

**A RETRIEVAL METHOD (DFM FRAMEWORK) FOR  
AUTOMATED RETRIEVAL OF DESIGN FOR ADDITIVE  
MANUFACTURING PROBLEMS**

A Dissertation  
Presented to  
The Academic Faculty

by

Sungshik Yim

In Partial Fulfillment  
of the Requirements for the Degree  
Doctor of Philosophy in the  
School of Mechanical Engineering

Georgia Institute of Technology  
May, 2007

**A RETRIEVAL METHOD (DFM FRAMEWORK) FOR  
AUTOMATED RETRIEVAL OF DESIGN FOR ADDITIVE  
MANUFACTURING PROBLEMS**

Approved by:

Dr. David W. Rosen, Advisor  
School of Mechanical Engineering  
*Georgia Institute of Technology*

Dr. Charles Eastman  
School of Architecture  
*Georgia Institute of Technology*

Dr. Janet K. Allen  
School of Mechanical Engineering  
*Georgia Institute of Technology*

Dr. Nelson C. Baker  
School of Civil Engineering  
*Georgia Institute of Technology*

Dr. Chris Paredis  
School of Mechanical Engineering  
*Georgia Institute of Technology*

Date Approved: March 1, 2007

## ACKNOWLEDGEMENTS

In completing this dissertation, I have been accompanied and supported by many people. I'd like to express my sincere gratitude for all of them.

First of all, I'd like to thank Dr. David Rosen who is my Ph.D advisor. Dr. Rosen provided me with this research opportunity and continuous support in completing this research work. I owe him lots of gratitude for guiding me both technically and philosophically.

Secondly, I'd like to thank my committee members Dr. Janet Allen, Dr. Chris Paredis, Dr. Charles Eastman, and Dr. Nelson Baker. The committee members provided me with critical guidance and comments which helped me realize the value of the research.

Thirdly, I'd like to thank Mr. Andrew Dugenske who provided me with employment and with a research opportunity in the factory information systems laboratory at Georgia Tech. Mr. Andrew Dugenske provided me with the insight for developing robust and economical software.

Fourth, I'd like to thank Dr. Jeff Donnell for helping me to write a clear and correct dissertation. Dr. Jeff Donnell supported me tremendously in developing this dissertation by correcting my language and suggesting appropriate words.

Fifth, I'd like to thank the members of the system realization laboratory, rapid prototyping and manufacturing institute, and factory information systems laboratory. The members in those labs provided me with tremendous resources to make progress in this research work.

Finally, I'd like to thank my family including my parents, sister, wife, and two sons for encouraging me to finish the research work. I especially thank my parents in supporting me both financially and mentally to complete the dissertation.

# TABLE OF CONTENTS

ACKNOWLEDGEMENTS .....	III
LIST OF TABLES .....	XIII
LIST OF FIGURES.....	XVI
SUMMARY .....	XXII
CHAPTER 1: INTRODUCTION .....	1
1.1 BACKGROUND .....	1
1.1.1 Additive Manufacturing .....	5
1.1.2 Geometric tailoring .....	10
1.1.3 Challenges and research objective .....	23
1.1.3.1 Robot Arm example .....	23
1.1.3.2 Challenges .....	28
1.1.4 Approach .....	36
1.2 RESEARCH QUESTIONS AND HYPOTHESES .....	41
1.3 VALIDATION.....	45
1.3.1 Theoretical structure validation.....	45
1.3.2 Empirical structure validation .....	46

1.3.3 Empirical performance validation .....	47
1.3.4 Theoretical performance validation .....	47
1.4 THESIS ORGANIZATION .....	47
CHAPTER 2: LITERATURE REVIEW .....	50
2.1 DESIGN FOR MANUFACTURING .....	50
2.1.1 Product-process fit.....	51
2.1.1.1 Feature recognition.....	52
2.1.1.2 Feature-based design.....	55
2.1.2 Simultaneous design of product and process .....	57
2.1.2.1 Feature representation .....	58
2.1.2.2 Integration Framework.....	59
2.1.2.3 Knowledge sharing.....	61
2.2 ENGINEERING INFORMATION MANAGEMENT .....	63
2.2.1 Design repository .....	67
2.3 SUMMARY .....	72

CHAPTER 3: SCOPE OF THE DESIGN AND PROCESS PLANNING DOMAINS AND DISCUSSIONS ON MANUFACTURING RULES (ORIGIN AND CHARACTERISTICS).....	73
3.1 INFORMATION SPACE FOR DESIGN AND PROCESS PLANNING DOMAINS.....	73
3.1.1 Design requirements.....	74
3.1.2 Process variables .....	78
3.1.3 Research scope .....	79
3.2 MANUFACTURING PROBLEM TEMPLATE (MPT).....	81
3.3 MANUFACTURING RULES.....	85
3.3.1 Origin and Background.....	85
3.3.1.1 Definition, Structure and Derivation.....	89
3.3.1.2 Characteristics .....	92
3.4 THE FUNCTIONALITIES OF MANUFACTURING RULES IN RELATING THE GIVEN DESIGN REQUIREMENTS TO THE RELEVANT DFM PROBLEMS THROUGH APPROPRIATE MPTs...	101
3.5 SUMMARY .....	103
CHAPTER 4: MANUFACTURING RULES DISCOVERY IN STEREO LITHOGRAPHY AND MAPPING BETWEEN DESIGN AND PROCESS PLANNING DOMAINS.....	105
4.1 RULE DISCOVERY IN STEREO LITHOGRAPHY .....	105
4.1.1 Manufacturing rules discovery for part orientation.....	108

4.1.2 Manufacturing rules discovery for layer thickness .....	116
4.1.3 Manufacturing rules discovery for overcure .....	118
4.1.4 Combining manufacturing rules for different process variables .....	119
4.1.5 Manufacturing rules generalization to values other than value 3 and 4.....	122
4.1.6 Relation between the discovered manufacturing rules and MPTs .....	123
4.2 RELATING DESIGN REQUIREMENTS TO RELEVANT DFM PROBLEMS .....	127
4.3 MATHEMATICAL STRUCTURE OF DESIGN TO MPT MAPPING AND UTILIZATION OF THE MAPPING IN RETRIEVING DFM PROBLEMS.....	132
4.3.1 Structure preserving properties .....	132
4.3.2 Retrieval and ranking using the mapping.....	137
4.4 SUMMARY .....	142
CHAPTER 5: INFORMATION MODELING (USING DESCRIPTION LOGICS).....	144
5.1 CHARACTERISTICS AND REQUIREMENTS.....	144
5.2 REPRESENTATION FORMALISMS .....	149
5.2.1 Logical reasoning system .....	152
5.2.2 Probabilistic reasoning system.....	154
5.2.3 Learning .....	155

5.2.4 Natural language processing system .....	156
5.2.5 Selection.....	157
5.2.6 Description logics.....	160
5.2.6.1 Basics .....	161
5.2.6.2 Utilization example .....	162
5.2.6.3 Computational complexities.....	164
5.2.6.4 Consistency and correctness of subsumption in DL .....	164
5.3 DESCRIPTION LOGICS IMPLEMENTATIONS.....	166
5.4 SOFTWARE ENVIRONMENT FOR ENCODING THE MANUFACTURING RULES.....	175
5.5 HYPOTHESES VALIDATION.....	185
5.6 SUMMARY .....	186
CHAPTER 6: RETRIEVAL METHOD (RETRIEVAL AND RANKING) .....	188
6.1 DFM FRAMEWORK: COMPONENTS.....	188
6.1.1 Overview .....	189
6.1.1.1 Specific requirements from storing algorithm .....	196
6.1.1.2 Specific requirements from retrieval algorithm .....	197

6.1.2 Information models and meta rule taxonomy .....	200
6.1.3 Storing algorithm.....	201
6.1.4 Retrieval algorithm.....	204
6.2 RETRIEVAL DEMONSTRATION USING ROBOT ARM EXAMPLE.....	210
6.2.1 Representation and information modeling .....	214
6.2.2 Storing .....	219
6.2.3 Retrieval .....	223
6.3 HYPOTHESES VALIDATION.....	227
6.4 SUMMARY .....	229
CHAPTER 7: VALIDATING DFM FRAMEWORK.....	231
7.1 OVERVIEW .....	231
7.2 TEST CASE SELECTION AND SAMPLES OF DESIGN REQUIREMENTS AND DFM PROBLEMS .....	235
7.2.1 Criteria.....	235
7.2.2 Wind tunnel surface design .....	236
7.2.3 Realization of the condition part of manufacturing rules.....	240
7.3 TEST CASE SET UP.....	248

7.3.1 DL applicability measure .....	248
7.3.2 Retrieval performance measure.....	251
7.3.3 Computational feasibility measure.....	260
7.4 TEST BED IMPLEMENTATION.....	264
7.5 RESULTS.....	270
7.5.1 Result for description logic’s applicability measure .....	270
7.5.2 Result for retrieval performance measure .....	272
7.5.3 Result for computational feasibility measure.....	284
7.6 HYPOTHESES VALIDATION.....	289
7.7 SUMMARY .....	290
CHAPTER 8: ACHIEVEMENT AND RECOMMENDATIONS .....	293
8.1 SUMMARY OF HYPOTHESES VALIDATION.....	293
8.1.1 Question 1 and Hypothesis 1.....	294
8.1.2 Question 2 and Hypothesis 2.....	295
8.1.3 Question 3 and Hypothesis 3.....	295
8.1.4 Question 4 and Hypothesis 4.....	296
8.2 THEORETICAL PERFORMANCE VALIDATION .....	296

8.3 RESEARCH CONTRIBUTIONS.....	298
8.3.1 Design for manufacture (DFM).....	298
8.3.2 Engineering information management .....	300
8.4 LIMITATIONS AND FUTURE WORKS .....	302
8.4.1 Limitations .....	302
8.4.2 Future work .....	304
8.4.2.1 Research issues and systematic guidelines for extending the retrieval method.....	304
8.4.2.2 Extending the design requirements .....	308
8.4.2.3 Extending the retrieval method for other manufacturing processes.....	308
8.4.2.4 Extending the retrieval method for other DFM knowledge .....	309
APPENDICES.....	310
APPENDIX A: ATOMIC CONCEPTS, ROLES, MANUFACTURING RULES, AND QS ENCODING IN ALE .....	310
APPENDIX B: EXPECTED RESULTED FOR DESCRIPTION LOGICS’ APPLICABILITY MEASURE .....	318
APPENDIX C: EXPECTED QUERY RESULTS FOR THE RETRIEVAL PERFORMANCE MEASURE .....	321
REFERENCES:.....	324

## LIST OF TABLES

Table 1.1 Word formulation of cDSP for geometric tailoring (GT) .....	13
Table 1.2 Math formulation of cDSP for geometric tailoring (GT).....	14
Table 1.3 Word formulation of rapid prototyping process planning (RP-PP) .....	15
Table 1.4 Math formulation of rapid prototyping process planning (RP-PP).....	16
Table 1.5 Examples of empirical models .....	29
Table 1.6 Example of feasible spaces of part orientation and layer thickness.....	31
Table 1.7 Research Questions and Hypotheses.....	42
Table 2.1 Gap summary .....	72
Table 3.1 Process variables and their feasible value spaces .....	79
Table 3.2 The research scope .....	81
Table 3.3 Tabular form of surface finish model.....	90
Table 3.4 Generalized tabular form of accuracy model .....	92
Table 3.5 Example tabular form of accuracy model .....	97
Table 3.6 Example of meta rule and rule .....	98
Table 3.7 Complex meta rule derivation example .....	99
Table 3.8 Research questions and hypotheses (1 and 2).....	104
Table 4.1 Template for meta rule discovery in Stereolithography.....	106
Table 4.2 Relation between meta rules and MPTs.....	125
Table 4.3 Research question and hypotheses (1 and 2).....	142
Table 4.4 New knowledge discovered in DFM.....	143
Table 5.1 Characteristics of method operation environment .....	147
Table 5.2 Requirements for successful functioning of method.....	147

Table 5.3 Representations and reasoning mechanisms .....	152
Table 5.4 Description logics and their computational complexity.....	164
Table 5.5 Atomic concepts.....	166
Table 5.6 Implementation of atomic concept in description logics .....	168
Table 5.7 Implementation of relation (role) in description logics.....	169
Table 5.8 Research questions and hypotheses (1 and 2).....	185
Table 5.9 New knowledge discovered in EIM.....	187
Table 6.1 Retrieval example.....	197
Table 6.2 Relation between accuracy measurement value range and feasible space of process variables in Stereolithography.....	211
Table 6.3 Example DFM problems.....	213
Table 6.4 Example queries (design requirements).....	213
Table 6.5 Atomic concepts and roles for robot arm example .....	214
Table 6.6 Retrieval result in robot arm example.....	224
Table 6.7 Ranking result for meta rule (opposite-perpendicular) in robot arm example	225
Table 6.8 Corresponding research questions and hypotheses .....	228
Table 6.9 New knowledge discovered in DFM.....	230
Table 7.1 Summary of testing strategy.....	234
Table 7.2 Testing criteria .....	236
Table 7.3 Accuracy measurements specification on surfaces.....	242
Table 7.4 Additional design requirements for rules and DFM problems.....	243
Table 7.5 Scenarios and sets of DFM problems in the repository .....	250
Table 7.6 Design requirements for querying DFM problems .....	256
Table 7.7 Example of query result for d.r.4 .....	257

Table 7.8 Accuracy models.....	261
Table 7.9 Incremental accuracy models.....	263
Table 7.10 Incremental rules.....	264
Table 7.11 Retrieved manufacturing rules.....	273
Table 7.12 Relation between hypotheses and testing.....	289
Table 7.13 The research objectives and the corresponding accomplishment.....	291
Table 8.1 Research questions and hypotheses.....	294
Table 8.2 Conditions that need to be satisfied for the retrieval method to perform correctly.....	297
Table 8.3 Specific contributions of the research in DFM and engineering information management.....	302
Table 8.4 Limitations.....	303
Table 8.5 Research issues for extending the retrieval method in other domains of design and manufacturing.....	305
Table 8.6 Systematic procedure of extending the retrieval method in a domain where subsumption relation is not supported.....	307
Table 8.7 Manufacturing processes.....	309

## LIST OF FIGURES

Figure 1.1 Stereolithography process.....	6
Figure 1.2 Robot Arm .....	7
Figure 1.3 Example of stair stepping effect .....	8
Figure 1.4 Effect of part orientation on stair stepping effect .....	9
Figure 1.5 Graphical illustration of geometric tailoring .....	11
Figure 1.6 MPGT/RP formulation from RP-PP and GT.....	18
Figure 1.7 Solution strategy to solve ‘System cDSP’ .....	20
Figure 1.8 Modified RP-PP and modified MPGT/RP Problem formulation .....	21
Figure 1.9 Robot arm geometry and design problem (GT).....	24
Figure 1.10 Robot Arm manufacturing problem (RP-PP) .....	25
Figure 1.11 Robot Arm DFM problem (MPGT/RP) .....	26
Figure 1.12 Robot Arm manufacturing problem after decomposition.....	27
Figure 1.13 Robot Arm DFM problem after decomposition .....	28
Figure 1.14 Two solution strategies .....	30
Figure 1.15 Sample retrieved DFM problem for robot arm example .....	35
Figure 1.16 The manufacturing rules taxonomy that map design and process planning domains .....	37
Figure 1.17 Example of retrieval scenario .....	37
Figure 1.18 Components of the retrieval method (DFM framework).....	38
Figure 1.19 Validation square .....	45
Figure 2.1 Traditional sequential approach of design and manufacturing.....	50
Figure 2.2 Summary of research that relate design to DFM knowledge.....	62
Figure 2.3 Interoperability problem classification[71] .....	66

Figure 3.1 Concepts for forming design requirements.....	76
Figure 3.2 Relation between concepts for forming design requirements.....	76
Figure 3.3 Example of design requirement formed by determined concepts and relations .....	77
Figure 3.4 Process variables in Stereolithography .....	78
Figure 3.5 Examples of manufacturing problem templates (MPT) .....	83
Figure 3.6 Surface finish model .....	87
Figure 3.7 example.....	88
Figure 3.8 Structure of manufacturing rule.....	89
Figure 3.9 Manufacturing rule example.....	89
Figure 3.10 Example of complex manufacturing rules .....	93
Figure 3.11 Example of complex manufacturing rule derivation using complex manufacturing rules.....	95
Figure 3.12 Hierarchical relation in condition and result part .....	96
Figure 3.13 Hierarchical relation in manufacturing rules .....	96
Figure 3.14 Terminology taxonomy for meta rule and rule.....	100
Figure 3.15 Example of hierarchical relation among meta rule and rule.....	101
Figure 3.16 Relation between design requirements, manufacturing rules, and manufacturing problem templates .....	102
Figure 4.1 0° and 90° orientations for cylindrical and flat surface .....	107
Figure 4.2 Example of meta rule discovery in Stereolithography.....	109
Figure 4.3 Example of meta rule derivation in Stereolithography.....	111
Figure 4.4 Meta rules in Stereolithography.....	113
Figure 4.5 Meta rule taxonomy in Stereolithography .....	114

Figure 4.6 Mapping design space and feasible space of part orientation.....	116
Figure 4.7 Meta rules for layer thickness and overcure .....	117
Figure 4.8 Example of meta rules for part orientation and layer thickness .....	118
Figure 4.9 Meta rule for overcure .....	119
Figure 4.10 Example of supplementing the meta rule .....	120
Figure 4.11 Example of supplemented meta rule taxonomy.....	121
Figure 4.12 Example of meta rules generalization for values 5~8.....	122
Figure 4.13 Illustration of meta rule and rules functionality.....	128
Figure 4.14 Example for illustrating functionality of meta rule hierarchy .....	130
Figure 4.15 Generic mapping between design and MPT .....	133
Figure 4.16 Example of structure preserving manufacturing rules.....	135
Figure 4.17 Properties for preserving the structure of the design requirements and MPTs taxonomy in the manufacturing rules taxonomy.....	136
Figure 4.18 Proposition and proof for correctness of taxonomy.....	139
Figure 4.19 Example of meta rules and Qs taxonomy .....	140
Figure 4.20 Retrieval and ranking.....	141
Figure 5.1 Tasks to be performed using representation .....	145
Figure 5.2 Relation between characteristics and requirements .....	148
Figure 5.3 Example of first order logic.....	150
Figure 5.4 Support of the formalisms in satisfying requirements .....	158
Figure 5.5 Description logic representation example.....	163
Figure 5.6 Example of simple design requirement formation.....	170
Figure 5.7 Meta rule representations by atomic concepts and roles .....	171
Figure 5.8 Description logic (ALE) representation of manufacturing rules .....	173

Figure 5.9 Implementation environments for design requirement and manufacturing rule taxonomy representation .....	175
Figure 5.10 Atomic concept implementation using Protégé .....	177
Figure 5.11 Role implementation using Protégé .....	178
Figure 5.12 Meta rule implementation using Protégé .....	179
Figure 5.13 Meta rule example .....	179
Figure 5.14 Meta rule taxonomy computation .....	180
Figure 5.15 Addition atomic concepts for representing Qs .....	181
Figure 5.16 Example of Q representation .....	182
Figure 5.17 Qs taxonomy .....	183
Figure 5.18 Example of mapping process planning to design requirements .....	184
Figure 6.1 DFM framework components .....	189
Figure 6.2 General procedures of storing and retrieving in DFM framework .....	190
Figure 6.3 DFM problem repository structure example .....	194
Figure 6.4 Storing algorithm .....	202
Figure 6.5 Retrieval algorithm .....	205
Figure 6.6 Metric for part orientation .....	206
Figure 6.7 Metric for layer thickness, overcure, and accuracy models .....	206
Figure 6.8 Robot Arm example .....	211
Figure 6.9 Simple meta rule for robot arm example .....	212
Figure 6.10 Atomic concept and role implementation for robot arm example .....	215
Figure 6.11 Example of implemented meta rule in robot arm example .....	216
Figure 6.12 Meta rule taxonomy computation for robot arm example .....	217

Figure 6.13 Example representation of DFM problem 3 .....	217
Figure 6.14 Example representation of design requirement (query) 2.....	218
Figure 6.15 Result of finding direct parent for each DFM problem in robot arm example .....	219
Figure 6.16 Example of rule representation for DFM problem 3 .....	220
Figure 6.17 Introduction of DFM problem 1 .....	221
Figure 6.18 Introduction of DFM problem 3 .....	221
Figure 6.19 Introduction of DFM problem 2 .....	222
Figure 6.20 Finding meta rule for each query in robot arm example.....	223
Figure 6.21 Computation example of ranking metric for part orientation and accuracy models usage .....	226
Figure 7.1 Wind tunnel surface schematic .....	237
Figure 7.2 Surface structure (compliant mechanism) .....	238
Figure 7.3 Relative orientations and the associated surfaces .....	241
Figure 7.4 Development of rules.....	244
Figure 7.5 Meta rules and rules taxonomy.....	246
Figure 7.6 Illustration of testing strategy .....	249
Figure 7.7 Example design problem and design requirements using robot arm.....	253
Figure 7.8 Example design problem and design requirements using wind tunnel surface .....	254
Figure 7.9 PC Mouse design problem.....	259
Figure 7.10 Graphical illustration of DFM framework test bed .....	265
Figure 7.11 Graphical user interface for forming query .....	267
Figure 7.12 Graphical user interface for displaying retrieval results.....	268
Figure 7.13 Example of expected taxonomy for set 2 of top to bottom scenario .....	271

Figure 7.14 Example of computed taxonomy for set 2 of top to bottom scenario.....	271
Figure 7.15 Example of repository structure.....	273
Figure 7.16 Retrieved DFM problem.....	274
Figure 7.17 Retrieved and formulated DFM problem.....	280
Figure 7.18 Reusing process of accuracy model.....	281
Figure 7.19 Summary of support in DFM problem formulation and solution generation .....	283
Figure 7.20 Computational feasibility of subsumption using ALE .....	285
Figure 7.21 Manufacturing rule taxonomy computation capability with increasing complexity in design requirements .....	285
Figure 7.22 Manufacturing rule taxonomy computation time with increasing complexity of design requirements and number of rules .....	286

## SUMMARY

*Problem:* The process planning task for a given design problem in additive manufacturing can be greatly enhanced by referencing previously developed process plans. However, identifying appropriate process plans for the given design problem requires appropriate mapping between the design domain and the process planning domain. Hence, the objective of this research is to establish mathematical mapping between the design domain and the process planning domain such that the previously developed appropriate process plans can be identified for the given design task. Furthermore, identification of an appropriate mathematical theory that enables computational mapping between the two domains is of interest. Through such computational mapping, previously developed process plans are expected to be shared in a distributed environment using an open repository.

*Approach:* The design requirements and process plans are discretized using empirical models that compute exact values of process variables for the given design requirements. Through this discretization, subsumption relations among the discretized design requirements and process plans are identified. Appropriate process plans for a given design requirement are identified by subsumption relations in the design requirements. Also, the design requirements that can be satisfied by the given process plans are identified by subsumption relations among the process plans. To computationally realize such mapping, a description logic ( $\mathcal{AL}\mathcal{E}$ ) is identified and justified to represent and compute subsumption relation. Based on this investigation, a retrieval method (DFM framework) is realized that enables storage and retrieval of process plans.

*Validation:* Theoretical and empirical validations are performed using the validation square method. For the theoretical validation, an appropriate description logic ( $\mathcal{AL}\mathcal{E}$ ) is identified and justified. Also, subsumption utilization in mapping two domains and realizing the DFM framework is justified. For the empirical validation, the storing and retrieval performance of the DFM framework is tested to demonstrate its theoretical validity.

*Contribution:* In this research, two areas of contributions are identified: DFM and engineering information management. In DFM, the retrieval method that relates the design problem to appropriate process plans through mathematical mapping between design and process planning domain is the major contribution. In engineering information management, the major contributions are the development of information models and the identification of their characteristics. Based on this investigation, an appropriate description logic ( $\mathcal{AL}\mathcal{E}$ ) is selected and justified. Also, corresponding computational feasibility (non deterministic polynomial time) of subsumption is identified.

## **CHAPTER 1: INTRODUCTION**

This chapter presents the overview of the dissertation. There are four sections; background, research questions and hypotheses, validation, and thesis organization. The background section presents a detailed description of the problem domain that this research attempts to address (section 1.1). Also, a brief description of the solution approach is presented. The research questions and hypotheses section presents a detailed description of the research issues, questions, and corresponding hypothesis (section 1.2). The research issues are identified through clear illustration of the challenges to be addressed in realizing the approach. The validation section describes the validation strategy (section 1.3). In this research, we propose to use the validation square, so a brief description of each component of the validation square is presented. The thesis organization section provides a road map of this dissertation (section 1.4).

### **1.1 Background**

The motivation of this research is to support the redesign and fabrication of existing parts using new technology and new materials for teams working in a distributed environment. The new technology is confined to layer-based additive manufacturing[1-3]. Since new technology and new material are used to fabricate an existing part, its geometry may need to be modified during the redesign process to retain important mechanical characteristics of the existing part. In this research, the term distributed environment refers to an environment where engineers are distributed culturally as well as geographically.

To support this type of a redesign and fabrication task, a previously developed design for manufacturing (DFM) method called geometric tailoring[4, 5] is identified as

an appropriate method. The key idea of the method is to design and fabricate a functional prototype that simulates the production part while having different material properties than the production part. The method provides a systematic procedure for simultaneously designing a part and a manufacturing process such that those two satisfy the design requirement for the prototype. In this method, the task of simultaneously designing product and process is performed through a series of problem formulations and solution generations. The design requirements and objective are formulated as a design problem by the designer and transferred to the process planning engineer. Then, the process planning engineer formulates an appropriate manufacturing problem that can determine process variables for the design problem. The design for manufacturing problem (DFM problem) is then formulated by combining the design problem and the manufacturing problem. Through solving the DFM problem, the solid geometry of prototype and process variables that satisfy the given design requirements are determined. In geometric tailoring, the DFM problem is decomposed into sub problems and they are solved to generate the solution for the DFM problem. Specific details are presented in the geometric tailoring section.

In using geometric tailoring for the redesign and fabrication of an existing product in a distributed environment, a couple of difficulties are identified. Those are formulating and solving the DFM problems. In formulating DFM problems, the difficulty is modeling the design requirements in terms of manufacturing process variables. Proper modeling of design requirements requires in depth knowledge of design and manufacturing. In solving the DFM problem, the difficulty is identifying appropriate solution strategy. In geometric tailoring, decomposition is provided as a

solution strategy that is a time- and effort-intensive task. Decomposition is required when feasible spaces of process variables are infinite. In a case where the feasible spaces of the process variables are discrete, the decomposition can be avoided by generating a solution for each discrete value of the process variable and then selecting the best solution. Utilizing decomposition for solving DFM problems that do not require decomposition is a huge waste of time and effort. Also, it could lead to an inaccurate solution. Therefore, selecting an appropriate solution strategy helps obtaining accurate solutions efficiently. In this research, two solution strategies are identified: decomposition and multiple solutions. To determine the proper solution strategy, the correct feasible spaces of process variables need to be identified and that requires in depth knowledge of design and manufacturing. Therefore, the two difficulties identified above can be summarized as developing design requirements models and identifying feasible spaces of process variables. Identifying feasible spaces of process variables is partially performing process planning. Developing design requirements models in terms of identified feasible spaces of process variables support further process planning by allowing design requirements evaluation. Therefore, proper formulation and solution generation of the DFM problem requires an expert's process planning knowledge.

The objective of this research is to support process planning for part redesign and fabrication. The approach is to support process planning by retrieving previously formulated and solved DFM problems that provide necessary process planning knowledge. In this research, such DFM problems are called relevant DFM problems. The approach takes advantage of the fact that there is only a finite number of ways to satisfy various design problems in terms of process planning. In this research,

manufacturing rules and their subsumption relations are identified as the entities that determine appropriate process plans for the given design problem. Therefore, the approach classifies DFM problems based on manufacturing rules such that the DFM problems in the same class share the same process planning problem formulation. The design problems are related to appropriate manufacturing rules first. Then, the manufacturing rules and their subsumption relations identify relevant DFM problems through determining appropriate process plans. The retrieval method, called the DFM framework, is proposed and developed here. The method is intended to provide services for storing and retrieving DFM problems using an open repository in a distributed environment.

In developing the method, it is crucial to use a formalism to represent and compare the concepts (design requirements) to determine subsumption relations. In this research, description logic (DL)[6, 7] is identified as the appropriate formalism. To realize the retrieval method properly, several research issues and tasks are identified and listed below.

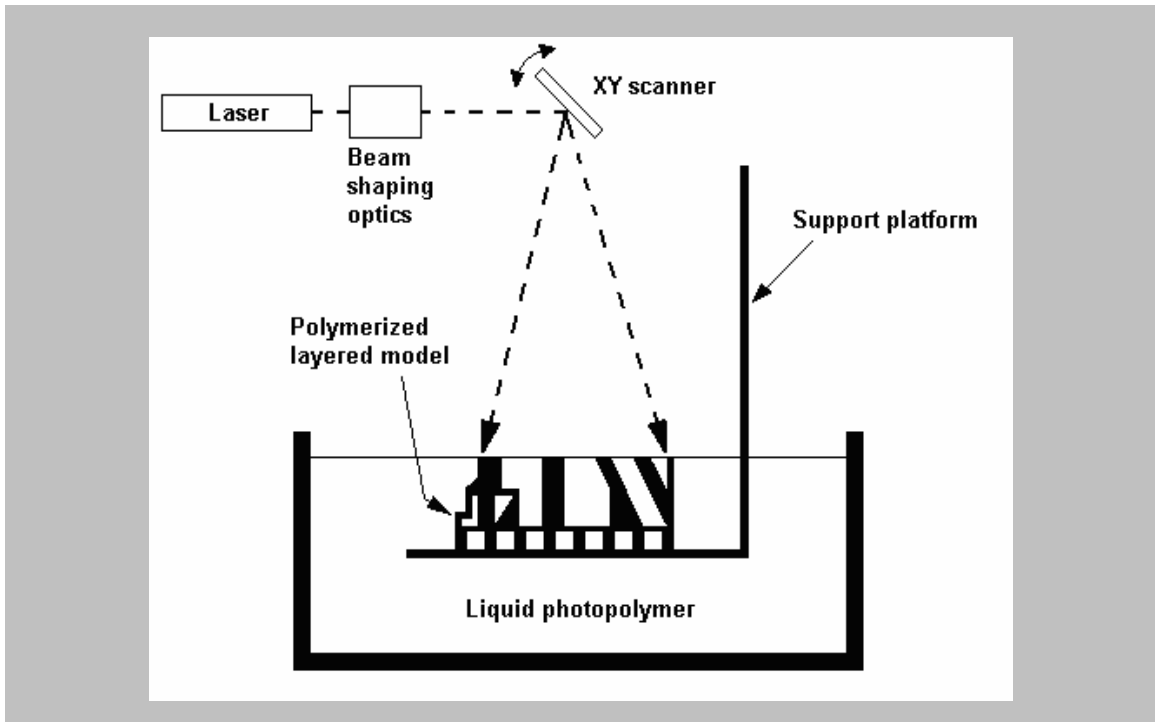
- Identify and justify information space for design problems and process planning domains
- Identify and justify the mathematical relation between design and process planning domains
- Identify and justify the appropriate DL for realizing the mapping between design domain and process planning domain
- Realize and justify the retrieval method

In this research, the retrieval method that is applicable in the design for additive manufacturing is realized by addressing the research issues listed above. Detailed discussions on the listed research issues are presented in section 1.1.4. The following paragraphs discuss details of background information including additive manufacturing, geometric tailoring, research objective, and approach.

### **1.1.1 Additive Manufacturing**

Additive manufacturing, also known as rapid prototyping (RP), refers to the process of fabricating parts layer-by-layer directly from the CAD model[1, 8, 9]. The key idea of the method is to fabricate a solid geometry by depositing material layer by layer. Hence, there is a fundamental difference between this technology and other traditional manufacturing technology such as machining (material removal) or casting (deform material).

Currently, there are about 30 RP technologies available for model fabrication based on various principles of additive manufacturing[4, 10]. Those include Stereolithography (SL), Selective Laser Sintering (SLS), Ink-Jet Printing, Fused Deposition Modeling (FDM), etc. These techniques can be classified based on two criteria; 1 the material used and 2 part building techniques. Figure 1.1 shows the fundamental fabrication technique of SL which is the most popular fabrication process among available RP technologies. The following paragraphs briefly describe the procedures.

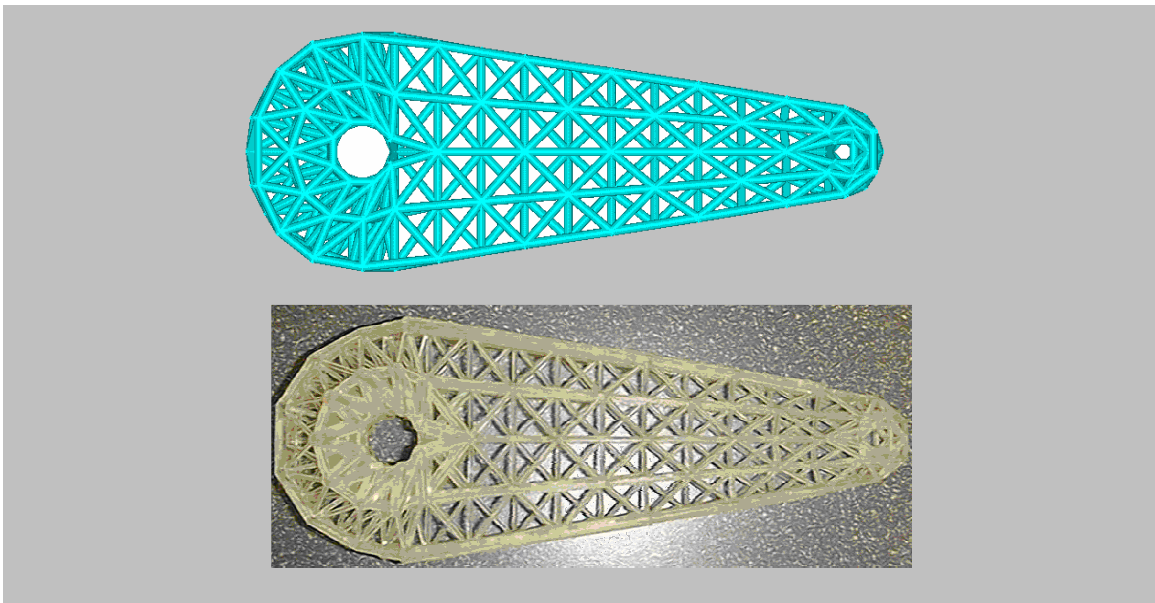


**Figure 1.1 Stereolithography process**

Stereolithography uses a platform, a laser and a liquid photopolymer that solidifies when exposed to the laser. First, the elevator (support platform) is located at some distance from the surface of the liquid equal to the first layer. Then, the laser beam scans the surface of the liquid following the contour of the slice. After contour scan, the interior of the contour is hatched by a hatch pattern. Once the hatching is completed, the elevator moves down to permit scanning the next layer. The final step is to remove the part from the liquid and place it in an oven to cure[4, 11, 12].

