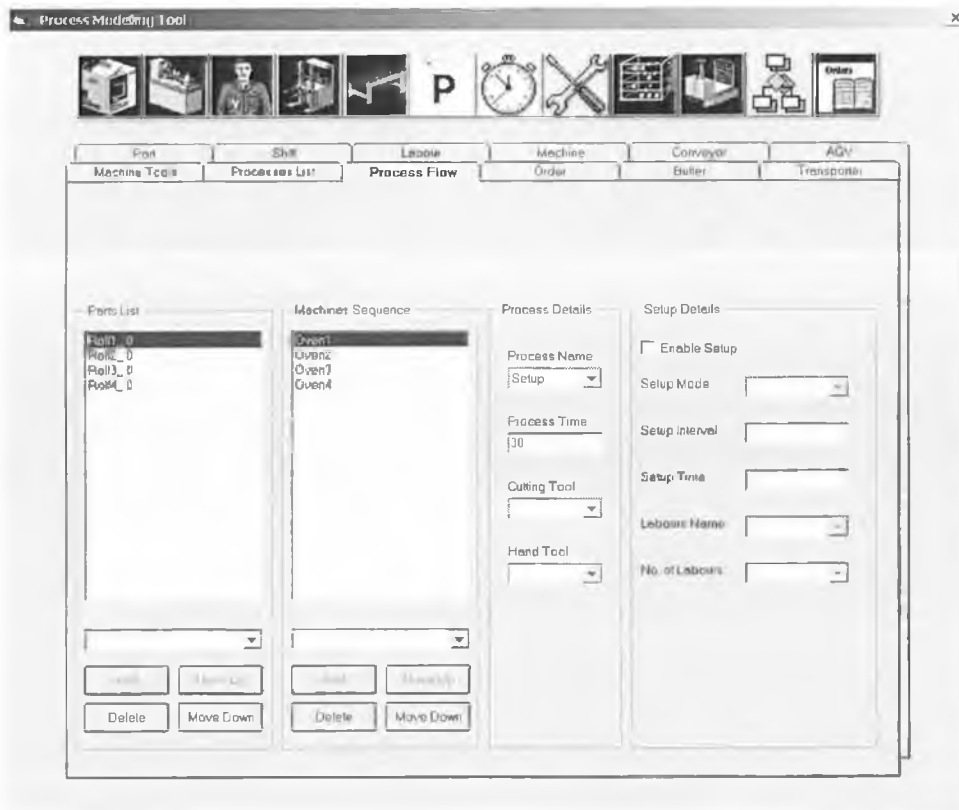


Figure 5.10: Defining the Process Flow of the Job Shop Model



Figure 5.11: Defining the Orders of the Job Shop Model

Once all the resources of the job shop model were identified, the process plan was defined using the *Process Flow Detail Dialog Page* and the *Orders Detail Dialog Page* as shown in Figure 5.10 and Figure 5.11. The *Process Flow Detail Dialog Page* in Figure 5.10 was used to define the sequence of the jobs by specifying the rolls names, the sequence of the Ovens needed to process the rolls and finally detailing the processes associated with the each Oven. The *Orders Detail Dialog Page* in Figure 5.11 was used to specify the lots size of each roll that needs to be processed using the same sequence of Ovens and processes.

5.3.1.2 Constructing the Job Shop Model Layout

Once all the resources of the shop floor model were defined using the Process Modelling Tool, the *Factory Layout window* of the VR-Simulator software was used to specify the location (x and y positions) of the resources within the shop floor to construct the layout of the Job Shop model.

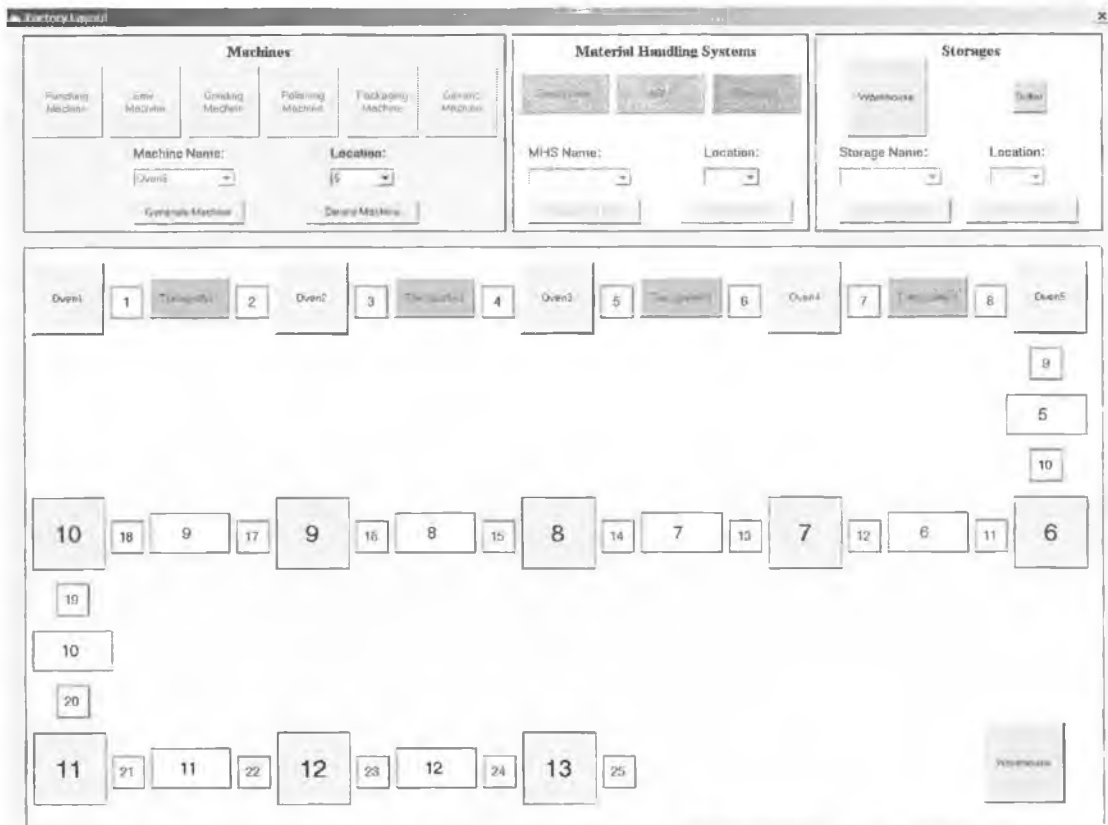


Figure 5.12: Constructing the Layout of the Job Shop Model

The “*Generic Machine Icon*” was used to represent the Ovens of the shop floor while the “*Transporter Icon*” was used to represent the required time to deliver the products between the Ovens. The locations of the Ovens, transporters and the warehouse were assigned by selecting the name the resource from its list and the location that needs to be occupied. Then, the buttons “*Generate Machine*”, “*Generate MHS*” and “*Generate Storage*” were used to allocate the resources as shown in Figure 5.12.

The time of each transporter was specified by clicking on the transporter icon and entering its value in minutes as show in Figure 5.13.



Figure 5.13: Specifying Transporters Names

5.3.1.3 Developing a Simulation Model of the Job Shop Model

After the resources and the layout of the Job Shop model were defined, the SMT of the VR-Simulator was used to develop a simulation model automatically of the shop floor in Witness as shown in Figure 5.14.

The simulation model of the shop floor was generated in three phases as following:

- **Define Phase:**

At this phase, all the resources of the Job Shop model are defined in Witness. The resources included Ovens, labourers, rolls, transporters and a warehouse. The generated resource names were displayed on the left side of the window of the simulation model as shown in Figure 5.14.

- **Display Phase:**

The x and y positions of the Ovens and the warehouse were used by the SMT to display them within the simulation model in Witness as shown in Figure 5.14.

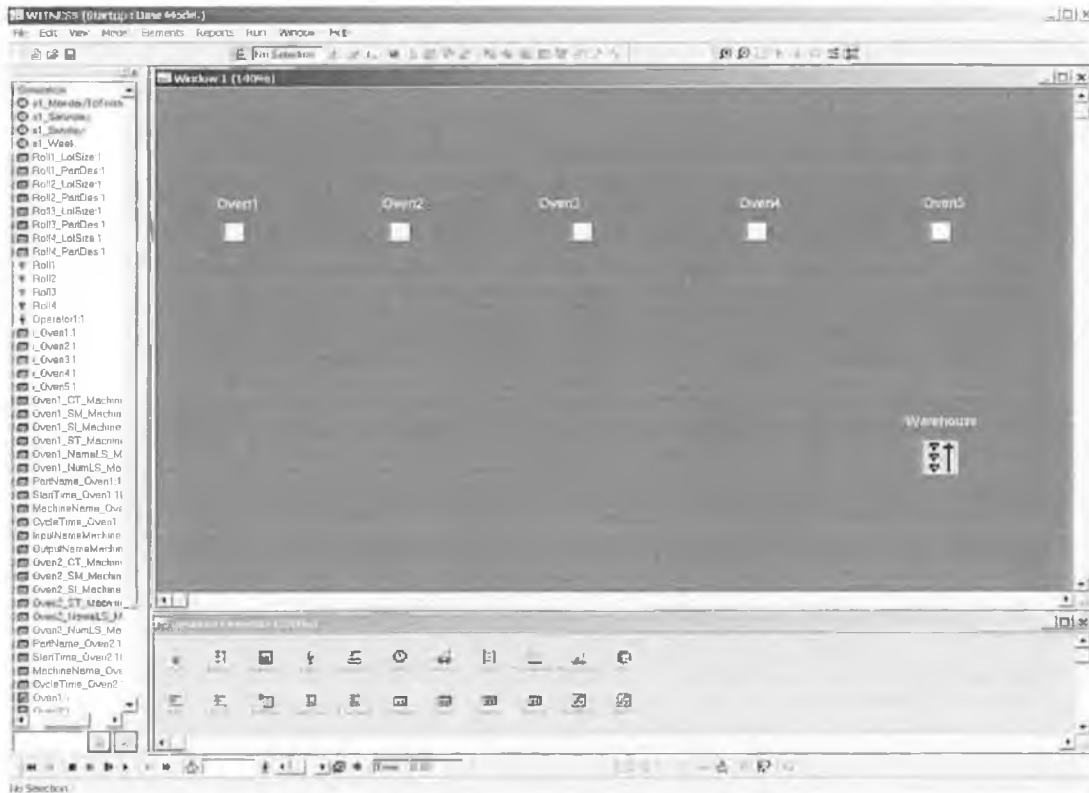


Figure 5.14: Simulation Model of the Shop Floor Generated Automatically

- **Detail Phase:**

A number of variables were generated automatically to be attached to the resources to hold the process plan of the Job Shop model. More details of how the SMT generates simulation models automatically in Witness were discussed in section 4.4.2.3.1.

5.3.1.4 Developing a Virtual Model of the Job Shop Model

The Virtual Modelling Tool (VMT) of the VR-Simulator software was used to generate a virtual model automatically of the Job Shop model. The VMT generated the virtual model in two phases as following:

▪ **Importing the resources to build the layout of the virtual model**

The first phase of generating the virtual model of the Job Shop was importing the virtual objects of the resources which were saved within the warehouse library of Superscape VRT. The types of the machines that were defined by the user using the PMT were used to specify the type of the machines to be imported from the warehouse library. Five Generic machines which represent the Ovens and a warehouse, were imported to construct the layout of the Job Shop model.

▪ **Assigning properties to the imported resources**

After each resource was imported from the warehouse library, some properties were assigned to Ovens and the warehouse which included their x, y positions and their names. Figure 5.15 shows a snapshot of the virtual Job Shop model. A special virtual object was created to be used to represent the Ovens. More details regarding how the virtual model can be generated automatically are given in section 4.4.5.2.

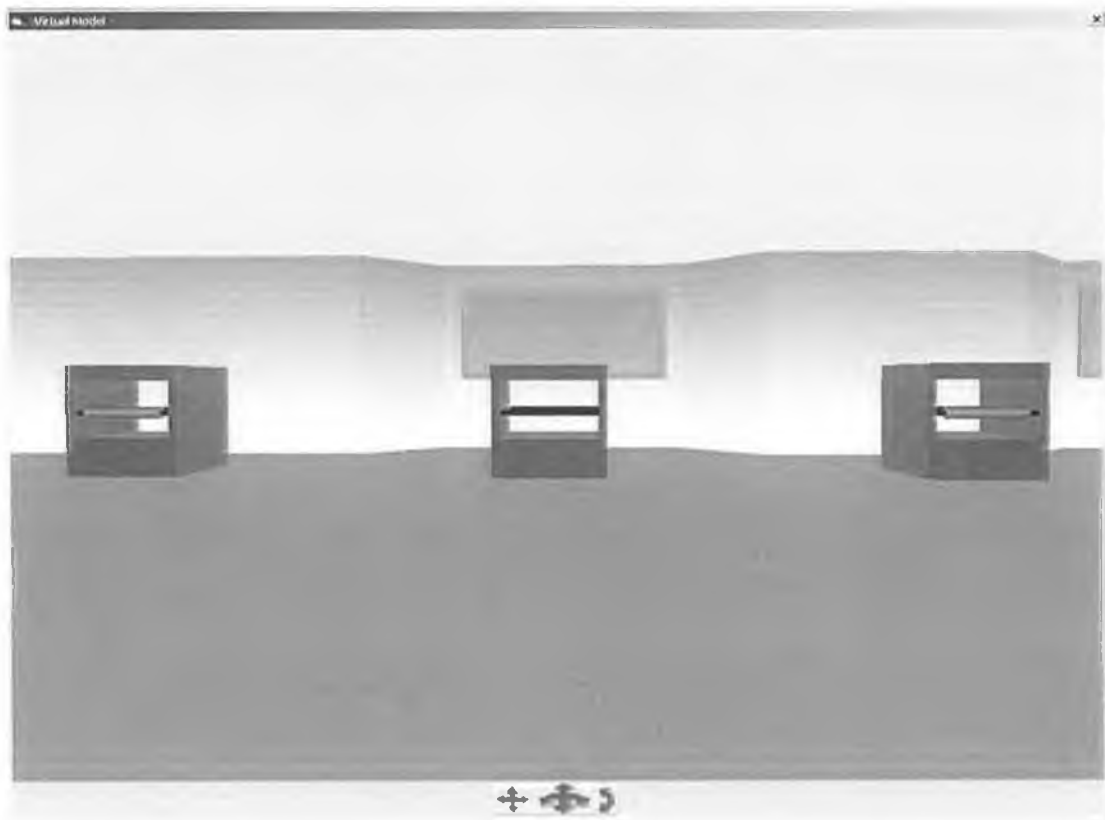


Figure 5.15: Virtual Model of the Job Shop Model

5.3.1.5 Loading the Process Plan of the Job Shop Model

Once the simulation model and the virtual model of the shop floor were generated, the process plan in Excel file was loaded by the “*Load Process Plan*” option, and the *Model Running Tool* was used to run the models for experiments with the Job Shop model as follows:

- Effects of having different numbers of labourers were investigated to see how the shop floor reacts.
- Different shifts have been applied to the labourers and the Ovens were investigated.
- Effects of changing the sequences of the processes on each Oven were investigated.
- Effects of adding more Ovens were investigated.

5.3.1.6 Results and Discussions

Gantt chart, Pie charts and Reports were used as output tools to display the statistical results of the Job Shop model to measure its performance.

▪ **Gantt Chart:**

The Gantt chart of the VR-Simulator was used to monitor the activities of the processes on the ovens as they happened against time. The Gantt chart helps to plan the tasks that need to be completed; also it can be used as a basic tool for scheduling when these tasks are carried out in the future.

Figure 5.16 shows some results of using the Gantt chart to show the activities of the Ovens where each colour represents a different type of process. For example, the red colour indicates the Oven is under maintenance and the white colour represents the cleaning process.

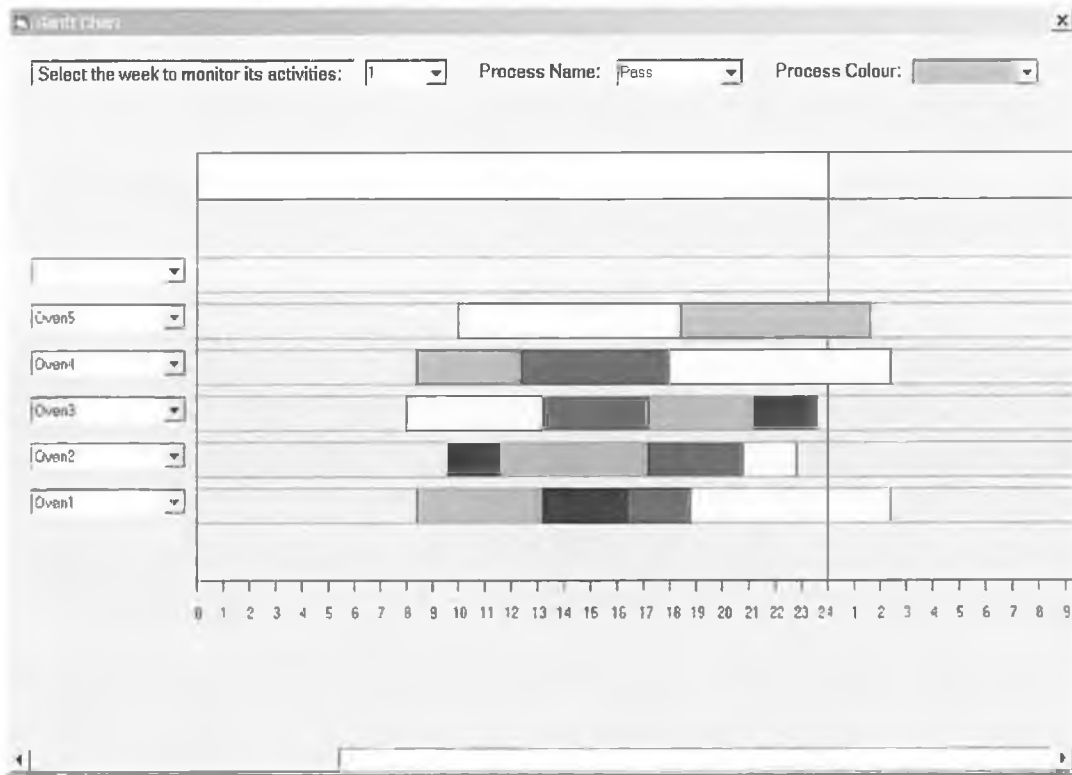


Figure 5.16: Gantt Chart Displaying the Running Time of the Processes Vs Time for the Job Shop Model

▪ **Pie Charts:**

Pie charts were used to display the utilisation of the Ovens and the labourers. Figure 5.17 shows the utilisation of Oven2. Each colour represents a different process as being assigned using the “Processes Detail Dialog”. The green colour stands for the *Pass* process that represents the time taken by Oven2 to coat the sheets since the start of the run.

The user can monitor the utilisation of the other machines of the job shop model by selecting the machine name from the machine names list. Figure 5.18 shows the utilisation of operator_1. The two states reflect the busy time and the idle time of the labourer.

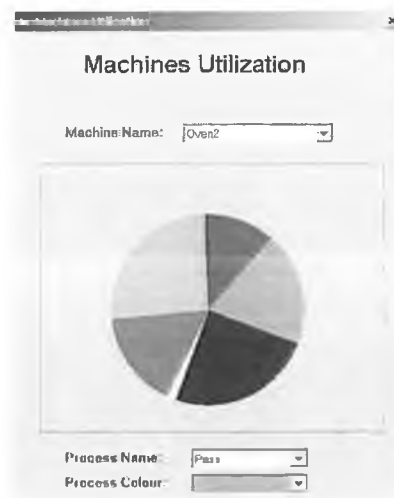


Figure 5.17: Utilisation of Oven2



Figure 5.18: Utilisation of Operator_1

▪ **Reports:**

Reports were used to display some of the statistics of the performance of the Job Shop model. Figure 5.19 shows a report of the processes for Oven3. The output analysing tools were used to measure the performance of the Job Shop model. The virtual model was used as a tool to verify the simulation model, understand the results and provide an environment to communicate the results. The real time interaction with the models were tested by using mouse and voice commands. A Head Mounted Display was used to display the virtual model of the job shop model.



Figure 5.19: Report of the Processes of Oven3

5.3.1.7 Verification and Validation of the Job Shop Model

The results of the Job Shop model were verified by an existing simulation package. The virtual model of the Job Shop model has been used also to verify the simulation model by providing a visual trace of events as they happen. This technique was used to point out any gross discrepancies in sending the rolls to the right ovens and the processing times.

Validation is the process of determining whether the model reflects reality. However, there are no validation techniques that give 100% certainty in the results of a model. The results of the Job Shop model were validated using actual shop floor data and consulting the experts of the modelled system.

5.3.2 Hypothetical Example of Flow Shop Model of a Manufacturing System

The second case study used to test the capabilities of the VR-Simulator was to develop a simulation model and a virtual model of a hypothetical example of a Flow Shop model that represents a work shop order to produce 2000 items/day of the product shown in figure 5.20.



Figure 5.20: Product Design

5.3.2.1 Flow Shop Model Representation

Figure 5.21 represents the diagram of how the products (sheets) are manufactured.

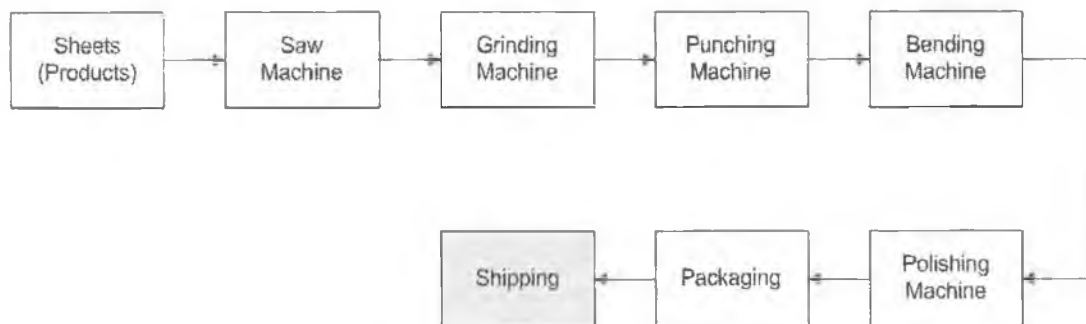
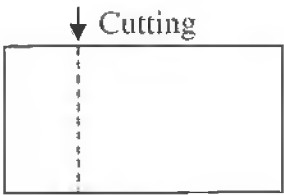
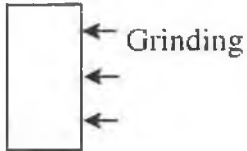

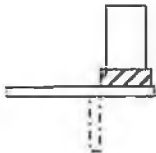
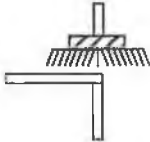
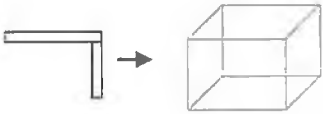



Figure 5.21: Flow Shop Model Representation

The manufacturing system is a Flow Shop model where all the products have to follow the same sequence to be processed by all the machines within the shop floor. It is assumed that the sheets are available within the shop floor.

The first process of the model is to cut the sheets into small sheets and then the grinding machine is used to smooth the cut sides. The punching machine is used to drill two holes within the sheet with different diameters. Once the holes are drilled, the bending machine is used to bend the sheet as shown in Figure 5.19. The polishing machine is used to polish the sheets before they are packaged. The packaging machine is used to package the finished products to be sent to the warehouse for shipping. Table 5.3 describes the processes of the work shop.

Table 5.3: Description of the Processes of the Work Shop

Process name	Description
Cutting	
Grinding	
Punching	
Bending	
Polishing	
Packaging	
Shipping	

5.3.2.2. Model Assumptions

The first assumption that has been made was specifying the required material handling systems to transfer the products from one location to another within the

shop floor. Figure 5.22 displays the material handling systems to be used for delivering the products between the resources of the shop floor.

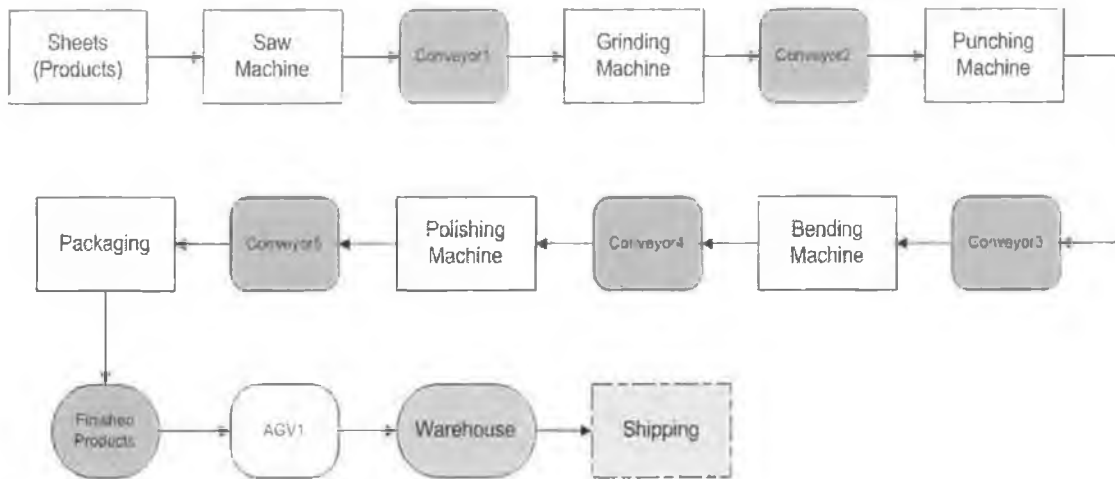


Figure 5.22: Material Handling Systems of the Flow Shop Model

Five conveyors were assumed to transfer the products between the machines while an AGV is used to deliver the finished products to the warehouse to be shipped. Some other assumptions were made to detail the resources of the Flow Shop model that include:

- Processing times of each machine.
- Breakdown details for the machines and conveyors.
- Setup details for the machines.
- Number of labourers needed to handle each machine.
- Shifts of the machines and labourers.
- Detailing the material handling systems.
- Creating a list of the processes.