There are several advantages to such additive manufacturing technologies compared to traditional manufacturing processes. First, it is faster. In most instances of additive manufacturing, the manufacturing processes are completely automated from the CAD model to fabrication. Secondly, it can generate complex shapes that are impossible to create using traditional manufacturing processes. This is because material is deposited

layer by layer. For example, a simple cube and a sculptured solid are equally easy to manufacture in additive manufacturing. Therefore, many of the geometric shapes that used to be impossible to fabricate are now feasible due to additive manufacturing technology. Figure 1.2 shows a robot arm that was designed and fabricated in the *Rapid Prototyping and Manufacturing Institute (RPMI)* at *Georgia Institute of Technology*. This new robot arm design is lighter than traditional robot arms yet it is strong enough to be suitable for high speed robots. Also, additive manufacturing technology can produce multi-material parts and parts with embedded electronics[13].

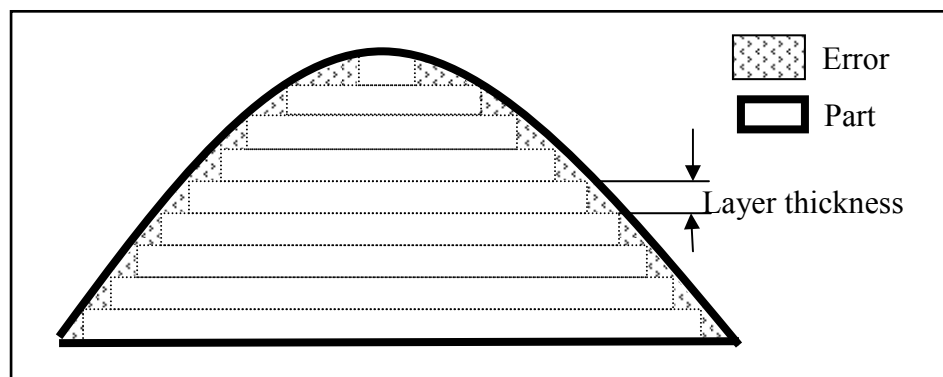


**Figure 1.2 Robot Arm**

So far, additive manufacturing technologies are mainly used to fabricate prototypes; however, direct manufacturing of parts using additive manufacturing technologies, called rapid manufacturing, is getting more attention recently[8, 14]. The idea of rapid manufacturing is not quite practical for most applications today due to its limitations of speed, part size, accuracy, surface finish, and material properties. However, there are

commercially available products that are manufactured by additive manufacturing technology. Those are usually low-volume and customized products such as hearing aid shells.

One of the most significant issues in manufacturing is satisfying the accuracy requirements specified on various geometric features within limited costs. Each manufacturing process has different capabilities in realizing various geometric features and satisfying accuracy requirements. Satisfying accuracy requirements in additive manufacturing has been under intensive investigation. The major cause of difficulty in satisfying accuracy requirements is the stair stepping effects shown in Figure 1.3[15-17].

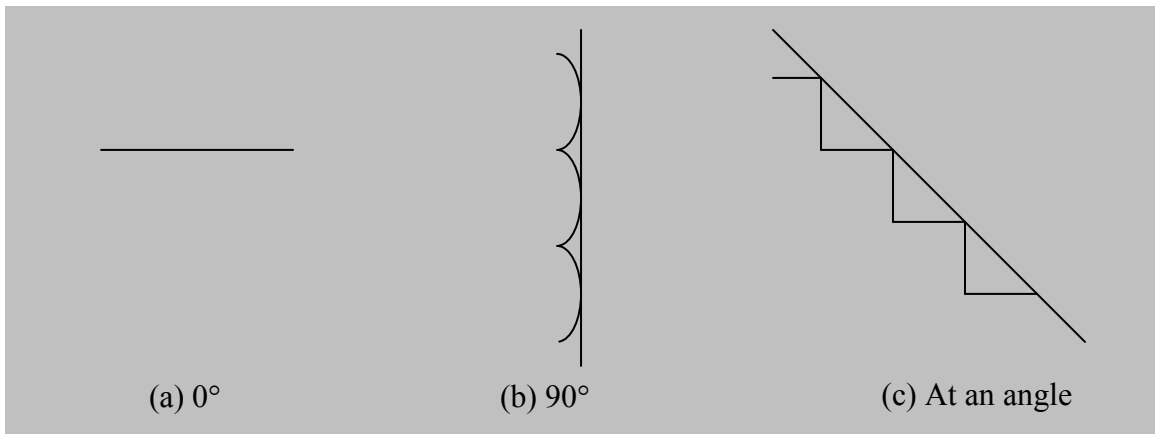


**Figure 1.3 Example of stair stepping effect**

The stair stepping effect in Figure 1.3 is caused by depositing material layer by layer. Hence, this effect is observed in curved or sloped surfaces. Lynn and West studied to identify the relationships between accuracy measurements specified on various surfaces and Stereolithography process variables[18-20]. In their experiments, mathematical models for relating accuracy measurements specified on various surfaces (flat, cylindrical, conical) to process variables are developed. Lynn showed that among the many process variables in Stereolithography, part orientation, layer thickness,

overcure, sweep period, and z-level wait have significant effect on accuracy. Among those process variables, part orientation and layer thickness have the most significant effect on the stair stepping effect.

As shown in Figure 1.3, a smaller layer thickness can definitely reduce the error caused by stair stepping effect. However, layer thicknesses are typically discrete variables at the machine level and only a few are available. Also, decreasing layer thickness increases the build time by increasing the number of layers to be fabricated. On the other hand, part orientation allows much more freedom than layer thickness in selecting an appropriate value[2, 20]. Figure 1.4 shows the relation between part orientation of flat surfaces and the stair stepping effect.



**Figure 1.4 Effect of part orientation on stair stepping effect**

In Figure 1.4, (a) represents a flat surface facing upward. This orientation generates minimum error because there is no stair stepping effect. In (b), the error introduced by fabricating a vertical surface is presented. The error is the space between the curved lines and the straight line. Such error is introduced by various process related facts such as the cured profile of the resin and the angle between the laser beam and the resin surface. In (c), the error introduced by fabricating a slanted flat surface is

presented. Among the three cases (a, b, and c), this introduces the biggest error. In other words the stair stepping effect is maximized.

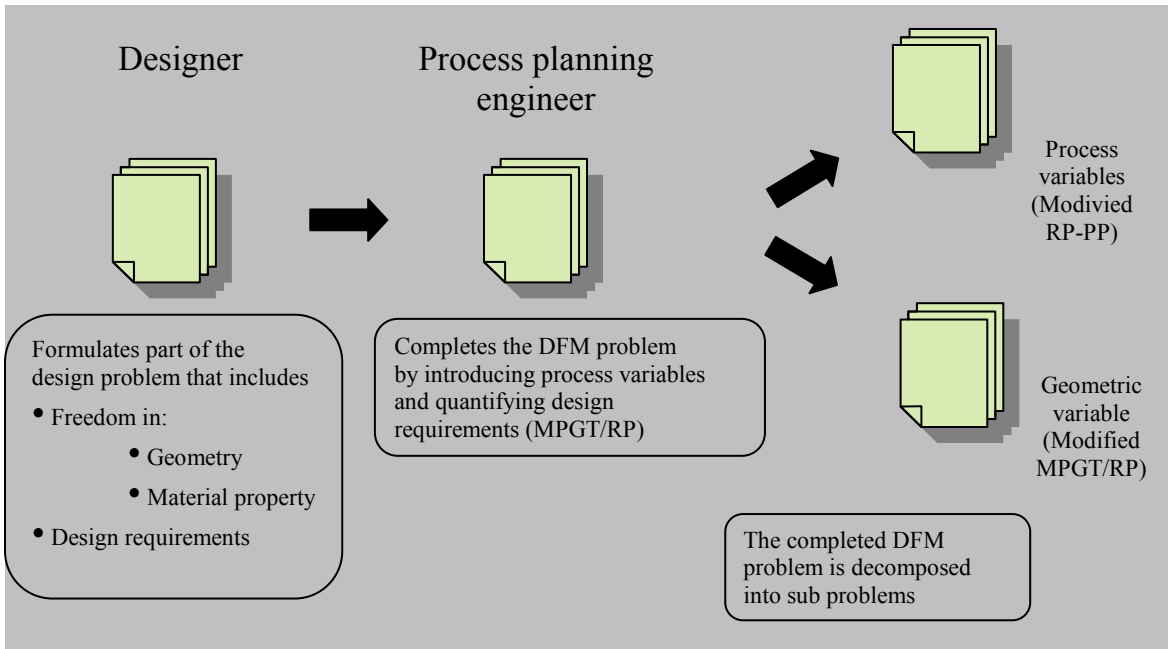
### **1.1.2 Geometric tailoring**

Geometric tailoring is defined as[5]:

*“The modification of some non-critical geometric features of a part to lower fabrication cost and time, and to produce functional prototypes that mimic the behavior of the production parts.”*

Hence, the key idea of the method is to produce a functional prototype that simulates a production part while having different material properties than the production part. The method is developed for manufacturing prototypes using rapid prototyping (RP) technology.

A graphical overview of geometric tailoring is shown in Figure 1.5. In this method, a designer sends design information such as design requirements and geometry, with some geometric freedom, to the process planning engineer. Then, the process planning engineer determines solid geometry and process variables such that the design requirements from the designer are satisfied.



**Figure 1.5 Graphical illustration of geometric tailoring**

In geometric tailoring, information transfer from the designer to the process planning engineer is accomplished by problem formulation. The problem formulation is in the form of a compromise Decision Support Problem (cDSP). A cDSP is a hybrid multi-objective problem formulation, and its objective is to explore the design space and improve selected concepts based on a set of goals, constraints, and bounds. It is mainly used to model decisions consisting of multiple goals that are often in conflict with one another. The structure of a cDSP is shown below[21]:

- *Given: A feasible alternative, assumptions, parameter values, and goals*
- *Find: Values of design and deviation variables.*
- *Satisfy: System constraints, system goals, and bounds on variables.*
- *Minimize: Deviation function that measures distance between goal targets and design point*

As shown in Figure 1.5, the procedure of geometric tailoring can be described as a series of problem formulation and solution generation steps. The partly formulated problem from the designer is completed by the process planning engineer. This is called a material process geometric tailoring/rapid prototyping (MPGT/RP) problem[5]. Then, the completed problem is decomposed into two problems; one for determining process variables and the other for determining geometric variables. The problem for determining process variables is called a modified rapid prototyping process planning (Modified RP-PP) problem. The problem for determining geometric variables is called the modified material process geometric tailoring (Modified MPGT/RP) problem. As a group, such formulated problems are called the design for manufacture (DFM) problems in this research. The following paragraphs describe details of each problem formulation and the solution strategy.

The MPGT/RP problem consists of two sub-problems; one is GT and the other is RP-PP. The MPGT/RP problem is formulated by combining the two sub-problems. The objective of GT is to develop a functional prototype that simulates the production part by adjusting non critical geometry with different material property than the production part. The word formulation is shown in Table 1.1.

**Table 1.1 Word formulation of cDSP for geometric tailoring (GT)**

<b>GIVEN:</b>	
<ul style="list-style-type: none"> <li>• Parametric CAD model of part</li> <li>• Functional property models</li> <li>• Target values for functional properties</li> <li>• Target values for process goals</li> </ul>	<ul style="list-style-type: none"> <li>• RP material properties</li> <li>• Production material properties</li> <li>• Target values of geometry variables</li> <li>• Target values of cost and time</li> </ul>
<b>FIND:</b>	
<b>System Variables:</b> Geometry variables	<b>Deviation Variables:</b> Deviation of goals from targets
<b>SATISFY:</b>	
<b>Goals:</b> Meet target functional properties Meet targets of geometry variables	<b>Constraints:</b> Meet geometry and/or assembly constraints
<b>Bounds:</b> Bounds for all system variables	
<b>MINIMIZE:</b>	
<b>Deviation Function:</b> Function of goal deviations	

The information required in GT problem formulation includes a parametric CAD model of the part, RP and production material properties, functional property models, and target values of functional, geometry, process, cost and time goals. The functional property model represents the quantified models that relate system variables to functional properties. The target values represent the designer’s preferences. Although the target values for design variables are specified, their feasible ranges are also specified by the designer. The process, cost, and time goals in Table 1.1 are affected only by the manufacturing process. However, the targets of these goals are specified to ensure that the manufactured prototype meets its project requirements.

The system variables are the geometric variables that have non-significant effect on the prototype’s functionality. Then, these variables are decided within the prototype’s

bounds when the formulated problem is solved. Deviation variables represent the deviation from the target goal values. Bounds represent the limit of the geometric variables that are specified by the designer.

The corresponding math formulation of the GT problem is shown in Table 1.2.

**Table 1.2 Math formulation of cDSP for geometric tailoring (GT)**

<b>GIVEN:</b>	
<ul style="list-style-type: none"> <li>• <math>FP_k = f(G_j, M)</math> (1.1)</li> <li>• <math>G_{Tjn}, FP_{Tjn}, PG_{Tjn}, C_{Tn}, T_{Tn}</math></li> </ul>	<ul style="list-style-type: none"> <li>• <math>M_T</math> material properties (prototypes)</li> <li>• <math>M_p</math> material properties (production)</li> </ul>
<b>FIND:</b>	
<b>System Variables:</b>	<b>Deviation Variables:</b>
$G_j$	$d_{i,n}^+, d_{i,n}^-$
<b>SATISFY:</b>	
<b>Constraints:</b>	
$G_m(G_j, M) = 0$ (1.2)	
$d_{i,n}^+ * d_{i,n}^- = 0$ (1.3)	
$d_{i,n}^+, d_{i,n}^- \geq 0$ (1.4)	
<b>Goals:</b>	
$G_{j,min}, G_{j,max}, FP_{k,min}, FP_{k,max}$ (1.5)	
<b>Bounds:</b>	
$G_{lbj} \leq G_j \leq G_{ubj}$ (1.6)	
<b>MINIMIZE:</b>	
<p><b>Archimedean Formulation:</b> <math>Z = \sum_{n=1}^4 \sum_{i=1}^{2J+2K} w_{i,n} (d_{i,n}^+ + d_{i,n}^-)</math> (1.7)</p> <p>Where <math>G_j</math>: geometric variable, <math>FP</math>: functional property, <math>PG</math>: production goal, <math>J</math>: number of geometric variables, <math>K</math>: number of functional properties, <math>n</math>: number of goal, <math>i</math>: number of variables</p>	

In Table 1.2, Young’s modulus and the tensile strength are considered to be the mechanical properties of interest for both the target and prototype materials. Response surface methodology and analytical modeling techniques are used to develop quantitative models relating the functional properties to geometric variables (Eq. 1.1). For example, the relation between a part’s stress and its geometry can be modeled using response surfaces.

The constraints are represented in equations (1.2~1.4). Equation (1.2) represents geometric constraints and equations (1.3 and 1.4) represent deviation constraints on variables. Equation (1.5) represents goals for geometry and functional properties. The bounds of the geometry and the objective function formulation are presented in equation (1.6) and (1.7) respectively. Equations 1.2~1.7 are generic and should be replaced with problem specific equations as the problem is formulated in a specific domain.

The objective of RP-PP problems is to fabricate a functional prototype such that it can satisfy the process, cost, and time goals. Its word formulation is shown in Table 1.3.

**Table 1.3 Word formulation of rapid prototyping process planning (RP-PP)**

<b>GIVEN:</b>	
<ul style="list-style-type: none"> <li>• Parametric CAD model of part</li> <li>• Target values for process goals</li> <li>• Production material properties</li> </ul>	<ul style="list-style-type: none"> <li>• RP material properties</li> <li>• RP process models</li> <li>• Target values of cost and time</li> </ul>
<b>FIND:</b>	
<b>System Variables:</b> RP process variables	<b>Deviation Variables:</b> Deviation of goals from targets
<b>SATISFY:</b>	
<b>Goals:</b> Meet target material property Meet targets of process goals Meet target cost and time	<b>Constraints:</b> Meet RP process constraints  <b>Bounds:</b> Bounds for all system variables
<b>MINIMIZE:</b>	
<b>Deviation Function:</b> Function of goal deviations	

The information required to formulate RP-PP includes the CAD model of the part, the production material properties, the RP material property models, the RP process models and the target values of process, cost, and time goals. In this problem, the production material properties, process, cost, and time goals are the targets to be accomplished. These values are specified in the GT problem formulation. Quantified

models for RP material property and process models are developed and used to relate the goals to system variables.

Mathematical formulation of RP-PP is shown in Table 1.4.

**Table 1.4 Math formulation of rapid prototyping process planning (RP-PP)**

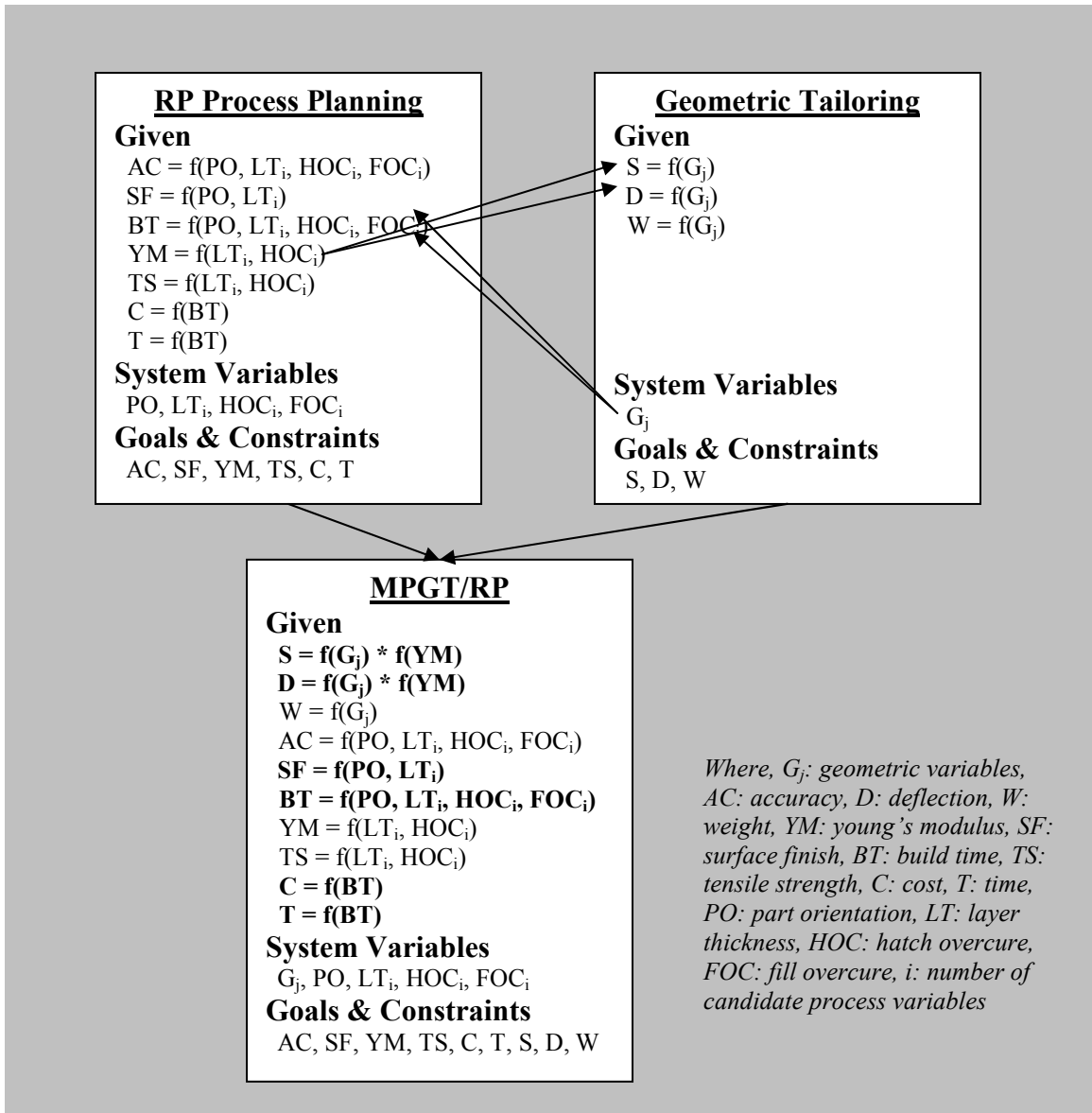
<b>GIVEN:</b>	
<ul style="list-style-type: none"> <li>• CAD model of part, <math>N_p</math></li> <li>• <math>AC_{Ton}, SF_{Tqn}, C_{Tn}, T_{Tn}, YM_{Tn}, TS_{Tn}</math></li> <li>• <math>AC_o = f(PO, LT, HOC, FOC, ZL, SP)</math></li> <li>• <math>BT = f(PO, LT, HOC, FOC, ZL, SP)</math></li> <li>• <math>C = f(BT)</math></li> </ul>	<ul style="list-style-type: none"> <li>• <math>ZL, SP = f(LT)</math></li> <li>• <math>SF_q = f(PO, LT)</math></li> <li>• <math>YM_p = f(LT, HOC)</math></li> <li>• <math>TS_p = f(LT, HOC)</math></li> <li>• <math>T = f(BT)</math></li> </ul>
<b>FIND:</b>	
<b>System Variables:</b>	<b>Deviation Variables:</b>
$PO (\alpha_x, \alpha_y, \alpha_z), LT_r, HOC_r, FOC_r$	$d^+_{i,n}, d^-_{i,n}$
<b>SATISFY:</b>	
<b>Constraints:</b>	
$x_p \leq x_b; y_p \leq y_b; z_p \leq z_b, LT_r \in [2, 4, 6, 8], d^+_{i,n} * d^-_{i,n} = 0, d^+_{i,n}, d^-_{i,n} > 0$	
<b>Goals:</b>	
$YM_{min}, YM_{max}, TS_{min}, TS_{max}, C, C, T, T, AC, SF$	
<b>Bounds:</b>	
$0^\circ \leq \alpha_x, \alpha_y, \alpha_z \leq 90^\circ, 2 \leq LT_r \leq 8, HOC_{lb} \leq HOC_r \leq HOC_{ub}, FOC_{lb} \leq FOC_r \leq FOC_{ub}$	
<b>MINIMIZE:</b>	
<p><b>Archimedean Formulation:</b> <math>Z = \sum_{n=1}^4 \sum_{i=1}^{2J+2K} w_{i,n} (d^+_{i,n} + d^-_{i,n})</math></p> <p>Where AC: accuracy, SF: surface finish, C: cost, T: time, YM: Young's modulus, TS: tensil strength, BT: build time, PO: part orientation, LT: layer thickness, HOC: hatch overcure, FOC: fill overcure, n: number of goals, i: number of variables,</p>	

The system variables are RP process variables that include part orientation (PO), slicing scheme, hatch overcure and fill overcure. Part orientation is the orientation (x, y and z orientation) in which the part is fabricated. The slicing scheme is the list of layer thicknesses ( $LT_r$ ) along the vertical axis that is used to fabricate parts. Hatch overcure

(HOC) and fill overcure (FOC) are the scanning variables that are specific to Stereolithography. The problem formulations in Tables 1.3 and 1.4 are generic. In case the fabrication process changes, the process variables need to be changed accordingly in the problem formulation (RP-PP). Similarly, the constraints also need to be defined for specific fabrication process and design requirements.

The goals in the RP-PP problem can be classified into three categories: meeting target material properties, meeting the process goals, and meeting target cost and time. Young's modulus and tensile strength fall in the material properties goal. Surface finish and tolerance requirements fall in the process goals. Finally, cost and time goals are for the whole RP process.

The MPGT/RP problem is then constructed by integrating GT and RP-PP with coupling considerations between them. A graphical illustration is shown in Figure 1.6.



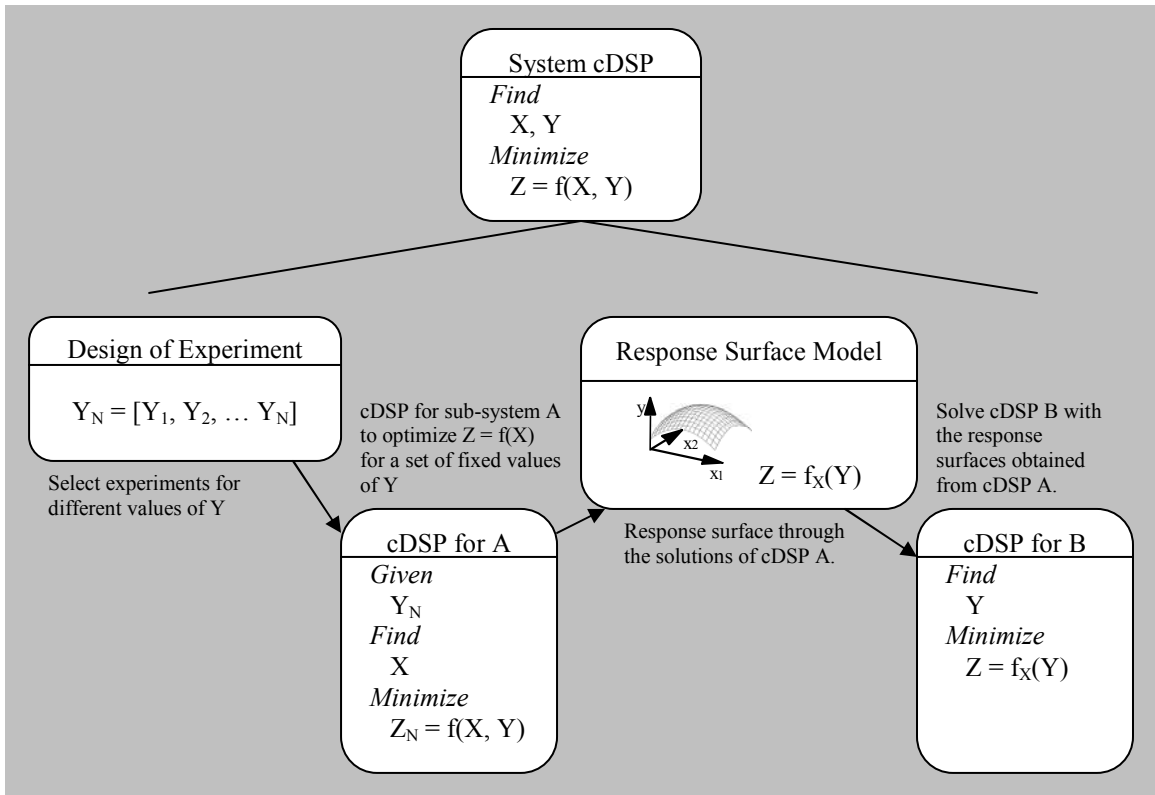
**Figure 1.6 MPGT/RP formulation from RP-PP and GT**

The objective of MPGT/RP is to achieve a desired functional and material property, part quality, cost and time by adjusting some of the non-critical dimensions of the part and by appropriate process planning of the RP machine. Its word and math formulation can be accomplished by simply integrating RP-PP and GT.

The difficulty in solving MPGT/RP is in its complex coupling conditions. Coupling conditions arise due to the coupling of cost, surface finish, stress and

displacement goals with system variables of the other sub-problems. For example, varying the geometry could influence build time (BT) due to changes in the height or width of the piece. Also, surface finish could be influenced by geometric change if the change influenced orientation of surface where surface finish is specified. Such examples are shown in Figure 1.6 with arrows. Stress and displacement are described as functions of geometric variables in GT. However, they are also functions of process variables in MPGT/RP because they are also functions of Young's modulus and Young's modulus is a function of process variables (system variables of RP-PP). Therefore, stress and displacement are also coupled goals. Such complex coupling conditions represent the worst case scenario of coupling between RP-PP and GT. The following paragraphs describe the solution strategy for solving a MPGT/RP problem.

The solution strategy is developed to solve a system-level cDSP that consists of two coupled cDSP's, 'cDSP for A' and 'cDSP for B'. Figure 1.7 shows a graphical illustration of this solution strategy.



**Figure 1.7 Solution strategy to solve ‘System cDSP’**

As shown in Figure 1.7, the solution for a system-level cDSP is obtained by sequentially solving ‘cDSP for A’ and ‘cDSP for B’. Design of experiments is used to generate a set of  $Y$  values which are used to solve ‘cDSP for A’ for each  $Y$  value. Then, response surfaces for  $f_X(Y)$  are developed for the data points  $(Y_i, Z_i)$  and they are used as the objective for solving ‘cDSP for B’. The solution for ‘cDSP for B’ is the solution for the system-level cDSP because solving for  $f_X(Y)$  is equivalent to solving for  $f(X, Y)$ .

This solution strategy is applicable to solving a MPGT/RP problem. First, two sub-problems are obtained after decomposing the MPGT/RP problem. They are modified RP-PP and modified MPGT/RP which correspond to ‘cDSP for A’ and ‘cDSP for B’ in Figure 1.7 respectively. Figure 1.8 shows the two problems (modified RP-PP and MPGT/RP).

<u>Modified RP-PP</u>	<u>Modified MPGT/RP</u>
<p><b>Given</b>  <math>AC = f(PO, LT_i, HOC_i, FOC_i)</math>  <math>SF = f(PO, LT_i)</math>  <math>BT = f(PO, LT_i, HOC_i, FOC_i)</math>  <math>YM = f(LT_i, HOC_i)</math>  <math>TS = f(LT_i, HOC_i)</math>  <math>C = f(BT) \quad S = f(YM)</math>  <math>T = f(BT) \quad D = f(YM)</math>  <math>G_{jN}</math></p> <p><b>System Variables</b>  <math>PO, LT_i, HOC_i, FOC_i</math></p> <p><b>Goals &amp; Constraints</b>  <math>AC, SF, YM, TS, C, T, S, D</math></p> <p><b>Minimize</b>  <math>Z_N = g(PO, LT_i, HOC_i, FOC_i, G_{jN})</math></p>	<p><b>Given</b>  <math>S = f_{RP-PV}(G_j)</math>  <math>D = f_{RP-PV}(G_j)</math>  <math>W = f_{RP-PV}(G_j)</math>  <math>AC = f_{RP-PV}(G_j)</math>  <math>SF = f_{RP-PV}(G_j)</math>  <math>YM = f_{RP-PV}(G_j)</math>  <math>TS = f_{RP-PV}(G_j)</math>  <math>C = f_{RP-PV}(G_j)</math>  <math>T = f_{RP-PV}(G_j)</math></p> <p><b>System Variables</b>  <math>G_{jN}</math></p> <p><b>Goals &amp; Constraints</b>  <math>AC, SF, YM, TS, C, T, S, D, W</math></p> <p><b>Minimize</b>  <math>Z_N = g_{RP-PV}(G_j)</math></p>

**Figure 1.8 Modified RP-PP and modified MPGT/RP Problem formulation**

The MPGT/RP problem is decomposed by decoupling stress, displacement, build time, and surface finish goals; this is accomplished by separating geometric variables and process variables. Equations (1.8)-(1.10) show the analogy between the RP-PP problem and ‘cDSP for A’.

$$X = \{PO, LT_i, HOC_i, FOC_i\} \text{ (RP process variables)} \quad (1.8)$$

$$Y_N = G_{jN} \text{ (Part geometry variable)} \quad (1.9)$$

$$F(X, Y) = g(PO, LT_i, HOC_i, FOC_i, G_{jN}) \quad (1.10)$$

As shown by equation 1.9, design of experiment is used to generate a set of geometric values. These values are then used to produce a number of part geometries. For each geometry, the modified RP-PP problem is solved to find the corresponding

process variables. The objective function for the modified RP-PP problem is the weighted sum of the goal (AC, SF, BT, YM, TS, C, T, S, D) deviations.

Equations (1.11) and (1.12) show the analogy between ‘cDSP for B’ and modified MPGT/RP. The objective function for modified MPGT/RP problem is the weighted sum of the goal (AC, SF, YM, TS, C, T, S, D) deviations.

$$Y = G_i \quad (1.11)$$

$$f_x(Y) = g_{PP-PV}(G_i) \quad (1.12)$$

In the solution strategy, modified RP-PP is the first problem and modified MPGT/RP is the second problem to be solved. The models ( $f_{RP-PV}(G_i)$ ) in the modified MPGT/RP problems in Figure 1.8 represent the response surface equations developed by the solutions from the modified RP-PP problem. By solving two problems (modified RP-PP and modified MPGT/RP), a solution for MPGT/RP is obtained.

As a conclusion, geometric tailoring provides a systematic way of designing and manufacturing a functional prototype in a distributed environment. There are two advantages identified. One is the realization of simultaneous design of product and process and the other is separation of design and manufacturing. However, there are still research issues that need to be addressed for the geometric tailoring method to be more practical in supporting distributed design and manufacturing. The following paragraphs describe the issues and objectives of this research.

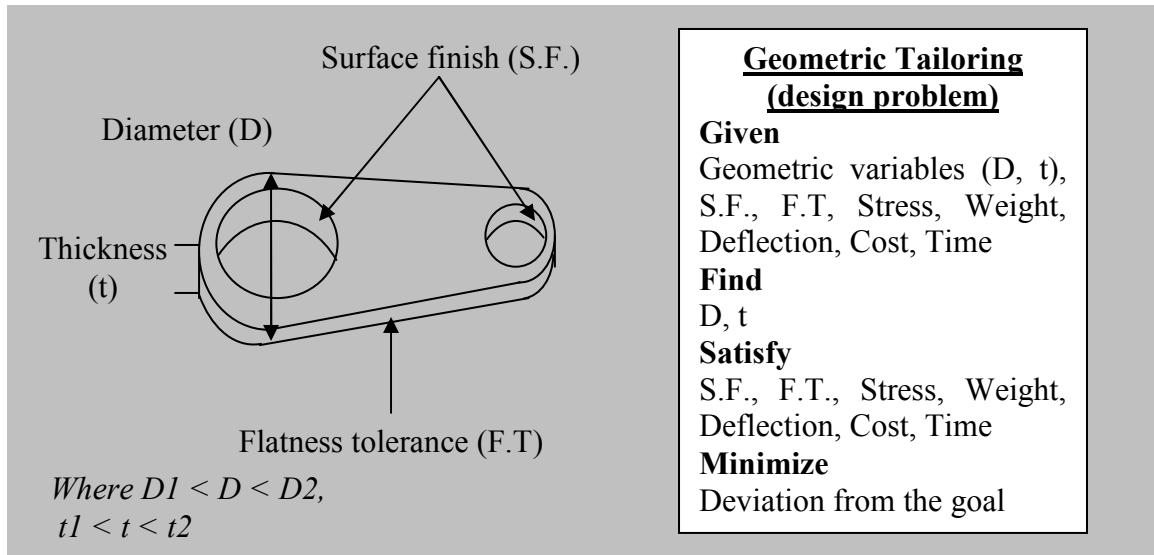
### **1.1.3 Challenges and research objective**

The motivation of this research is to support redesign and fabrication of an existing product while using new technology and material and while working in a distributed environment. This objective can mostly be satisfied by using geometric tailoring. Geometric tailoring provides a systematic procedure for fabricating a functional prototype with different material properties than production part. The method modifies non-critical geometric dimensions of the prototype to retain important mechanical characteristics of a production part. However, there are a couple of challenges that need to be addressed in order for geometric tailoring to be practical to use in a distributed environment. The challenges are formulating and solving DFM problems (MPGT/RP) which require in depth knowledge in process planning.