Table 5.4 through 5.7 show the assumptions that have been made to detail the machines, conveyors, AGVs and buffers of the Flow Shop model.

Table 5.4: Modelling Assumptions for Machines

Machine Name	Saw Machine	Grinding Machine	Punching Machine	Bending Machine	Polishing Machine	Packaging Machine
Cycle Time (min)	30	40	60	25	40	25
Labourers Type	Labour	Labour	Labour	Labour	Labour	Labour
Labourers Quantity	1	1	1	1	1	1
Breakdown Mode	No. of Opera.	No. of Opera.	No. of Opera.	No. of Opera.	No. of Opera.	No. of Opera.
Breakdown Interval	400	500	400	500	400	500
Repair Time	60	50	30	30	20	40
Labourers Names	Any	any	Any	any	any	any
Labourers Quantity	2	2	1	2	1	3
Setup Time			20			
Labourers Name			Any			
Labourers Quantity			1			
Shifts	9am-6pm	9am-6pm	9am-6pm	9am-6pm	9am-6pm	9am-6pm

Table 5.5: Modelling Assumptions for Conveyors

Conveyor Name	Con1	Con2	Con3	Con4	Con5
Length In Parts	10	10	10	10	10
Movement Index Time	1	1	1	1	1
Breakdown Mode					
Breakdown Interval					
Repair Time					
Labourers Names					
Labourers Quantity					
Shifts					

Table 5.6: Modelling Assumptions for AGVs

AGV Name	AGV1
Capacity In Parts	5
Speed With Load	1
Speed Without Load	1.5
Breakdown Interval	500
Repair Time	100

Table 5.7 Modelling Assumptions for Buffers

Buffer Name	Capacity in Parts
B1	10
B2	10
....	10
B12	10
Warehouse	500

The tools of the VR-Simulator were used to build a simulation model and a virtual model of the Flow Shop model as follows.

5.3.2.3 Defining the Flow Shop Model

The PMT was used to define the resources and the process plan of Flow Shop model based on the assumptions that have been made as explained in the previous sections. Before the PMT was used, an Excel file was created to be used as a database for the Flow Shop model. The Excel file included a number of Excel data sheets in which the information of each resource was saved within an individual Excel data sheet. The shifts, labourers and the products were defined as explained in the Job Shop model. Figure 5.23 shows two cutting tools were defined using the *Machine Tools Detail Dialog Page* to be used by the punching machine to drill two holes in the sheets.

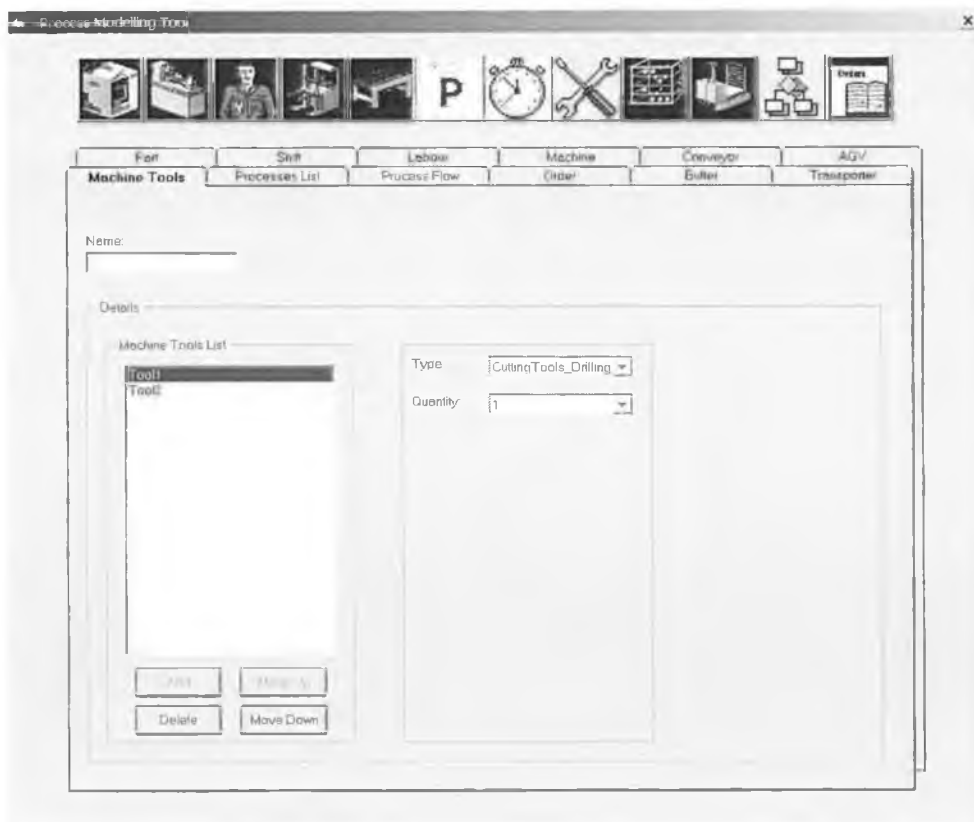
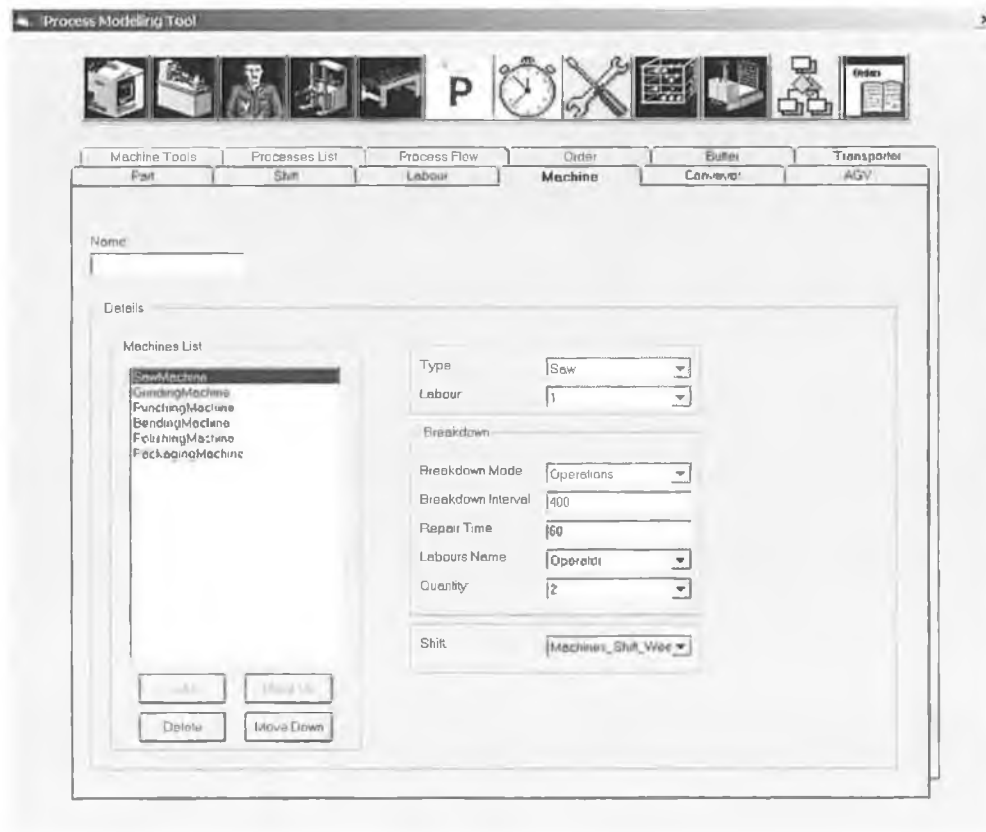


Figure 5.23: Defining the Machine Tools of the Flow Shop Model

Six machines were defined using the *Machine Detail Dialog Page* as Figure 5.24 shows. The machines were cutting machine, a grinding machine, a punching machine, a bending machine, a polishing machine and a packaging machine. Table 5.4 was used to detail each machine.



5.24: Defining the Machines of the Flow Shop Model

The material handling systems of the Shop Flow model consisted of five conveyors and an AGV. Figure 5.25 shows the *Conveyors Detail Dialog Page* that was used to define the conveyors based on the assumptions in Table 5.5.

The *AGV Detail Dialog Page* as shown in Figure 5.26 was used to detail the AGV based on the assumptions in Table 5.6.

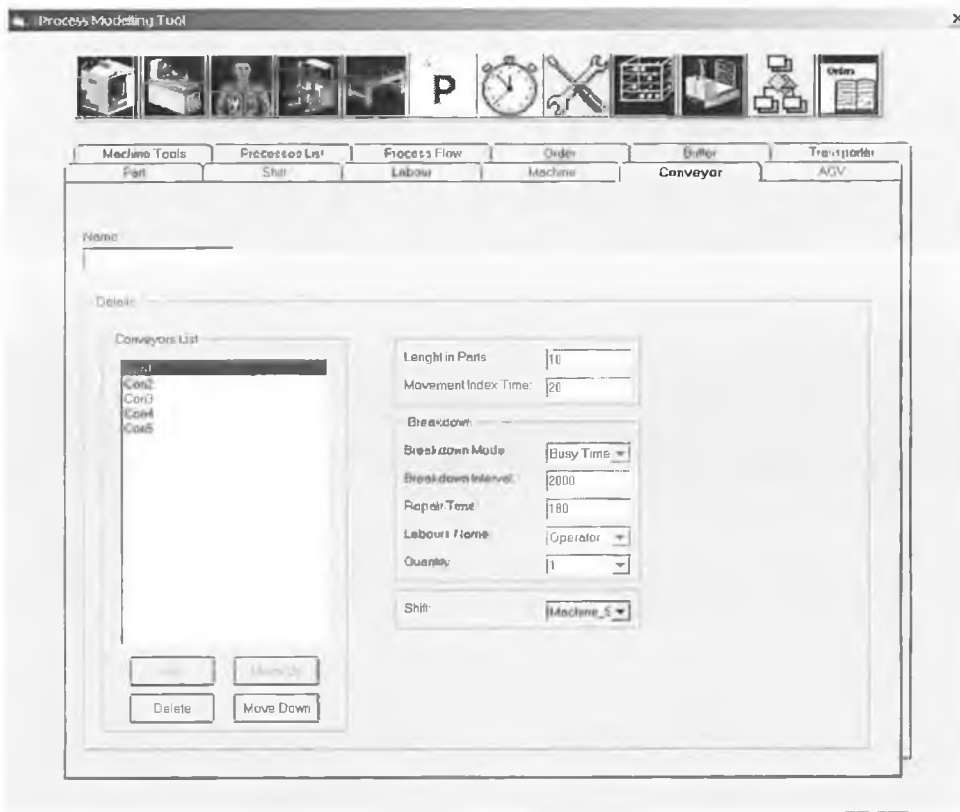


Figure 5.25: Defining the Conveyors of the Flow Shop Model



Figure 5.26: Defining the AGVs of the Flow Shop Model

The *Processes List Detail Dialog Page*, as shown in Figure 5.27 was used to create a list of all the processes that can be performed using the machines to process the parts.

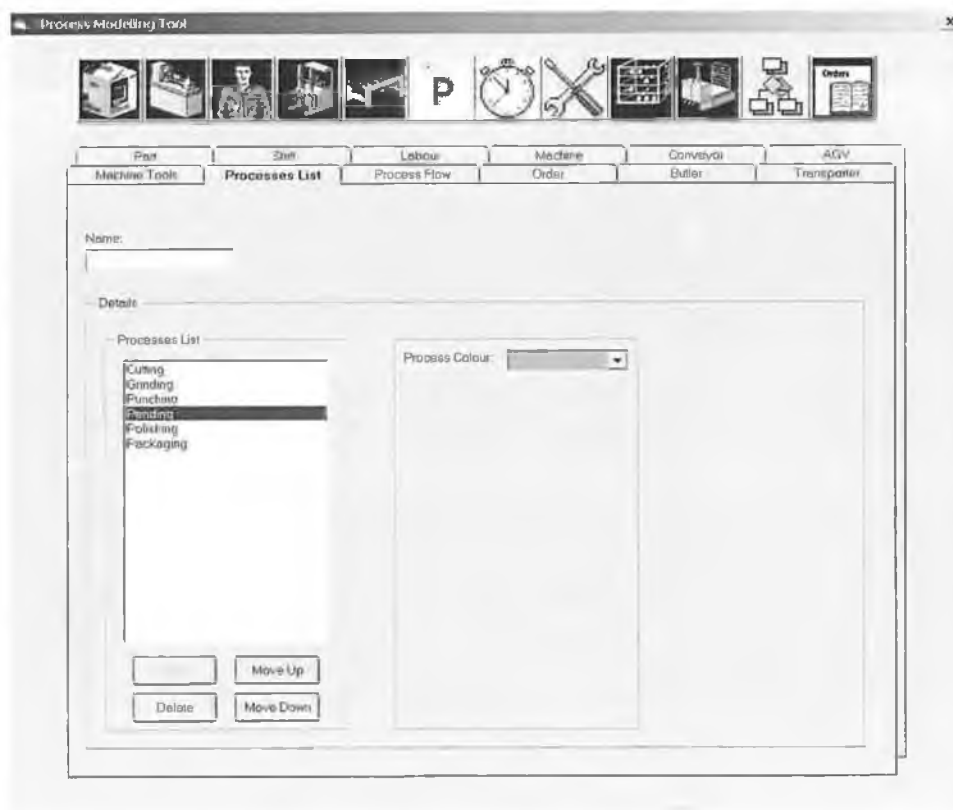


Figure 5.27: Defining the Processes of the Flow Shop Model

The *Buffer Detail Dialog Page*, as shown in Figure 5.28 was used to define 12 buffers and a warehouse. The buffers were located in front of and after each conveyor to store some of the products of the flow shop model until the next destination is free.

Table 5.7 was used to detail the buffers and the warehouse. Once all the resources were defined, the *Process Flow Detail Dialog Page* was used to define the sequence of how each product is manufactured and to provide details regarding processes of the machines as shown in Figure 5.29.

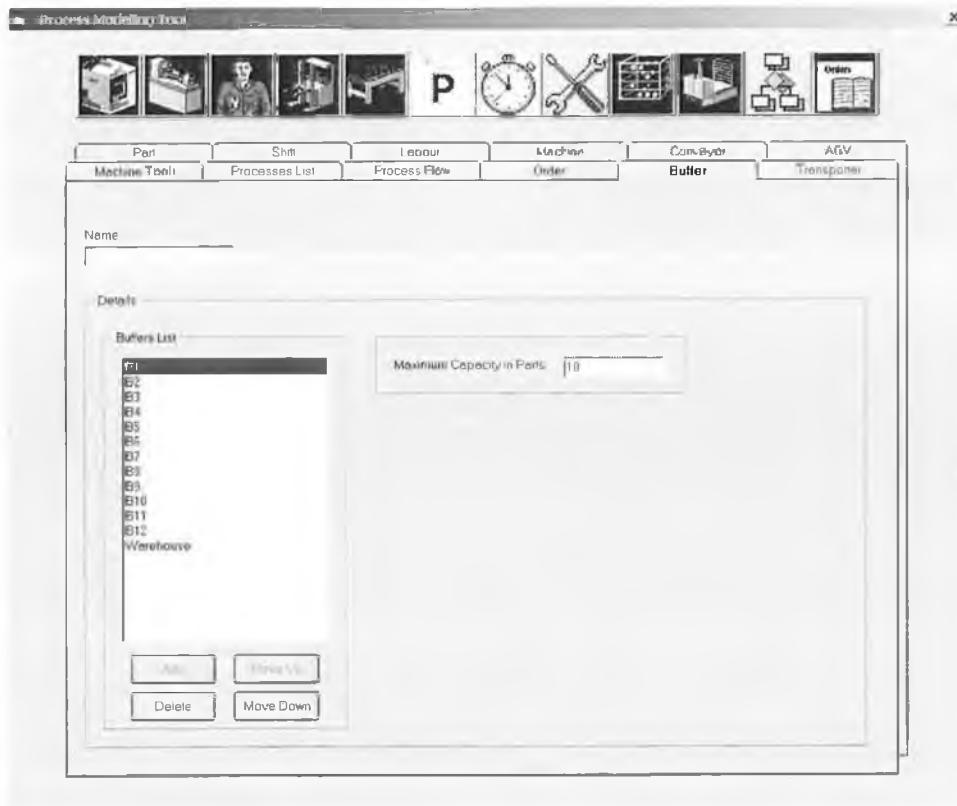


Figure 5.28: Defining the Buffers of the Flow Shop Model

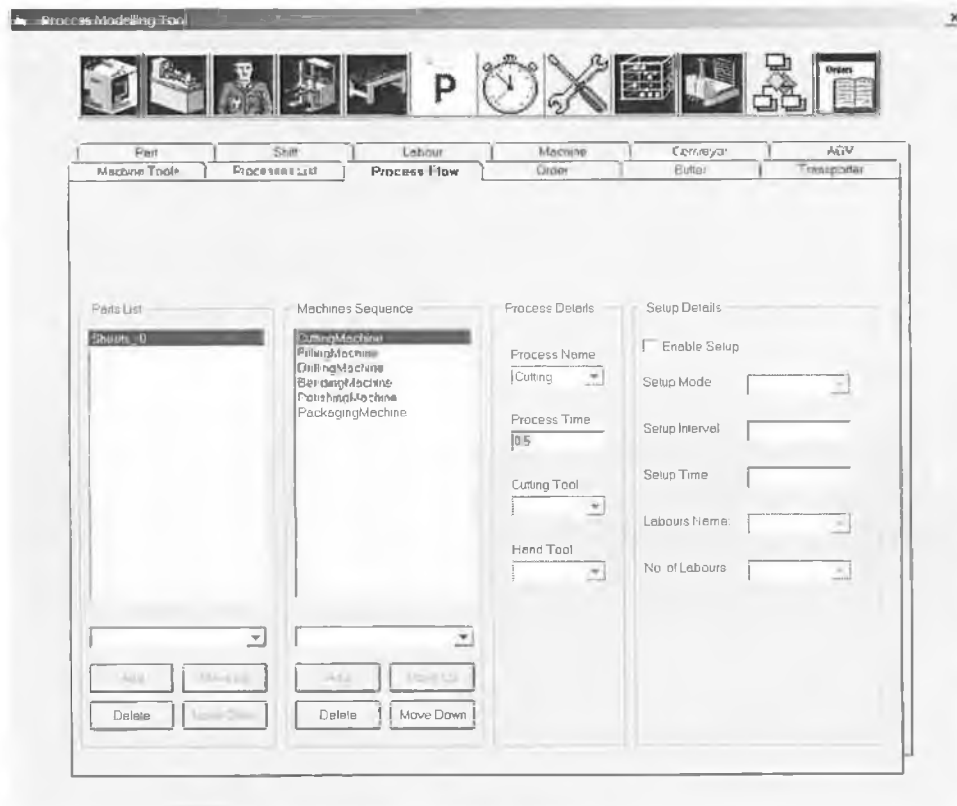
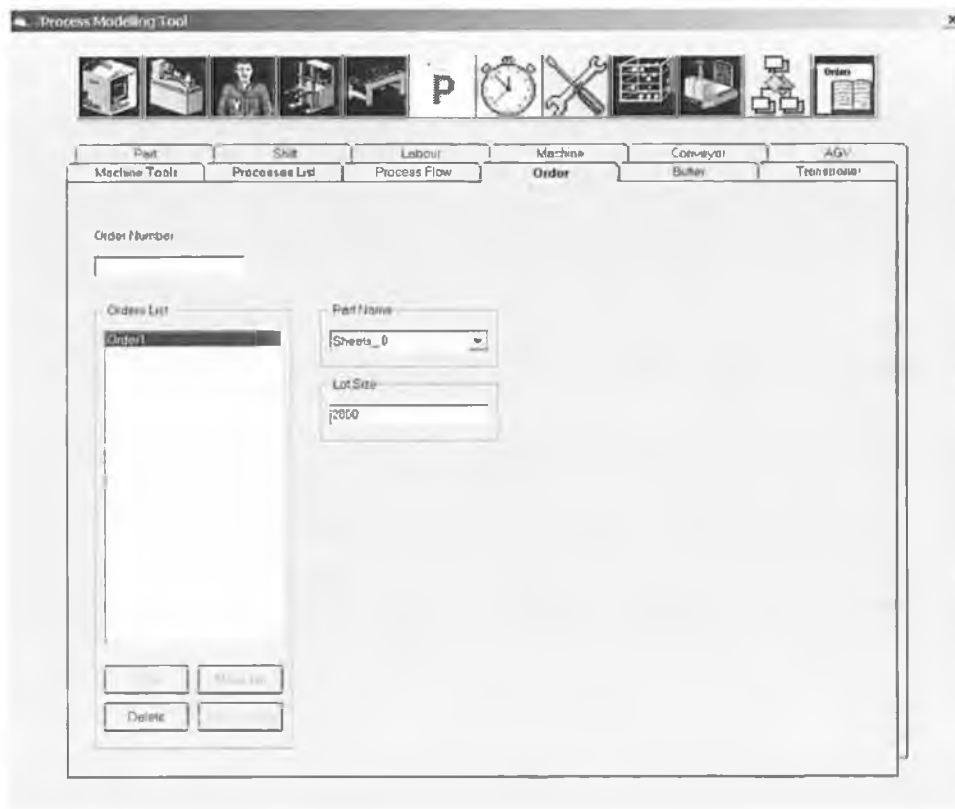


Figure 5.29: Defining the Process Flow of the Flow Shop Model

The *Order Detail Dialog Page* was used to specify the orders of each product that needs to be manufactured as shown in Figure 5.30. The order included the product name and the lot size of each product.



5.30: Defining the Orders of the Flow Shop Model

5.3.2.4 Constructing the Layout of the Flow Shop Model

After the Process Modelling Tool was used to define the resources and the process plan of the Flow Shop model, the *Factory Layout window* was used to construct the layout of the modelled system to specify the locations of the machines, conveyors and buffers. Figure 5.31 shows a snapshot of the layout of the flow shop model.

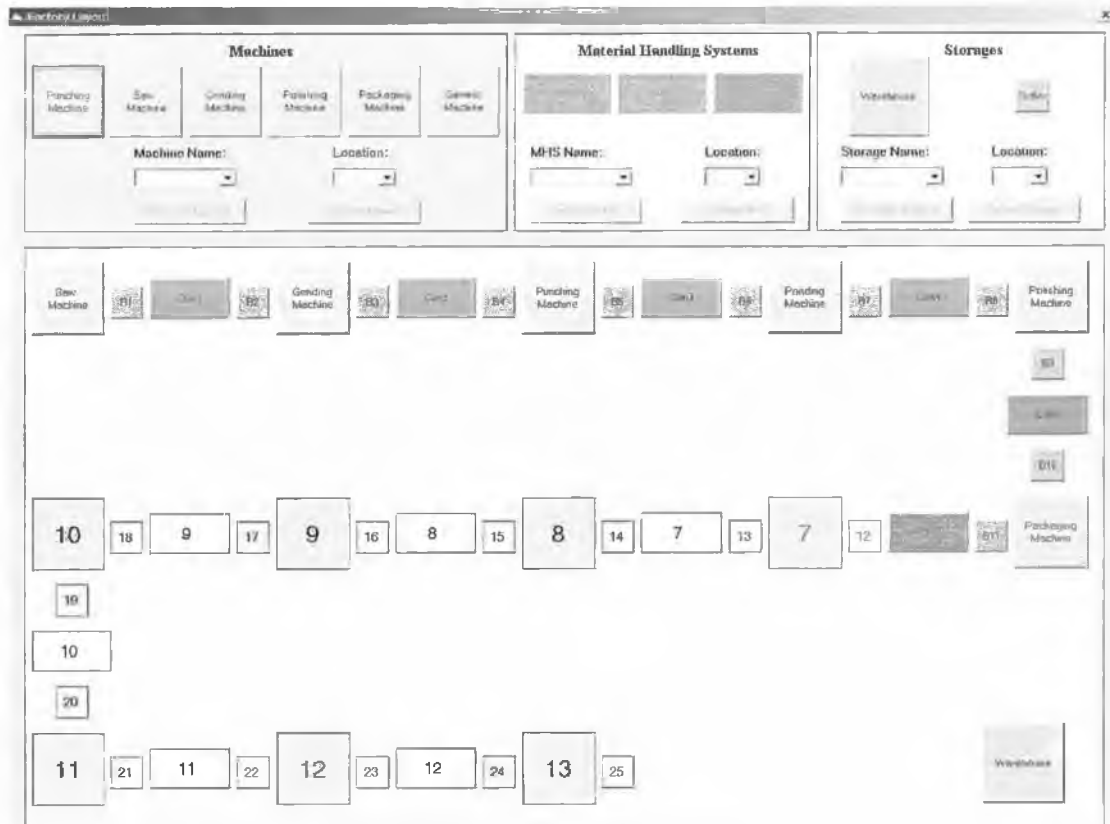


Figure 5.31: Constructing the Layout of the Flow Shop Model

5.3.2.5 Developing a Simulation Model of the Flow Shop Model

After the resources and the layout of the Flow Shop model were defined, the SMT of the VR-Simulator was used to develop a simulation model automatically of the shop floor in Witness as shown in Figure 5.32. The simulation model of the shop floor was generated in three phases as following:

- **Define Phase:**

At this phase, all the resources of the flow shop model were defined in Witness. The resources included products (sheets), saw machine, grinding machine, punching machine, bending machine, polishing machine, packaging machine, labourers, twelve buffers, four conveyors and a warehouse.