To illustrate specific challenges in geometric tailoring, robot arm example is used to demonstrate procedures of formulating and solving DFM problem first. Then, the identified challenges are presented.

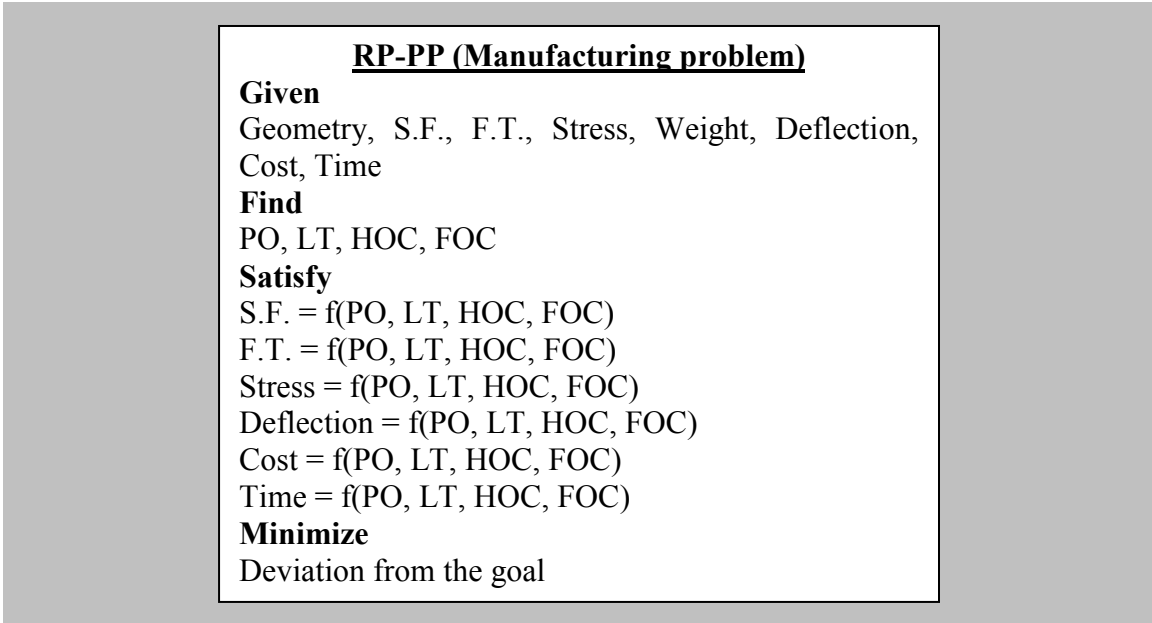
#### **1.1.3.1 Robot Arm example**

Figure 1.9 shows the geometry and design problem (GT) for the robot arm example[4].



**Figure 1.9 Robot arm geometry and design problem (GT)**

The information in Figure 1.9 is specified by the designer. The designer provides some freedom in geometry by assigning allowable ranges for outer diameter (D) and thickness (t) of the robot arm. Then, designer further specifies design requirements including surface finish, flatness tolerance, stress, weight, deflection, cost and time. The objective is to determine values for the given geometric variables and produce the part such that it satisfies the given design requirements. The information in Figure 1.9 is then transferred to the manufacturing site where the process planning engineer receives it. The process planning engineer formulates the manufacturing problem (RP-PP) for the given design problem. The objective of the manufacturing problem is to systematically identify the feasible space of process variables that influence the design requirements in the design problem. Figure 1.10 shows the manufacturing problem that process planning engineers formulate for the design problem in Figure 1.9.

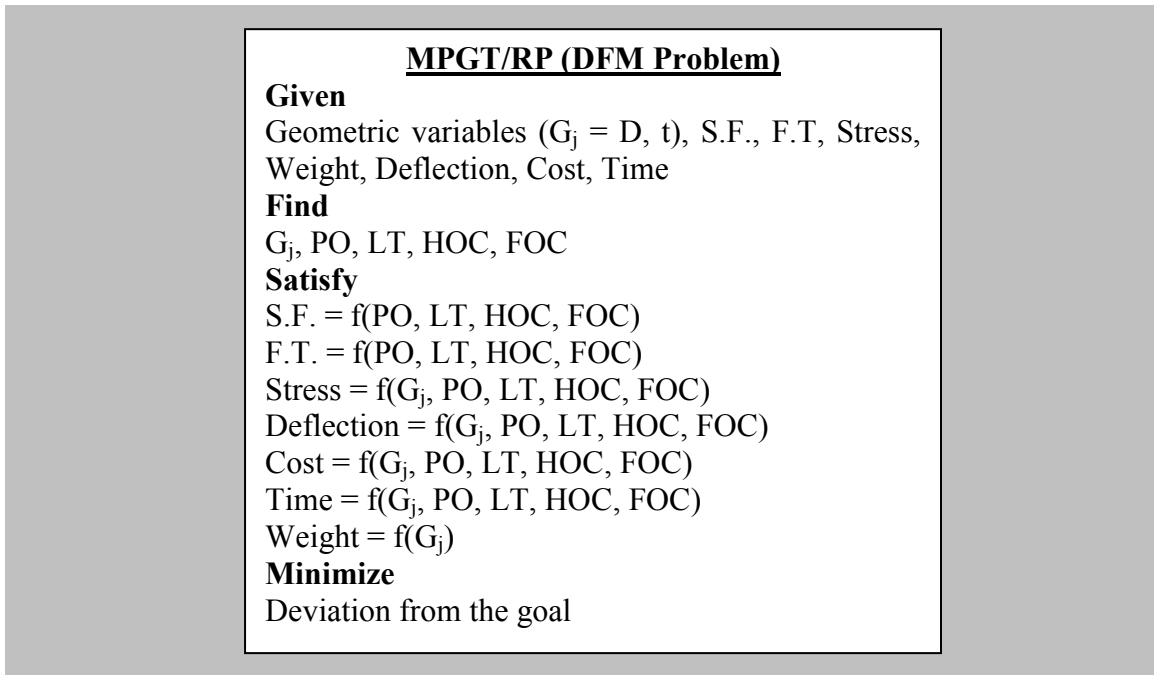


**Figure 1.10 Robot Arm manufacturing problem (RP-PP)**

The feasible space of process variables is identified by determining the process variables and their space that influence the quality of design requirements. In Figure 1.10, the manufacturing problem is formulated for the SLA process. Selected process variables are PO, LT, HOC, and FOC. Then, all the design requirements except weight are identified as being influenced by process planning (functions of all process variables). Accuracy measurements (surface finish and flatness tolerance) are functions of process variables due to the stair stepping effect discussed in the additive manufacturing section in this chapter. Stress and deflections are functions of process variables because the modulus of elasticity is a function of the process variables. Cost and time are functions of process variables because part build time is a function of process variables.

By assuming the limited knowledge of process planning engineer, the feasible spaces of the process variables are all assumed to be infinite in Figure 1.10. Once the manufacturing problem is formed, the next task is formulating the DFM problem

(MPGT/RP) by combining the design problem and the manufacturing problem. Figure 1.11 shows the DFM problem formulation.



**Figure 1.11 Robot Arm DFM problem (MPGT/RP)**

The DFM problem is formulated by a couple of steps. The first step is identification of design requirement dependencies on geometric variables and process variables by combining design problem and manufacturing problem. The second step is developing models of design requirements for solution generation process. The design requirements models are usually empirical models that are used for evaluation of design requirement values during the solution generation process. In Figure 1.11, design requirements including stress, deflection, cost, and time are identified as geometric variables and process variables dependent requirements. Accuracy requirements such as S.F. and F.T. are identified as process-variables-dependent requirements. Also, the weight requirement is determined to be dependent only on geometric variables.

Then, the formulated DFM problem is decomposed into two sub problems based on the solution strategy in Figure 1.7. One is a modified manufacturing problem (modified RP-PP) and the other is a modified DFM (modified MPGT/RP) problem. Figures 1.12 and 1.13 show the modified manufacturing problem and the modified DFM problem respectively.

**Modified RP-PP (Manufacturing problem)**

**Given**  
Geometries ( $G'$ ), S.F., F.T., Stress, Weight, Deflection, Cost, Time

**Find**  
PO, LT, HOC, FOC

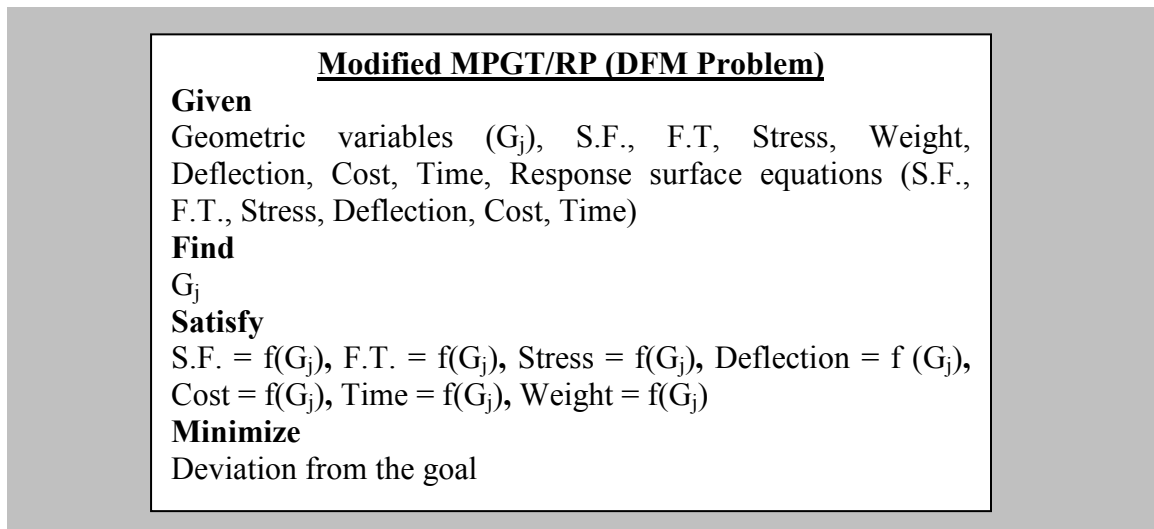
**Satisfy**  
 $S.F. = f(PO, LT, HOC, FOC)$   
 $F.T. = f(PO, LT, HOC, FOC)$   
 $Stress = f(PO, LT, HOC, FOC)$   
 $Deflection = f(PO, LT, HOC, FOC)$   
 $Cost = f(PO, LT, HOC, FOC)$   
 $Time = f(PO, LT, HOC, FOC)$

**Minimize**  
Deviation from the goal

**Figure 1.12 Robot Arm manufacturing problem after decomposition**

In the modified manufacturing problem, process variables are determined for each sample of geometry ( $G'$ ). Therefore, the system variables are only process variables. The samples of geometry are developed by performing design of experiment. More specifically, samples of values for  $D$  and  $t$  are determined, then the geometry of the robot arm is created for each combination of sampled values of  $D$  and  $t$ . Once the process variables for each geometry are determined, corresponding values of design requirements are also computed. Then using the sampled geometric values ( $D, t$ ), response surface equations are developed for each design requirement. Finally, the modified DFM

problem (modified MPGT/RP) is formulated using response surface equations as shown in Figure 1.13.



**Figure 1.13 Robot Arm DFM problem after decomposition**

In Figure 1.13, the design requirements are modeled as functions of geometric variables only as expected. Through solving this modified DFM problem, a solid geometry of the robot arm and the corresponding process variables are determined.

#### 1.1.3.2 Challenges

The challenge identified during the robot arm example is the need for process planning knowledge that can be used to properly formulate and solve a DFM problem. Proper process planning requires in depth knowledge in both design and manufacturing. Hence, the objective of this research is to retrieve previously formulated and solved DFM problems to support process planning. The following paragraphs discuss the details.

Two specific challenges are identified in formulating and solving a DFM problem: development of design requirement models in DFM problem formulation and generating solutions to DFM problem. In modeling design requirements, the process

variables that influence design requirements are identified from the manufacturing problem. Then, empirical models of the design requirements are developed in terms of geometric variables and identified process variables. Lynn developed empirical models for various tolerances in terms of Stereolithography process variables[22, 23]. West developed an empirical model for surface finish on a flat surface in terms of Stereolithography process variables[20]. Table 1.5 shows an example of such empirical models.

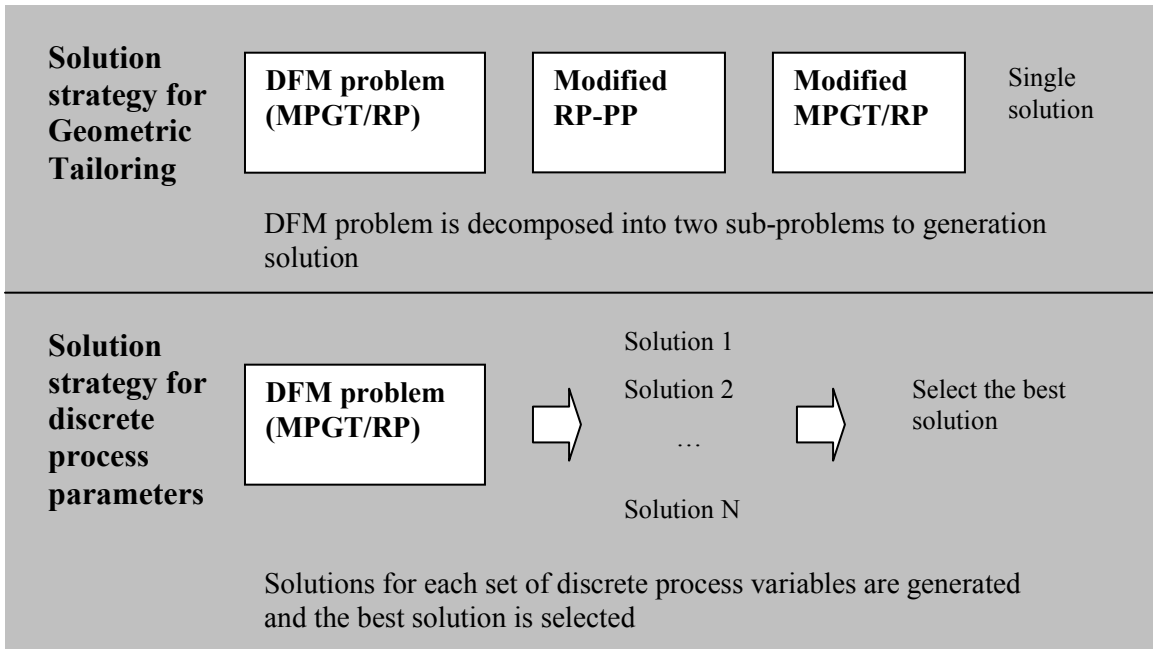
**Table 1.5 Examples of empirical models**

Design requirements	Empirical models
Concentricity of the conical surface	$0.0002 * ZL + -1.8094 * HO + 1.0861 * FO + (-0.0028) * SP + 3.8040 * HO * HO + (-49.9638) * FO * FO + 0.0002 * SP * SP + 0.0633 * ZL * HO + 0.2287 * HO * SP + 0.0081$
Parallelism of planar surfaces	$(-0.0010) * ZL + -3.4506 * HO + 0.9557 * FO + (-0.0011) * SP + 299.9033 * HO * HO + 0.0001 * ZL * SP + 263.9449 * HO * FO + 0.1047 * HO * SP + 0.0061$
Perpendicularity of planar surfaces	$(-0.00149) * ZL + -2.31720 * HO + 0.55254 * FO + (-0.00125) * SP + 0.00005 * ZL * ZL + 0.00009 * SP * SP + 185.12015 * HO * FO + 0.01352$
Where ZL = Z level wait time(sec), HO = Hatch overcure (in), FO = Fill overcure (in), SP = Sweep period (sec)	

In Table 1.5, examples of empirical models for the design requirements including concentricity on conical surface, parallelism tolerance, and perpendicularity tolerance on planar surfaces are presented. Those models are developed in terms of process variables only. Hence, the models in Table 1.5 are generic and independent of specific design problems. Those models can be reused in various design problems by augmenting them to accommodate specific geometry[4]. Details are presented in chapter 7. Other design requirement models such as stress and deflection are specific to design problems. Hence, those design requirements models are typically not reusable.

Developing the models in Table 1.5 requires procedures such as setting up the experiment, data collection, empirical model generation using response surface methodology, and validation of the models. Therefore, developing design requirement models is a time and knowledge intensive task.

In the robot arm example, the solution for the formulated DFM problem is produced by decomposition of the problem with the assumption that the process variables are completely unknown (infinite feasible space). However, the decomposition can be avoided and the formulated DFM problem can be directly solved if the feasible space of the process variables is known to be discrete. In a case where the process variables are all discrete, the solution for each discrete process variable value can be generated and the best one can be selected. If any one of the process variables is continuous, decomposition is utilized. In this research, these two ways of generating solutions are called solution strategies. Figure 1.14 shows the two solution strategies.



**Figure 1.14 Two solution strategies**

As shown in Figure 1.14, the solution strategy becomes simpler as the feasible space of process variables decreases. In this research, the characteristics of process variables refer to the condition of each process variable as being continuous or discrete. Correspondingly, there are two solution strategies: for continuous and one for discrete process variables. The difficulty is to identify an appropriate solution strategy for the formulated DFM problem. More specifically, the difficulty is in identifying the feasible space of process variables. In this research, the feasible space of a process variable means the identified process variables and their feasible space for satisfying design requirements. Table 1.6 shows examples of the feasible spaces of process variables.

**Table 1.6 Example of feasible spaces of part orientation and layer thickness**

Process variables	Feasible spaces
Part orientation	Infinite, 4, 3, 2, and 1
Layer thickness	2, 4, and 8 mils

In table 1.6, examples of feasible spaces for part orientation and layer thickness are presented. The feasible space of part orientation can be one of infinite, 4, 3, 2, and 1. The feasible spaces of layer thickness can be one of 2, 4, and 8 mils or combinations of them. Hence, the feasible spaces of process variables can be either infinite or discrete. The feasible space of process variables is usually determined during the manufacturing problem formulation where process variables that are influential to design requirements are identified. With the process planner's in depth knowledge in design and manufacturing, appropriate feasible spaces of process variables are identified accurately, which enables appropriate solution strategy selection. Therefore, the challenge in selecting an appropriate solution strategy is the need to have in depth knowledge about design and manufacturing.

In short, formulating and solving a DFM problem requires in depth knowledge of both design and manufacturing, which has been the major issue in DFM research. Consequently, the challenge in applying the geometric tailoring method to the redesign and fabrication of an existing product while using new technology and material in distributed environment becomes the challenge of formulating and solving DFM problems. More specifically, the difficulties in formulating and solving a DFM problem are modeling design requirements and identifying an appropriate solution strategy for solving the DFM problem. To model the design requirements properly, correct identification of the feasible space of process variables that influence the design requirements is crucial. Determining an appropriate solution strategy means identification of the correct feasible space of process variables. Therefore, properly formulating and solving DFM problems requires accurate identification of the feasible space of process variables. Determining the feasible space of process variables is basically partially performing process planning. This partial process planning is performed in manufacturing problem formulation (Figures 1.10) in geometric tailoring. During the manufacturing problem formulation, the feasible space of process variables and their relation to design requirements are identified. Then, the completion of process planning and part design is done at solution generation of DFM problem.

The objective of this research is to support the designers and process planning engineers as they perform process planning by retrieving previously formulated and solved DFM problems in a distributed environment. More specifically, the retrieved DFM problems support process planning by providing design requirement models and feasible spaces of process variables. In this research, the retrieved DFM problem

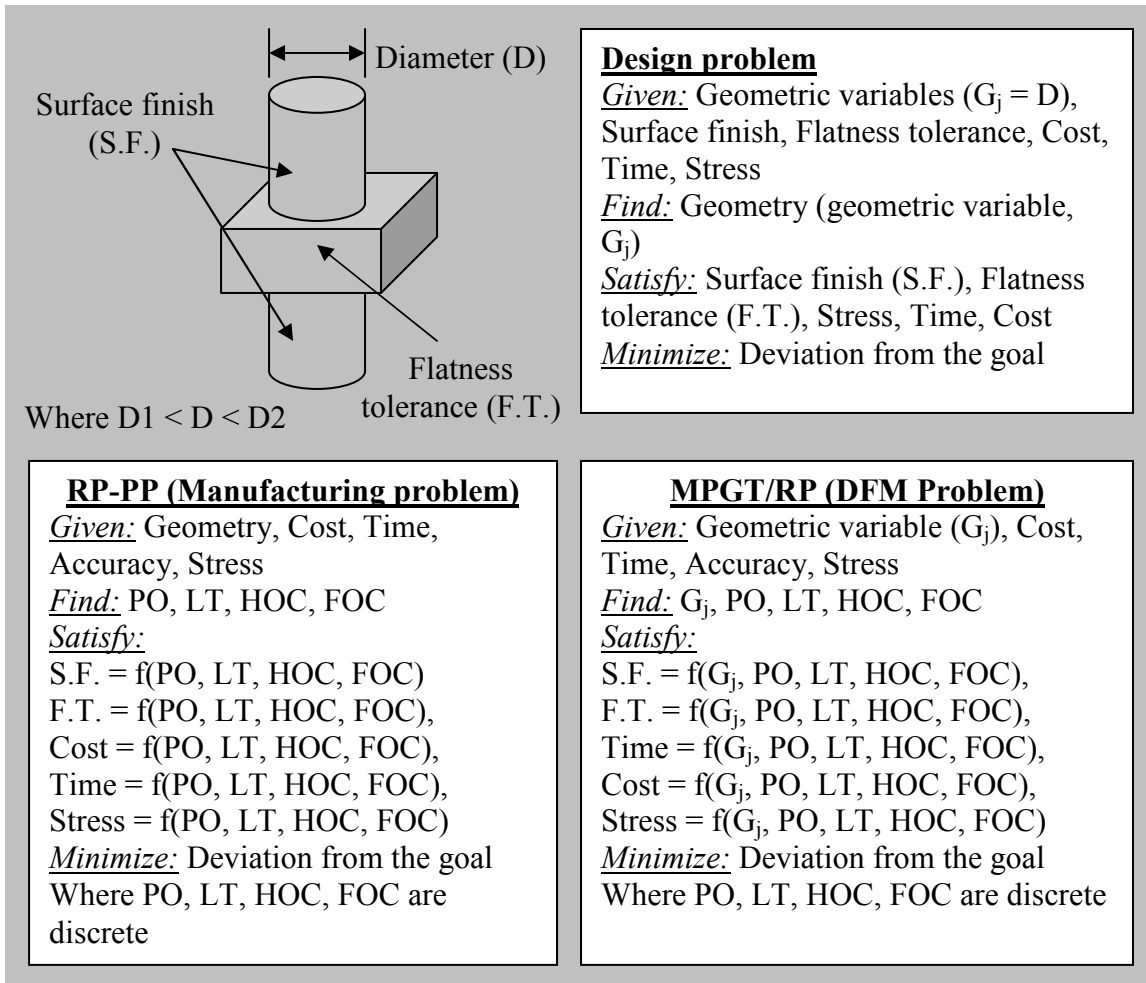
includes the complete history of DFM problem formulation and solution generation (design problem, manufacturing problem, and DFM problem). To properly support process planning, the retrieved DFM problems should show following items:

1. The design requirements from the design problem that influence process planning
2. Appropriate manufacturing problem for the given design problem
  - a. Feasible space of process variables and their relation to the design requirements
    - i. Appropriate solution strategy
    - ii. Models of design requirements that are needed to formulate a DFM problem
3. Formulated and solved DFM problem
  - a. Actual design requirements models developed in terms of feasible space of process variables
  - b. Demonstration of solution strategy utilization that is corresponding to the feasible space of process variables

By showing 1, the designer or process planning engineer understands the design requirements in the given design problem that influence process planning. By showing 2, correct partial process planning for the given design requirements is informed. By showing 3, design requirement models are delivered to support the DFM problem formulation and correct solution strategy utilization is demonstrated. Through 3, completion of process planning and part design is demonstrated. In this research, a DFM problem that provides above items is called a relevant DFM problem. The underlying

principle that enables the retrieval of relevant DFM problems is the fact that there are only finite ways to satisfy various design problems in terms of process planning.

To illustrate the benefit of retrieving previously developed DFM problems, an example case of previously developed DFM problem is introduced in Figure 1.15. In Figure 1.15, there are design, manufacturing, and DFM problem formulation. In this research, the retrieved DFM problem refers to the entire history of problems formulation and solution generation for a design problem. Hence, DFM problem include design, manufacturing, and DFM problem with solution for the DFM problem. The manufacturing problem in Figure 1.15 is assumed to be applicable to the design problem of the robot arm shown in Figure 1.9. Also, the robot arm example in Figure 1.9 is assumed to be the given design problem and the appropriate manufacturing and DFM problems are expected to be formulated. The following paragraphs describe details of the benefit.



**Figure 1.15 Sample retrieved DFM problem for robot arm example**

First, the manufacturing problem in Figure 1.15 provides the appropriate feasible spaces of process variable including part orientation, layer thickness, and overcure their relation to the given design requirements. Hence, the designer or process planning engineer do not have to have in depth knowledge to formulate an appropriate manufacturing problem. The feasible spaces of process variables are in the form of examples shown in Table 1.6.

Then, the retrieved DFM problems in Figure 1.15 shows the actual models of accuracy requirements including surface finish specified on cylindrical surfaces and a

flatness tolerance specified on a flat surface. Those models are in the form shown in Table 1.5. Hence, the designers or process engineers do not have to develop such models instead those models can be reused.

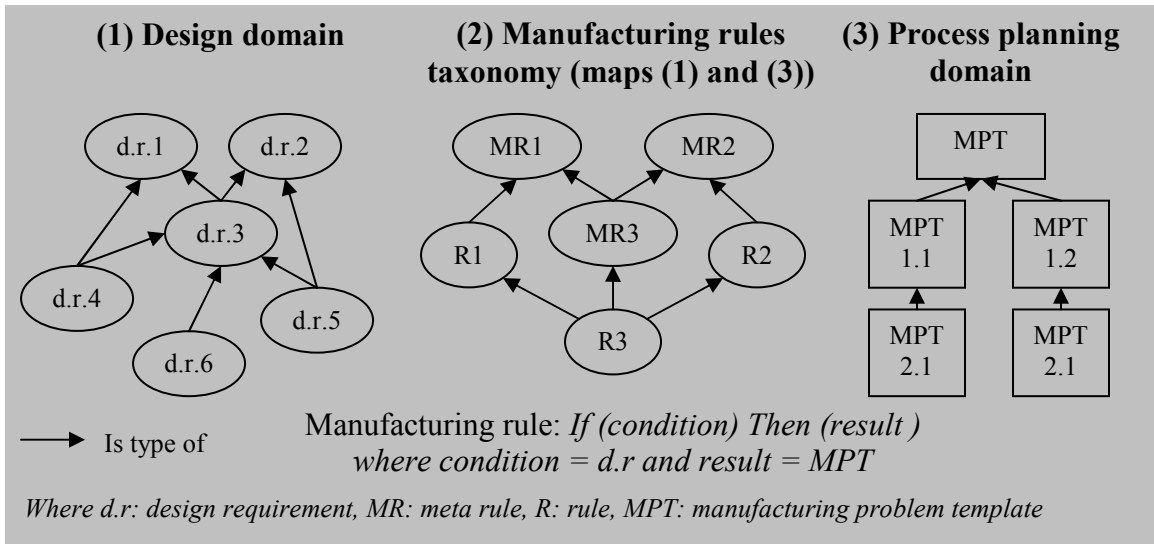
Further more, the solution for the DFM problem in Figure 1.15 shows how to search (solution strategy) the determined feasible spaces of process variables to determine exact values for the geometric and process variables. As shown in the manufacturing problem in Figure 1.15, the feasible spaces of process variables are discrete. Therefore, the corresponding solution strategy should be multi solution (Figure 1.14) and the decomposition should be avoided. An inexperienced designer or process planning engineer who could not formulate appropriate manufacturing problem could have decomposed the formulated DFM problem and introduced inaccuracies.

In short, relevant DFM problems can greatly enhance the formulation and solution generation of DFM problems through supporting process planning (providing feasible spaces of process variables and accuracy models). In depth example of DFM problem reuse is presented in chapter 7.

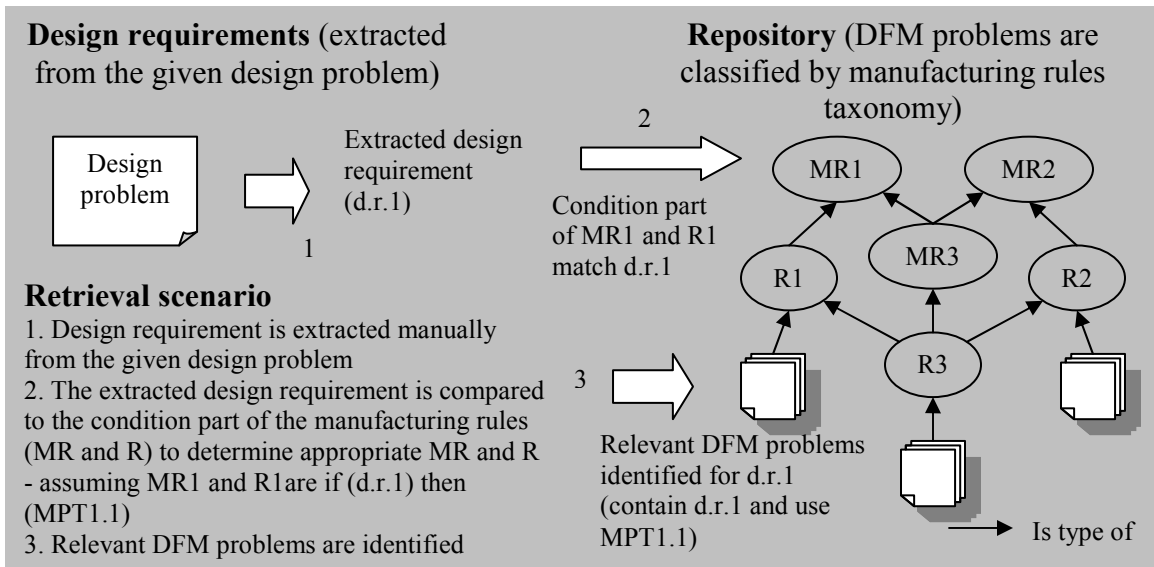
#### **1.1.4 Approach**

In this research, a retrieval method called DFM framework is proposed and developed. The key idea is to relate the given design problem to the relevant DFM problems through determining appropriate process plans. An example of such process plan is the manufacturing problem shown in Figure 1.15. This type of process plans is called the manufacturing problem template (MPT). During research, it is found that the manufacturing rules taxonomy is an entity that maps design problems to appropriate MPTs. Hence, the approach is to classify DFM problems by manufacturing rules such

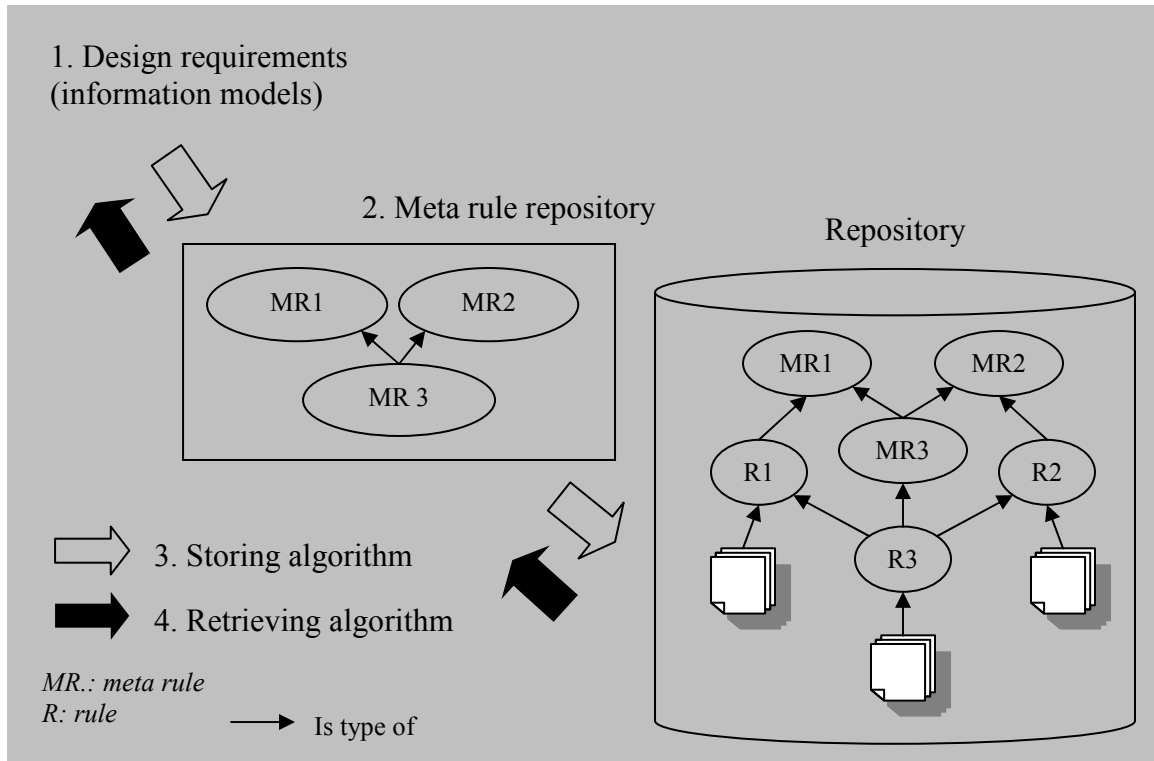
that the given design problem can be related to the relevant DFM problems through the manufacturing rules taxonomy. Figures 1.16, 1.17 and 1.18 illustrate the key ideas.



**Figure 1.16 The manufacturing rules taxonomy that map design and process planning domains**



**Figure 1.17 Example of retrieval scenario**



**Figure 1.18 Components of the retrieval method (DFM framework)**

The details of the key ideas illustrated using Figures 1.16, 1.17, and 1.18 are presented in chapters 3 and 4. The following paragraphs present brief discussions of the key concepts of the retrieval method.

As discussed before, the approach is to relate the given design problem to the relevant DFM problems through identifying appropriate process plans (MPTs) for the given design problem. To achieve this, the first task is to discover a consistent and correct mapping relation between design problems and process plans. Figure 1.16 illustrates this relation. In Figure 1.16, the manufacturing rules taxonomy is described as an entity that maps the design and process planning domains. The design requirements (d.rs) and MPTs constitute the design and process planning domains respectively. The MPTs determine feasible spaces of process variables. In this research, the design

requirements for the MPTs are computed by using the empirical models that determine exact values of process variables for the given design requirements. Furthermore, it is found that there are subsumption relations among the d.rs and MPTs as shown in Figure 1.16. Then, mapping between the d.rs and MPTs is accomplished by the manufacturing rules taxonomy. The structure of a manufacturing rule is shown in Figure 1.16. The condition part is d.r and the result part is MPT. Two kinds of manufacturing rules are identified: meta rules and rules. The difference is in the condition parts. Meta rules have more abstract condition parts than do rules. In this research, meta rules are collected up front and rules are derived using the collected meta rules. Due to the subsumption relation in d.rs and MPTs, there are subsumption relations in manufacturing rules. This subsumption relation in manufacturing rules allows computing a manufacturing rules taxonomy to be computed and the taxonomy then maps the design and process planning domains. Specifically, the subsumption relation in the design requirements maps the design requirements to the MPTs, and subsumption relations in the MPTs maps the MPTs to the design requirements. The details are presented in chapter 3.

To illustrate the way the mapping relation in Figure 1.16 is used in the retrieval method, an example is discussed in Figure 1.17. In Figure 1.17, the repository is structured by the manufacturing rules taxonomy. Specifically, the meta rules classify the rules and the rules classify the DFM problems. Then, the design requirement d.r.1 is manually extracted from the given design problem and compared to the condition part of the manufacturing rules in the repository. The manufacturing rules MR1 and R1 are identified to have condition parts that match d.r.1. In this example, the condition part of MR1 and R1 are assumed to be the same. Hence, DFM problems under R1 are identified

as relevant DFM problems for the given design problem. The DFM problems under R1 use MPT1.1 for process planning because their design requirements contain d.r.1. Then, the retrieved DFM problems provide feasible spaces of process variables (based on MPT1.1) and design requirements models in terms of the determined feasible spaces of process variables. Furthermore, the identified DFM problems demonstrate formulating and solving a DFM problem for the design requirements d.r.1. This is the basic idea of how the mapping relation in Figure 1.16 is used to retrieve relevant DFM problems. The details are discussed in chapter 4.

Figure 1.18 shows the components of the retrieval method (DFM framework). In Figure 1.18, the method consists of four components; 1. an information model for representing design requirements, 2. a meta rule taxonomy, 3. a storing algorithm and 4. a retrieval algorithm. The design requirements are extracted from the design problem manually. Then, the appropriate manufacturing rule for the design requirements is determined by first, representing the design requirement with information models and second, comparing them to the condition parts of the meta rules and deriving an appropriate rule. The determined manufacturing rules are then, used to navigate the manufacturing rule taxonomy in the repository to locate the relevant DFM problems. The meta rules are collected up front to dynamically derive rules and structure the repository. The details are presented in chapter 3.

In essence, the retrieval method maps design and process planning domains through manufacturing rules and their subsumption relations. Then, the design requirements are related to the relevant DFM problems through the established mapping. Description logic is selected as an appropriate mathematical theory for realizing the

established mapping. To ensure applicability of the retrieval method in design for additive manufacturing, the following research tasks need to be completed.

1. Identify and justify information space for design problems and process planning domains
2. Identify and justify the mathematical relation between design and process planning domains
3. Identify and justify the appropriate DL for realizing the mapping between design and process planning domain
4. Realize and justify the retrieval method based on 1, 2, and 3

By addressing research issue 1, the scope of the retrieval method can be identified. By addressing research issue 2, the mapping between design and process planning can be formalized by mathematical foundation. By addressing issue 3, the mapping between design and process planning can be realized by DL. By addressing issue 4, the retrieval method that is applicable in the design for additive manufacturing can be realized. The establishment of the mathematical relation and description logics utilization are discussed in more detail through research questions and hypotheses in the following section.

## **1.2 Research questions and hypotheses**

Four research questions and hypotheses are identified in this research as shown in Table 1.7. The questions Q1 and Q2 are established to address research issues 1~3 above. The questions Q3 and Q4 are established to address research issue 4 above. The following paragraphs describe the specific details of the research issues in each research question.

**Table 1.7 Research Questions and Hypotheses**

Research Questions		Hypotheses
Representation	Q1. How should design requirements be represented?	H1. Among the various expressive description logics, there is one description logic that provides minimum expressivity for representing design requirements
	Q2. How should the design and process planning domains be mapped?	H2. Subsumption in DL enables mathematical mapping between design domain and process planning domain
Retrieval	Q3. How should DFM problems be stored?	H3. Subsumption in DL enables systematic and consistent structuring of the repository
	Q4. How should DFM problems be retrieved and ranked?	H4. Subsumption in DL and the ranking metric enables retrieval and ranking of the relevant DFM problems

The first research question and hypothesis (Q1 and H1) are established to identify appropriate description logic for representing the design requirements. In this research, the design requirements are represented and reasoned with to relate the given design requirements to relevant DFM problems. There are several description logics with varying expressivity. As the expressivity in description logics increases, the corresponding computational complexity in its inference increases. Hence, the description logic that provides minimum expressivity is desired. To validate hypothesis H1, the information models for design requirements are developed and analyzed to determine the minimum expressivity for representing the design requirements. To achieve this, the information spaces for the design and process planning domains are identified and justified first. Then, the manufacturing rules are collected within the identified information space to discover design requirements. Finally, the discovered design requirements are carefully analyzed to realize information models that are used to determine the minimum expressivity. Identification of information space and discovery of the manufacturing rules are discussed in chapter 3 and 4 respectively. Then, selecting appropriate description logic is discussed in chapter 5.

The second research question and hypothesis (Q2 and H2) are established to realize systematic mapping of the design and process planning domains. As discussed previously, the design problems are related to the relevant DFM problems through appropriate process plans (MPTs). Therefore, correctly relating the design problems to the relevant DFM problems requires correctly relating design problems to appropriate MPTs. Establishing the correct relation between the design problems and MPTs requires correctly mapping the design and the process planning domains. To map the two domains correctly, accurate relationship between the two domains needs to be identified, and justified. As discussed previously, the manufacturing rules and their subsumption relations are used to map the two domains in this research. An in depth investigation regarding manufacturing rules is performed. The investigation includes discovery and justification of their structure, derivation, and mathematical properties. Through the investigation, the mathematical relation (subsumption relation) among the manufacturing rules is identified and justified. This investigation and justification serves as the theoretical validation of hypothesis 2. Then, the theoretical validity of hypothesis 2 is demonstrated by using the subsumption algorithm in DL. Specifically, the manufacturing rules taxonomy is computed using the subsumption in DL so that the subsumption in DL can be used to map the design and process planning domains. Details of investigations and validations are presented in chapter 3, 4, and 5.

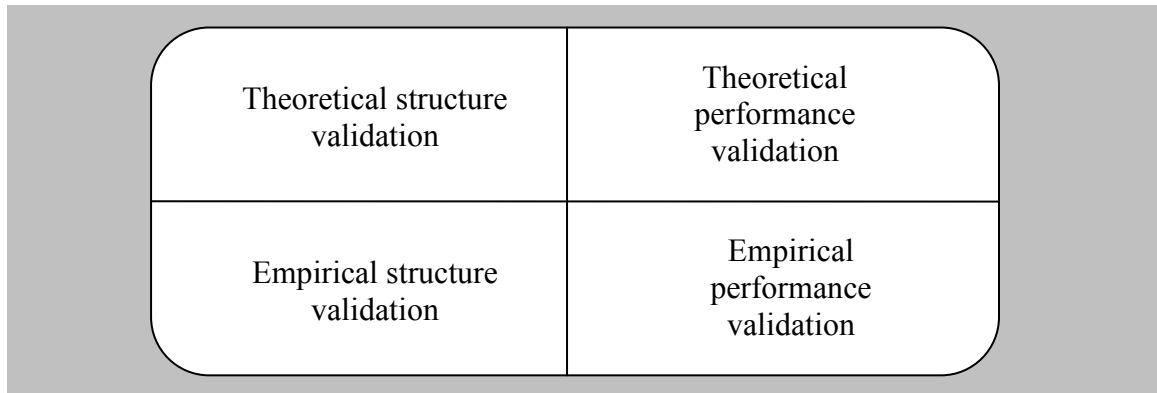
The third research issue is the systematic structuring of the repository. The DFM problem repository needs to be structured such that relevant DFM problems can be retrieved from the given design requirements. As discussed previously, the DFM problems are classified by manufacturing rules. Consequently, the repository needs to be

structured by the manufacturing rules taxonomy. The challenge is to compute the manufacturing rules taxonomy correctly and consistently. Hence, hypothesis H3 states that subsumption in DL enables systematic storing of DFM problems by computing manufacturing rules taxonomy correctly and consistently. The approach for validating this hypothesis is to develop an algorithm that structures the repository by computing manufacturing rules taxonomy using subsumption in DL. The theoretical validation of the corresponding hypothesis (H3) is achieved by determining the appropriate description logic for representing the design requirement so that subsumption can be used. The theoretical and empirical validation of this hypothesis is presented in chapters 6 and 7 respectively.

The fourth research issue is in retrieving and ranking the relevant DFM problems from the given design requirements. To achieve this, the relevant DFM problems need to be identified and ranked. Through realizing the mapping between design and process planning domains, the relevant DFM problems for the given design requirements can be identified. Then using the ranking metric, the identified DFM problems can be ranked. Hence, the hypothesis H4 states that the subsumption in DL and the ranking metric enable retrieval and ranking of the relevant DFM problems. To validate this hypothesis, an algorithm that uses subsumption and ranking metric is realized. The theoretical validation of corresponding hypothesis (H4) is achieved by proving the applicability of the description logics and the justification of the ranking metric. The theoretical and empirical validation of this hypothesis will be shown in chapters 6 and 7 respectively.

### 1.3 Validation

The validation square is proposed for methodology validation and shown in Figure 1.19[24].



**Figure 1.19 Validation square**

As shown in Figure 1.19, the validation square consists of four components: theoretical structure, empirical structure, empirical performance, and theoretical performance. In theoretical structure validation, the internal consistency of the retrieval method is discussed by theoretically validating the hypotheses in Table 1.7. In empirical structure validation, the justification of the selected test case is discussed. In empirical performance validation, the usefulness of the retrieval method is empirically demonstrated. In the theoretical performance validation, the usefulness of the method is claimed beyond the domain that the retrieval method is developed for. The following sections discuss utilization of each component in this research.

#### 1.3.1 Theoretical structure validation

This task is performed by theoretically validating the hypothesis 1 and 2 in Table 1.7. This is because hypotheses 3 and 4 are theoretically valid as far as description logics

are applicable for representing design requirements and mapping design and process planning domain.

Determining the applicability of description logics is identifying the description logic that provides minimum expressivity. From this identification, the corresponding computational complexity of subsumption can be determined. To identify the minimum expressive description logic, the information space for design requirements in design for additive manufacturing is identified first. Then, information models for design requirements are developed using the information in the identified information space. Through development of information models, minimum expressivity for representing design requirements is identified. Then, the mathematical relation between design and process planning domains is identified. In this research, manufacturing rules and their subsumption relations are identified as a mathematical relation that maps the design and process planning domains. Each of above procedures is justified by a literature. Chapters 3, 4 and 5 provide detailed discussions.

### **1.3.2 Empirical structure validation**

This validation justifies the test case for empirical performance validation. The most important criterion for selecting a test case is its geometric complexity. The geometry needs to be complex enough to realize samples of design requirements and DFM problems that enable testing the entire manufacturing rules discovered in this research. The design and fabrication of a surface that goes into a wind tunnel is a chosen project for generating DFM problems and design requirements for empirical validation. Those generated problems and design requirements are then used to test the method's

storing and retrieval performance. The detailed justification for selecting this test case is presented in chapter 7.

### **1.3.3 Empirical performance validation**

The focus of this task is to empirically validate the method's ability in storing and retrieving the relevant DFM problems. First, the applicability of description logic is demonstrated. This is performed by demonstrating the consistent and correct manufacturing rules taxonomy computation capability. Second, the retrieval performance is demonstrated. This is performed by demonstrating the correct storing and retrieval capability using samples of design requirements and DFM problems. Finally, the computational feasibility is investigated. The subsumption computation and manufacturing rules taxonomy computation time are measured with increasing problem size and number of manufacturing rules. Details are presented in chapter 7.

### **1.3.4 Theoretical performance validation**

The method is developed by assuming that increasing the complexities in design requirements will decrease the feasible space of the process variable that satisfies the design requirements. We believe this is true in general for the domain of design for manufacture. Hence, we claim that the method should be applicable in design for manufacture with moderate modification.

## **1.4 Thesis Organization**

In chapter 2, literature reviews are presented. The literature review is divided into two areas; design for manufacture and engineering information management. Through the literature review, the gaps in each area are identified and summarized.

In chapter 3, the information spaces for design and process planning are identified and justified first. Then, detailed discussions of manufacturing rules are presented. The origin, background, definition, and characteristics of manufacturing rules are presented. Through scoping the information spaces of the two domains and establishing foundation for manufacturing rules, this chapter contributes toward theoretical validation of hypotheses 1 and 2.

In chapter 4, the discovery of manufacturing rules in Stereolithography is discussed first. Then, the discovered manufacturing rules are used to realize the mathematical relation between design and process planning domains. The mathematical properties of manufacturing rules that enable the mathematical mapping of the two domains are discussed too. Through identifying the mathematical relation between the two domains, this chapter contributes toward theoretical validation of hypotheses 1 and 2.

In chapter 5, description logics' implementation of information models is discussed. From the collected manufacturing rules in chapter 4, design requirements are collected and their information models are developed. The information models are developed such that their subsumption relations can be captured by description logics. Through this modeling, the cause of subsumption in design requirements and manufacturing rules is identified. Then, the appropriate description logic for capturing this subsumption relation is identified and justified. The description logics' implementation of design requirements representation and manufacturing rules taxonomy computation are demonstrated. Theoretical validity of hypotheses 1 and 2 are discussed.

In chapter 6, the entire method is discussed. First, the algorithms for storing and retrieval are introduced. The derivation and justification of the ranking metric are

presented next. Then, a demonstration of how the method relates the design requirements to relevant DFM problems is presented with a robot arm example. In the example, several DFM problems and design requirements are used to demonstrate the method's capability. Then, the theoretical validity of hypotheses 3 and 4 is discussed.

In chapter 7, the empirical validation of the method is presented. The usefulness of the retrieval method (DFM framework) is demonstrated. First, the justification of the test case (wind tunnel duct surface design) is presented. Then, a set of design requirements is presented for the test case set up. The test case set up for the three measures is discussed. The three measures include DL applicability, retrieval performance, and computational feasibility. A software implementation of DFM framework is introduced with details of how information models and algorithms are implemented. Finally, the test case results and hypotheses validations are discussed. Through demonstrating the usefulness of the retrieval method, the theoretical validity of hypotheses is demonstrated.

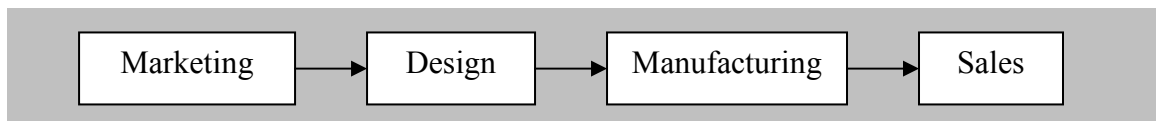
In chapter 8, conclusions are presented. First, the hypotheses validation is summarized. Then, the specific contributions in each area including DFM and engineering information management are presented. Finally, the limitations and future work is presented.

## CHAPTER 2: LITERATURE REVIEW

In this chapter, literature review for two research issues including DFM and engineering information management (EIM) are presented. In DFM, various ways of representing and relating design to DFM knowledge are described. In EIM, various techniques regarding information representation and reasoning for information retrieval are discussed. Finally, summary of the gap that is identified in the literature review will be presented with possible explanation for existing distinctions between needs and capabilities.

### 2.1 Design for manufacturing

Design for manufacturing can be defined as the practice of designing products with manufacturing in mind. Its goal is to reduce the costs of manufacturing a product and to improve the ease with which that product can be made. The traditional sequential approach has been followed by many companies (Figure 2.1). However, this approach does not predict the impact that design has on the downstream processes[25, 26].



**Figure 2.1 Traditional sequential approach of design and manufacturing**

The cost of the product includes the design costs, manufacturing costs, and product redesign costs. The manufacturing costs can be broken into three categories[25, 27]:

- Labor (direct and indirect): 2-15% of total (cost)
- Materials and manufacturing processes: 50-80% of total

- Overheads: 15-45%

Typically, 80% of the manufacturing costs are determined during design although design takes approximately 10%. Hence, the manufacturing costs cannot be improved more than 20% just by improving manufacturing. Therefore, a significant reduction of overall costs through the sequential approach shown in Figure 2.1 is severely limited.

Herrmann also indicated that ignoring downstream issues (or producing poor estimates) leads to poor product designs that may cause unforeseen problems and excessive costs downstream[28]. Often, redesign is performed to correct unforeseen problems during the design, but the cost of redesign can be prohibitive. Therefore, the importance of DFM cannot be overly emphasized.

Susman defined the term design for manufacturing as an effort by designers and manufacturers to improve the product-process fit or to increase the degree to which the products and process are designed simultaneously[29]. Product-process fit is about guiding or evaluating the design such that technical or economic feasibility can be determined during the design process. Simultaneous design of product and process is about information oriented integration of design and manufacturing. According to such goals, the literature review in DFM is divided into two groups in this section.

### **2.1.1 Product-process fit**

The dominant approach in DFM improvement of product-process fit is to incorporate manufacturing concerns into the design process, with the goal of improving product quality, decreasing product cost, and reducing product development time. In short, the objective of such an approach is to ensure that the designer considers manufacturing issues during the design stage.

The implemented DFM systems (mostly software applications) should guide the user through the design so that a part is compatible with a process, or it should provide the user with feedback so that the user can decide if the part needs to be modified. Usually, these types of DFM systems encode knowledge about economic and technological feasibility into rules, algorithms and metrics to evaluate the design. Two major ways of evaluating the design are identified; feature-based and feature recognition. The following paragraphs describe research efforts in each area.

#### 2.1.1.1 Feature recognition

In feature recognition geometric reasoning is performed on part geometry to identify important features[11, 30-35].

Huikang and co-authors discuss various issues in automating process planning tasks and issues in integrating the process planning task with commercial CAD/CAM software[32]. The author argues that feature extraction and feature-based process planning are two important tasks for process planning. The feature extraction algorithm, which extracts features with embedded manufacturing knowledge, is developed. Then the extracted feature is fed into the process planner. The machining features are classified into three broad categories each with machining significance specific to NC machining, so that when extracted they are useful in making process-planning decisions. Using such systems, it is demonstrated that setup planning, operation sequencing and tool selection is automated based on feature shape, feature locations, tool access directions and feasibility of work piece locating and clamping.

Xiaomin and Shah present an interactive and iterative process planning framework, called ASUPPA, which focuses on providing intelligent assistance to a

human engaged in process planning[35]. The author believes that complete automation of process planning is impossible because in practice optimality metrics for process plans are context-sensitive, and because there is significant organizational resistance to approaches that completely eliminate humans from the process planning. The framework (ASSUPA) detects machining features from the geometry by using another framework called the ASU Features Test bed. Then, the extracted features are used to generate a machining process plan. Using the generated process plan, ASSUPA engages in a loop with the user to revise the process plan.

McAdams and Bidkar developed a mathematical framework to automatically evaluate the manufacturability of injection-molded and die-cast parts[31]. Their framework consists of two major components. One is the logical algorithm for the general problem of feature recognition. The other is the implemented mathematical and numerical algorithm to solve key outstanding challenges in feature recognition for manufacturability analysis. The feature recognition method decomposes the part into elemental cubes and the manufacturability of each cube is used to evaluate the manufacturability of the whole part. Through this framework, parting surfaces, undercuts, holes, and bosses in the context of an injection-molded or die-cast part are located

Roberts and co-authors developed a feature-based technique for automated manufacturability evaluation of machining process. In this work, the author introduced a new type of feature called a resource based flexible form manufacturing feature[36]. This feature is used as a form of high level operation plan with which accurate estimates of production cost and time can be made. The author developed a feature recognition

algorithm termed Objective Driven Clustering. The algorithm first generates feature primitives that are operational the sub-plans for sub-regions of a part. Then, primitives are selected and clustered by heuristics, constraints and a user defined evaluation objective to form manufacturing features. The methodology is demonstrated by showing that it can accommodate a part with complex surfaces and interacting form features.

Midha and Smith discussed a system to assist material and process selection while optimizing the design for a part to be manufactured by the cold compaction process[37]. The feature recognition mechanism in this system is implemented as rule based design geometry evaluation program that assesses the part's suitability for cold compaction. B-rep formatted design data conforming to STEP standard is employed in this work. Material and process variable selections are carried out using a hybrid system employing Bayesian neural network and heuristics. Through developing such system that aids material selection and manufacturability analysis during the early design stage, it is claimed that part designers can reduce the number of design iterations, improve the quality of designs and minimize trials required in powder metallurgy part manufacture.

Lu and co-authors presented a volume-based geometric reasoning and visualization approach to support design evaluation during preliminary design[38]. The system extracted from the part model the underlying geometric characteristics which usually affect part quality or increase manufacturing difficulties. All the information was then presented to designers in a form which could be easily interpreted via visualization techniques. The presented system focused on thermal and flow related problems in die-casting.

### 2.1.1.2 Feature-based design

The goal of feature-based design is to represent the design using important geometric or manufacturing features so that the represented features are used to evaluate the design[19, 37, 39-43].

Rehman and co-authors developed a method for modeling costs throughout the design phase of a product's life-cycle, from conceptual to detailed design[11]. This method is developed by incorporating knowledge-based and case-based approaches. In the method, design knowledge is used to identify features from design descriptions. Then extracted features are linked to production knowledge through the black board framework of problem solving, which incorporates both case-based and rule-based reasoning. The author argues that the approach to design evaluation has the advantage of allowing management to make more accurate bid estimates, of encouraging designers to design to cost and of reducing the amount of design rework, hence reducing the product's time to market and controlling product cost.

Lin developed a knowledge-based design critique system for manufacture and assembly of rotational machined parts at the early design stage[19]. Feature-based design is utilized to represent rotational parts with machining features and relationships between them. To support such representations, taxonomies of features and relationships are developed. The design guidelines are encoded into a design critique system that is implemented by the rule-based system. The author argues that an early critique to rotational part design and reduce the problems and adjustments that may occur in the manufacturing and assembly processes of a product's life-cycle.

Welp and co-authors introduced a method that estimates part cost at the conceptual design stage[43]. The features in conceptual design stage are represented using an object oriented design method. Then, the classes containing concept modeling elements with driving cost attributes concerning technical and process properties are developed. The cost generation mechanism implemented in the knowledge-based application is then mapped to the class structure framework to determine the cost.

Gao and Sharma argue that the most of the important cost related design decisions are taken in the early design stages[40]. The author presents a method (FBCDS) that evaluates and analyzes at the early design stage. The method is based on a process planning system and an embedded expert system which resolves the abstract data associated with the early design stage. Such abstract data is composed of manufacturing features for machining processes and is represented with STEP AP224 standards. The method incorporates a feature library developed using STEP AP224.

So far we have discussed various research efforts in product-process fit in DFM. More specifically, feature-based design and feature recognition methods to evaluate design to determine technical and economic feasibility are discussed.

However, in these methods, researchers have to formulate the requirements of the investigated manufacturing process into knowledge (rule or algorithm), then develop software systems to allow designers to analyze manufacturability during the design stage. The approach seems to work well for processes with few requirements like rapid prototyping and for the conceptual design stage by using approximation codes. However, automated analysis meets difficulties if a part or a fabrication process is complex. For a complex part, complex interactions of three-dimensional geometries have largely

prevented formalization of this knowledge. For a complex fabrication process, it is rather difficult, if not impossible, to formulate all of a fabricator's experience and decisions into rules and algorithms on the designer side[15].

### **2.1.2 Simultaneous design of product and process**

Concurrent engineering is defined as “a systematic approach to the integrated, concurrent design of products and their related processes, including manufacturing and support”[15, 44]. Recently, research in concurrent engineering has drawn tremendous attention due to rapid developments in information technology. Numerous research efforts have been focused on different methods of integrating product and process planning. Eversheim classified these research efforts into three groups[45]. The first group is organizational structure-oriented integration, which is focused on the formation of multidisciplinary design teams such that their collaboration is enhanced. The second group is process-oriented integration through simultaneous execution of product and process design. The third group is information-oriented integration, which focuses on realizing information flow between design and manufacturing by improved data exchange. This research focuses on information-oriented integration of design and manufacturing.

Jin identified three major issues in information-oriented integration research. The first issue is task decomposition and representation[46]. Task decomposition is about modularizing tasks into sub-tasks that can be divided in a way that minimizes interaction between sub-tasks. Task representation is about defining design/manufacture tasks and sub-tasks in a form that computers can easily handle to identify the interactions among the sub-tasks and tasks. The second issue is the need of a communication infrastructure

to enable communications among distributed tasks and sub-tasks. Once the tasks and sub-tasks are completed, the results need to travel to either higher tasks or next sequential tasks in the distributed environment. Hence, a sophisticated communication infrastructure is required to facilitate information flow in such an environment. The third issue is communication coordination. It is required to resolve dependencies among sub-tasks.

Among the numerous research efforts in information oriented integration, three sub areas are identified as relevant to this research; feature representation[47-50], integration framework for distributed environment[15, 51-62], and knowledge sharing[63-66]. The following paragraphs describe each area in detail.

#### 2.1.2.1 Feature representation

In DFM, the manufacturing feature information needs to be identified from the geometry so that the design can be evaluated using encoded DFM knowledge (rules and algorithms). So far almost all computer aided design (CAD) systems transfer only geometric information using standards such as IGES, STEP, DXF, and VDAFS[48]. However, there is a lack of standards to define features unambiguously and uniformly. Some research efforts have been focused on developing feature representations.

D'Souza and co-authors developed a feature representation language called N-Rep for feature definition that includes topology, topological relationships, geometry, geometric relationships, variables and parametric relationships[48]. N-Rep uses a constraint graph to represent features. The nodes represent the shape entities while the edges represent the shape conditions. This is intended to be an application-neutral unified representation for feature data exchange, designed by features and feature

recognition. The representation is tested for compatibility with the feature representation of STEP AP224 for machining process.

Ma and co-authors introduced a representation called associative feature. This representation is intended to integrate knowledge-oriented tools and CAD application[47]. In this representation, an object oriented technique is used to model features, and they are related to various geometric entities. Then, the geometry with associative features is used to evaluate the technical feasibility of the design. For the case study, cooling channels in the design of plastic injection moulds with a CAD tool are modeled as an illustrating associative feature type.

Those feature representations are limited to only geometric entities and do not provide sufficient expressivity to represent DFM tasks.

#### 2.1.2.2 Integration Framework

Integration framework is about developing a DFM method for distributed environment. It usually involves information modeling, agent (distributed tasks) identification, metric and algorithm development.

Sambu and Rosen introduced a DFM method called geometric tailoring[4, 5, 67]. The authors indicate that the challenge in fabricating a functional prototype is in the mismatch of material property for prototype and production part. This becomes more challenging in distributed environments especially, when designers have little or no knowledge about manufacturing prototypes. The key idea of the method is to modify the dimensions of prototype parts to match key characteristics of production parts, such as stress and deflection behaviors. In geometric tailoring, the designer provides some freedom in the geometry and the manufacturer utilizes such freedom to identify process

variables that satisfy the given design requirements. The information transfer from design to manufacturer is in problem formulation. Through this procedure, geometric tailoring eliminates design and manufacturing iteration.

Chen and co-authors developed a method that enhances simultaneous design of product and process[15]. The author treats such tasks as a two-objective optimization problem with two teams: design and manufacture. The design team aims to optimize the overall product functionality and the manufacture team aims to minimize the total manufacturing costs. The key idea of the method is to solve the optimization problem by exploring three game-type team paradigms (non-cooperative, cooperative, leader/follower). The method consists of satisfaction metric, three dual-team based models, and computation algorithms. Through the study of the welded beam design, the method is tested and demonstrated.

Sormaz and co-authors introduce a methodology that integrates CAM software with rule-based process planning[61]. The author indicates that process planning in machining involves selection of machining operations, sequencing of machining operations, selection of cutting tools, determining setup requirements, calculation of cutting variables, tool path planning and NC code generation. Among them process sequencing is the significant activity that determines the cost and time. It is experience-based and relies heavily on the judgment of the machinist. In the method, a rule-based system is used as an application neutral and process sequence generating mechanism. Then the generated process sequences are used by distributed CAM software.

Those frameworks are specific to certain DFM tasks and are not quite extensible for other purpose such as sharing DFM knowledge in a distributed environment.

### 2.1.2.3 Knowledge sharing

Some research efforts are focused on updating and sharing DFM knowledge (rules and algorithms) in distributed environments. The following paragraph describes such research in detail.

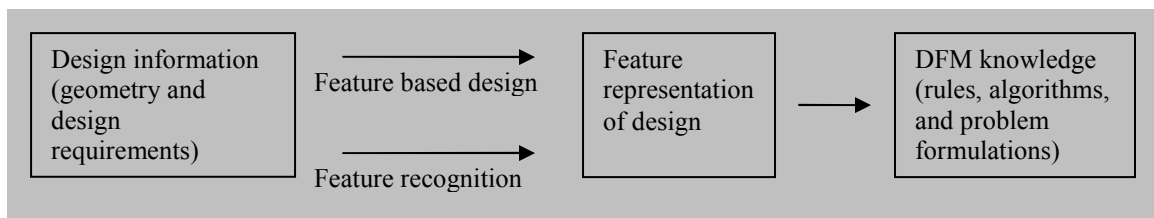
Tharakan and Shah introduce generic domain independent shell for manufacturability that is configurable and customizable to any domain or process[66]. The author explains the importance and necessity of having a framework that provides a mechanism to easily modify existing domain knowledge and that allows domains to be selectively populated for being competitive. The shell considers manufacturability assessment in its entirety from the global level of the entire manufacturing domain to the process level and finally to the user/workshop level. Among the many components, the interaction module, knowledge base, and the evaluation module are the three main modules. The interaction module allows a domain expert to update domain knowledge. This updated knowledge is stored in knowledge base module. Then the evaluation module evaluates the design. Case studies for milling and injection molding are proposed.

Stauffer and Rule describe the lack of and need for a consistent format for representing DFM guidelines[65]. The author further argues that a standard DFM guidelines structure would provide a consistent context for better understanding these guidelines. A template is proposed in the form of taxonomy based on variable cost of the part. The variable cost is comprised of its material, labor and utility costs and for each of these, the factors that influence the cost are defined. The author claims that this template can provide a framework for organizing and presenting DFM guidelines in a context that

the engineer can use in a more effectively. The templates are validated through two test cases: guidelines for electromechanical machining and guidelines for plastic injection molding.

Hodgson and co-authors proposed a knowledge acquisition method for acquiring design as well as shop floor manufacturing knowledge[64]. The method captures part information, part records, case studies, model improvement, design issues, and production issues. Such information is then encoded into a knowledge base during the design and production time. Through case-base retrieval, a query is performed to retrieve relevant cases of design or manufacturing process plans.

So far we have discussed various ways of evaluating designs to determine technical and economic feasibility. Such research efforts can be summarized as shown in Figure 2.2.



**Figure 2.2 Summary of research that relate design to DFM knowledge**

As shown in Figure 2.2, there are research efforts in feature based design, feature recognition, feature representation, framework development, and DFM knowledge sharing. However, there has been little or no research effort focused on capturing and sharing DFM knowledge through a formal approach. Some studies discussed DFM knowledge update and capturing. However, this research mostly depend on ad-hoc methods and lack formal approach. Without a formal approach, it is extremely difficult

to guarantee extensibility and reusability of DFM knowledge especially in a distributed environment.

One possible explanation of this deficiency is that the DFM systems are mainly developed to evaluate the design not to share DFM knowledge. Because of this, a majority of the DFM knowledge is localized and used for evaluating designs in a specific domain[50]. Recently, researchers in the DFM area identified the necessity and importance of capturing and sharing DFM knowledge. However, the research is still at an early stage and needs tremendous attention to deliver fruitful result to industry.

## **2.2 Engineering Information Management**

Due to rapid developments in information technology, the use of globally distributed engineering design teams continues to increase as companies aim to boost profits and decrease lead times by leveraging knowledge from dispersed locations. Such collaboration involves information from diverse disciplines and results in a collaborative set of tasks among multidisciplinary, distributed design teams[68]. For designing and developing complex products, the design tasks are often decomposed into smaller tasks so that the designers can focus on smaller and manageable tasks. However, such decomposition often results in inefficiencies and difficulties in the communication and integration of product information between designers[69]. Undoubtedly, information management is a critical issue for such collaboration to be successful. Studies show that information management problems in the automotive supply chain are estimated to cost nearly \$1 billion U.S. dollar per year[70].

Engineering information management (EIM) systems initially provided workgroup file management for product definition information in the form of

CAD/CAM/CAE files; these systems evolved to managing structured product meta-data. Structured meta-data is computer processible and amenable to efficient storage and retrieval by database management systems; it is often driven by the underlying business processes it supports. In early applications, such meta-data were used to manage part control information that referenced CAD/CAM/CAE files. Today, this has advanced to support business processes such as engineering releases, product bills of material definition, and change management processes. This evolution has elevated the applicability of EIM systems throughout the product life cycle, where structured engineering contents drives many business processes including production, supply chain fulfillment, service engineering, and sales configuration. Such EIM systems are needed in enhancing collaboration among distributed designers and engineers for the following reasons[71]:

- Design and manufacturing are information intensive tasks and generate immense amounts of digital information
- There is a need to share information in diverse disciplines

However, current EIM research still needs to address many issues including interoperability, knowledge modeling, knowledge storing, knowledge retrieval, etc.

Product information modeling is recognized as the most essential step toward better integration of distributed resources. It defines data elements and relationships among them. A product information model describes its attributes and relationships (e.g., inheritance, aggregation, classification) to other data elements. Popular methods include entity-relationship method (ER) and object-oriented method (OO)[50].

Product information modeling has received significant attention in the area of relational databases, information systems, and knowledge representations. Relational database applications in business-process have provided foundations for data modeling that are applicable to engineering information modeling to a limited degree. However, the complex nature of the engineering information demands a great deal of investigation and research. Hence, EIM is a relatively new area compared to business-process information management[71].

Today, product development is more often done in geographically distributed environments. Out sourcing is not only done for manufacturing but also for product development efforts. Product development across companies, and even within a single company is often done using a heterogeneous software tool environment. Due to such an environment, the internet and intranet are supplanting paper and telephone as a means of information exchange. Therefore, it is critical for a design software tool to support formal representation, capture, and exchange of product information[72]. A classification of interoperability problems and information exchange that summarizes the essential problems in engineering is shown in Figure 2.3[71].

- Differing format – Identical objects are described with different languages. Syntactical translation between the different systems must be completed
- Differing representation – The same objects are represented in different systems with different representations. Objects with identical relevant properties A B-rep and a CSG model occupying the same volume. The representation must be converted between x and y
- Differing behavior – Objects with identical static properties, but acting differently under operations.
- Incompatible content – Objects with different properties relevant static properties. Negotiation outside the scope of the transaction.

**Figure 2.3 Interoperability problem classification[71]**

In addressing many of these problems, information plays the major role as the key integrator in product development. Numerous research efforts from academia, government, and industry have contributed to addressing such problems. More specifically, major research efforts have contributed toward developing computer-based product and process models for exchanging information throughout the development process.