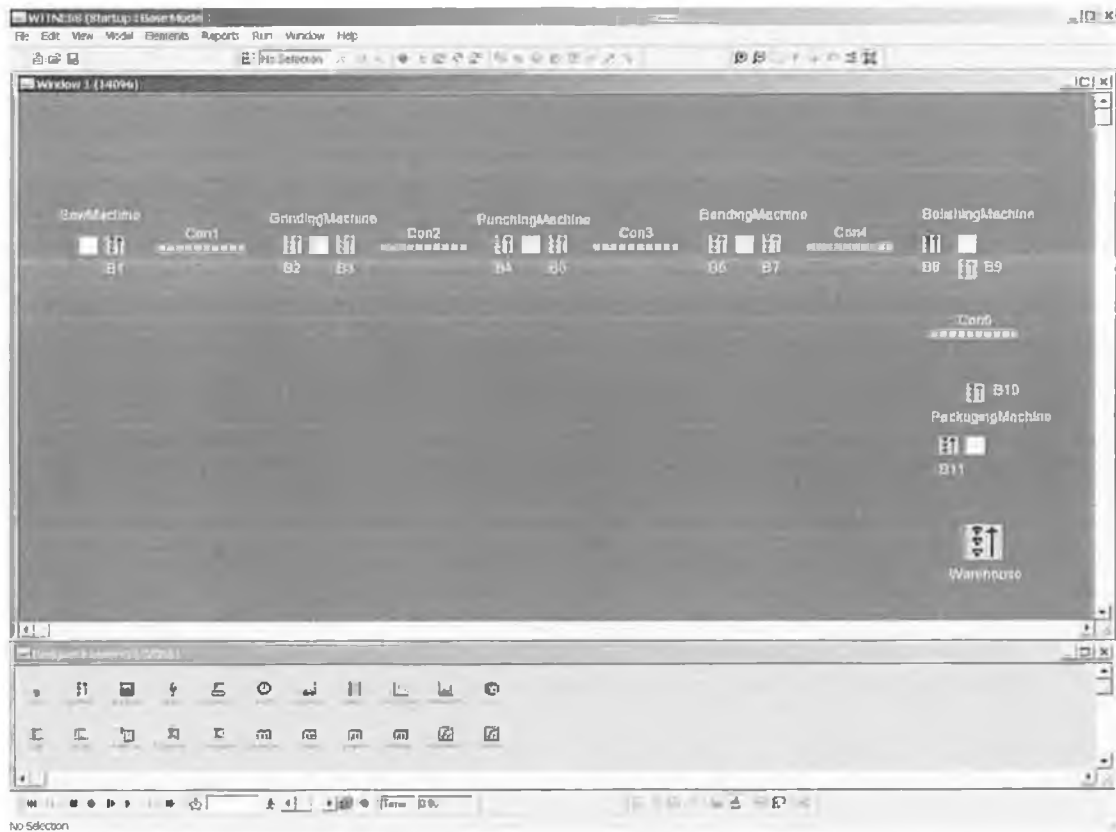


Figure 5.32: Simulation Model of the Flow Shop Model Generated Automatically

▪ **Display Phase:**

The x and y positions of the machines, conveyors, buffers and the warehouse were used by the SMT to display them within the simulation model in Witness as shown in Figure 5.32.

▪ **Detail Phase:**

A number of variables were generated automatically to be attached to the resources to hold the process plan of the Flow Shop model. At this stage, the information in Tables 5.4, through 5.7 were used to detail the resources.

5.3.2.6 Developing a Virtual Model of the Flow Shop Model

The Virtual Modelling Tool (VMT) of the VR-Simulator software was used to generate a virtual model automatically of the Flow Shop model. The VMT generated the virtual model in two phases as following:

- **Importing the resources to build the layout of the virtual model**

The first phase of generating the virtual model of the Flow Shop was importing the virtual objects that represent the machines, conveyors and the buffers from the warehouse library of Superscape VRT.

- **Assigning properties to the imported resources**

After each resource was imported from the warehouse library, some properties were assigned to the resources which included their x, y positions and name of each resource. Figure 5.33 shows a snapshot of the virtual flow shop model.

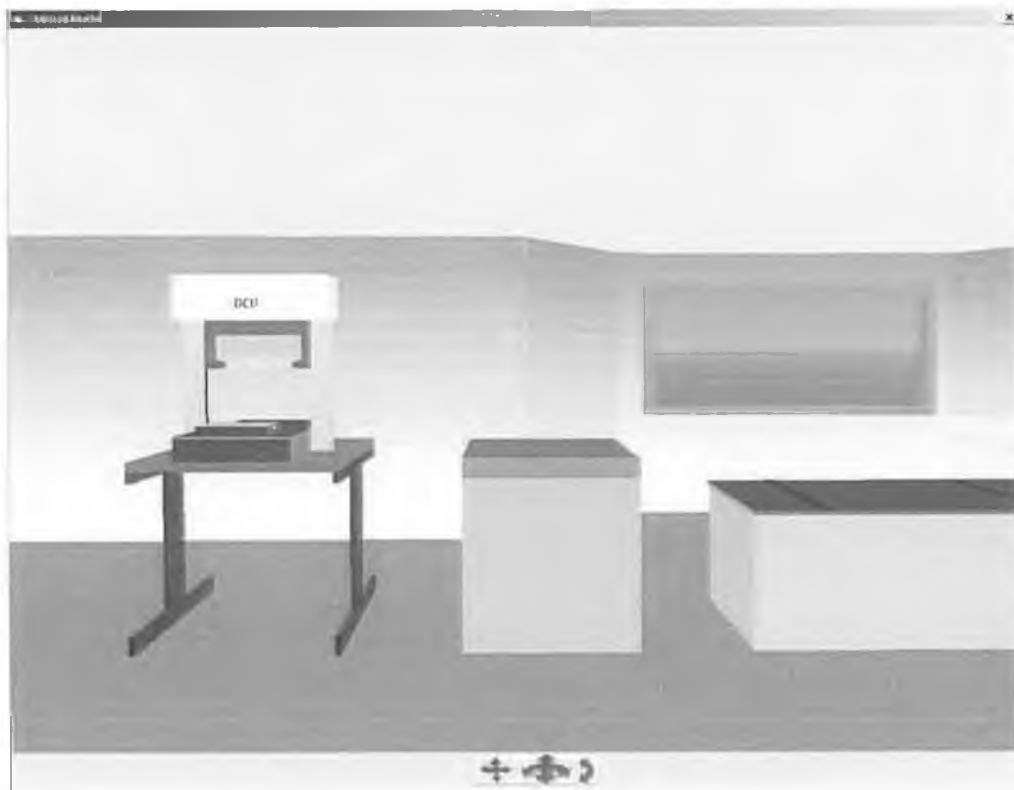


Figure 5.33: Virtual Model of the Flow Shop Generated Automatically

5.3.2.7 Loading the Process Plan of the Flow Shop Model

Once the simulation model and the virtual model of the shop floor were generated, the process plan which was stored within the excel file was loaded to run the following experiments:

- Effects of having different numbers of labourers were investigated to see how the flow shop model reacts.
- Different shifts have been applied to the labourers as the machines were investigated.
- Effects of changing the cycle times of the machine were investigated.
- Effects of changing the capacity of the buffers were investigated.
- Effects of applying breakdowns were investigated.

5.3.2.8 Results and Discussion

Gantt chart, Pie charts and Reports were used as output tools to display the statistical results of the Flow Shop model to measure the performance of the modelled system.

- **Gantt chart:**

The Gantt chart of the VR-Simulator was used to monitor the activities of the processes on the machines as they happened against time. The Gantt chart helps to plan the tasks that need to be completed; also it can be used as a basic tool for scheduling when these tasks are carried out in the future.

Figure 5.34 presents some results of using the Gantt chart to show the activities of the machines where each colour represents a different process type. For example, the red colour represents the status of the machine under maintenance, the green colour represents the busy time of the machine and the blue colour represents the repair time.

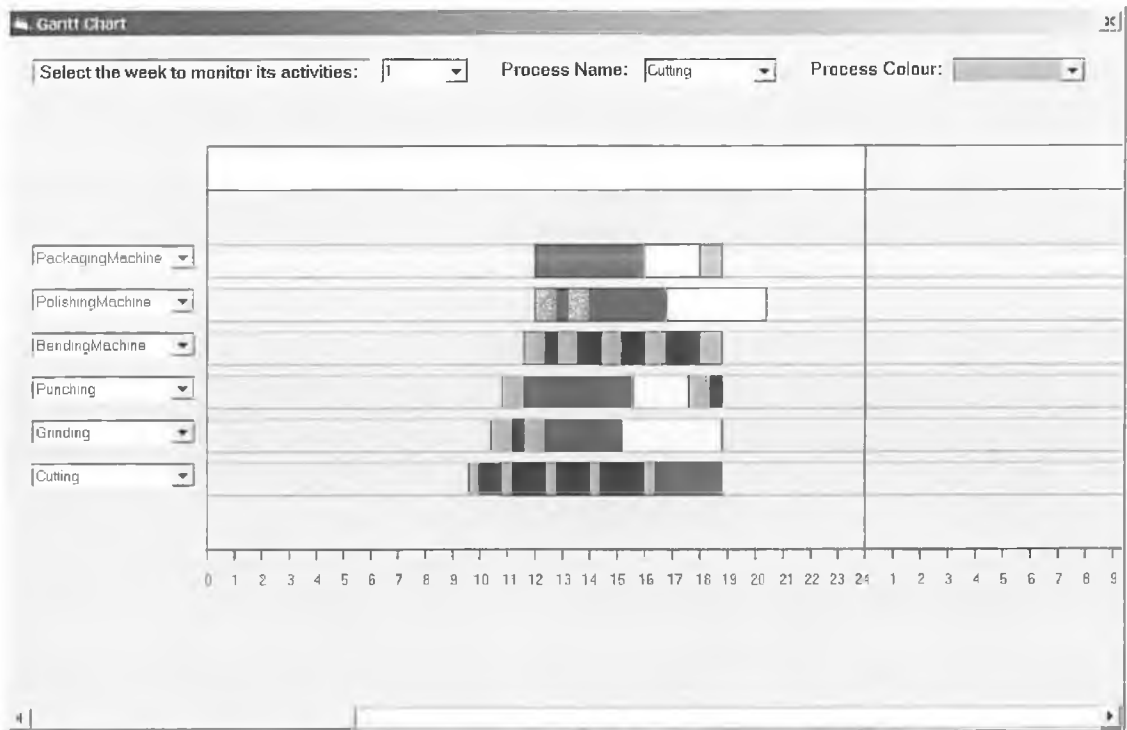


Figure 5.34: Gantt chart Displaying the Running Time of the Machines Vs Time for the Flow Shop Model

▪ **Pie Charts:**

Pie charts were used to display the utilisation of the machines, conveyors and the labourers of the flow shop model. Figure 5.35 represents the utilisation of saw machine.

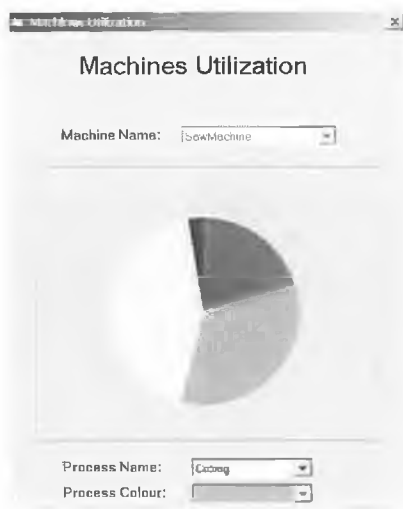


Figure 5.35: Utilisation of the Saw Machine

The Pie chart of the machines has four colours where each colour represents a different status. The green colour represents the main process of the machine which is cutting. The yellow colour represents the idle time of the machine. The red colour represents the time when the machine is broken down, the blue colour represents the repair time of the machine.

▪ Reports:

The Reports output tool of the VR-Simulator software was also used to examine the performance of the resources in the model and to provide the user with relevant information about their interaction, details and status. Reports can help to identify areas where the user may be able to improve the model's operation. Reports have been collected at different times about all the resources of the Flow Shop model. Figure 5.36 shows a report about the saw machine and Figure 5.37 shows another report about the sheets (products).

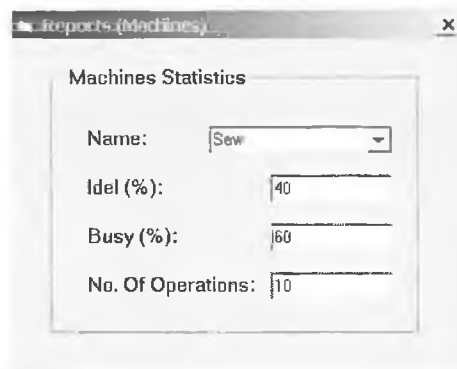


Figure 5.36: Reports about the Saw Machine

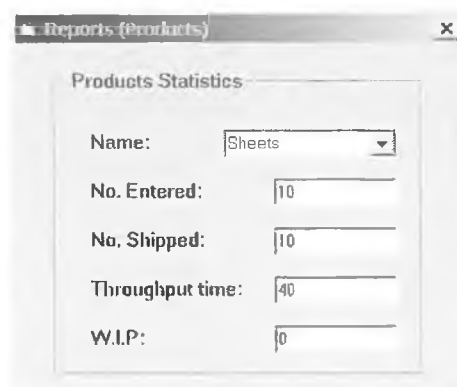


Figure 5.37: Reports about the Sheets (Products)

Different experiments were conducted to measure the performance of the Flow Shop model in each case by changing some assumptions, which included processing times, shifts, capacity of buffers and the parameters of the conveyors. The output analyzing tools were used to monitor the statistical analysis of the resources to determine performance of the job shop model. The virtual model was used as a tool to verify the simulation model, understand the results and provide an environment to communicate the results. The real time interaction with the models was tested using mouse and voice commands. A Head Mounted Display was used to display the virtual model of the flow shop model.

5.4 MANUFACTURING SYSTEMS

The VR-Simulator was designed and developed as a tool to address manufacturing system design and operational problems and to support the users in decision-making processes. Therefore in order to explore the application domain of the VR-Simulator, two main types of discrete manufacturing systems were studied as described below:

Job Shop: In a **Job Shop** system, a wide variety of different products in low volume are produced. The products are released to the production system in batches of one or more parts. For each batch (job) the production sequence and processing time at each workstation in the sequence are known. Due date may also be known. Job Shop manufacturing systems present significant modelling challenges due to their complexity and versatility.

Flow Shop: A **Batch Flow Shop** or **Flow Shop** is similar to Job shop in that many different types of discrete parts are produced in batches. However in a Flow Shop different product batches visit workstations in the same sequence. Basically the flow is unidirectional, following the same route through the system. The emphasis on this system is to remove bottlenecks and make effective use of resources.

5.5 THE VR SIMULATOR LIMITATIONS AND BENEFITS

As outlined before, the VR-Simulator software was designed and developed to allow users to develop simulation models and virtual models automatically of manufacturing systems from a valid Process Models. In addition the VR-Simulator software, with a number of tools, enables the user to study a manufacturing system and interact with the generated models in real time using voice commands and VR devices. As a result the VR-Simulator has inherited a number of limitations and benefits which became more evident when models were developed for a Job shop and a Flow Shop. These are discussed in the coming subsections.

5.5.1 General Features of the VR-Simulator Software

User-Friendliness:

The VR-Simulator software has been developed to be user-friendly to enable the user to model and study a manufacturing system without a need for extensive training. Non-experts users, who do not have good knowledge about simulation modelling and programming, can easily employ the VR-Simulator to develop simulation models and virtual models of shop floor activities.

Modelling the Resources:

The VR-Simulator provides the user with the required resources for modelling manufacturing systems. These resources have been chosen based on the resources available in the simulation packages available on the market today, such as Witness. The Process Modelling Tool provides the users with a multiple detail dialog pages interface that allows them to model each resource individually and apply some characteristics to it.

Modelling the Material Handling Systems:

In manufacturing systems in order to transfer parts from one location to another different material handling systems are commonly used. The VR-Simulator software allows the user to model three types of material handling systems, namely AGVs, Conveyors and Transporters. These systems represent operating characteristics which

must be considered in modelling activities. For instance, in modelling a conveyor, one must address several issues such as: how the load is transported, how its capacity determined and how loading and unloading of the conveyor occur. Also, in conveyor analysis, decision such as conveyor speed, accumulation size must be considered. The VR-Simulator provides a detailed method of modelling such systems to increase the flexibility in modelling different manufacturing systems.

Modelling Flexibility:

The VR-Simulator establishes a relatively high degree of flexibility in modelling different systems such as Job Shop and Flow Shop manufacturing systems. However, to develop an accurate model for complex aspects of a system requires further development of the functions and resources of the VR-Simulator.

Ease of Model Development:

Instead of writing programs, model development is now an interactive process involving mouse clicks, list box selections, and similar paraphernalia. The tools of the VR-Simulator enable the user to develop simulation models and virtual models of shop floor activities automatically.

Process representations:

Using the Process Flow detail page and Factory Layout window of the VR-Simulator one can simply specify the relationship between the resources in a model. However further flexibility in this field is required to allow a better presentation of cyclic flows in a model.

Visualization:

The VR-Simulator software provides the user with a virtual environment to simulate the activities of a shop floor in 3D presentation. The virtual environment can be used as a tool to verify the simulation model.

Interactivity:

The user can interact with the simulation models and the virtual models that have been generated automatically in real time using mouse, keyboard, joystick and

microphone. The virtual models can be displayed using Head Mounted Display (HMD).

Interface to other software tools:

The VR-Simulator software has been successfully integrated with other tools such as Witness, Superscape VRT and Excel. More software tools supporting OLE automation can be integrated to the VR-Simulator to increase its capabilities.

Output Analysis:

The VR-Simulator has a set of output tools that can be used to display statistical analysis of shop floor activities to measure their performance.

5.5.2 The Limiting Factors of the VR-Simulator Software:

Process Flow Logic:

The process flow logic describes the movement of parts through the system from one operation to another. The process flow logic is predefined using the Process Flow detail dialog page and the Factory Layout window. This means that parts are routed to particular operations via a predefined route. This method of modelling the flow logic proved to be very efficient for models where the flow logic is fixed as presented in the case studies of the Job Shop and the Flow Shop.

However in modelling manufacturing systems with product varieties and routing sequences increase, the complexity of making a flow decision increases as well. Such systems will have some situations where there are multiple outputs and each output has a different set of destinations from which to choose. Therefore, it was necessary to consider a number of assumptions and estimations in order to model parts movements in the system.

As a result, the modelling of the flow logic must be improved to facilitate a more efficient way to determine the part movements in the system. For instance, the incorporation of conditional statements to model the flow logic (i.e. Alternative atoms) could greatly help to address this limitation. Using the conditional statements,

the user would be able to define conditions or decision logics and when the conditions are realised then parts would proceed to the specified destinations in the system. Nevertheless, the current flow logic that can be constructed using the VR-Simulator is efficient enough for models with a typical degree of flow complexity.

Job assignment logic:

A Job is an activity performed by a number of resources on parts. A Job is characterised by the input parts and the specific Auxiliary or Manpower resources configurations. The job assignment logic is used to define the sequence by which a job is performed by particular resources in the model. This logic is defined in the VR-Simulator by a random selection rule. This means that the machines, conveyors and AGVs execute whichever job is available. The availability of a job is based on the availability of the input parts, auxiliary and manpower resources. The random assignment of jobs has limited the VR-Simulator capability in scheduling and sequencing jobs in complex manufacturing systems.

Resource allocation logic:

The resource allocation logic is used to select a resource from all those resources that are waiting to perform tasks. The allocation logic is performed randomly in the VR-Simulator. In this way a resource is selected randomly among other resources waiting for the same process. The random allocation of resources has proved to be a very efficient discipline since no difficulties were observed in any of the generated models. However, sometimes it would be necessary to define a specific allocation for a resource which suggests a method for selecting a resource from among several resources. Therefore incorporating other resource allocation rules such as highest priority operation and longest waiting operation could greatly increase the modelling flexibility of the VR-Simulator. Nevertheless the random allocation of resources is a suitable selection criterion in many cases.

Interactivity:

The real time interaction with the models is limited to breakdown, repair, stop and run the machines. More interaction commands can be added to give more flexibility to the user to interact with other resources such as labourers and material handling

systems. This can be useful in some decision making situations to command a resource to perform a particular job. For example, giving a command to an AGV to collect some parts and deliver them to another location.

Visualisation:

The current virtual model is limited to display the resources in 3D presentation within the virtual environment. The virtual machines do not simulate the actual process where the lamps, sounds and simple tool animation on the machines are used to give an indication about the status of each machine. Complex animation can be added to the machines to simulate the actual processes. Also, the 3D presentation of the AGVs in the virtual model is limited to show the AGV beside the collecting or delivering points. More animation can be added to simulate the actual movement of the AGV for transferring the products within the virtual environment.

Shop Floor Layout:

The layout that can be generated using the VR-Simulator is limited to a number of machines, material handling systems, and buffers with predefined locations. More flexibility needs to be added to allow the user to increase the number of resources as well as the ability to change the locations of each resource.

5.5.3 The Scope of the VR Simulator Application

The VR-Simulator presented a very flexible way to model the real case study of the Job Shop and hypothetical example of the Flow Shop. It was relatively simple to translate these systems to simulation models and virtual models using the three main tools of the VR-Simulator software (PMT, SMT and VMT). After introducing the two case studies using the VR-Simulator, it was found that the tools in the VR-Simulator are adequate to mimic the dynamic behaviour of these systems. The generated models were suitable for analysing and understanding these systems. Having developed models for different manufacturing systems, it was realised that using the VR-Simulator one can build models for relatively complex systems. However as the complexity of the systems increases in terms of flow, operational and

behavioural logics, it would be necessary to further enhance the VR-Simulator functionalities to be able to formalise and capture the system's dynamic behaviour.

Overall, the functionality of the VR-Simulator falls between the Spreadsheet that is used by companies to formalise shop floor data and a sophisticated simulation tool that requires high expertise to develop detailed models. The Simulator allows the user with the least expertise to develop simulation models and virtual models for manufacturing systems.

5.6 CONCLUSIONS

This chapter has discussed the evaluation of the VR-Simulator software. Two case studies were presented to identify the capabilities and the limitations of the developed software. The general features of the VR-Simulator software and the scope of its applications were also presented.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 THESIS SUMMARY

The aim of this thesis was to design and develop a new software that allows non-expert personnel to develop simulation models and virtual models of manufacturing systems automatically to study their behaviours and is summarised as follows:

- Limitations of the simulation tools available on the market today that can be used to model and analyse manufacturing systems were discussed. The objective of this research was to overcome these limitations.
- The case study involving integrating of the simulation with the virtual model of a real manufacturing system showed that developing simulation and virtual models of manufacturing systems require programming and modelling skills. Hence, a new software was developed to enable non-expert personal to automatically develop simulation and virtual models on their own.
- The thesis presented the three technologies (Simulation, Process Modelling and Virtual Reality) that have been applied successfully to design, model and analyse manufacturing systems. A literature review of using these technologies in manufacturing application has also been presented.
- The thesis presented a new approach of integrating a simulation model with a virtual model to address a real manufacturing system. The presented approach can be followed to integrate existing simulation tools with virtual reality tools that support OLE automation to allow users to interact with the simulation models and

the virtual environments in real time using voice commands and displaying the virtual model using Head Mounted Display.

- A real case study and a hypothetical example of manufacturing systems were presented in this thesis to evaluate the capabilities of the VR-Simulator software.

6.2 CONCLUSIONS

The overall conclusions that can be drawn from this thesis are as follows:

1. The main objective of this research has been achieved by developing a software called VR-Simulator which aids to address the design and validation of manufacturing systems. The developed software has the following features:
 - Easy to use by non-technical personnel.
 - Has its own Process Modelling tool to define a manufacturing system.
 - Enables non-expert users to develop simulation models automatically in Witness environment of manufacturing systems.
 - Enables non-expert users to develop virtual models automatically in Superscape environment of manufacturing systems.
 - Provides the expert users with a tool to shrink the time of developing simulation and virtual models of manufacturing systems.
 - Create an environment to simulate the activities of the simulation model in 3D presentation within the virtual model in real time.
 - Permits users to interact with the simulation model and the virtual model in real time using input devices (e.g. keyboard, joystick, mouse, voice commands).
 - Supports Virtual Reality devices (e.g. Head Mounted Display).
 - Has its own Gantt chart that has been developed by the author using VB to monitor the activities of the processes on the machines as they happen against time.
 - Provides the user with other output tools (e.g. Pie charts, Timeseries) to display the statistical analyses of the generated models.

- The VR-Simulator software is an open system that can be integrated with scheduling tools or linked to a real manufacturing system.
2. A new approach was developed by the author that integrates a simulation model with a virtual model of manufacturing systems. The developed approach enables users to interact with the models in real time and display the virtual environment using HMD. The developed approach can be followed to develop a simulation model automatically in any simulation packages that supports OLE automation.

6.3 FUTURE WORK

In order to make the VR-Simulator software a commercial system, the following issues need to be addressed:

- Improving user friendliness of the Simulator:

As mentioned before, the main users of this tool are the non-technical personnel who are often unfamiliar with software in their work. Further development can make the VR-Simulator even more User-Friendly.

- Improving operational logics:

In order to enhance the general features of the VR-Simulator, further improvements of some operational logics could be undertaken. Examples of these logics are Process Flow logics, Job assignment logics and Resource allocation logics. These improvements would provide the VR-Simulator with a higher degree of modelling flexibility and ease of model development.

- Virtual Reality devices:

The VR-Simulator software supports Head Mounted Display (HMD) to view the virtual models. More VR devices can be added to enhance the interaction with the virtual model. For example, Gloves can be added to allow the user to change the

virtual operational machine tools or to interact with the virtual machines. The VR devices can provide great features to the VR-Simulator to be used as a training tool.

▪ Improving visualisation:

Further work can be done to enhance the virtual model presentation and animation as following:

- Constructing more realistic shapes of the current virtual resources.
- Adding more virtual machines such as lathe and assembly machine to the warehouse library of Superscape VRT.
- Adding more features to the virtual machines to simulate more realistic processes of how the products are manufactured.
- Adding movement animation to the AGVs.

▪ Factory Layout:

The Factory Layout tool of the VR-Simulator can be improved to allow more freedom of constructing the layout of manufacturing systems.

▪ Interactivity:

The interaction with the models of the VR-Simulator is limited to stop, run, breakdown and repair the machines. More commands for interacting can be added.

▪ Linking the VR-Simulator to a real manufacturing system:

The VR-Simulator software with some efforts can be linked to a real system to examine the status of its behaviour and also to control it for optimisation and scheduling purposes.

- *Developing an independent simulation engine:*

The VR-Simulator uses the simulation engine of Witness to process the data of the simulation elements. A special simulation engine can be developed to make the VR-Simulator works as standalone application.

- *Adding more features to the VR-Simulator to support operations management:*

Features can be added to the VR-Simulator software to increase its capability and functionality to be used as a tool for operations management aspects of manufacturing systems.