ISO 10303 (commonly known as STEP) has provided a set of standardized product models for exchanging engineering product data between software applications used through the life-cycle of product[73, 74]. The initial objective of STEP is to provide standardized product model for CAD applications such that the standard can be used to exchange CAD data in an application neutral form. STEP is – “a neutral mechanism capable of completely representing product data throughout the life-cycle of a product,...The completeness of this representation makes it suitable not only for neural file exchange, but also as a basis for implementing and sharing database and archiving.”

The development of this standard initiated in 1984 in ISO. Its application areas are not confined to exchange of CAD data but are found in many engineering disciplines such as manufacturing feature representations (AP224).

However, STEP has some limitations. First, STEP loses all the history about the geometry when it transfers data. Therefore, it only captures a snap shot of geometry and does not allow one to track the changes in engineering information and the evolution of the product. Secondly, it is quite complex to use. In other words, STEP demands a stiff learning curve and expensive tools. One possible explanation for this is that STEP tries to capture all the possible information regarding geometry and more. Hence a huge amount of information needs to be identified and encoded. Consequently, using STEP requires expensive tools and education[71].

Among the numerous research efforts in engineering information management, efforts toward design repository are identified as relevant. The following section describes various research efforts that have contributed toward development of a design repository.

### **2.2.1 Design repository**

The main objective of a design repository is to store and retrieve design rationale information such as how and why certain product is designed in a particular way. When information is retrieved from the repository, the designer should be able to see the evolutionary nature of the product knowledge and information throughout the design process. Therefore, a design repository is not intended to be a part catalog[71, 75]. Szykman and co-authors describes characteristics of a design repository in the following list.

- *“Design repositories attempt to capture a more comprehensive product representation than traditional CAD databases, including the kinds of knowledge such as function, behavior, rationale, etc”*
- *“Design repositories tend to be heterogeneous by containing information such as schemata, data structure, video, and other types of information”*
- *“Capabilities of supporting the design processes are built into the design repository. Such capabilities include search for components/assemblies that satisfy required functions, explicit representation of physical and functional decompositions and mapping between them, automated reasoning about the design and more”*

Consequently, a relational database approach is not suitable for a design repository. Artificial intelligence (AI) techniques are often employed. The following paragraphs describe various research efforts in design repository.

There have been some research efforts toward retrieving designs by detecting similarities in geometry and using a case-based retrieval approach[76-83].

Ramani and co-authors introduced a method for a reconfigurable shape search system for 3D engineering models[78, 79]. The method takes a shape query and converts it into feature vectors and skeletal graphs using algorithms such as voxelization, skeletonization, and skeletal graph extraction. Then the method uses the Euclidean distance of the feature vectors, as well as the distance between skeletal graphs to measure the shape similarity. The method is implemented and tested by populating 3D shapes in the database.

Lu and co-authors introduced a case-based reasoning (CBR) method for retrieving machining process plans[77]. The author argues that one effective way of developing a new machining process plan is to retrieve similar cases of process plans. The method consists of feature-based representations, an index of parts, feature hierarchy for cutting process, and a similarity metric. The domain of the method is confined to axisymmetric part machining.

Regli and co-authors applied a CBR technique in mechanical bearing design[76, 80, 83]. A system is implemented for similar design case retrieval and adaptation to the current problem. The cases are indexed with bearing load direction and shaft housing diameters. Then the *Nearest Neighbor Matching Algorithm* (NNM; Kolodner,1993) is used to determine the similarity of the cases. The retrieved cases are adapted to the current problem by approach that combines parametric and constraint satisfaction techniques. The method is tested with rolling bearing design problems.

Other research in this area includes CBR for retrieving conflict resolution in distributed design[84], conceptual design information[85], etc. In those studies, CBR approach provided what has been expected. However, there are limitations in the current CBR research. One of the critical issues is its extensibility. As shown in the literature, the CBR requires representing the domain knowledge, indexing cases, and detecting similarities. All these procedures are performed in ad-hoc fashion in current CBR research[26]. In other words, there is no formalism for representing and reasoning cases. Consequently, extending previously built design repository using CBR is extremely difficult, if impossible, especially in a distributed environment.

There are some design repository researches based on an ontological approach[86-89]. In the ontological approach, domain knowledge is explicitly and formally represented. Then the retrieval is performed by semantic inference using rule-based logics or ontological matching. The following paragraphs describe some of the representative research.

Kim and coauthors introduced method for retrieving electromechanical components for simulation purposes[86]. The author emphasizes the importance of the cataloging systems being able to support engineers in selecting and evaluating electromechanical components and subsystems. Hence the method allows one to describe the intended use as a query. Then, the query is refined based on the domain ontology to retrieve multimodal information of electromechanical components for simulation and selection. The domain knowledge is represented using knowledge representation environment LOOM[90, 91]. LOOM organizes concepts and relations into a taxonomy using subsumption algorithm. The method uses the query capability of LOOM to retrieve component information.

Ramani and co-authors introduce an approach of building a design repository by using ontology engineering and natural language processing (NLP)[89]. Based on the domain ontology, NLP is used to extract the design knowledge from the text document to annotate, index, and retrieve. Due to the complexity and expense of a full NLP, simplified NLP algorithms are developed to process text. Storing and retrieval performance is tested using a few years of class projects at Perdue University.

Regli and co-authors introduce a method of retrieving mechanical devices design to aid conceptual design[87, 88]. The method utilizes description logics to represent

domain knowledge for automated reasoning service. The method consists of three components; conceptual design interface, design semantics representations, and reasoning mechanism. The conceptual design interface is used to capture the design knowledge based on the design semantics representations. Then, captured knowledge is encoded into description logics to store/retrieve conceptual design cases. Engineering functions are primarily represented for indexing conceptual design cases. The method is tested using electronic sensor design example.

Other research in ontology usage in engineering includes service discovery in PLM environment[92-94], product data description[95-101], application integration[72, 102-105], etc. The ontological approach for modeling product information and developing a design repository has received increasing attention due to increasing popularity of semantic web and XML web services (Li). The semantic web tries to capture, exchange, utilize and manage a large collection of disparate knowledge for various domains using domain ontology. Such attempts are directly applicable in developing an engineering design repository.

In the ontological approach, knowledge representation formalisms are used to represent and reason the domain knowledge in some researches. However, researchers currently fail to provide applicability analysis of formalisms in the domain where they are applied. For example, utilization of the semantic web language OWL (Web Ontology Language) is basically using formalism description logics. However, none of the research that uses such language provides critical analysis such as expressivity or computational feasibility of such formalism. Without those analyses, it is quite difficult

to predict the system performance especially in guaranteeing extensibility of the design repository.

### 2.3 Summary

The gap is identified in DFM and EIM. Table 2.1 summarizes the gap in each area.

**Table 2.1 Gap summary**

Area	Gap
DFM	Formal approach of relating design information to DFM knowledge to share and manage DFM knowledge in distributed environment
EIM	Applicability analysis of formalisms in the domain where they are applied. More specifically, description logics applicability analysis in design for additive manufacturing

## **CHAPTER 3: SCOPE OF THE DESIGN AND PROCESS PLANNING DOMAINS AND DISCUSSIONS ON MANUFACTURING RULES (ORIGIN AND CHARACTERISTICS)**

As discussed in Figure 1.16 in section 1.1.4, the retrieval method relates the design problems to the relevant DFM problems through mapping the design and process planning domains. Manufacturing rules and their subsumption relations are identified as the entities that enable such mapping in this research. In this chapter, the two domains (design and process planning) are identified and justified (section 3.1). Through this identification, the scope of this research is identified. Based on this scope, manufacturing problems templates (section 3.2) and manufacturing rules (section 3.3) are discussed. Specifically, the foundations of MPTs, design requirements, meta rules, and rules are established. Finally, the functionalities of meta rules and rules in relating the design requirements to the relevant DFM problems through appropriate MPTs are discussed (section 3.4). By defining the scope of the research and establishing the foundations for manufacturing rules, this chapter contributes toward the theoretical validation of hypotheses 1 and 2.

### **3.1 Information space for design and process planning domains**

In this section, we identify an information space for the design and process planning domains. Identifying the design information space involves identifying the design requirements that can objectively relate any given design problem to the appropriate process planning. Those requirements are identified to be the accuracy requirements, including surface finish and various tolerances. For the process planning domain, process variables for Stereolithography, including part orientation, layer

thickness, and overcure are identified. During research, Stereolithography is used to discover manufacturing rules that are general in additive manufacturing and are specific to Stereolithography. The following sections present the details of the information space for each domain.

### **3.1.1 Design requirements**

The general design information space is a huge area that includes functions, geometry, features, accuracy requirements, time, costs, etc. From such a huge information space, accuracy requirements are identified to objectively relate the given design problems to the appropriate process plan in additive manufacturing. In this research, those accuracy requirements include surface finish and 13 tolerances[22, 23]. Other design requirements such as cost, time, and geometry are not included, because the ways those design requirements influence process planning in additive manufacturing are specific to the design problems. For example, the cost requirement can influence process variables differently depending on the complexity and size of the geometry, preferences, restrictions, etc. Modeling and representing such subjective information is a huge research area and is not the focus of this research.

Regardless of the design problems or design requirements, the way the accuracy requirements influence the process planning is consistent. For example, satisfying the surface finish requirements by selecting appropriate process variable values is not influenced by other design requirements such as cost or time. This is because the relation between accuracy measurements and process planning are determined by physics and mathematics rather than preferences. Many of the empirical models that relate the accuracy measurement to process plans are developed at Georgia Tech[20, 22, 23, 106].

Therefore, accuracy requirements and associated information are selected to objectively relate design problems to the process planning domain.

Generally, satisfying the accuracy requirements is the most significant issue in manufacturing process planning[1, 3]. In additive manufacturing, the most significant hindrance in meeting accuracy requirements is the stair stepping effect as discussed in chapter 1. Therefore, process variables in additive manufacturing are chosen such that the stair stepping effects are minimized on the surfaces where accuracy requirements are specified. As discussed in section 1.1.1, part orientation and layer thickness are the most influential process variables that can minimize the stair stepping effects. Part orientation is a process variable that is used to indicate the orientation of a part. Layer thickness is the thickness of each deposited layer (Figure 1.3). The two variables are universal process variables in layer-based additive manufacturing [2, 107]. Details on those process variables are presented in the next section.

Hence, appropriate values of part orientation and layer thickness are selected to minimize the stair stepping effects on the important surfaces. Difficulties arise when those surfaces are differently oriented. In some cases, it is infeasible to identify part orientation values that can satisfy all the accuracy requirements on differently oriented surfaces. For such cases, layer thickness can be controlled in an attempt to satisfy accuracy requirements. However, feasible values for layer thickness are typically limited to few discrete values and decreasing the layer thickness greatly increases part build time. Hence, alleviating the stair stepping effect using layer thickness is very limited. Not only the orientations of surfaces but also, the severity of the restrictions imposed on accuracy measurement impact decisions concerning part orientation values. For example, the part

orientation value can be restricted to a single value if one of the accuracy measurements is so restricted that it can only be satisfied by a single part orientation value.

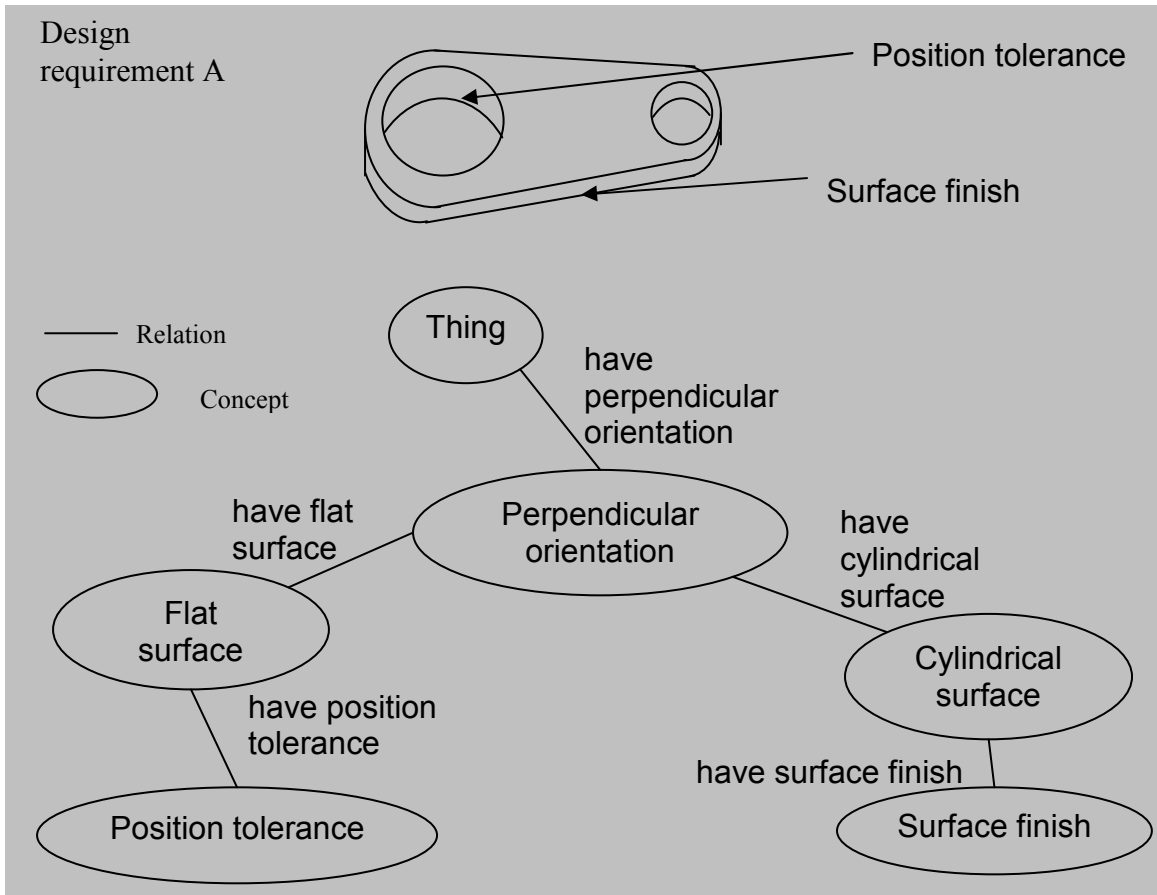
In short, satisfying the accuracy measurement is greatly hindered by the stair stepping effect and it can be greatly alleviated by proper part orientation value. To select the proper part orientation, design requirements such as surface types, relative orientations of surfaces, accuracy requirements, and restrictions imposed on accuracy requirements are critical[2, 20, 22, 23]. Therefore, those concepts form the information space for design requirements that are relevant to additive manufacturing. Figures 3.1 and 3.2 show the concepts and relations respectively. Figure 3.3 shows an example of formulating a design requirement using concepts and relations in Figures 3.1 and 3.2 respectively.

- Types of surface
  - Flat, Cylindrical, Conical, etc
- Relative orientations of surface where accuracy measurements are specified
  - Opposite, Perpendicular, Angled
- Accuracy measurement
  - Surface finish, Flatness tolerance, Position tolerance, etc
- Restriction level

**Figure 3.1 Concepts for forming design requirements**

- have surface finish, have position tolerance, have flatness tolerance, etc
- have opposite orientation, have perpendicular orientation, have angled orientation
- have flat surface, have cylindrical surface

**Figure 3.2 Relation between concepts for forming design requirements**



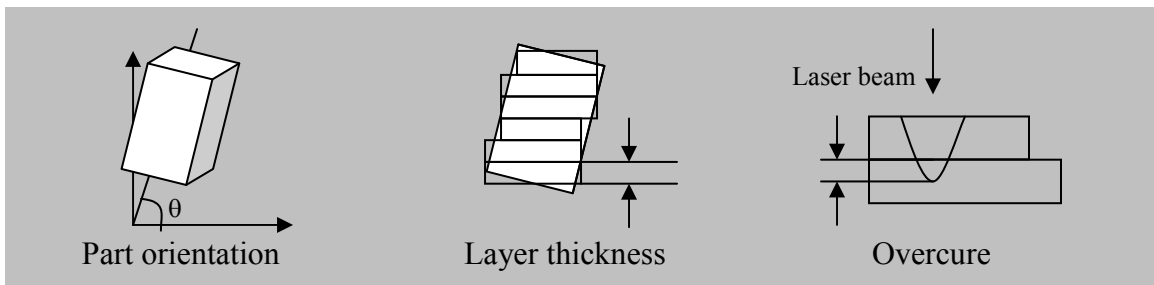
**Figure 3.3 Example of design requirement formed by determined concepts and relations**

In Figure 3.1, the concepts that are required to express design requirements are presented in a taxonomic structure. Those taxonomies include surface type, accuracy measurements, relative orientations, and restriction levels. In Figure 3.2, the relations that link or combine the concepts in Figure 3.1 to form design requirements are presented. Then in Figure 3.3, an example of design requirement formulation is presented using the robot arm example. First, the surfaces and associated accuracy measurements are identified for formulating design requirements. Then, the design requirement is formed by adding concepts to an abstract concept ‘thing’. In Figure 3.3, a

node represents a concept and an edge represents a relation. In plain words, such design requirements can be expressed as “*Surface finish is specified on cylindrical surface and position tolerance is specified on flat surface and the relative orientation of the two surfaces is perpendicular*”. In this example, the restriction level is omitted and is discussed in a later section. In short, the concepts in Figure 3.1 and 3.2 are combined to form the information space for the design domain in this research.

### 3.1.2 Process variables

As mentioned previously, Stereolithography is selected as the manufacturing process to be studied for collecting manufacturing rules in this research. Hence, process variables and their feasible spaces for Stereolithography are selected for the process planning domain. In this research, the phrase feasible space means the possible ranges of values. Figure 3.4 presents a graphical illustration of the major process variables in Stereolithography.



**Figure 3.4 Process variables in Stereolithography**

In Figure 3.4, part orientation, layer thickness, and overcure are presented. As discussed previously, part orientation is the orientation of the part. Hence, the values are angles as shown in Figure 3.4. Layer thickness means thickness of the material deposition layer as shown in Figure 3.4. Overcure is the depth of penetration of the laser

beam that bonds two layers together. Overcure only exists in laser-based additive manufacturing processes including Stereolithography, Selective laser sintering, etc. In laser-based additive manufacturing, a laser beam is used to cure liquid resin or powder particles to deposit the material. Overcure means depth of penetration of the laser beam below the current layer to bond the current layer to the previous layer as shown in Figure 3.4. Table 3.1 presents the feasible values for each process variable.

**Table 3.1 Process variables and their feasible value spaces**

Process variables	Feasible values
Part orientation (PO)	Infinite (continuous)
Layer thickness (LT)	Discrete
Overcure (OC)	Discrete

The feasible values of part orientation are infinite. In other words, there is total freedom in selecting any part orientation value. Layer thickness is discrete. Typically, there are only few (2~4) thickness values available at the machine level. Overcure is also discrete. Overcure is typically determined by various process conditions including the laser scan speed, the material properties of liquid resin, etc. In short, the information space for process planning in Stereolithography is formed by the process variables and their feasible values shown in Table 3.1.

### **3.1.3 Research scope**

In this research, the given design problems are related to the relevant DFM problems through appropriate process plans. Therefore, the scope of the retrieval method and research is determined by identifying types of design problem and process plans that the retrieval method relates. Based on the previous two sections (section 3.1.1 and 3.1.2), the scope of this research is identified in this section.

As discussed in section 1.1.4, the design requirements are manually extracted from the given design problems. In this research, the design requirements that are formed by the concepts in Figures 3.1 and 3.2 are manually extracted for the retrieval of the relevant DFM problems. Such design requirements are typically available at the detailed design stage[108].

At the detailed design stage, uncertain geometry is finalized and the detailed process plans are determined. Hence, the design problems at this stage contain approximate geometry and detailed design requirements shown in Figures 3.1 and 3.2. Furthermore, the design problems are limited to single and solid mechanical part design.

The scope of process planning is limited to the layer-based additive manufacturing processes. In other words, the retrieval method retrieves relevant DFM problems to support process planning for the layer-based additive manufacturing processes. Those manufacturing processes include Stereolithography, Selective laser sintering, Fused deposition modeling, 3D printing, etc.

The scope of the process planning further limits the scope of the design problems. This is because additive manufacturing technologies are typically used to fabricate highly customized parts that demand complex geometry such that the geometry is not realizable by conventional manufacturing technologies. The research scope is summarized in Table 3.2.

**Table 3.2 The research scope**

Design problem	Process planning
<ul style="list-style-type: none"> <li>• Design problems for detail design stage               <ul style="list-style-type: none"> <li>○ Uncertain geometry need to be finalized and the corresponding process plans need to be determined</li> </ul> </li> <li>• Design problems that require:               <ul style="list-style-type: none"> <li>○ Highly customized parts</li> <li>○ No mass production</li> <li>○ Demands complex geometry</li> </ul> </li> <li>• Solid mechanical and single part</li> </ul>	Layer-based additive manufacturing including Stereolithography, Selective laser sintering, Fused deposition modeling, 3D printing, etc

Based on the identified design and process planning domains, the following sections discuss the manufacturing problem template and the manufacturing rules.

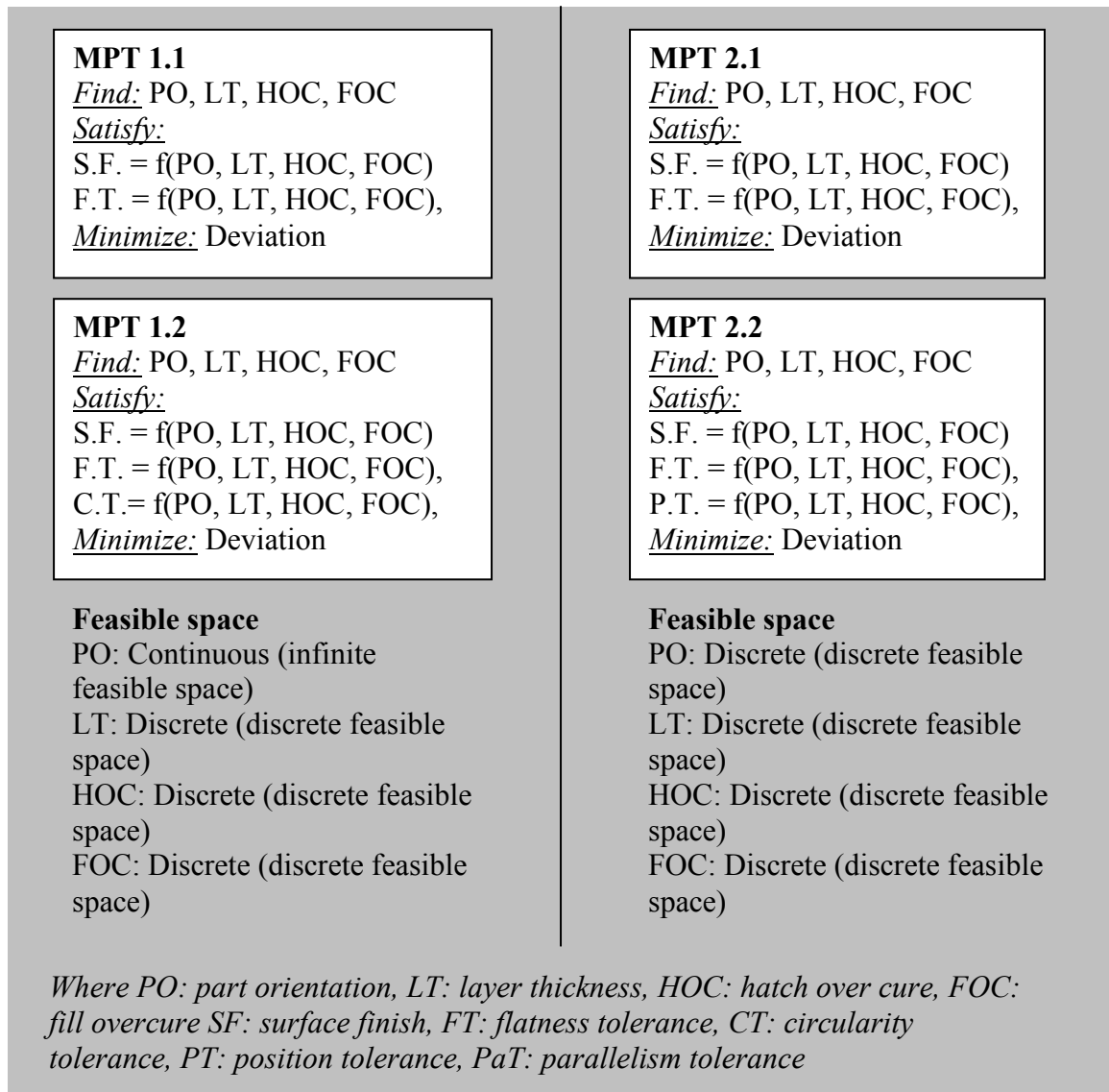
### **3.2 Manufacturing problem template (MPT)**

In this section, the manufacturing problem template (MPT) is introduced. The MPTs are the constituents of process planning domain in Figure 1.16. The MPTs determine the feasible spaces of process variables and their relation to the given design requirements. The following paragraphs discuss the details of MPTs.

The difficulty in formulating and solving a DFM problem is in modeling the design requirements and identifying an appropriate solution strategy for solving the DFM problem. To model the design requirements properly, correct identification of the feasible space of process variables that influence the design requirements is crucial. Determining an appropriate solution strategy means identification of correct feasible space of process variables as discussed in chapter 1. Therefore, properly formulating and solving DFM problems require accurate identification of feasible space of process

variables. Determining feasible space of process variables is basically partially performing process planning. This partial process planning is performed in manufacturing problem formulation (Figures 1.10, Tables 1.2 and 1.3) in using geometric tailoring. Then, process planning and design is completed at solution generation of DFM problem.

The formulated manufacturing problems are usually applicable to ranges of design problems. For example, the manufacturing problem in Figure 1.15 is applicable to the design problem of a robot arm (Figure 1.9) as well as its own design problem (Figure 1.15). This is because there is only a finite number of ways to satisfy various types of design problems in terms of process planning. The MPTs are pre-formulated manufacturing problems that are known to be applicable to the ranges of design problems for partial process planning. For the generality of MPTs, only accuracy measurements (surface finish and tolerances) are considered for their design requirements (excluding cost, time, stress, etc). Examples of MPTs are shown in Figure 3.5.



**Figure 3.5 Examples of manufacturing problem templates (MPT)**

In Figure 3.5, MPTs determine the feasible space of process variables and accuracy requirements that should be modeled as functions of the feasible space of process variables. For example, MPT 1.1 determine feasible space of part orientation, layer thickness, hatch overcure, and fill overcure to be infinite, discrete, discrete, and discrete respectively. It also determines that surface finish and flatness tolerance are functions of part orientations, layer thickness, hatch overcure and fill overcure. In the

case of MPT 2.1 and 2.2, the feasible space of process variables is all discrete. Therefore, the accuracy models are functions of the feasible space of the discrete process variables in MPT 2.1 and MPT 2.2. If these MPTs are available during the DFM problem formulation, appropriate manufacturing problems can be selected relieving the designers or engineers from developing them from scratch. Then, selected MPT relates the accuracy measurements to feasible spaces of process variables that need to be considered for the given design problem.

In the robot arm example discussed in Figures 1.9 ~ 1.13, let's assume that MPT 2.1 is an appropriate manufacturing problem. Then, MPT 2.1 shows feasible spaces of process variables (discrete) and their relations to accuracy measurements (surface finish and flatness tolerance). Further more, MPT 2.1 determines an appropriate solution strategy for the DFM problem to be the solution strategy for discrete process variables. In short, MPTs provide feasible spaces of process variables and their relations to accuracy measurements for the given design requirements. However, MPTs do not provide actual accuracy models and demonstration of solution strategy utilization.

In this research, manufacturing rules and their subsumption relations are identified as entities that determines appropriate MPTs for the given design problem. Hence, previously formulated and solved DFM problems are classified by manufacturing rules such that DFM problems under the same manufacturing rules share the same MPT. In other words, the DFM problems under the same manufacturing rule have the same design requirements (*given* part of MPT) and accuracy models that are expressed by the same feasible space of process variables. The following section presents detailed discussions on manufacturing rules.

### **3.3 Manufacturing rules**

In this section, the background information of manufacturing rules such as origin, definition, derivation, and characteristics are discussed. The following sections discuss details.

#### **3.3.1 Origin and Background**

In this research, the manufacturing rules and their subsumption relations determine appropriate MPTs for a given design requirements. The MPTs are basically the feasible spaces of process variables. Currently, empirical models of accuracy requirements are used to determine exact values of process variables for a given accuracy requirement. In this research, those empirical models are used to identify the design requirements and the corresponding MPTs. The manufacturing rules are then entities that map those identified design requirements and MPTs in a form of a rule. The following paragraphs describe the details.

Quantitative models of accuracy measurement (surface finish and tolerances) can greatly enhance process planning tasks. A significant amount of work is done at Georgia Tech in obtaining such models.

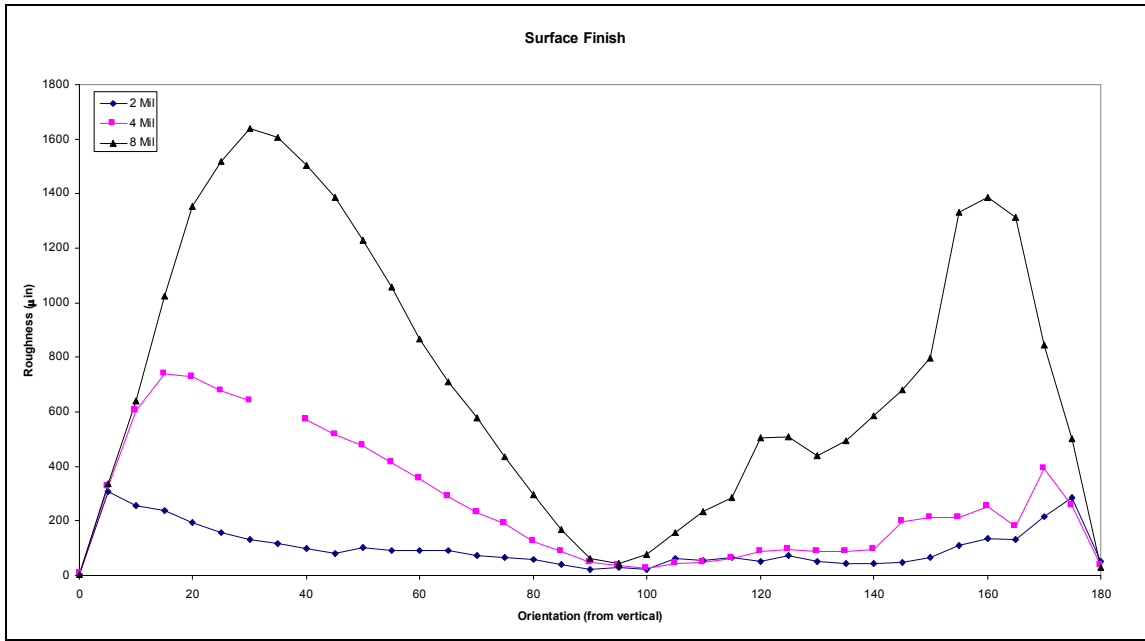
Lynn performed a number of experiments on an SLA 250 (SLA machine) – SOMOS 7110 (resin) and developed mathematical models to predict tolerances of SLA prototypes[22, 23]. The first set of experiments performed by Lynn was screening experiments. From the screening experiment results, Lynn identified hatch overcure, fill overcure, sweep period, z-level wait, layer thickness and part orientation to be the RP variables that have significant effect on part accuracy. A Face Centered Composite design of experiments is used to determine the next set of experiments (main

experiments) used to generate response surface models. Different types of accuracies are studied, which include flatness, parallelism, perpendicularity, concentricity, circularity, and positional tolerance.

West performed surface finish experiments on SLA 250-SOMOS 7110 and developed quantitative models to predict surface finish[20]. The RP variables that affect surface finish are part orientation and layer thickness. By using a similar experimental method, Sambu generated quantitative models for surface finish of the parts built on SLA 3500-SL 7500[106].

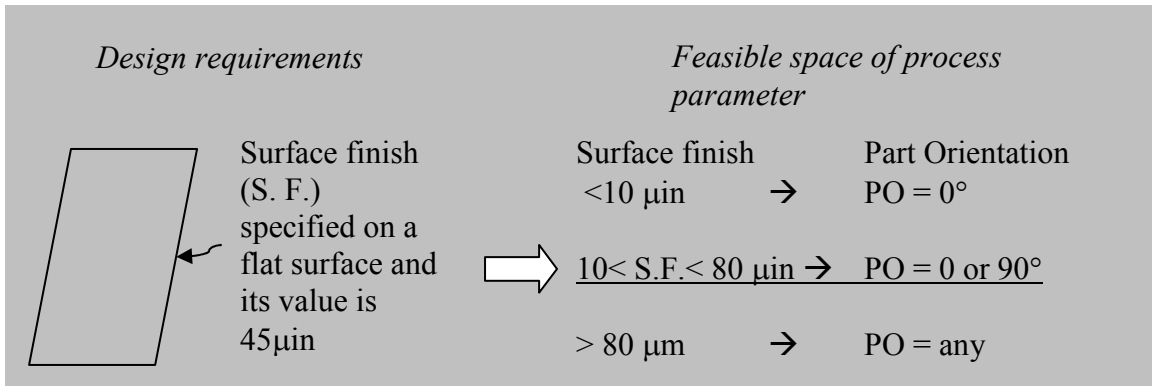
Such models are developed to compute exact values of process variables to fabricate a part such that the part satisfies the initial design requirements.

Such models can not only be used to compute the exact process variables but also, to determine the feasible space of process variables. For example, Figure 3.6 shows the plot of surface finish values versus part orientations for a flat surface[20].



**Figure 3.6 Surface finish model**

In Figure 3.6, the vertical axis represents surface finish and the horizontal axis represents surface orientation values. The three curves correspond to various layer thickness values. For the illustration purpose, the surface finish quality in Figure 3.6 can be classified into three categories; surface finish values less than 10  $\mu\text{in}$ , between 10  $\mu\text{in}$  and 80  $\mu\text{in}$  and over 80  $\mu\text{in}$ . The corresponding part orientation values for satisfying such requirements are  $0^\circ$ ,  $90^\circ$ , and any orientation other than  $0^\circ$  or  $90^\circ$  respectively. In short, the accuracy models shown in Figure 3.6 can be used to identify the feasible space of process variables from the given design requirements. For example, assume that the surface finish requirement is specified on a flat surface and its value is 45  $\mu\text{in}$  as shown in Figure 3.7. To satisfy such design requirements, the surface orientation has to be either  $0^\circ$  or  $90^\circ$ . Therefore, the search space for part orientation is confined to  $0^\circ$  and  $90^\circ$ .



**Figure 3.7 example**

In this research, the goal for utilizing the empirical models is to determine the feasible spaces of process variables, not exact values. Specifically, the feasible spaces to be determined are either infinite or discrete. If discrete, the discrete values of the process variables are of interest. Empirical models are useful for determining the exact values of process variables. To determine the feasible spaces of process variables, the empirical models need to be manually interpreted as shown above. Even after manual interpretation, finding the feasible spaces of process variables that satisfy multiple accuracy requirements requires manual work and computations using the empirical models. To avoid such manual tasks and computations, entities called manufacturing rules are proposed. The manufacturing rules are derived from tabular representations of empirical models such as Figure 3.6. In the tabular form, the accuracy requirement ranges are mapped to feasible spaces of process variables (discrete and continuous). Through manufacturing rules, the accuracy requirements in the design problems are directly related to feasible spaces of process variables. The following paragraphs describe the definition, structure and systematic procedure of derivation of the manufacturing rules.