Publications of Tariq Saad Mujber

Papers:

- (1) Developing of an Interactive Virtual Space Station, *10th International Conference on Human - Computer Interaction (HCI2003)*, 22-27, June 2003, Crete, Greece.
- (2) Virtual Reality Applications in Manufacturing Process Simulation, *International Conference on Advances in Materials and Processing Technology (AMPT2003)*, 8-11 July 2003, Dublin City University, Dublin, Ireland.
- (3) Integration of Simulation Model to Virtual Manufacturing Environment, *20th International Manufacturing Conference (IMC-20)*, 3rd September 2003, Institute of Technology, Cork, Ireland.
- (4) Trends in Simulation and Virtual Reality Applications in Manufacturing, *3rd International Conference on Advanced Manufacturing Technology (ICAMT 2004)*, 11-15 May 2004, Kuala Lumpur, Malaysia.
- (5) A New Hybrid Dynamic Modelling Approach For Process Planning, *3rd International Conference on Advanced Manufacturing Technology (ICAMT 2004)*, 11-15 May 2004, Kuala Lumpur, Malaysia.
- (6) Developing A 3D-Simulator To Increase The Performance Of Manufacturing Systems, *14th Internal Conference on Flexible Automation and Intelligent Manufacturing (FAIM2004)*, 12-14 July 2004, Ryerson University, Toronto, Canada.
- (7) Using Virtual Reality in Manufacturing Systems, *Days Of Science In Rousse*, 29-30 October 2004, University of Rousse, Rousse, Bulgaria
- (8) Addressing Factory Design and Planning Using Simulation and Virtual Reality Technologies, *The 15th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2005)*, 18-20 July 2005, Bilbao (Bizkaia) Spain.
- (9) Development of Models Generating Tool Software to Address Manufacturing Systems, *11th International Conference on Human Computer Interaction (HCI International 2005)*, 22-27 July, 2005, Las Vegas, Nevada USA .
- (10) Design and Development of a Decision Support System for Manufacturing Systems, *I-PROMS 2005, Virtual Conference, 2005*.

Journals:

- (1) Virtual Reality Applications in Manufacturing Process Simulation, *Journal of Material Processing Technology*, 2004.
- (2) A New Hybrid Dynamic Modelling Approach For Process Planning. *Journal of Material Processing Technology*, 2005.

References

- [1] Cathal Heavey, Shafi M. S. Khanian, Development of A Process Model Simulator, *Journal of Simulation Modelling Practice And Theory*, Vol.10, P. 13-33, 2002.
- [2] Mikell P. Groover, *Automation, Production Systems, And Computer Integrated Manufacturing*, Prentice-Hall Inc., USA, 1987.
- [3] Ronald Askin and Charles Standridge, *Modeling and Analysis of Manufacturing Systems*, John Wiley & Sons. Inc., Canada, 1993.
- [4] Bonfatti, F., Monari, P.D. And Mussini, B., EPSYLON: An Enhanced Process Model To Support Lean & Green Manufacturing, *Proceeding of Product Data Technology Days*, Watford, UK, 1998.
- [5] Schlenoff, C., Knutilla, A. And Ray, S., *Unified Process Specification Language Requirements For Modelling Process*, NISTIR 5910, National Institute of Standards And Technology, Gaithersburg, MD, 1996.
- [6] Johansson, H.J., Mchugh, P., Pendlebury, A.J., Wheeler III, W.A., *Business Process Reengineering Breakpoint Strategies For Market Dominance*. Chichester: John Wiley & Sons Ltd, 1993.
- [7] Harrel, C. And K. Tumay. *Simulation Made Easy- A Manager's Guide*. Norcross, Georgia, U.S.A., Industrial Engineering And Management Press, Institute of Industrial Engineers, 1995.
- [8] Nethe, A And Stahlmann, H.D.. *Survey of a General Theory of Process Modelling*. In *International Conference on Process Modelling*. 1999.
- [9] Bustard, D., Kawalek, P., And Norris, M., *System Modelling For Business Process Improvement*. Boston. London., Artech House. P. 47-49, 1999.
- [10] Succi, G., Predonzani, P., Vernazza, T., *Business Process Modelling With Objects, Costs, and Human Resources*. *System Modelling For Business Process Improvement*. 1999.
- [11] Phalp, K.T., *the CAP Framework for Business Process Modelling*. *Information and Software Technology*, P.731-744, 1998.
- [12] Hammer, M., *Reengineering Work: Don't Automate, Obliterate*. *Harvard Business Review*, P. 104-112, 1990.

- [13] Davenport, T., Short, J., The New Industrial Engineering: Information Technology and Business Process Redesign. Sloan Management Review , 11-27, 1990.
- [14] Grover, V., And Malhotra, M.K., Business Process Reengineering: A Tutorial on The Concept, Evolution, Method, Technology And Application. Journal of Operations Management. P.193-213, 1997.
- [15] Hammer, M. And Champy, J. Reengineering The Corporation: A Manifesto For Business Revelotion, London: N. Brealey Publishing, 1993.
- [16] Cross, K.F., Feather, J.J., and Lynch, R.L., Corporate Renaissance, The Art of Reengineering. Blackwell Publishers, 1994.
- [17] Currie, W.L., Revisiting Management Innovation And Change Programmes: Strategic Vision Or Tunnel Vision?. Omega, Volume 27, Issue 6, P.647-660, 1999.
- [18] O'Neill, P., And Sohal, A.S., Business Process Reengineering, A Review Of Recent Literature. Technovation, . P. 571-581, 1999.
- [19] O'Sullivan, D., and Dooley, L., Decision Support System for The Management of System Change. Technovation, Volume 19, Issue 8. P. 483-493, 1999.
- [20] Love, P.E.D., And Gunasekaran, A., Process Reengineering: A Review of Enablers. International Journal of Production Economics. P. 183-197, 1997.
- [21] [http://www.latonabg.com/web/latonanew/home.nsf/vPagesLookup/products_bpm_products_aristoolset~en/\\$FILE/aris_toolset_en.pdf](http://www.latonabg.com/web/latonanew/home.nsf/vPagesLookup/products_bpm_products_aristoolset~en/$FILE/aris_toolset_en.pdf)
- [22] Technology, N.I.S.T., Integration Definition For Information Modelling (IDEF1X)., National Institute of Standards And Technology, 1993.
- [23] <http://www.bisoft.com.au/dtsheetWFM.pdf>
- [24] IDEF3 Process Flow And Object State Description Capture Method Overview, (Report)., 2000. IDEF Family of Methods Homepage: <Http://WWW.Idef.Com/Overviews/Idef3.Htm>.
- [25] <http://www.qualitydigest.com/march97/html/qssoft.htm#Anchor7735>

- [26] IDEF4 Object-Oriented Design Method Overview, (Report), 2000. IDEF Family of Methods Homepage: <Http://WWW.Idef.Com/Overviews/Idef4.Htm>.
- [27] Carrie, A. and Plaia, A. "Application and assessment of IDEF3 - process flow description capture method" in International Journal of Operations and Production Management, Vol. 15(1) pp 63-73.
- [28] IDEF5 Ontology Description Capture Overview, (Report), 2000. IDEF Family of Methods Homepage: <Http://WWW.Idef.Com/Overviews/Idef5.Htm>.
- [29] Charles Mclean, Swee Leong, Charley Harrell, Philomena M. Zimmerman, Roberto F. Lu, Simulation Standards: Current Status, Needs, And Future Directions, Proceedings of The 2003 Winter Simulation Conference, 2003.
- [30] Pegden, C.D., Shannon, E.R., and Sadowski, P.R., Introduction To Simulation Using SIMAN. Second Edition. , Mcgraw-Hill, International Editions, Industrial Engineering Series, 1991.
- [31] T. S. Mujber And M.S.J Hashmi, Trends In Simulation And Virtual Reality Applications In Manufacturing, ICAMT 2004, Kuala Lumpur, Malaysia, 2004.
- [32] Banks, J., Introduction To Simulation, Proceedings of The Winter Simulation Conference J. A. Joines, R. R. Barton, K. Kang, And P. A. Fishwick, Eds., 2000.
- [33] Banks, J., Carson, J., Nelson, B. L., Nicol, D. M. Discrete-Event System Simulation, Prentice-Hall, Inc., NJ, 2001.
- [34] Law, A. M., Kelton, W. D. Simulation Modelling and Analysis, 3rd Ed., Mcgraw-Hill, New York, 2000.
- [35] Pritsker, A.A.B., Introduction to Simulation And SLAM II, Halsted Press And Systems Publishing Corporation, 1984.
- [36] Shannon, R.E, Systems Simulation-The Art and Science, Prentice-Hall, 1995.

- [37] Kochhar, A.K., Computer Simulation of Manufacturing System – 3 Decades of Progress, Proceedings of The Third European Simulation Congress, Edinburgh, UK, P. 3-9, 1989.
- [38] Jerry Banks, Handbook of Simulation Principles, Methodology, Advances, Applications, and Practice, Engineering & Management Press, 1998.
- [39] Savolainen, T., D. Beeckmann, P. Groumpos, and H. Jagdev. Positioning For Modelling Approaches, Methods And Tools, Computer In Industry, Vol. 25, P. 225-262, 1995.
- [40] Harrell, C., and K. Tummy. Simulation Made Easy: A Manager's Guide, Industrial Engineering And Management Press, Norcross, Ga, 1996.
- [41] Harrel, C. And K. Tumay. Simulation Made Easy- A Manager's Guide. Norcross, Georgia, U.S.A., Industrial Engineering And Management Press, Institute of Industrial Engineers, 1995.
- [42] Arisha, A., Intelligent Shop Scheduling For Semiconductor Manufacturing, Phd Thesis, Dublin City University, Ireland, 2003.
- [43] Johnson, J. Creating Chaos, American Programmer, Vol. 8, No.7, 1995.
- [44] Deaver, R.A., Selecting A Manufacturing Simulation System, CIM Review, Vol.3, No.3, P. 6-8, 1987.
- [45] Pidd, M. Choosing Discrete Simulation Software, OR Insight, Vol. 2, No. 3, P. 22-23, 1989.
- [46] A. Arisha And M. El Baradie, On The Selection Of Simulation Software For Manufacturing Application, Proceedings Of The Nineteenth International Manufacturing Conference (IMC19), Queen's University Belfast, Belfast, P. 495 – 507, 2002.
- [47] Banks, J., (Editor) Handbook of Simulation. Principles, Methodology, Advances, Applications, and Practice. , John Wiley & Sons Inc, 1998.
- [48] Law, A., Kelton, D.D., Simulation Modelling & Analysis. Second Edition, Mcgraw-Hill, Inc, 1991.
- [49] [Http://Www.Wolverinesoftware.Com](http://Www.Wolverinesoftware.Com)
- [50] [Http://Minuteman Software P.O. Box 131 Holly Springs, NC27540 USA](http://Minuteman Software P.O. Box 131 Holly Springs, NC27540 USA)
- [51] [Http://Www.Caciasl.Com/Simscript/Simscript_Prod_Features.Html](http://Www.Caciasl.Com/Simscript/Simscript_Prod_Features.Html)

- [52] Witness, From Lanner [Http://Www.Lanner.Com](http://www.lanner.com)
- [53] Arena, ([Http://Www.Sm.Com/](http://www.sm.com/))
- [54] Promodel ([Http://Www.Promodel.Com](http://www.promodel.com))
- [55] Simul8 ([Http://Www.Simul8.Com](http://www.simul8.com))
- [56] Extend ([Http://Imaginethatinc.Com/](http://imaginethatinc.com/))
- [57] David Krahl, THE EXTEND SIMULATION ENVIRONMENT, Proceedings of The 2002 Winter Simulation Conference, 2002
- [58] Flexsim [Http://Www.Flexsim.Com](http://www.flexsim.com)
- [59] Shannon, R.E. Knowledge Based Simulation Techniques For Manufacturing. International Journal of Production Research, 5(26): P. 953-973, 1988.
- [60] Maria, Anu., Introduction To Modeling And Simulation. Proceedings of The 1997 Winter Simulation Conference, 1997, P. 7-14, 1997.
- [61] Law, A. M. And W. D. Kelton. Simulation Modelling and Analysis, 2nd Ed., Mcgraw-Hill, New York, 1991.
- [62] Bank, J., J. And R. R. Gibson. Don't Simulate When: Ten Rules For Determining When Simulation Is Not Appropriate, IIE Salutations, 1997.
- [63] Sutherland, I. The Ultimate Display. In Proceedings of IFIPS Congress, New York, P. 506 – 508, 1965.
- [64] K. Pimentek and K. Teixeira. Virtual Reality: Through the Newlooking Glass. Windcrest/Mcgraw-Hill Inc: PA, First Edition, 1993.
- [65] Porter, S. Interview: Jaron Lanier, Computer Graphics World, Vol. 14, No. 4, P. 61-70, 1992.
- [66] Longhurst, C. Event-Driven Visual Effects In Flight Simulation Virtual Environments, Virtual Reality Applications, Academic Press, San Diego, P. 231-244, 1995.
- [67] Hollands, R.J. And E.A. Trowbridge. A PC-Based Virtual Reality Arthroscopic Surgical Trainer, Simulation In Synthetic Environments 1996, The Society For Computer Simulation, San Diego, Vol. 28, No. 2, P. 17-22, 1996.
- [68] Expert System. Virtual Reality Application. Vol. 12, No. 2, P. 174-175, 1995.