### 3.3.1.1 Definition, Structure and Derivation

A manufacturing rule is a rule that identifies the feasible space of process variables from the given design requirements. Its structure and examples are shown in Figures 3.8 and 3.9 respectively.

**If (*condition*) then (*result*)**  
where *condition*: design requirements (surfaces with accuracy measurements specification)  
*result*: feasible space of process parameter (MPT)

**Figure 3.8 Structure of manufacturing rule**

- If (Surface finish is specified on flat surface and its value is less than 10  $\mu\text{in}$ ) then (part orientation needs to be  $0^\circ$ )
- If (Surface finish is specified on flat surface and its value is less than 80  $\mu\text{in}$ ) then (part orientation can be  $0$  or  $90^\circ$ )
- If (Surface finish is specified on flat surface and its value is greater than 80  $\mu\text{in}$ ) then (part orientation can be any orientation)

**Figure 3.9 Manufacturing rule example**

As shown in Figure 3.8, the manufacturing rule consists of condition and result parts. The design requirements are specified in the condition part and the corresponding process variable space is specified in the result part. Hence, design requirements are mapped to the corresponding feasible space of process variables through manufacturing rules.

A systematic derivation of manufacturing rules can be accomplished by explicitly representing the relation between process variables and accuracy measurement in a tabular form. An example of such a tabular form is shown in Table 3.3. Table 3.3 is an explicit representation of the relation between surface finish and the feasible space of

process variables (part orientation and layer thickness) presented in Figure 3.6. In Table 3.3, part orientation values including 0°, 45°, 90°, 145°, and 160° with corresponding three layer thicknesses (2, 4, and 8 mils) are selected. Then, the surface finish value for each combination of part orientation and layer thickness is presented. The rows in Table 3.3 are in the ascending order of surface finish value. By arranging the rows in this way, the feasible space of process variables can be easily determined. For example, a surface finish value less than or equal to 70µin can be satisfied by the combinations 90°-4 mils and all above in Table 3.1. A surface finish value less than or equal to 10µin can be satisfied by a part orientation of 0° with any layer thickness.

**Table 3.3 Tabular form of surface finish model**

Surface Orientation	Layer thickness	Surface finish on flat surface
0°	2 mil	3 µin
0°	4 mil	5 µin
0°	8 mil	10 µin
90°	2 mil	30 µin
45°	2 mil	70 µin
90°	4 mil	70 µin
145°	2 mil	75 µin
90°	8 mil	80 µin
160°	2 mil	200 µin
145°	4 mil	300 µin
160°	4 mil	350 µin
45°	4 mil	550 µin
145°	8 mil	800 µin
45°	8 mil	1400 µin
160°	8 mil	1450 µin

Therefore, manufacturing rules can be explicitly and systematically identified through such tabular form. Table 3.3 can be developed for various combinations of accuracy measurement and types of surfaces.

In additive manufacturing, the major hindrance to satisfying the accuracy requirement is the stair stepping effect shown in Figure 1.4[2, 107]. In Figure 1.4, there is no stair stepping effect in the  $0^\circ$  orientation. For the  $90^\circ$  orientation, there are some stair stepping effects and the stair stepping effect is maximized in the sloped orientation. This is exactly what is shown in Figure 3.6. At the  $0^\circ$  orientation, the best surface finish is obtained. At around  $90^\circ$ , the next best surface finish is obtained.

However, there are exceptions because of the layer thicknesses. For instance, combinations of  $45^\circ$ -2 mils and  $145^\circ$ -2 mils produced better surface finish than  $90^\circ$ -8 mils in Table 3.1. This is because surface orientation values other than  $0^\circ$  or  $90^\circ$  with the smallest layer thickness can produce better surface finish than  $90^\circ$  with largest layer thickness. However considering surface finish and various tolerances in general, it is known that the  $0^\circ$  orientation satisfies accuracy requirements better than the  $90^\circ$  orientation and the  $90^\circ$  orientation satisfies better than sloped orientations. The  $0^\circ$  and  $90^\circ$  orientations for the various surfaces are discussed in a later section. Also, a smaller value of layer thickness satisfies accuracy requirements better than a larger value of layer thickness for any fixed surface orientation[2, 20, 22, 23, 107]. Therefore, generalization of Table 3.3 can be made based on the above discussion. A generalized Table 3.3 is shown in Table 3.4

**Table 3.4 Generalized tabular form of accuracy model**

Surface Orientation	Layer Thickness	Accuracy measurement on surface
0°	2 mil	< value 1
	4 mil	< value 2
	8 mil	< value 3
90°	2 mil	< value 4
	4 mil	< value 5
	8 mil	< value 6
0° ± x	2 mil	< value 7
	4 mil	< value 8
	8 mil	< value 9
90° ± y	2 mil	< value 10
	4 mil	< value 11
	8 mil	< value 12
Others	2 mil	< value 13
	4 mil	< value 14
	8 mil	< value 15

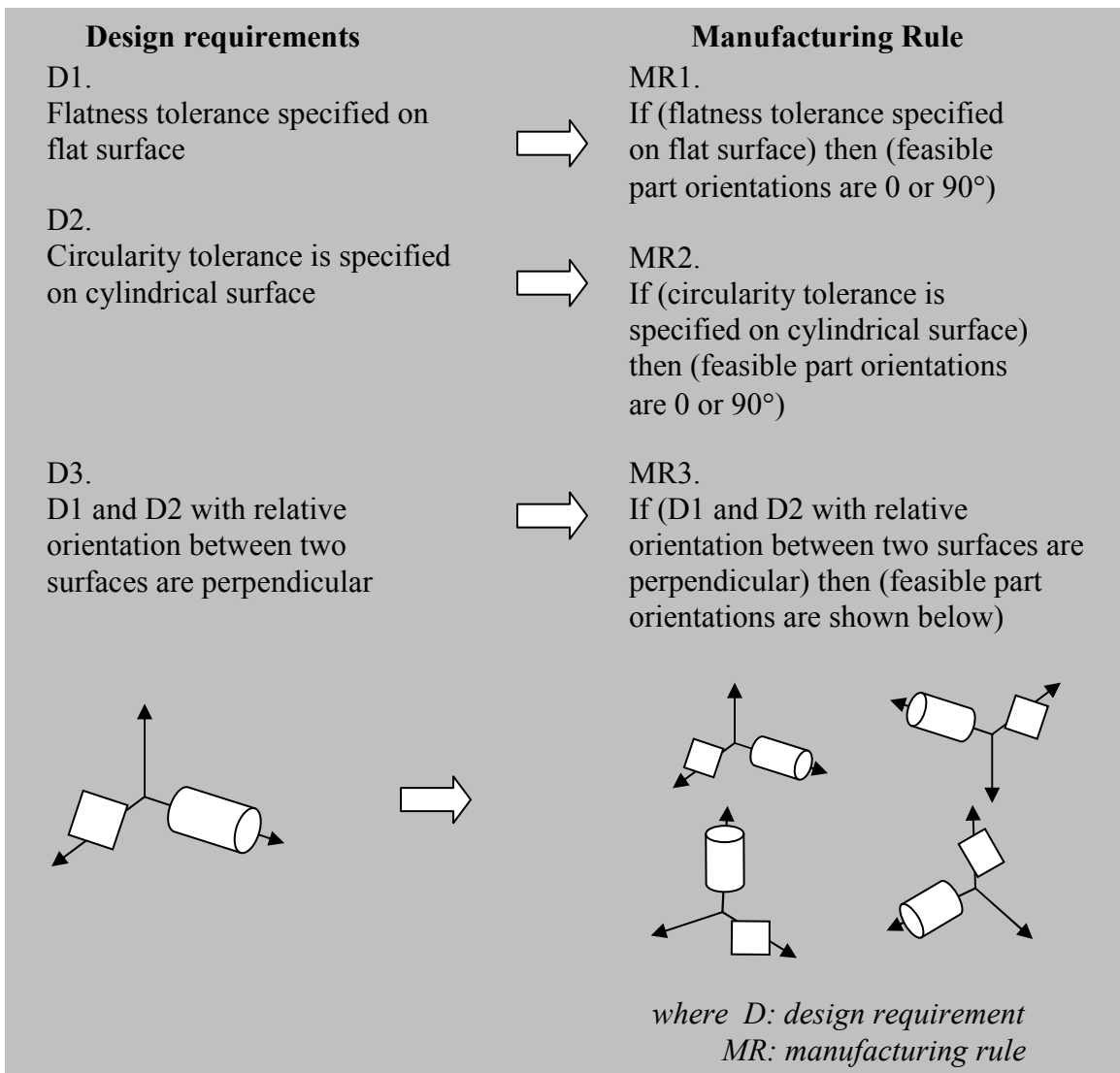
In Table 3.4, the third column should be filled with accuracy measurement value ranges from the empirical models for any specific combination of accuracy measurement and surface type. The surface orientation values  $0^\circ \pm x$  and  $90^\circ \pm y$  represent ranges of angles. For instance,  $0^\circ \pm 5$  means surface orientation ranges from  $-5^\circ$  to  $5^\circ$ . The values in the third column are not in the order of size. In other words, “value 4” could be smaller than “value 3”. Also, “value 7” and “value 10” could be smaller than “value 6” for small values of x and y. In the following section, utilization of generalized Table 3.4 in deriving manufacturing rules is discussed.

### 3.3.1.2 Characteristics

The examples shown in Figure 3.7 are simple manufacturing rules. In other words, each manufacturing rule only describes the feasible space of process variables for a single combination of surface and accuracy requirements.

However, multiple such combinations influence the feasible space of process variables that satisfy all the given design requirements. Consequently, manufacturing

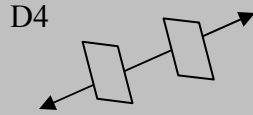
rules for multiple such combinations become more complex. Figure 3.10 shows the examples of such manufacturing rules with corresponding design requirements. In Figure 3.10, simple manufacturing rules are presented first. Then, the derivation of more complex manufacturing rules is shown. In this example, part orientation is considered as a process variable, and its possible values are assumed to be either 0 or 90° for any given design requirements.



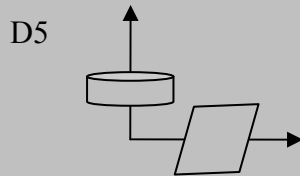
**Figure 3.10 Example of complex manufacturing rules**

As shown in Figure 3.10, the design requirements D1 and D2 can be satisfied by orienting the surface either  $0^\circ$  or  $90^\circ$ . However, satisfying design requirement D3 requires satisfying D1 and D2 where the relative orientation of the two surfaces is perpendicular. Therefore, the overall feasible orientations that satisfy D3 are shown in MR3 in Figure 3.10. The complex manufacturing rules can also be combined to derive more complex manufacturing rules. Figure 3.11 shows an example.

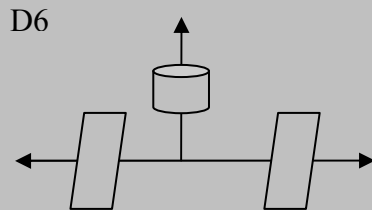
### Design requirements



Flatness tolerances are specified on each of two flat surface that are oriented opposite



Surface finish is specified on a cylindrical surface and flatness tolerance is specified on a flat surface and the relative orientation is perpendicular



Flatness tolerances are specified on each flat surface that are oriented opposite AND Surface finish is specified on cylindrical surface and its relative orientation to flat surfaces is perpendicular

where *PO*: part orientation

*D*: design requirement

*MR*: manufacturing rule

*PO1,PO2,PO3*: values of part orientation

### Manufacturing Rule

MR4.

IF (two flatness tolerance are specified on two flat surfaces) and (their relative orientation is opposite) THEN (possible PO is PO1)

MR5.

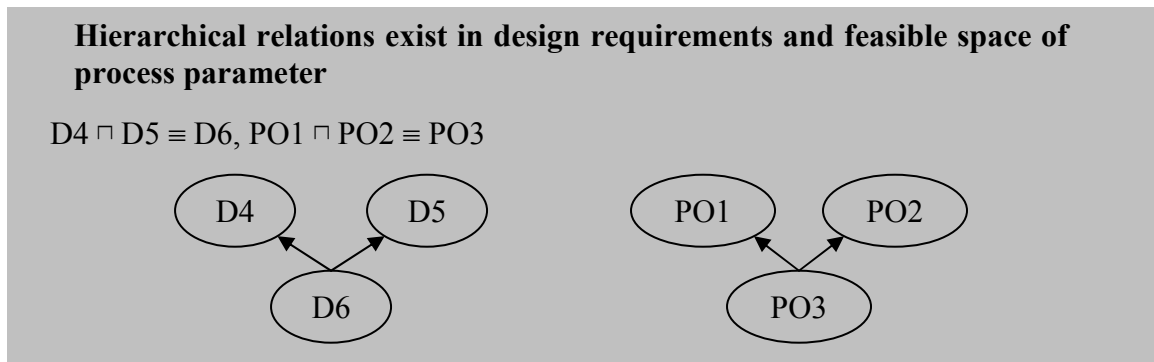
IF (surface finish is specified on cylindrical surface and flatness tolerance is specified on flat surface) and (their relative orientation is perpendicular) THEN (possible PO is PO2)

MR6.

IF ((two flatness tolerance are specified on two flat surfaces each) and (their relative orientation is opposite)) AND ((surface finish is specified on cylindrical surface and flatness tolerance is specified on flat surface) and (their relative orientation is perpendicular)) THEN (possible PO is PO3) WHERE  $PO3 = PO1 \cap PO2$

**Figure 3.11 Example of complex manufacturing rule derivation using complex manufacturing rules**

In Figure 3.11, the design requirement D6 is a combination of D4 and D5. The part orientation values that satisfy D4 and D5 can be identified by an approach that is similar to that used for MR3 in Figure 3.10. The part orientation values are written as PO1 and PO2 in Figure 3.11 for simplicity. Then, the appropriate part orientation for satisfying D6 is the intersection of PO1 and PO2 because the part orientation for D6 needs to satisfy both D4 and D5. In short, the more complex manufacturing rules can be derived by combining simpler ones. Also, their relationships are hierarchical through their condition and result parts as shown in Figures 3.12 and 3.13.



**Figure 3.12 Hierarchical relation in condition and result part**



**Figure 3.13 Hierarchical relation in manufacturing rules**

In this research, two types of manufacturing rules are identified: one is manufacturing rules shown above and the other is meta manufacturing rules. From here on, meta manufacturing rules are referred to as meta rules and manufacturing rules are referred to as rules. Meta rules are abstracted forms of rules in terms of accuracy

measurements and types of surfaces. For example, each row in Table 3.4 represents a rule, and there are such tables for each combination of surface type and accuracy measurement. If the rules that share the same result part (feasible process variables) are collected from various such tables, the corresponding meta rule can be derived. Examples are shown in Tables 3.5 and 3.6. The data in Table 3.5 is arbitrarily chosen for illustration purposes. Table 3.5 shows the tabular form of the accuracy models for three combinations of surface and accuracy measurement; surface finish on flat surface, circularity tolerance on cylindrical surface, and position tolerance on flat surface. If the rules for highlighted parts in Table 3.5 are collected, then a corresponding meta rule can be derived as shown in Table 3.6

**Table 3.5 Example tabular form of accuracy model**

Part Orientation	Layer Thickness	Surface finish on flat surface	Circularity tolerance on cylindrical surface	Position tolerance on conical surface
0°	2 mil	< 1μin	< 4μin	< 5μin
	4 mil	< 2μin	< 8μin	< 10μin
	8 mil	< 3μin	< 12μin	< 15μin
90°	2 mil	< 4μin	< 16μin	< 20μin
	4 mil	< 5μin	< 20μin	< 25μin
	8 mil	< 6μin	< 24μin	< 30μin
Other than 0° or 90°	2 mil	< 7μin	< 28μin	< 35μin
	4 mil	< 8μin	< 32μin	< 40μin
	8 mil	< 9μin	< 36μin	< 45μin

**Table 3.6 Example of meta rule and rule**

Rule	Meta rule	
If (surface finish is specified on flat surface and its value is restricted to be less than 6 $\mu\text{in}$ ) then (part orientation can be either 0° or 90° and layer thickness can be 2, 4, or 8 mils)	If (accuracy measurement is specified on a surface and it is restricted to be less than a value that requires part orientation to be either 0° or 90° and layer thickness to be 2, 4, or 8 mils) then (part orientation can be either 0° or 90° and layer thickness can be 2, 4, or 8 mils)	
If (circularity tolerance is specified on cylindrical surface and its value is restricted to be less than 24 $\mu\text{in}$ ) then (part orientation can be either 0° or 90° and layer thickness can be 2, 4, or 8 mils)		
If (position tolerance is specified on conical surface and its value is restricted to be less than 30 $\mu\text{in}$ ) then (part orientation can be either 0° or 90° and layer thickness can be 2, 4, or 8 mils)		

In Table 3.6, rules corresponding to highlighted part of Table 3.5 are collected on the left column. Their result parts are all the same and their condition parts are all about having accuracy measurement specified on any surface with some restriction level. Therefore, the condition part can be abstracted such that an accuracy measurement is specified on any type of surface, and it can be restricted such that the feasible part orientation is 0° or 90°. Such abstraction is in the condition part of the meta rule shown in the right column. Its result part is the same as the rules in the left column.

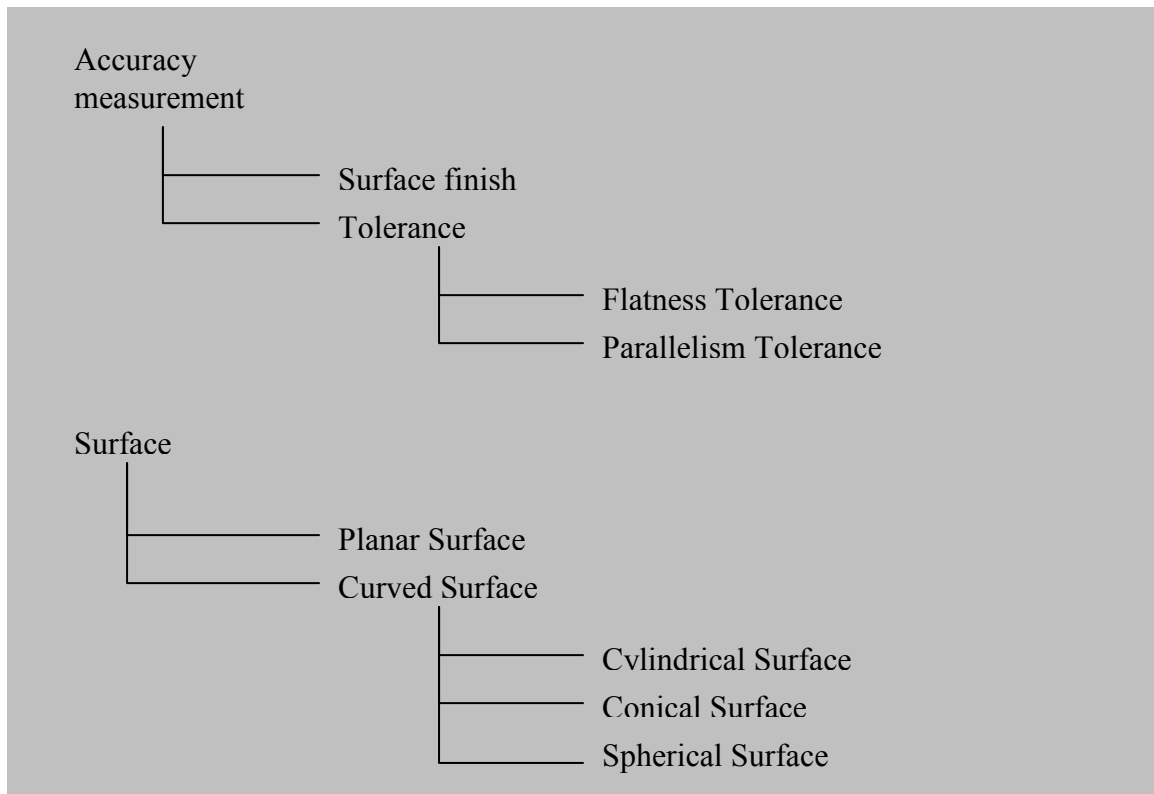
The meta rule shown in Table 3.6 is an example of a simple one. The complex meta rules can be derived by similar processes that derive the complex rules shown in Figures 3.10 and 3.11. Examples of complex meta rules are shown in Table 3.7.

**Table 3.7 Complex meta rule derivation example**

Meta rule	Rule
<u>MR 1</u> IF ((accuracy measurements are specified on two surfaces) and (their relative orientation is opposite) and (accuracy measurement values are restricted to be less than value 1)) THEN (possible PO is PO1)	<u>R 1</u> IF ((two flatness tolerance are specified on two flat surfaces) and (their relative orientation is opposite) and (accuracy measurement is restricted to be less than value 1)) THEN (possible PO is PO1)
	<u>R 2</u> IF ((flatness tolerance is specified on flat surface and parallelism tolerance is specified on flat surface) and (their relative orientation is opposite) and (accuracy measurement is restricted to be less than value 1)) THEN (possible PO is PO1)
<u>MR 2</u> IF ((accuracy measurements are specified on two surfaces) and (their relative orientation is perpendicular) and (accuracy measurement values are restricted to be less than value 1)) THEN (possible PO is PO2)	<u>R 3</u> IF ((surface finish is specified on cylindrical surface and flatness tolerance is specified on flat surface) and (their relative orientation is perpendicular) and (accuracy measurement values are restricted to be less than value 1)) THEN (possible PO is PO2)
	<u>R 4</u> IF ((position tolerance is specified on conical surface and parallelism tolerance is specified on flat surface) and (their relative orientation is perpendicular) and (accuracy measurement values are restricted to be less than value 1)) THEN (possible PO is PO2)
<u>MR 3</u> IF ((accuracy measurements are specified on two surfaces and their relative orientation is opposite) and (accuracy measurements are specified on two surfaces and their relative orientation is perpendicular) and (accuracy measurement values are restricted to be less than value 1)) THEN (possible PO is PO3) WHERE PO3 = PO1 $\cap$ PO2	<u>R 5</u> IF (((two flatness tolerance are specified on two flat surfaces each) and (their relative orientation is opposite)) and ((surface finish is specified on cylindrical surface and flatness tolerance is specified on flat surface) and (their relative orientation is perpendicular)) and (accuracy measurement values are restricted to be less than value 1)) THEN (possible PO is PO3) WHERE PO3 = PO1 $\cap$ PO2
where MR: Meta rule R: Rule, PO: part orientation	

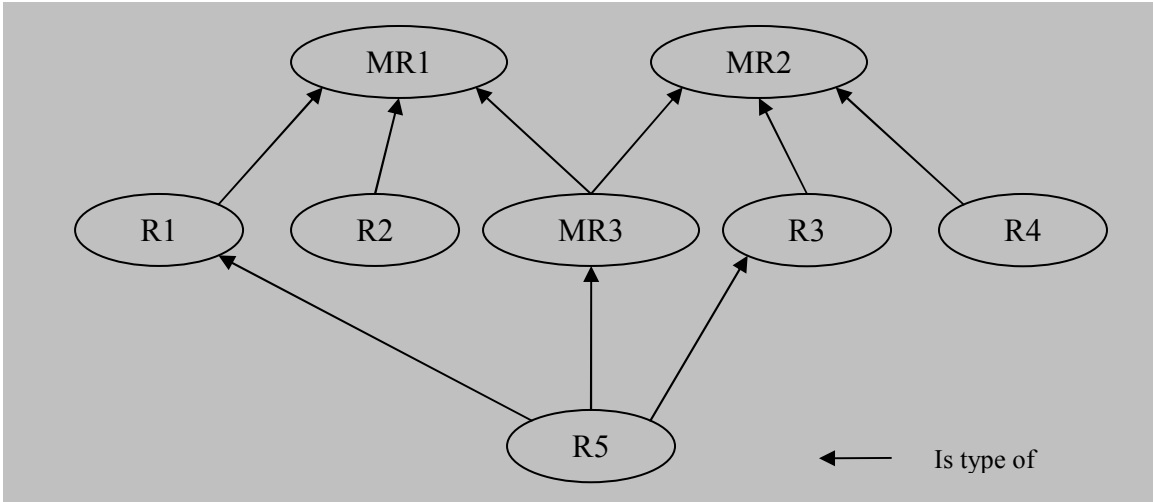
In Table 3.7, part orientation is considered as a process variable. Five rules (R 1~5) are used to demonstrate the derivation of the corresponding meta rules (MR 1~3). As shown in Table 3.7, a more complex meta rule (MR 3) can be derived from simpler ones (MR 1,2) through a similar process as shown in Figures 3.10 and 3.11. To illustrate the relation between meta rules and rules, Figure 3.14 shows the hierarchical nature of

condition part through accuracy measurement and types of surfaces. Figure 3.14 shows two taxonomies; one for accuracy measurement and one for types of surfaces[22].



**Figure 3.14 Terminology taxonomy for meta rule and rule**

The condition part of the meta rule consists of the root of these taxonomies and the relative orientations of surfaces. The condition part of the rules is then, a combination of various types of surface and accuracy measurements. Through the taxonomy in Figure 3.14, a hierarchical relation can be drawn between meta rules and rules such that the meta rule subsumes rules. Also, a hierarchical relation among meta rules can be drawn by similar reasoning in Figures 3.10 and 3.11. The hierarchical relation among meta rules and rules is shown in Figure 3.15. The hierarchy in Figure 3.15 is constructed using the meta rules and rules shown in Table 3.7.



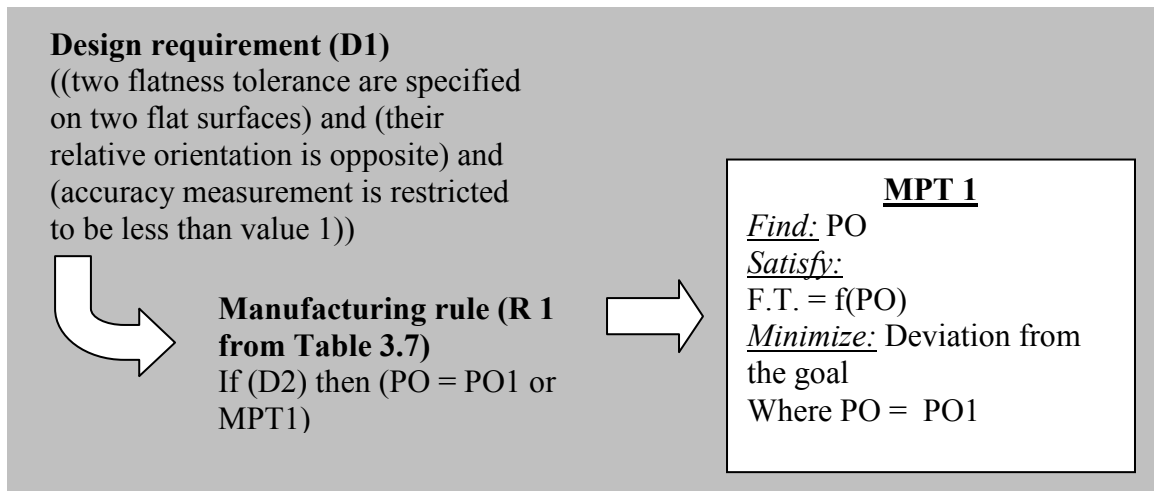
**Figure 3.15 Example of hierarchical relation among meta rule and rule**

From Table 3.7, meta rules 1 and 2 are different through their condition part. Meta rule 3 however, is the intersection of meta rules 1 and 2 through their condition parts. The condition parts of rules 1 and 2 are more specific than that of meta rule 1. Similarly, the condition parts of rules 3 and 4 are more specific than that of meta rule 2 and the condition part of rule 5 is more specific than that of meta rule 3. The condition part of rule 1 is different than that of rule 3; however the condition part of rule 5 is the intersection of those of rules 1 and 3. Figure 3.15 basically summarizes the above relations among manufacturing rules. The following section describes the details of functionalities of manufacturing rule (meta rule and rule) using manufacturing problem templates (MPTs).

### **3.4 The functionalities of manufacturing rules in relating the given design requirements to the relevant DFM problems through appropriate MPTs**

As discussed in chapter 1, relevant DFM problems should provide the appropriate solution strategy and models of accuracy measurements that are required for formulating

and solving a new DFM problem. In this research, the manufacturing rules and their subsumption relation determine the appropriate MPTs for the given design requirements. Then, the DFM problems are classified by the manufacturing rules such that the problems under the same manufacturing rules share the same MPT. The given design requirements are then, related to the relevant DFM problems through manufacturing rules and their subsumption relations. Figure 3.16 shows an example.



**Figure 3.16 Relation between design requirements, manufacturing rules, and manufacturing problem templates**

In Figure 3.16, the rule R1 (from Figure 3.11) is identified as an appropriate rule for D1 by comparing D1 to the condition part of R1. Then, MPT 1 is determined as an appropriate MPT by R1. This is because the feasible spaces of the process variables that R1 determines match MPT 1. Then the DFM problems classified under R1 are retrieved as relevant DFM problems for D1. The retrieved DFM problems possess design requirements D1 and feasible space of part orientation PO1 (share MPT1). Also, the retrieved DFM problems possess accuracy model flatness tolerance on flat surface.

Therefore, the retrieved DFM problems provide a solution strategy corresponding to PO1 and accuracy model for D1.

In this research, DFM problems are classified by rules such that DFM problems under the same rule share the same MPT for process planning. Also, the meta rules and rules are indexed by their condition parts (design requirements). Then, the given design requirements are compared to those indexes to locate relevant DFM problems. During this procedure, meta rules and rules lead the design requirement to those DFM problems that have appropriate solution strategy and to models of accuracy measurements respectively. Also, the subsumption relations among meta rules and rules supports retrieving relevant DFM problems. Hence manufacturing rules and their subsumption relation basically relate the given design requirements to relevant DFM problems through mapping between design and process planning domains (MPTs). These details are discussed in the next chapter using manufacturing rules discovered in Stereolithography.

### **3.5 Summary**

In this chapter, the scope of the research is identified in Table 3.2. Then, the manufacturing problem templates are discussed in section 3.2 within the identified scope of the process planning. In section 3.3, the foundation of the manufacturing rules including origin, definition, structure, derivation, and characteristics are discussed. Through this discussion, the theoretical foundations of the manufacturing rules are established. Finally, the relation between manufacturing problem templates and manufacturing rules is discussed in section 3.4. Through this discussion, the functionalities of manufacturing rules in mapping design and MPTs are discussed.

In Table 3.8, the research questions and hypotheses (1 and 2) are presented.

**Table 3.8 Research questions and hypotheses (1 and 2)**

Questions	Hypotheses
Q1. How should design requirement be represented?	H1. Description logics enable representation of design requirements
Q2. How should the design and process planning domains be mapped?	H2. Subsumption in DL enables mathematical mapping between design domain and process planning domain

Research question 1 and hypothesis 1 are about selecting appropriate description logics for representing design requirements. Hence, identifying the information space in design and process planning contributes toward the theoretical validation of hypothesis 1. Research question 2 and hypothesis 2 are about realizing the mathematical mapping between design and process planning using subsumption in DL. In this research, manufacturing rules and their subsumption relations map the two domains. Hence, establishing the theoretical foundation of the manufacturing rules contributes toward the validation of hypothesis 2.

## **CHAPTER 4: MANUFACTURING RULES DISCOVERY IN STEREOLITHOGRAPHY AND MAPPING BETWEEN DESIGN AND PROCESS PLANNING DOMAINS**

In this chapter, the mapping between design requirements and MPTs in Figure 1.16 is established. First, the discovery of the manufacturing rules is discussed (section 4.1). The manufacturing rules for process variables in Stereolithography are discovered and combined. Then, the task of relating the design requirements to the relevant DFM problems through the manufacturing rules and their subsumption relations is discussed using examples (section 4.2). Through those examples, the mapping between design and process planning domains is realized and formalized (section 4.3). By discovering the manufacturing rules and establishing mapping between the two domains, this chapter contributes toward theoretical validation of hypotheses 1 and 2. Finally, the new knowledge that is contributed toward DFM by establishing a formal mapping between the two domains (section 4.4) is discussed.

### **4.1 Rule discovery in Stereolithography**

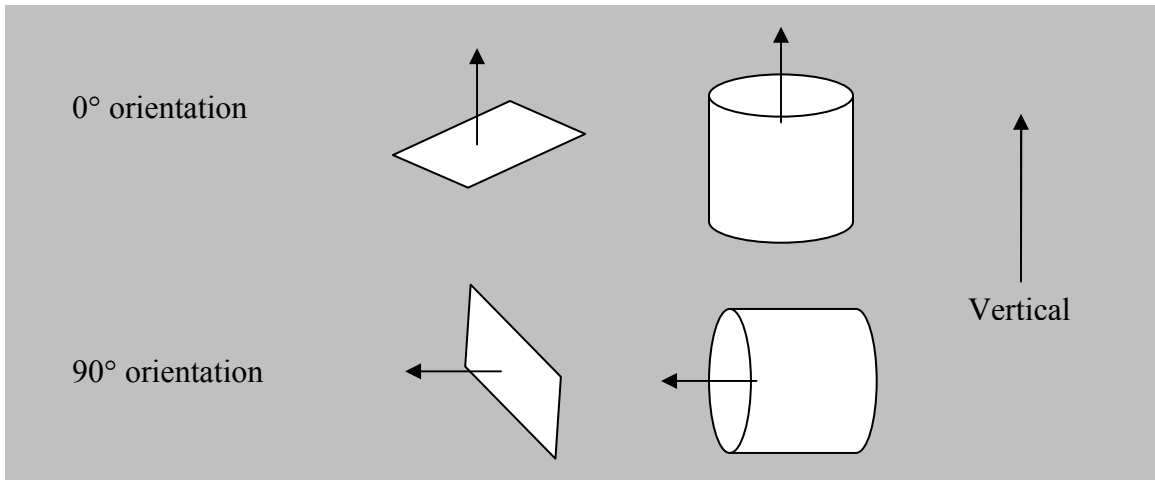
In this research, Stereolithography is chosen for discovering manufacturing rules that apply to Stereolithography as well as to other additive manufacturing processes. As discussed previously, three major process variables in Stereolithography are part orientation, layer thickness, and overcure[2, 107]. Among the three process variables, part orientation is the only continuous variable. The other two variables are typically discrete at the machine level. The following paragraphs describe manufacturing rules collection for each process variable.

For the part orientation and layer thickness, a template for deriving meta rules is developed. Table 4.1 shows the template.

**Table 4.1 Template for meta rule discovery in Stereolithography**

Part orientation	Layer thickness	Accuracy measurement value range
0°	Thin	Value 1
	Thick	Value 2
90°	Thin	Value 3
	Thick	Value 4
0° ± x	Thin	Value 5
	Thick	Value 6
90° ± y	Thin	Value 7
	Thick	Value 8
Others	Thin	Value 9
	Thick	Value 10

Table 4.1 shows feasible values for each process variable. For part orientation, feasible values are chosen to be 0°, 90° and any ranges of orientations including 0° ± x, 90° ± y, and other. For layer thickness, feasible values are chosen to be thin and thick. As discussed previously, a 0° part orientation satisfies tighter accuracy requirement than 90°. Also, a 90° orientation satisfies tighter accuracy measurements than any orientation other than 0° or 90°. Figure 4.1 shows the 0° and 90° orientations of flat and cylindrical surfaces.



**Figure 4.1 0° and 90° orientations for cylindrical and flat surface**


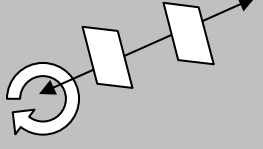

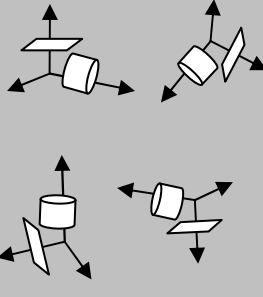

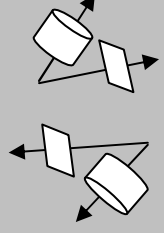
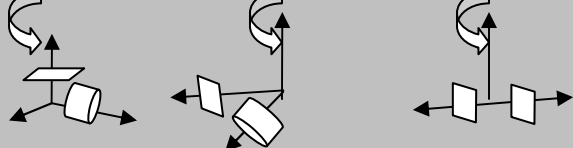
Smaller value of layer thickness satisfies tighter accuracy measurements better than larger values for a fixed surface orientation. In Table 4.1, two layer thicknesses are considered because layer thickness is typically discrete and two layer thicknesses are sufficient to demonstrate manufacturing rule collection. The values in the third column in Table 4.1 are the accuracy measurement values that can be satisfied by corresponding feasible spaces of process variables in the first and second column. “value 3” could be smaller than “value 2” and the “value 5” could be smaller than “value 4”. However, “value 1” is always less than “value 2” and “value 3” is always less than “value 4”. This is because tighter accuracy requirements can be satisfied by smaller layer thickness with the same surface orientation[2, 20, 22]. The feasible space of part orientation is determined by the relative orientations of surfaces where accuracy measurements are specified. The feasible space of layer thickness is determined by the value specified for the accuracy measurements.

In the following sections, the manufacturing rules discovery for part orientation, layer thickness, and overcure are discussed. Then, combination of the manufacturing

rules for three process variables and the generalization of the manufacturing rules are discussed. Finally, the relation between discovered manufacturing rules and MPTs is discussed.

#### **4.1.1 Manufacturing rules discovery for part orientation**

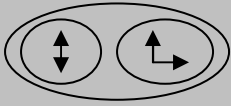
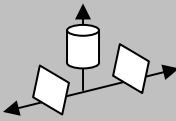
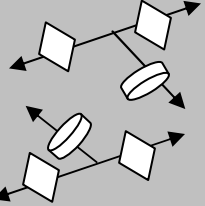
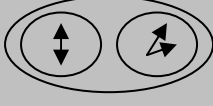
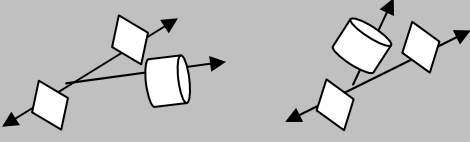
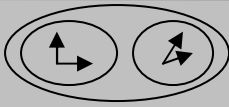
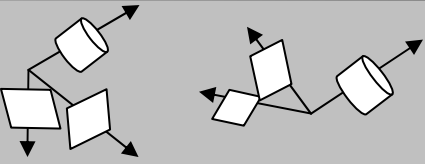
For the discovery of manufacturing rules for part orientation, the values in Table 4.1 are considered to be in the range;  $\text{value } 3 \leq \text{value} \leq \text{value } 4$ . In other words, only  $0^\circ$  or  $90^\circ$  of surface orientation can satisfy any accuracy measurements specified on any type of surface. The values that  $0^\circ$  orientation satisfies are typically lower than the values that  $90^\circ$  orientation can satisfy. Hence, values around 1 and 2 typically demand  $0^\circ$  orientation. In that case, feasible part orientation becomes constant and there can be only one meta rule. However, values 3 and 4 can be satisfied by two orientations ( $0^\circ$  and  $90^\circ$ ); and because of this, there are a lot more manufacturing rules to be discovered. Hence, values 3 and 4 are selected for discovering manufacturing rules. Then, the generalization of the discovered manufacturing rules for the other values is discussed in a later section. The corresponding meta rule becomes “*If (accuracy measurement is specified on surface and its restriction is in the range; value 3  $\leq$  value  $\leq$  value 4) then (feasible part orientations are  $0^\circ$  or  $90^\circ$ )*”. Based on this meta rule, more meta rules can be discovered by combining surfaces with accuracy measurement in various orientations. First, let’s consider two surfaces with accuracy measurements. Figure 4.2 shows the three meta rules that are identified by considering only two surfaces. The following paragraphs describe the details of the discovery.

If	Then
MR1  Accuracy measurements are specified on surfaces and their relative orientation is opposite	 Infinite feasible part orientations (both surfaces are in $90^\circ$ )
MR2  Accuracy measurements are specified on surfaces and their relative orientation is perpendicular	 Four feasible part orientations (one or both surfaces are in $90^\circ$ )
MR3  Accuracy measurements are specified on surfaces and their relative orientation is angled	 Two feasible part orientations (both surfaces are in $90^\circ$ )
Part orientations with respect to vertical axis are considered as single orientation	

**Figure 4.2 Example of meta rule discovery in Stereolithography**

As discussed above, only two orientations are feasible to satisfy the individual accuracy measurement;  $0^\circ$  and  $90^\circ$ . When the surfaces with accuracy measurements are combined in various relative orientations, the overall part orientation needs to be set such that individual surfaces are oriented either  $0^\circ$  or  $90^\circ$ . Due to such restrictions, there are three types of relative orientations that generate distinctive overall feasible part orientations when two surfaces are combined. Those three relative orientations are






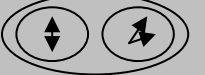











opposite, perpendicular, and angled as shown in Figure 4.2. For opposite orientation (MR1), neither surface can have  $0^\circ$  orientation because the other surface will have  $180^\circ$  orientation. Hence, both surfaces have to be in  $90^\circ$  orientation to satisfy accuracy measurement on both surfaces. This constrains the normal vectors (normal vector for flat surface and axis for cylindrical surface) of the two surfaces to lie in a horizontal plane and generate infinite feasible part orientations by rotating the normal axis. For perpendicular orientation (MR2), either surface can have  $0^\circ$  orientation because the other surface will be in  $90^\circ$  orientation. Also, both surfaces can be in  $90^\circ$  orientation. By adding up all the possible orientations as shown in Figure 4.2, four feasible part orientations are determined. For angled orientation (MR3), neither surface can have  $0^\circ$  orientation because it forces the other surface to be in neither  $0^\circ$  nor  $90^\circ$  orientation. Therefore, the two surfaces have to be oriented in  $90^\circ$ . Consequently, there are two overall orientations that satisfy this condition as shown in Figure 4.2. In this research, the part orientations formed by rotating the part with respect to the vertical axis as shown at the bottom of Figure 4.2 are considered as single orientation. This is because the ways those part orientations influence design requirements or other process variables are the same. More meta rules can be derived by combining the meta rules shown in Figure 4.2. Figure 4.3 shows the examples.

	<b>If</b>		<b>Then</b>
<p>MR4</p>  <p>Accuracy measurements are specified on surfaces and their relative orientation is opposite and perpendicular</p>	 <p>Cylindrical surface in 0°</p>  <p>All three surfaces in 90°</p> <p>Three feasible part orientations</p>		
<p>MR5</p>  <p>Accuracy measurements are specified on surfaces and their relative orientation is opposite and angled</p>	 <p>Two feasible part orientations (all three surfaces are in 90°)</p>		
<p>MR6</p>  <p>Accuracy measurements are specified on surfaces and their relative orientation is perpendicular and angled</p>	 <p>Two feasible part orientations (all three surfaces are in 90°)</p>		

**Figure 4.3 Example of meta rule derivation in Stereolithography**

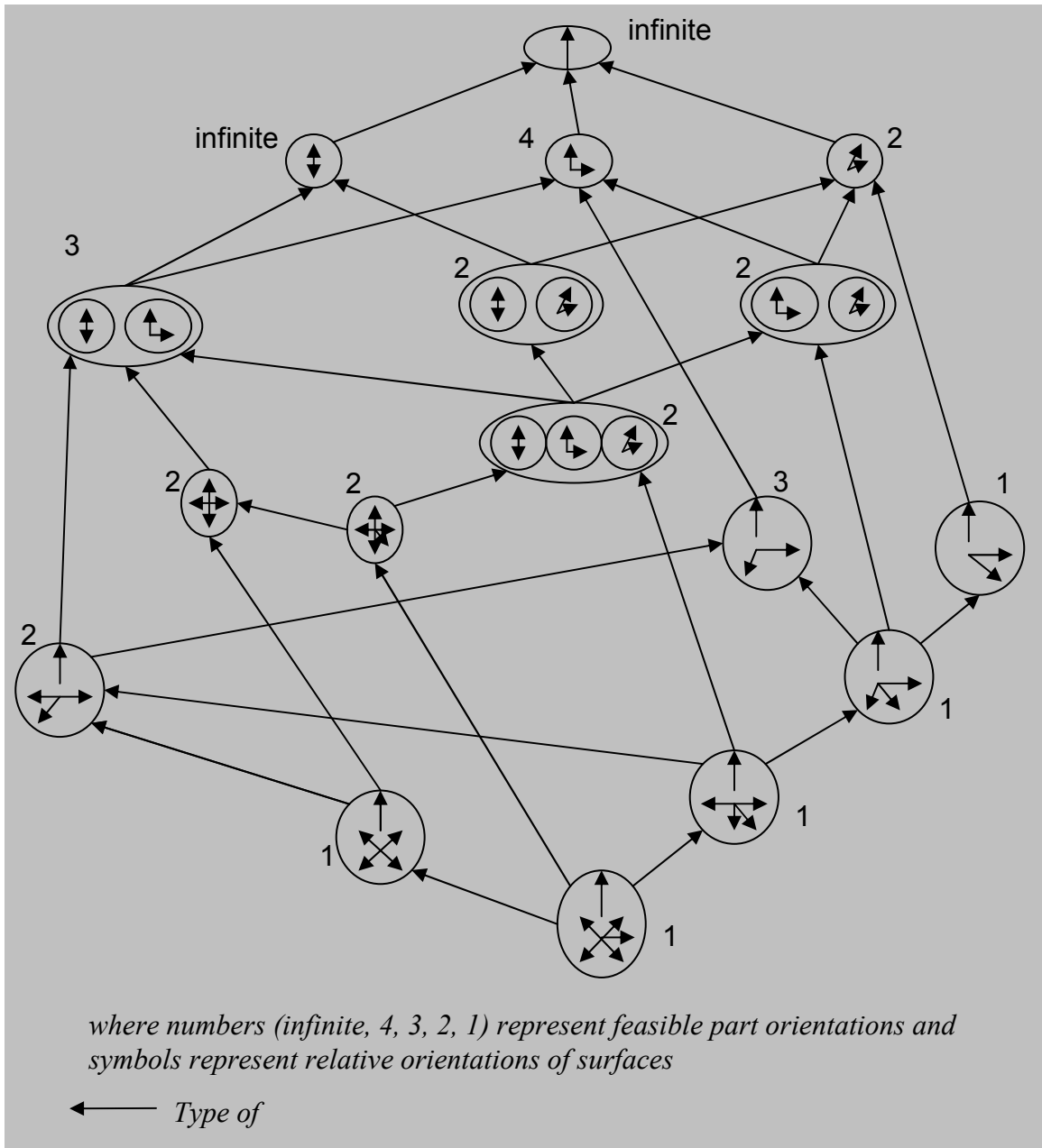
The meta rules in Figure 4.3 are derived for three or more surfaces, and their relative orientations are derived by combining simpler orientations in Figure 4.2. The first row (MR4) in Figure 4.3 shows the meta rule for three or more surfaces having relative orientations opposite and perpendicular. The result part of this row can be derived by similar reasoning in Figure 3.11. This result part is the intersection of the result parts of MR1 and MR2 in Figure 4.2. The explanation for this is that the condition part of MR4 is the combination of condition parts for MR1 and MR2. Hence, the result part of MR4 needs to be the intersection of the result parts of MR1 and MR2 to satisfy the condition parts of both MR1 and MR2. Similarly, MR5 and MR 6 can be derived by

combining the individual condition parts and finding the intersection of the result parts for MR1, MR2, and MR3. Therefore, more and more complex meta rules can be derived by following the above procedure. By following such a procedure, 17 meta rules are discovered and shown in Figures 4.4 and 4.5.

Rule #	Representative symbol	If (accuracy measurements are specified on surfaces and their relative orientations are ...)	Then (number of feasible part orientations are ...)
MR 1		Single surface	Infinite
MR 2		Opposite	Infinite
MR 3		Perpendicular	4
MR 4		Angled	2
MR 5		Opposite, Perpendicular	3
MR 6		Opposite, Angled	2
MR 7		Perpendicular, Angled	2
MR 8		Opposite, Perpendicular, Angled	2
MR 9		Two Opposite, Perpendicular	2
MR10		Two Opposite, Perpendicular, Angled	2
MR11		3D Perpendicular	3
MR12		3D Angled	1
MR13		3D Perpendicular, Angled	1
MR14		3D Opposite Perpendicular	1
MR15		3D Two Opposite	1
MR16		3D Opposite, Perpendicular, Angled	1
MR17		3D Two Opposite, Angled, Perpendicular	1

where relations among number of feasible part orientations are: infinite  $\supset$  4  $\supset$  3  $\supset$  2  $\supset$  1

**Figure 4.4 Meta rules in Stereolithography**

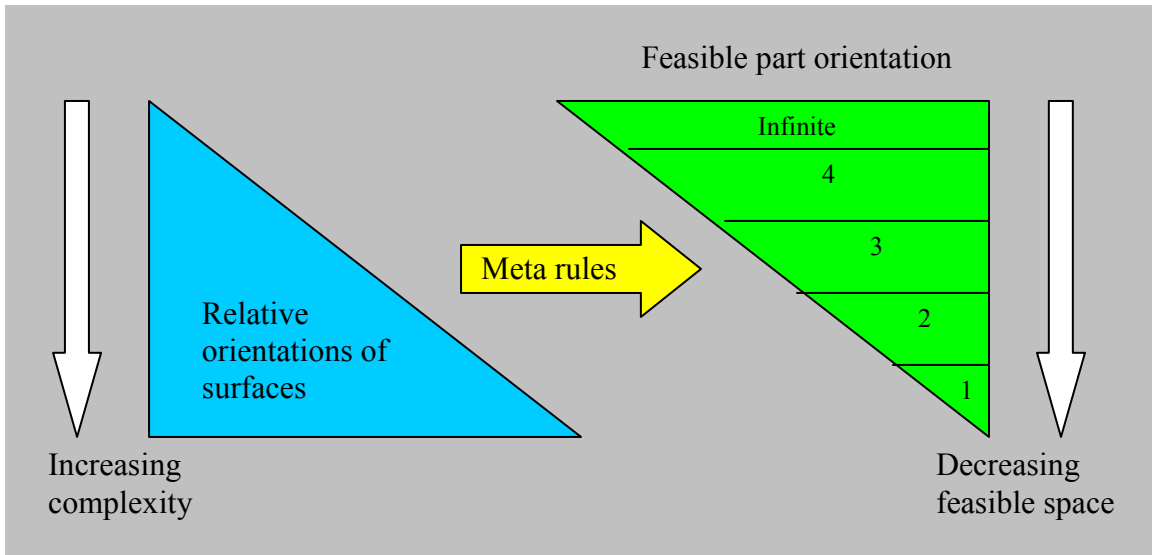


**Figure 4.5 Meta rule taxonomy in Stereolithography**

Figure 4.4 shows the individual meta rules and Figure 4.5 shows the hierarchical relations among them. In Figure 4.4, the second column shows the symbols for the relative orientation characteristics in the design requirement. The third column represents the relative orientations of the surfaces. The fourth column shows the number of feasible

part orientations. As discussed in Figures 3.10, 3.11, 4.2, and 4.3, the number of feasible part orientations represents various types of orientations; there are hierarchical relations among them.

The meta rule taxonomy in Figure 4.5 is constructed based on the hierarchical relations among the condition parts of the meta rules in Figure 4.4. As discussed previously, the hierarchical relations exist among meta rules because their condition parts are formed by various combinations of the three primitive relative orientations; opposite, perpendicular, and angled. In Figure 4.5, the three primitive orientations are combined in 2 and 3 dimensional space to construct various complex relative orientations. The taxonomy is formed such that the top represents the simplest relative orientations and the bottom represents the most complex orientations. Hence, the nodes become more complex as one moves from top to bottom. The numbers next to each node represent the feasible space of part orientation. As one moves from the top of the taxonomy to the bottom, the feasible space of part orientation decreases. In the taxonomy, the feasible space of part orientation changes from infinite to discrete (4, 3, and 2) and discrete to constant (1). The corresponding solution strategies are discussed in Figure 1.7. Therefore, the meta rules shown in Figures 4.4 and 4.5 are capable of relating design requirements to DFM problems with various solution strategies, including a solution strategy for continuous or discrete process variables. A graphical illustration of this is shown in Figure 4.6.



**Figure 4.6 Mapping design space and feasible space of part orientation**

In Figure 4.6, there are two triangles. The triangles on the left and right represent design requirement and feasible space of part orientation, respectively. As one moves downward on the left triangle, the design requirements become complex. As one moves down on the right triangle, the feasible space of part orientation decreases. Then the manufacturing rules are a mapping between the two triangles. Figure 4.6 is another view of the meta rule taxonomy in Figure 4.5 that illustrates the role of manufacturing rules between the design requirements and the feasible space of part orientation. So far the meta rule discovery for part orientation is discussed. The following paragraphs describe the meta rule discovery for layer thickness.

#### **4.1.2 Manufacturing rules discovery for layer thickness**

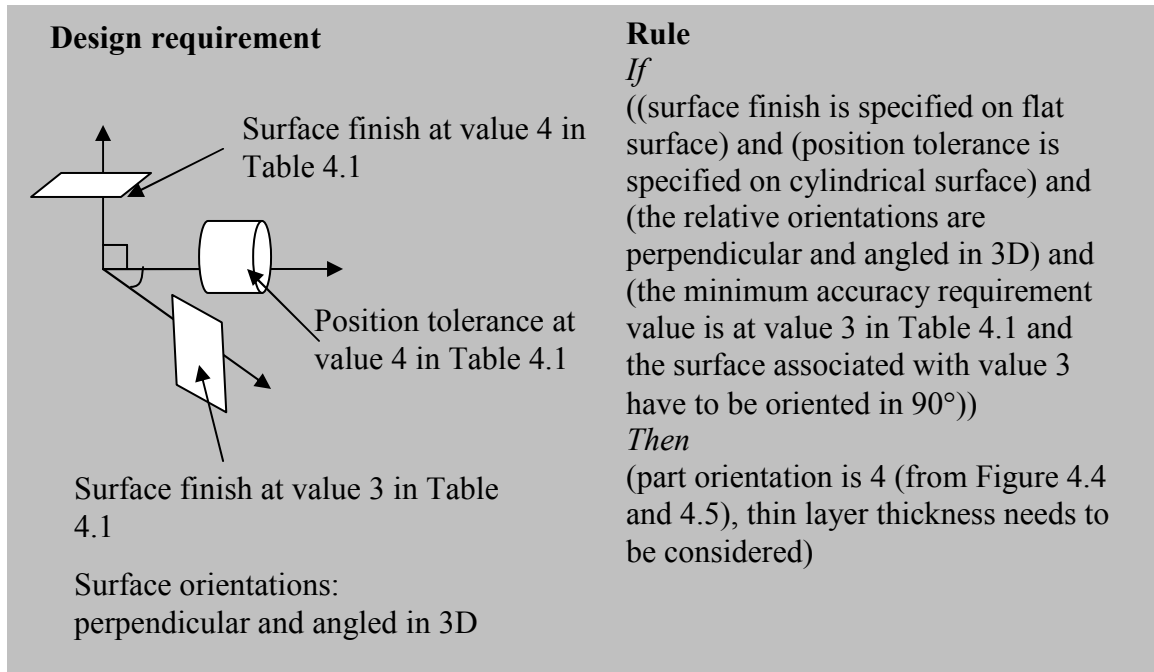
Meta rules for layer thickness determine the minimum layer thickness that needs to be considered during process planning rather than minimum feasible spaces. This is because of two reasons. First, values for layer thickness are determined by the severity of accuracy requirement values rather than the complexity of the design requirements. For

instance, selecting accuracy measurement value on any surface to be value 3 in Table 4.1 results in small layer thickness if that surface has to be in  $90^\circ$ . Secondly, the determined minimum layer thickness does not have to be used for the entire part fabrication [10, 16, 17, 20]. The layer thickness can be still variable depending on the geometry of the part and other reasons such as cost, time, etc. Therefore, meta rules discovered for layer thickness can determine minimum layer thickness to be considered but not the feasible spaces. Based on the above discussions, two meta rules are discovered using Table 4.1. They are shown in Figure 4.7.

- (a). If (there is a surface that has to be oriented in  $90^\circ$  and accuracy measurement value is at value 3 in Table 4.1) then (thin layer thickness need to be considered)
- (b). Otherwise, layer thickness is unknown (discrete)

**Figure 4.7 Meta rules for layer thickness and overcure**

The meta rules in Figure 4.7 determine the minimum layer thickness that needs to be considered for process planning from the accuracy measurement value and surface orientation. This makes sense because part orientation needs to be determined first to determine the appropriate layer thickness in additive manufacturing process planning. The meta rules in Figure 4.7 basically show that small value of layer thickness needs to be considered if there are surfaces that have to be in  $90^\circ$  with “value 3”. Otherwise, the layer thickness is unknown. To illustrate this better, an example is discussed in Figure 4.8.



**Figure 4.8 Example of meta rules for part orientation and layer thickness**

The example in Figure 4.8 demonstrates a case where the minimum layer thickness can be identified. In the example, surface finish is specified on two flat surfaces and position tolerance is specified on the cylindrical surface. Due to the relative orientations of the surfaces, the flat surface with surface finish value 3 has to be in 90° orientation (meta rule 11 in Figure 4.4). Thus, a small layer thickness needs to be considered according to meta rule (a) in Figure 4.7.

#### 4.1.3 Manufacturing rules discovery for overcure

For the overcure, there is no generally proven fact that thinner overcure can satisfy accuracy requirements better than thicker overcure. Instead of thin or thick, there is usually an optimal value of overcure that satisfies the given accuracy requirements[22, 23]. Such optimal values are typically determined through the empirical models that relate the accuracy measurement value to the part orientation, layer thickness, and

overcure. However, the relations between overcure and accuracy measurements are highly non-linear and are not very well established. Therefore, it is extremely difficult to discover manufacturing rules at the current stage of research in additive manufacturing. In this research, two meta rules for the overcure are discovered and shown in Figure 4.9.

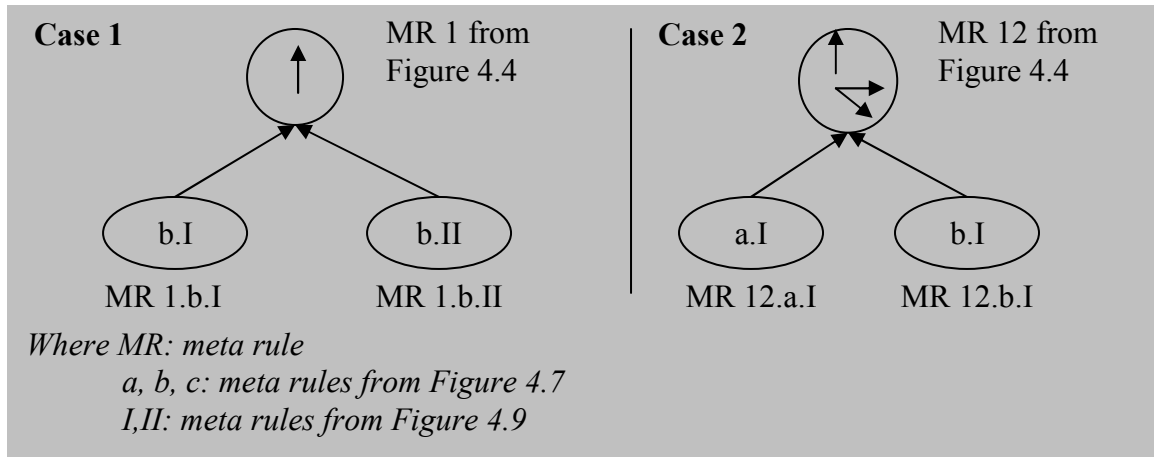
- I. If (there are surfaces other than flat surface in  $0^\circ$  orientation) then (overcure needs to be considered as process parameter for satisfying the given accuracy requirements)
- II. Otherwise, overcure is unknown

#### **Figure 4.9 Meta rule for overcure**

In Figure 4.9, two meta rules (meta rule I and II) for overcure are introduced. Overcure needs to be considered as a process variable when the design requirements call for surfaces other than flat surface in the  $0^\circ$  orientation. This is because overcure influences all the accuracy requirements achievement with surfaces other than flat surface in the  $0^\circ$  orientation. Hence, the meta rules determine whether overcure needs to be considered as a process variable or not based on the above. Compared to the part orientation and layer thickness, the influence that overcure has on achieving accuracy requirements is minimal[22, 23].

#### **4.1.4 Combining manufacturing rules for different process variables**

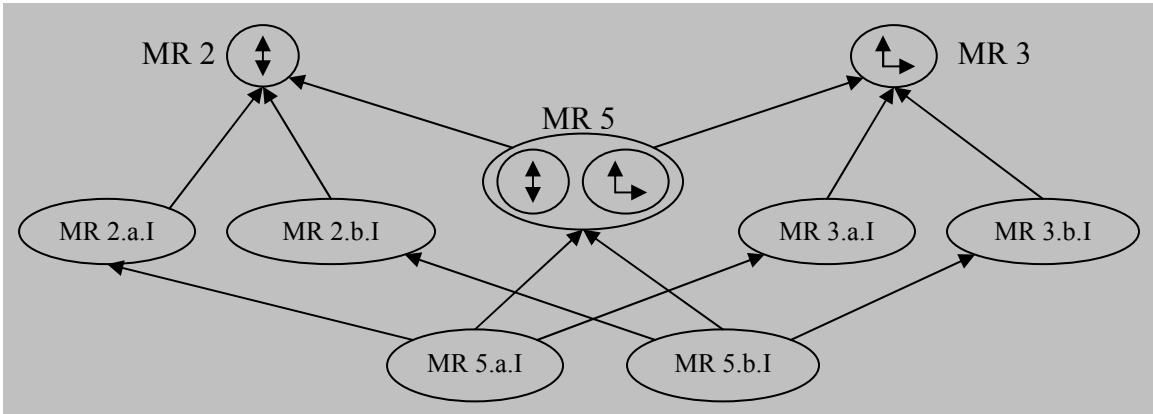
In this research, the meta rules in Figures 4.7 and 4.9 are used to supplement the meta rule taxonomy in Figure 4.5. For each node in Figure 4.5, the appropriate meta rules in Figure 4.7 and 4.9 are attached. Figure 4.10 shows an example.



**Figure 4.10 Example of supplementing the meta rule**

In Figure 4.10, two representative cases of supplemented meta rules are shown. In both cases, the meta rules in Figures 4.7 and 4.9 are combined (a.I, a.II, b.I, b.II) and selectively attached to meta rules in Figure 4.5. The combined supplementary meta rules including a.I, a.II, b.I, and b.II are realized by combining meta rules in Figures 4.7 and 4.9. For instance, the condition part of a.I is formed by adding the condition part of meta rule “a” from Figure 4.7 and the condition part of “I” from Figure 4.9. Hence, the condition part of a.I is “(there is a surface that has to be oriented in  $90^\circ$  and accuracy measurement value is at value 3 in Table 4.1) and (there are surfaces other than flat surface in  $0^\circ$  orientation)”. Among the four possible supplementary meta rules, only three (a.I, b.I, and b.II) are considered in Figure 4.10. This is because meta rules from Figures 4.7 and 4.9 are supplemented to the meta rules in Figure 4.5 selectively. For example, the supplementary meta rule (a) in Figure 4.7 cannot be supplemented to MR 1 in case 1. This is because there is only a single surface orientation in MR 1 and the surface can be either in  $0^\circ$  or  $90^\circ$  orientation to satisfy accuracy requirements. Hence, the meta rule (a) in Figure 4.7 is not applicable. The combined meta rules b.I and b.II are

supplemented to MR 1. In case 2, the meta rule “II” in Figure 4.9 cannot be supplemented because there are always surfaces that are in 90° orientation. Hence, meta rule “II” in Figure 4.9 is not applicable in case 2. The combined meta rules a.I and b.I are supplemented to realize MR 12.a.I and 12.b.I. All the meta rules in Figure 4.4 belong to case 2 except MR 1. Figure 4.11 shows an example of supplemented meta rule taxonomy.



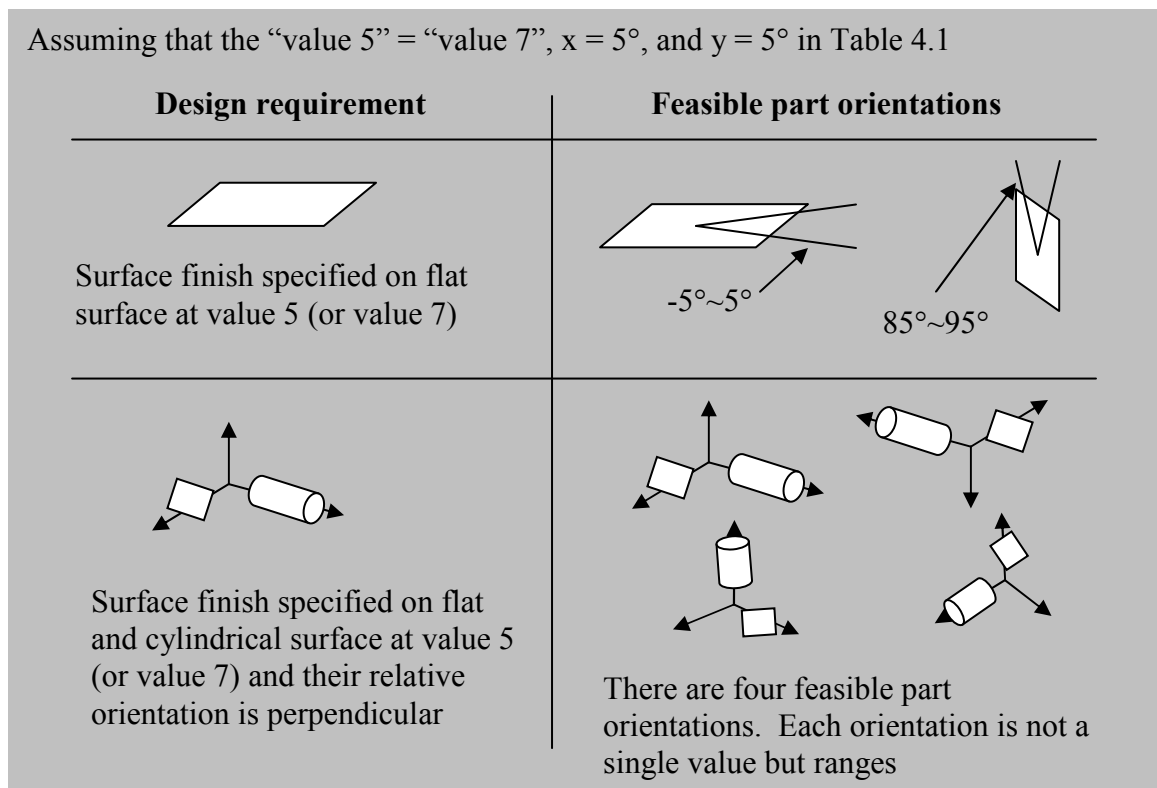
**Figure 4.11 Example of supplemented meta rule taxonomy**

In Figure 4.11, supplementary meta rules are shown for MR 2, 3, and 5. The meta rules MR 2, 3, and 5 are from Figure 4.4. As discussed before, each node of meta rule MR x is supplemented by supplementary meta rules to realize supplemented meta rules MR.x.y.I where x ranges from 1 to 17 and y ranges from a to b. Among the supplemented meta rules, there are hierarchical relations (among MR x.a.I and among MR x.b.I). This is because the hierarchical relations among MR x.a.I and x.b.I are formed by condition and result parts of MR x. For example, MR 2.a.I subsumes MR 5.a.I due to relative orientation in the condition part and feasible space of part orientation in the result part. Similarly, MR 2 subsumes MR 5. The supplemented meta rules determine the feasible spaces of part orientation, the minimum value for layer thickness

and whether overcure needs to be considered or not. Complete supplemented meta rules are developed in  $\mathcal{AL}\mathcal{E}$  and in the Appendix A.

#### 4.1.5 Manufacturing rules generalization to values other than value 3 and 4

So far, the meta rules are discovered for values 3 and 4 in Table 4.1. However, those meta rules can be generalized to values 5~8 in Table 4.1. Figure 4.12 shows an example.



**Figure 4.12 Example of meta rules generalization for values 5~8**

In Figure 4.12, the values 5 and 7 in Table 4.1 are assumed to be the same. Also,  $x$  and  $y$  are assumed to be  $5^\circ$ . Then, the first design requirement shows that the surface finish is specified on a flat surface at “value 5”. The corresponding feasible surface orientations are ranges including  $-5^\circ \sim 5^\circ$  and  $85^\circ \sim 95^\circ$  instead of  $0^\circ$  and  $90^\circ$ . The second

design requirement shows that the surface finish is specified on the flat and cylindrical surfaces where their values are “value 5”. Also, the relative orientation of the surfaces is perpendicular. Then, the corresponding feasible part orientations are four ranges instead of four discrete values because accuracy requirement on each surface can be satisfied by ranges of surface orientation rather than single value. Then, the corresponding meta rule for the second design requirement becomes “*If ((accuracy measurements are specified on flat and cylindrical surfaces at value 5) and (the relative orientations of surfaces is perpendicular)) then (feasible part orientations are four ranges)*”. By the similar procedure, the meta rules for the values 5~8 are derived and it is found that the meta rules for values 5~8 are the same as the meta rules for values 3~4 except the result part. As shown in Figure 4.12, the result part of the meta rules for values 5~8 are ranges rather than discrete (4 ranges versus 4). Hence, the result parts of the meta rules for values 5~8 are infinite, 4 ranges, 3 ranges, 2 ranges and 1 range instead of infinite, 4, 3, 2, and 1. The feasible space of part orientations that are determined by the meta rules for values 5~8 are all infinite (ranges). In short, the derivation of meta rules for values 5~8 in Table 4.1 is basically the same as derivation for values 3~4. Also, derivation of the meta rules for more x and y values (ex: 5, 10, 15, etc) are basically the same. Hence, only the meta rules for values 3~4 are discussed in this research.