- [69] Tushar H, Dani, Marwan Fathallah, And Rajit Gadh, "COVARDS: An Architecture For A Conceptual Virtual Design System", DE-VOL.67, Design For Manufacturability ASME 1994.
- [70] Iqbal Mohamed, Computer Aided Manufacturing System Modelling And Development Using Virtual Reality, PhD Thesis, Dublin City University, 2000.
- [71] Tariq Mujber, Tamas Szecsi, M.S.J Hashmi, Virtual Reality Applications In Manufacturing Process Simulation, Journal of Material Processing Technology 155-156 P. 1834-1838, 2004.
- [72] Manetta, R. & Blade., (1995), "Glossary of Virtual Reality Terminology", International Journal of Virtual Reality, Vol.1 No. 2, 1995.
- [73] Bolas, M.T. (1994). Human Factors In The Design of An Immersive System. IEEE Computer Graphics and Applications, 14, Pp 55-59.
- [74] ENRICO GOBBETTI, RICCARDO SCATENI, VIRTUAL REALITY: PAST, PRESENT AND FUTURE, Virtual Environments In Clinical Psychology And Neuroscience, 1998, Amsterdam, Netherlands.
- [75] Falby, J.S., Zyda, M.J., Pratt, D.R., And Mackey, R.L. NPSNET: Hierarchical Data Structures for Real-Time Three-Dimensional Visual Simulation. Computers And Graphics 17, 1 , P. 65-69, 1993.
- [76] Turner, R., Song, L., and Gobbetti, E. Metis: An Object-Oriented Toolkit For Constructing Virtual Reality Applications. In Proceedings Sixth Eurographics Workshop On Programming Paradigms In Graphics (Conference Held In Budapest, Hungary, F.Arbab And P.Slusallek, Eds., Eurographics Workshop Proceedings Series, Eurographics Association, P. 79-90, 1997.
- [77] Hudson, T.ÊC., Lin, M.ÊC., Cohen, J., Gottschalk, S., And Manocha, D. V-COLLIDE: Accelerated Collision Detection For VRML. In VRML 97: Second Symposium On The Virtual Reality Modeling Language (New York City, NY), R.Carey And P.Strauss, Eds., ACM SIGGRAPH / ACM SIGCOMM, ACM Press, 1997.
- [78] Ruspini, D.C., Kolarov, K., And Khatib, O. The Haptic Display of Complex Graphical Environments. In SIGGRAPH 97 Conference

- Proceedings, T.Whitted, Ed., Annual Conference Series, ACM SIGGRAPH, Addison Wesley, Pp. 345-352, 1997.
- [79] Buttolo, P., Oboe, R., and Hannaford, B. Architectures for Shared Haptic Virtual Environments. *Computers and Graphics* 21, 4, 421-432, 1997.
- [80] DELMIA, [Http://Www.Delmia.Com](http://www.Delmia.Com)
- [81] R. S. Tiruvannamalai, *Virtual Reality in Design and Manufacturing Applications*, 2002.
- [82] Melissa Saadoun, *Virtual Manufacturing and Its Implications*, France, 1999.
- [83] Barrus, J.W., *the Virtual Workshop: A Simulated Environment For Mechanical Design*, Ph.D. Dissertation, Massachusetts Institute of Technology, September, 1993.
- [84] Trika, S.N., Banerjee, P., And Kashyap, R.L., *Virtual Reality Interfaces For Feature-Based Computer-Aided Design Systems*, 1996.
- [85] Virtual Reality Applications Centre, [Http://Www.Vrac.Iastate.Edu/Research_Archive/Manufacturing/Factory/Index.Php](http://www.Vrac.Iastate.Edu/Research_Archive/Manufacturing/Factory/Index.Php)
- [86] *Manufacturing Training*, Sunrise Company, [Http://Sunrisevr.Com](http://Sunrisevr.Com)
- [87] Anthony P. Waller, John Ladbrook, *Experiencing Virtual Factories of The Future*, 2002.
- [88] Schaefer, D., Borgmann And C., Scheffter, D., *Factory Planning And The Potential of Virtual Reality*, 2001.
- [89] Matthew W. Rohrer, *Seeing Is Believing: The Importance of Visualization In Manufacturing Simulation*, Winter Simulation Conference, 2000.
- [90] Robinson, Stewert. *Simulation Model Verification And Validation: Increasing The Users Confidence*, Winter Simulation Conference, 1997.
- [91] NIST, [Http://Www.Nist.Gov/Public_Affairs/Gallery/Vrmanu.Htm](http://www.Nist.Gov/Public_Affairs/Gallery/Vrmanu.Htm)
- [92] Bayliss, G.M., Bower, A., Taylor, R.I., And Willis, P.J., *Virtual Manufacturing, Presented At CSG 94 – Set Theoretic Modelling Techniques And Applications*, S, 1994.
- [93] S. Jayaram, H. Connacher, K. Lyons, *Virtual Assembly Using Virtual Reality Techniques*, *Computer Aided Design*, P. 557-584, 1997.

- [94] R. Tesic, P. Banerjee, Exact Collision Detection Using Virtual Objects In Virtual Reality Modeling of A Manufacturing Process, J. Manufacturing System, P. 367-376, 1999.
- [95] W.B. Lee, C.F. Cheung, J.G. Li, Applications of Virtual Manufacturing In Materials Processing, P. 416-423, 2001
- [96] Matthew W. Rohrer, Seeing Is Believing: The Importance of Visualization In Manufacturing Simulation, Winter Simulation Conference, 2000.
- [97] Jerry Banks, The Future of Simulation Software: A Panel Discussion, Proceedings of The 1997 Winter Simulation Conference, 1997.
- [98] T.S. Mujber, T. Szecsi And M.S.J Hashmi, Integration of Simulation Model to Virtual Manufacturing Environment, IMC20, Dublin Ireland, 2003.
- [99] Kamat, V.R., And Martinez, J.C. "Validating Complex Construction Simulation Models Using 3D Visualization", Systems Analysis Modelling Simulation, Vol. 43, No. 4, Taylor & Francis Group, London, United Kingdom, P. 455-467, 2003.
- [100] E. Lin Et Al., Contribution to Virtual Manufacturing Background Research, [Http://Www.Isr.Umd.Edu/Labs/CIM](http://www.isr.umd.edu/labs/cim), 1995.
- [101] R. Tesic And P. Banerjee, Motion Modeling In A Virtual Reality-Aided Factory Simulator, Flexible Automa. Intell. Manuf. (FAIM) Conf, Atlanta, 1996.
- [102] F. Dai, Virtual Reality For Industrial Applications, Springer-Verlag, Berlin, Heidelberg, 1998.
- [103] N. Shahmanesh, Time Machines For Today | On How The Virtual Factory Allows Automotivecompanies To See And Change The Future of Their Production Process, Automotive Eng. P. 66-69, 1999.
- [104] A. M. Price, Virtual Product Development Case Study of The T-45 Horizontal Stabilizer, 39th AIAA/ASME/ASCE/AHS/ASC Struct., Struct. Dyn. Mater. Conf., Long Beach, CA, 1998.
- [105] R. C. Braun, P. J. Donohue And W. G. Mary, Manufacturability Tools In The Product Development Environment, 39th AIAA/ASME/ASCE/AHS/ASC Struct., Struct. Dyn. Mater. Conf., Long Beach, CA, 1998.

- [106] C. Hindman and K. B. Ousterhout, A Virtual Design System For Sheet Metal Forming, *J. Mater. Processing Tech.* 84, 1, P. 107-116, 1998.
- [107] D. A. Bodner Et Al., Virtual Prototyping of Electronics Assembly Systems, *SME Electron. Manuf. Eng.* 14, 3, P. 1-5, 1999.
- [108] H. Jack and M. Karlesky, Virtual Manufacturing Laboratory, *Ann. ASEE Conf.*, Seattle, WA, 1998.
- [109] George Chryssolouris, Et. Al. A Novel Virtual Experimentation Approach To Planning And Training For Manufacturing Processes—The Virtual Machine Shop, *INT. J. COMPUTER INTEGRATED MANUFACTURING*, VOL. 15, NO. 3, P. 214–221, Taylor & Francis, 2002.
- [110] Schaefer, D., Borgmann And C., Scheffter, D., Factory Planning and The Potential of Virtual Reality, Conference at Chalmers, Gothenburg, Sweden, October 4th-5th, 2001
- [111] T.S. Mujber, T. Szecsi And M.S.J Hashmi, A New Hybrid Dynamic Modelling Approach For Process Planning, ICAMT, Kuala Lumpur, 2004.
- [112] Vineet R. Kamat And Julio C. Martinez, GENERAL-PURPOSE 3D ANIMATION WITH VITASCOPE, Proceedings of The 2004 Winter Simulation Conference, 2004.
- [113] Charles Mclean, Swee Leong, Charley Harrell, Philomena M. Zimmerman, Roberto F. Lu, Simulation Standards: Current Status, Needs, And Future Directions, Proceedings of The 2003 Winter Simulation Conference, 2003.
- [114] Pegden, C. D., R. E. Shannon, and R. P. Sadowski. Introduction To Simulation Using SIMAN, 2nd Ed., McGraw-Hill, New York, 1995.
- [115] Law, A. M., And W. D. Kelton. Simulation Modelling and Analysis, 2nd Ed., McGraw-Hill, New York, 1995.
- [116] Banks, J., J. S. Carson II, and B. L. Nelson. Discrete-Event System Simulation, 2nd Ed., Prentice Hall, Upper Saddle River, N.J, 1996.
- [117] Osman Balci, Verification, Validation and Accreditation of Simulation Models, Proceedings of the 1997 Winter Simulation Conference, USA, 1997.
- [118] Sargent, Robert, G., Verifying and Validating Simulation Models, Proceedings of the 1996 Winter Simulation Conference, 1996.

- [119] Banks, Jerry, Haddock, Jorge, McKay, Kenneth N., and Rothenberg, Jeff, Panel Session: Validation: Expanding The Boundaries, Proceedings of The 1988 Winter Simulation Conference, P. 402-407, 1988.
- [120] Balci, Osman, Validation, Verification, And Testing Techniques Throughout The Life Cycle of a Simulation Study, Annals of Operations Research, 1994.
- [121] Sargent, Robert, G., Some Subjective Validation Methods Using Graphical Displays of Data, Proceedings of The 1996 Winter Simulation Conference, 1996.
- [122] Banks, Jerry, Carson, John, S. And Nelson, Barry, L., Discrete-Event System Simulation, Prentice-Hall International Series, 1996.
- [123] Law, Averill, and Kelton, David, Simulation Modeling And Analysis, McGraw-Hill, Inc., 1991.
- [124] Kleijnen, Jack, P.C., Verification and Validation of Simulation Models, European Journal of Operational Research 82, P.145-162, 1995.
- [125] Carson, John S., Verification and Validation: A Consultant's Perspective, Proceedings of the 1989 Winter Simulation Conference, 1989.
- [126] T.S. Mujber, T. Szecsi And M.S.J Hashmi Addressing Factory Design And Planning Using Simulation And Virtual Reality Technologies, FAIM 2005, Spain, 2005.
- [127] Superscape Inc. Superscape VRT 5.6 User's Guide. U.K.: Superscape VR Plc, Third Edition, 1998.
- [128] Vlatka Hlupic. Simulation Software: An Operational Research Society Survey of Academic and Industrial Users. Proceedings of the 2000 Winter Simulation Conference, 2000.
- [129] Lanner Group, Inc. (1998). Witness 2000, User Manuals.
- [130] Glenn Drake, Jeffrey S. Smith: Simulation System for Real-Time Planning, Scheduling, and Control, Winter Simulation Conference, 1996.

APPENDICES

APPENDIX A

The attached CD of this thesis includes the source code of the VR-Simulator software divided into seven files as following:

1. Source code of the Graphic User Interface (GUI) of the VR-Simulator:

This file contains the source code which was used to design and develop the GUI of the VR-Simulator software.

2. Source code of the Process Modelling Tool (PMT) algorithms:

This file contains the source code of the algorithms which were used to collect information from the user of a manufacturing system to define resources and the process plan of the modelled system.

3. Source code of the algorithms for generating simulation models automatically in Witness:

This file contains the source code of the algorithms which were used to generate simulation model automatically in Witness.

4. Source code of the algorithms for generating virtual models automatically:

This file contains the source code of the algorithms which were used to generate virtual model automatically in Superscape environment.

5. Source code of the algorithms for loading the process plan:

This file contains the source code of the algorithms which were used to load the process plan from the VR-Simulator to the simulation model.

6. Source code of the factory layout algorithms:

This file contains the source code of the algorithms which were used to read the layout of a manufacturing system and then provide the positions data of the resources to the SMT and VMT to construct the layout of the modelled system.

7. Source code of the statistical algorithms:

This file contains the source code of the algorithms which were used to read the status of the resources of the simulation model to collect and display the statistics of the modelled system.

8. Source code of the SCL algorithms:

This file contains the source code of Superscape Control Language (SCL) algorithms which were used in Superscape environment to generate virtual objects to construct a virtual model of a manufacturing system.