#### **4.1.6 Relation between the discovered manufacturing rules and MPTs**

As discussed previously, the manufacturing rules are used to classify DFM problems such that problems under the same rule share the same MPT for process planning. In this research, there are 20 MPTs identified. The 20 MPTs correspond to five feasible values of part orientation (infinite, 4, 3, 2, 1), two values of layer thickness

(thin, thick), and two cases of overcure (part of the process planning for satisfying accuracy requirements or not). However, 12 out of 20 MPTs are utilized in this research because the discovered meta rules only determine feasible spaces of process variables for 12 MPTs. The meta rules MR 2.x.I ~ 17.x.I where x ranges from a to b determine overcure to be part of the process variables for satisfying accuracy requirements. Hence, those meta rules determine 10 (5 part orientation, 2 layer thickness, and single overcure) MPTs. Also, the meta rules MR 1.b.I and 1.b.II determine part orientation to be infinite, layer thickness to be always unknown, and overcure to be either part of the process planning for satisfying the given accuracy requirements or not. Hence, those meta rules determine 2 MPTs. Table 4.2 presents the relation between the meta rules and MPTs. In Table 4.2, the feasible spaces of process variables for each MPT are shown.

**Table 4.2 Relation between meta rules and MPTs**

Meta rule	MPT
MR 2.(a or b).I	MPT 1.1 and 1.2 PO: infinite LT: For MR x.a.I, LT = thin (MPT 1.1), For MR x.b.I, LT = thick (MPT 1.2) OC: part of the process planning for satisfying accuracy requirements
MR 3.(a or b).I	MPT 2.1 and 2.2 PO: 4 LT: For MR x.a.I, LT = thin (MPT 2.1), For MR x.b.I, LT = thick (MPT 2.2) OC: part of the process planning for satisfying accuracy requirements
MR (5,11).(a or b).I	MPT 3.1 and 3.2 PO: 3 LT: For MR x.a.I, LT = thin (MPT 3.1), For MR x.b.I, LT = thick (MPT 3.2) OC: part of the process planning for satisfying accuracy requirements
MR (4, 6, 7, 8, 9, 10).(a or b).I	MPT 4.1 and 4.2 PO: 2 LT: For MR x.a.I, LT = thin (MPT 4.1), For MR x.b.I, LT = thick (MPT 4.2) OC: part of the process planning for satisfying accuracy requirements
MR (12, 13, 15, 16, 17).(a or b).I	MPT 5.1 and 5.2 PO: 1 LT: For MR x.a.I, LT = thin (MPT 5.1), For MR x.b.I, LT = thick (MPT 5.2) OC: part of the process planning for satisfying accuracy requirements
MR 1.b.(I or II)	MPT 6.1 and 6.2 PO: infinite LT: unknown OC: For MR 1.b.I, OC = part of the process planning for satisfying given accuracy requirements (MPT 6.1), for MR 1.b.II, OC = not part of the process planning for satisfying given accuracy requirements (MPT 6.2)
<p><i>Where MR x.y.z: meta rule (x ranges from 1 to 17 according to Figure 4.4, y ranges from a to b according to Figure 4.7, z ranges from I to II according to Figure 4.9)</i>  <i>MPT: manufacturing problem template, PO: part orientation, LT: layer thickness, OC: overcure</i></p>	

In Table 4.2, 34 meta rules are related to 12 MPTs. As discussed previously, the relations are identified by the feasible spaces of process variables that meta rules determine. For example, MPT 3.1 is related to the meta rule MR 5.a.I because MPT 3.1 and MR 5.a.I determine the same feasible spaces of process variables. There are multiple meta rules that determine the same MPT. For example, MR (4, 6, 7, 8, 9, 10).a.I are related to MPT 4.1. This is because cardinality of relation between design requirements to MPT is many to one. Also, there are subsumption relations among MPTs. The MPT  $x.y$  with lower  $x$  subsumes higher  $x$  for  $x$  in 1 to 5. For example MPT 4.y subsumes 5.y where  $y$  is the same. This is because the feasible space of part orientation of MPT 4.x ( $PO = 2$ ) subsumes that of MPT 5.x ( $PO = 1$ ) where values of LT and OC are the same in MPT 4.x and 5.x. The significance of this hierarchical relation is that the MPT  $x_1.y$  is applicable to the design requirement that require MPT  $x_2.y$  where  $x_1 < x_2$ . For instance, assume that the appropriate MPT for the given design requirement is MPT 4.1. Then, MPT 1.1, 2.1, and 3.1 are all applicable to the given design requirement. This is because the space of process variables that MPT 1.1, 2.1, and 3.1 searches includes the space of process variables that MPT 4.1 searches. This characteristic is useful in relating the given design requirements to the relevant DFM problems. More detailed discussions are presented in the next section. The relation between rules and MPTs is not discussed because rules do not further determine feasible spaces of process variables. The following section describes the detailed functionalities of meta rules, rules and manufacturing rules taxonomy.

## **4.2 Relating design requirements to relevant DFM problems**

In this section, two examples are used to illustrate the functionalities of manufacturing rules and their hierarchical relations. The first example illustrates the functionalities of meta rules and rules and the second example illustrates functionalities of the manufacturing rule hierarchy. Through these examples, the ability of the manufacturing rules and their subsumption relations in mapping the design and process planning domains is discussed.

The first example is shown in Figure 4.13. In Figure 4.13, two rules and a meta rule are developed such that the meta rule subsumes the two rules through their condition part. Their hierarchical relation is also shown. Under each rule, it is assumed that there are DFM problems whose design requirements match the condition part of the corresponding rule. Finally, there is a given design requirement. The following paragraphs describe how the given design requirements are related to relevant DFM problems through meta rules and rules.

**Given design requirement:** (Circularity tolerance is specified on conical surface and flatness tolerance is specified on flat surface) and (their relative orientation is perpendicular)

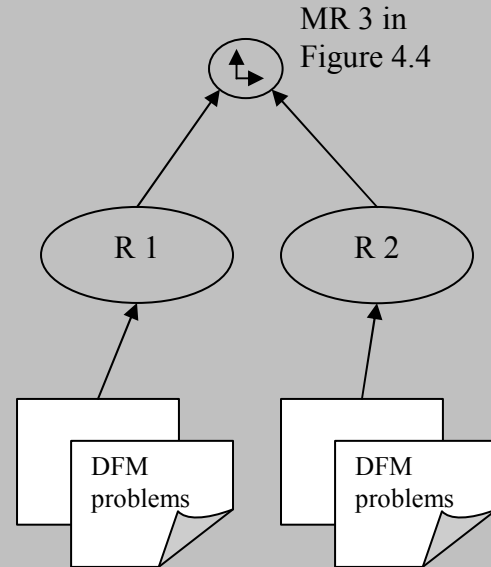
**R 1:**

**If** ((Position tolerance is specified on cylindrical surface and flatness tolerance is specified on flat surface) and (their relative orientation is perpendicular))  
**then** (four feasible part orientations are available)

**R 2:**

**If** ((Circularity tolerance is specified on conical surface and flatness tolerance is specified on flat surface) and (their relative orientation is perpendicular))  
**then** (four feasible part orientations are available)

where R: rule, MR: meta rule



**Figure 4.13 Illustration of meta rule and rules functionality**

As previously discussed, meta rules and rules determine the feasible space of process variables. The difference is that the rules have more specific information in their condition parts in terms of accuracy measurements and surface types. In this research, the meta rules classify rules and rules classify DFM problems. The rules are classified such that rules under a meta rule share the same result part and their condition part is more specific than that of the meta rule in terms of accuracy measurements and surface types. The DFM problems are classified under rules such that their design requirements match those of the rules. An example is shown in Figure 4.13. In this way, DFM problems under same meta rule share the same solution strategy and DFM problems under a rule share the same accuracy measurement models. For example, the given

design requirements in Figure 4.13 exactly match the condition part of R2. Hence, DFM problems under R2 provide solution strategy and accuracy models needed for formulating and solving a new DFM problem for the given design requirements. The DFM problems under R1 can provide part of the models of accuracy measurement needed; however, they provide an appropriate solution strategy. Therefore, DFM problems under R1 are also relevant DFM problems.

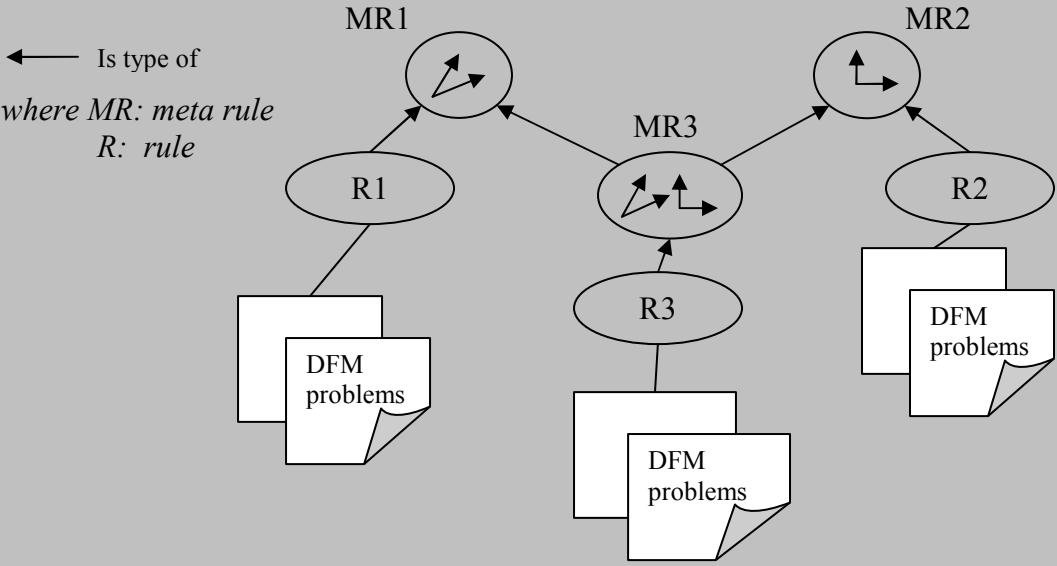
In short, meta rules lead design requirements to DFM problems that have appropriate solution strategies and rules lead design requirements further to the DFM problems that have appropriate models of accuracy measurements. The following example discusses the functionalities of the hierarchical relations among meta rules in relating design requirement to DFM problems.

Figure 4.14 shows an example that illustrates the functionalities of this meta rule hierarchy. In Figure 4.14, three meta rules (MR1, 2, and 3), three rules (R1, 2, and 3), and the given design requirements are presented. The hierarchical relations among the manufacturing rules are also shown. Meta rules and rules are developed such that each rule 1, 2, and 3 is subsumed by each meta rule 1, 2, and 3, respectively. Then, it is assumed that there are DFM problems under each rule such that their design requirements match the condition parts of the rules. The following paragraphs describe the details of functionality of meta rule hierarchy.

**Given design requirement:** ((surface finish and flatness tolerance is specified on flat surface) and (position and parallelism tolerance is specified on cylindrical surface) and (relative orientation is angled and perpendicular) and (3 <value <4 in Table 4.1 ))

**Manufacturing rule taxonomy:**

- MR1: If ((relative orientation is angled) and (3 <value <4 in Table 4.1)) then (2 feasible part orientations are available)
- MR2: If ((relative orientation is perpendicular) and (3 <value <4 in Table 4.1)) then (4 feasible part orientations are available)
- MR3: If ((relative orientation is angled and perpendicular) and (3 <value <4 in Table 4.1)) then (2 feasible part orientations are available)
- R1: If ((surface finish is specified on flat surface) and (position tolerance is specified on cylindrical surface) and (relative orientation is angled) and (3 <value <4 in Table 4.1 )) then (2 feasible part orientations are available)
- R2: If ((flatness tolerance is specified on flat surface) and (circularity tolerance is specified on conical surface) and (relative orientation is angled) and (3 <value <4 in Table 4.1)) then (4 part orientations are available)
- R3: If ((surface finish and flatness tolerance is specified on flat surface) and (position and parallelism tolerance is specified on cylindrical surface) and (relative orientation is angled and perpendicular) and (3 <value <4 in Table 4.1)) then (2 feasible part orientations are available)



**Figure 4.14 Example for illustrating functionality of meta rule hierarchy**

In this research, we attempt to develop a method that identifies the following items from the given design requirements.

- Part or all of the given design requirements that influence the feasible space of process variables
- Feasible space of process variables
- Solution strategy
- Models of accuracy measurements
- Examples (DFM problems)

The goal is not just to provide examples (DFM problems) but also to provide underlying reasons for formulating and solving the DFM problem in a particular way. Therefore, the relevant DFM problem must address the given design requirement. In case there are no DFM problems that address the given design requirement entirely, the DFM problems that address the requirement partly should be identified. During the research, it is found that DFM problems that address part of the given design requirement can still provide appropriate solution strategies and accuracy models.

In Figure 4.14, the appropriate manufacturing rules (MR3 and R3) for the given design requirements can be detected by comparing the given design requirements to the condition part of the manufacturing rules. Then, MR3 leads the given design requirement to those DFM problems that yield appropriate solution strategies and R3 leads to those DFM problems that have required accuracy models. Hence, the DFM problems under R3 provide an appropriate solution strategy and accuracy models. Now, let's assume that there are no DFM problems under R3 in Figure 4.14. Then, the given design requirements can be broken down into pieces that match the condition part of MR1 and MR2. The DFM problems under R1 and R2 may not provide all the accuracy models needed; however, they provide an appropriate solution strategy. The appropriate solution

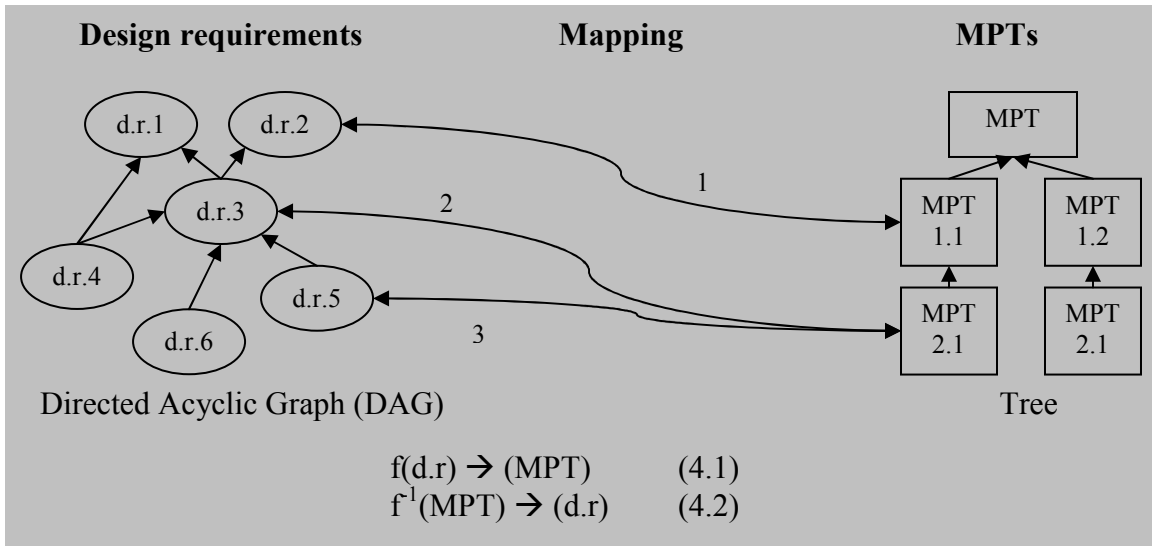
strategy for the three meta rules (MR1, 2, and 3) is the one for discrete values for process variables according to Figures 4.4 and 1.14. Hence, DFM problems under R1 and R2 provide an appropriate solution strategy. In short, the subsumption relation in manufacturing rules relates the given design requirements to the relevant DFM problems through the mapping between the design and process planning domains (MPTs). The following section discusses the details of this mapping.

### **4.3 Mathematical structure of design to MPT mapping and utilization of the mapping in retrieving DFM problems**

In this section, establishment of the mathematical mapping between the design domain and the process planning domain is discussed. Then, the desired retrieval and ranking procedures are explained by the established mapping relation. The following sections describe the details.

#### **4.3.1 Structure preserving properties**

In this section, the mathematical mapping between design and MPT is discussed. As demonstrated in section 4.2, the manufacturing rules taxonomy maps the two domains. This role of manufacturing rules taxonomy is formalized as a mathematical structure in this section. To better illustrate this, Figure 4.15 shows a generic graphical illustration of the design and process planning domains and their mapping.



**Figure 4.15 Generic mapping between design and MPT**

In Figure 4.15, the design requirements are in a directed acyclic graph. As discussed in section 3.4, the meta rules and rules are indexed by their condition parts (design requirements). Also, the manufacturing rules taxonomy shown in Figures 4.5 and 4.11 are constructed based on their indexes. Therefore, the structure of the manufacturing rules taxonomy is the structure of the design requirements. As shown in Figures 4.5 and 4.11, the manufacturing rules taxonomy is presented as a directed acyclic graph (DAG). Consequently, the structure of the design requirements is a DAG.

In Figure 4.15, the MPTs are in a tree structure. As discussed in section 4.1.6, there are subsumption relations among MPTs through the feasible space of part orientation. Hence, the subsumption relations among MPTs are simple and represented as a tree structure as shown in Figure 4.15. The subsumption relations are described such that  $MPT_{x1.y}$  subsumes  $MPT_{x2.y}$  where  $x1 < x2$  (Table 4.2).

The manufacturing rules map the two domains by mapping the two taxonomies (DAG and Tree). The curved arrows between the two taxonomies represent the

manufacturing rules. For instance, the arrows 1, 2, and 3 map d.r.2, 3, and 5 to MPT1.1, 2.1, and 2.1 respectively.

As discussed in section 4.2, the subsumption relations in the condition part of the manufacturing rules map the design requirements to appropriate MPTs. This is concisely represented as Equation 4.1. Also, the subsumption relations in MPTs map the MPTs to appropriate design requirements. This is concisely represented as Equation 4.2.

To illustrate this better, let's consider d.r.2, d.r.3, and d.r.5 and corresponding MPTs in Figure 4.15. The MPTs that can satisfy d.r.5 can be found by identifying the manufacturing rules whose condition parts are equivalent to or subsume d.r.5 (in this case arrows 1, 2, and 3). Then, the result parts of the identified manufacturing rules determine MPT2.1 and 1.1 as appropriate MPTs. Similarly, the appropriate d.rs for MPT1.1 can be found by identifying manufacturing rules that are equivalent and subsumed by the manufacturing rules that have MPT1.1 in their result parts (in this case arrows 1, 2, and 3). Then, the condition parts of the identified manufacturing rules determine d.r.2, 3, and 5 as appropriate d.rs.

For a successful mapping, the manufacturing rules taxonomy should preserve the structure of the design requirements taxonomy and MPTs taxonomy. In other words, subsumption relations in the design requirements,  $d.r.(2 \sqsupset 3 \sqsupset 5)$ , and in the MPTs,  $MPT(1.1 \sqsupset 2.1)$ , should be preserved in the manufacturing rules taxonomy;  $arrow(1 \sqsupset 2 \sqsupset 3)$ . Examples in Figure 4.16 illustrate this.

**Structure preserving manufacturing rules (case 1)**

1. Arrow 1: If (d.r.2) then (MPT1.1)
2. Arrow 2: If (d.r.3) then (MPT2.1)
3. Arrow 3: If (d.r.5) then (MPT2.1)

The subsumption relations in arrows are  $1 \sqsupset 2 \sqsupset 3$  through their condition part (d.r.2  $\sqsupset$  3  $\sqsupset$  5)

Also,  $MPT1.1 \sqsupset 2.1$

**Structure not preserving manufacturing rules (case 2)**

1. Arrow 1: If (d.r.2) then (MPT2.1)
2. Arrow 2: If (d.r.3) then (MPT1.1)
3. Arrow 3: If (d.r.5) then (MPT2.1)

The subsumption relations in arrows are  $1 \sqsupset 2 \sqsupset 3$  through their condition part (d.r.2  $\sqsupset$  3  $\sqsupset$  5)

However, arrow  $1 \sqsupset 3 \rightarrow MPT2.1 \sqsubset 1.1$

**Figure 4.16 Example of structure preserving manufacturing rules**

Figure 4.16 shows examples of manufacturing rules that preserve and do not preserve the structure of the design requirements and MPTs taxonomy. In case 1, the subsumption relation of manufacturing rules matches the subsumption relation of the condition (d.r) and result (MPT) parts. In case 2, the subsumption relation of the manufacturing rules matches the subsumption relation of the condition parts but not the result parts. Preserving the structure means the agreement of subsumption relation in condition and result parts of manufacturing rules. To correctly map design and process planning domains, this structure preserving property is important. The following paragraph discusses this in detail.

The feasible spaces of the process variables should never increase in size with increasing complexity of design requirements in the collected manufacturing rules. For instance, mapping d.r.5 with MPT1.1 (arrow 3) and d.r.2 with MPT2.1 (arrow 1) where  $d.r.2 \sqsupset d.r.5$  and  $MPT1.1 \sqsupset MPT2.1$  results in increasing the complexity of design

requirements (d.r.2  $\rightarrow$  d.r.5) increasing the feasible spaces of process variable (MPT2.1  $\rightarrow$  MPT1.1). This mapping makes no sense because satisfying more complex design requirement (d.r.5) must require a stricter process plan (MPT2.1). Therefore, the structure of the manufacturing rules taxonomy should preserve the structure of the design requirements and MPTs taxonomy in order to correctly map the design and process planning domains. In this research, such requirements are formalized as mathematical properties. Figure 4.17 shows the properties.

Let

1. MR1 = If (d.r.1) then (MPT1)
2. MR2 = If (d.r.2) then (MPT2)

*Where d.r.: design requirement, MPT: manufacturing problem template, MR: manufacturing rule*

**Properties**

1. There is many to one relation in design requirements and MPTs
2. If (d.r.1 = d.r.2) then (MPT1 = MPT2)
3. If (d.r.1  $\supset$  d.r.2) then (MPT1  $\supset$  MPT2)

**Figure 4.17 Properties for preserving the structure of the design requirements and MPTs taxonomy in the manufacturing rules taxonomy**

To explain the properties in Figure 4.17, design requirements (d.rs) and MPTs are treated as quantifiable concepts such that the subsumption relation among those can be expressed using set theory operators (ex: d.r.1  $\supset$  d.r.2). Property 1 restates Table 4.2. Property 2 states that if the condition parts (d.r) of the two manufacturing rules are the same then their results parts (MPT) should also be the same. This is because there is only one MPT that is the most relevant to any particular design requirement, and the manufacturing rules always determine the most relevant MPT. Property 3 states that if there is a subsumption relation in the design requirements then there is either a

subsumption or equivalent relation in the result part (MPT). There cannot be subsumption relations in opposite directions between the condition and the result part (ex:  $d.r.1 \sqsupseteq d.r.2$  and  $MPT1 \sqsubseteq MPT2$ ). This is because the feasible spaces of the process variables decrease with increasing complexity of the design requirements. Therefore, satisfying the properties in Figure 4.17 ensures that the collected manufacturing rules will never increase the feasible spaces of those process variables with increasing complexities in the design requirement. Consequently, the manufacturing rules taxonomy preserves the structure of the design requirements and the MPTs taxonomy.

In this research, the collected manufacturing rules satisfy the properties in Figure 4.17. The meta rules in Figure 4.5 show that increasing the complexity of the relative surface orientations never increases the feasible space of part orientation. Such properties are inherent in the rules because the condition parts of the rules are combinations of the condition parts of the meta rules and the specific surface types and accuracy requirements. In general, the properties in Figure 4.17 are believed to be presented in the manufacturing rules that are discovered in design for additive manufacturing. Hence, we claim that the manufacturing rules taxonomy in additive manufacturing preserves the structure of the design requirements and MPTs taxonomy. Consequently, the subsumption relations in the design requirements can relate the design requirements to the relevant DFM problems through identifying appropriate MPTs.

#### **4.3.2 Retrieval and ranking using the mapping**

In this research, the mathematical mapping discussed in section 4.3.1 is utilized to retrieve and rank the relevant DFM problems. To better illustrate, the two taxonomies (design requirements and MPTs) are combined to form a Qs taxonomy. The Qs

taxonomy is introduced to better illustrate the inverse mapping (from MPT to design requirements). The quantity  $Q$  is formed by combining the condition (d.r) and the result (MPT) parts of a manufacturing rule using the intersection operator ( $d.r \sqcap \text{MPT}$ ). The  $Q$  can be interpreted as “a thing that has both d.r and MPT”. However, the  $Q$  is a pure mathematical quantity and there is no specific interpretation to the real world.

The  $Q$ s taxonomy is expected to be the same as the design requirements and manufacturing rules taxonomy if the manufacturing rules possess the properties in Figure 4.17. In other words, the subsumption relations among  $Q$ s should obey the subsumption relations of design requirements. The proof is shown in Figure 4.18.

The proposition in Figure 4.18 states that the subsumption relations in design requirements are the same as the subsumption relations in  $Q$ s. This is because the subsumption relation directions in the condition and result parts of two manufacturing rules are never opposite. Hence, the design requirements taxonomy and manufacturing rules taxonomy should be the same as the  $Q$ s taxonomy if the manufacturing rules hold properties in Figure 4.17.

**Proposition:**

Let

- $MR1 = \text{If (d.r.1) then (MPT1)}$
- $MR2 = \text{If (d.r.2) then (MPT2)}$
- $Q1 = \text{d.r.1} \sqcap \text{MPT1}$
- $Q2 = \text{d.r.2} \sqcap \text{MPT2}$

Then, the following three properties can be shown to hold for a collection of manufacturing rules if properties in Figure 4.17 hold

1. If  $(\text{d.r.1} = \text{d.r.2})$  then  $(Q1 = Q2)$
2. If  $(\text{d.r.1} \neq \text{d.r.2})$  then  $(Q1 \neq Q2)$
3. If  $(\text{d.r.1} \sqsupset \text{d.r.2})$  then  $(Q1 \sqsupset Q2)$

Where *MR*: manufacturing rule, *d.r.*: design requirements, *MPT*: manufacturing problem template

**Proof**

In general

$$Q1 \sqsupset Q2 \rightarrow Q2 \sqcap \neg Q1 = \emptyset$$

By substituting  $Q1$  and  $Q2$ , we get

$$(\text{d.r.2} \sqcap \text{MPT2}) \sqcap \neg(\text{d.r.1} \sqcap \text{MPT1}) = \emptyset$$

$$(\text{d.r.2} \sqcap \text{MPT2}) \sqcap (\neg \text{d.r.1} \sqcup \neg \text{MPT1}) = \emptyset$$

$$((\text{d.r.2} \sqcap \text{MPT2}) \sqcap \neg \text{d.r.1}) \sqcup (\text{d.r.2} \sqcap \text{MPT2}) \sqcap \neg \text{MPT1} = \emptyset$$

For above equation to be valid the following conditions have to be satisfied

$$(a). (\text{d.r.2} \sqcap \text{MPT2}) \sqcap \neg \text{d.r.1} = \emptyset$$

$$(b). (\text{d.r.2} \sqcap \text{MPT2}) \sqcap \neg \text{MPT1} = \emptyset$$

Once (a) and (b) are satisfied then,  $Q1 \sqsupset Q2$  is proved

**Property 1:** *Given:*  $\text{MPT1}$  and  $\text{MPT2}$  are the same because  $\text{d.r.1} = \text{d.r.2}$ .

*Proof:*  $Q1 \sqsupset Q2$  and  $Q2 \sqsupset Q1$  should be true because  $Q1 = Q2$ . By following above general procedure,  $Q1 \sqsupset Q2$  and  $Q2 \sqsupset Q1$  are true because  $\text{d.r.1} = \text{d.r.2}$  and  $\text{MPT1} = \text{MPT2}$

**Property 2:** *Given:*  $\text{d.r.1} \neq \text{d.r.2}$

*Proof:* For  $(Q1 \neq Q2)$  to be true,  $(Q1 \not\subset Q2)$  and  $(Q2 \not\subset Q1)$  should be true.

Also,  $(Q1 \sqcap \neg Q2 \neq \emptyset)$  and  $(Q2 \sqcap \neg Q1 \neq \emptyset)$  have to be true. By following above general procedure, it is found that (a) is always violated for both cases due to the fact that  $\text{d.r.1}$  and  $\text{d.r.2}$  are different. Therefore,  $Q1 \neq Q2$

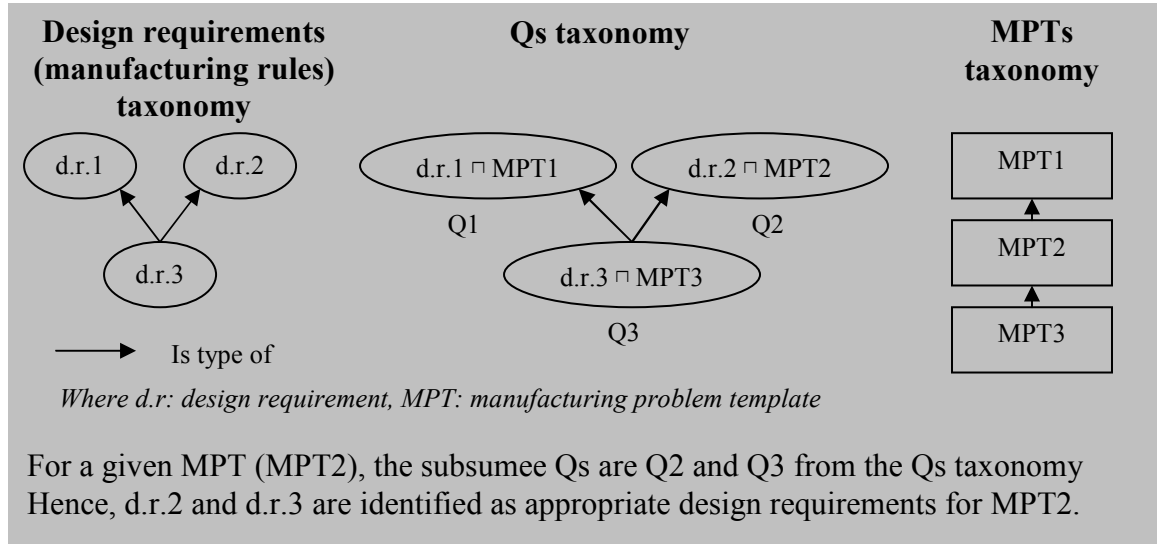
**Property 3:** *Given:*  $\text{MPT1} \sqsupset \text{MPT2}$  or  $\text{MPT1} = \text{MPT2}$  because  $\text{d.r.1} \sqsupset \text{d.r.2}$

*Proof:* By following the general procedure above, it is found that (a) is satisfied because  $\text{d.r.1} \sqsupset \text{d.r.2}$ . Also, (b) is satisfied for either  $\text{MPT1} \sqsupset \text{MPT2}$  or  $\text{MPT1} = \text{MPT2}$ . Therefore,  $Q1 \sqsupset Q2$

**Figure 4.18 Proposition and proof for correctness of taxonomy**

In Figure 4.18, three mathematical properties and their corresponding proofs are shown. The proof of each property is shown at the bottom of Figure 4.18. Each proof uses the general proof procedure shown at the beginning of the entire proof.

By constructing the Qs taxonomy, inverse mapping from MPT to design requirements can be demonstrated. Figure 4.19 shows an example.



**Figure 4.19 Example of meta rules and Qs taxonomy**

In Figure 4.19, the example of design requirements, MPTs and Qs taxonomy is shown. For this example, it is assumed that d.r.1, 2, and 3 are paired with MPT1, 2, and 3 respectively in the manufacturing rules. Hence, Qs are formed as shown in Figure 4.19. Then, the Qs taxonomy can be used to query appropriate d.rs for a given MPT. For instance, d.rs for MPT2 can be found by identifying Qs that are subsumees of MPT2. Those are Q2 and Q3. Hence, d.r.2 and 3 are identified as appropriate d.rs for MPT2. In short, the design requirements taxonomy (manufacturing rules taxonomy) can be used to identify appropriate MPTs (forward mapping  $f(d.r.) \rightarrow (MPT)$ ) and the Qs taxonomy can be used to identify appropriate d.rs (inverse mapping  $f^{-1}(MPT) \rightarrow (d.r.)$ ). In this research,

the forward mapping supports identifying the relevant DFM problems as discussed in Figures 4.13 and 4.14. Then, the inverse mapping supports ranking the identified DFM problems.

Among the DFM problems that are identified by forward mapping, the problems that use an MPT that closely matches the needed MPT for the given design requirement support process planning better as discussed in section 4.1.6. Therefore, the retrieved DFM problems should be ranked based on MPTs. Consequently, ability to identify DFM problems and d.rs from the given MPT is important. The inverse mapping discussed in Figure 4.19 supports this.

In short, forward mapping supports identifying the relevant DFM problems and inverse mapping supports ranking the identified DFM problems. Figure 4.20 describes such relation in a concise mathematical notation.

**Given:**

$d.r = \{d.r.1, d.r.2, \dots, d.r.n\}$ ,  $MPT = \{MPT1, MPT2, \dots, MPTn\}$ ,  
A given design requirement = g.d.r

**Retrieval and ranking:**

1. From  $f(g.d.r) = d.MPT$  (determined MPTs), a set of appropriate MPTs (d.MPT) is determined for g.d.r
2. Rank m in d.MPT by their subsumption relation (ex:  $MPT1 \sqsupseteq MPT2$  then MPT1 is ranked higher than MPT2)
3. For each m in d.MPT,  $d.r = \{d \mid d = \{x \mid x \sqsupseteq g.d.r, x \in f^{-1}(m)\}\}$

**Figure 4.20 Retrieval and ranking**

In Figure 4.20, the forward mapping is described with item 1 where a set of appropriate MPTs (d.MPT) is determined for the g.d.r. Then, the MPTs are ranked based on their subsumption relations. For each MPT (m) in d.MPT, the corresponding design

requirements are found by inverse mapping and subsumption relation with g.d.r. This is shown as item 3 in Figure 4.20.

#### 4.4 Summary

In this chapter, a mathematical mapping between the design and the process planning domains is established and justified. First, the manufacturing rules for each process variable in Stereolithography are discovered and combined (section 4.1). Those manufacturing rules are used to construct a manufacturing rules taxonomy in Figures 4.5, 4.7, and 4.9. Then, the task of relating design requirements to relevant DFM problems is demonstrated through examples (section 4.2). Through the examples, the formal mapping between design and process planning is realized (section 4.3). The mathematical properties that the collected manufacturing rules must possess to enable the realized mapping are identified and discussed (Figure 4.16).

In Table 4.3, the research question and hypotheses (1 and 2) are restated.

**Table 4.3 Research question and hypotheses (1 and 2)**

Questions	Hypotheses
Q1. How should design requirement be represented?	H1. Description logics enable representation of design requirements
Q2. How should the design and process planning domains be mapped?	H2. Subsumption in DL enables mathematical mapping between design domain and process planning domain

By collecting manufacturing rules, the design requirements (condition part of manufacturing rules) are also collected. To validate hypothesis 1, the collected manufacturing rules should be analyzed carefully to identify and justify selection of appropriate description logics. Hence, the manufacturing rules collection and justification in this chapter contributes toward theoretical validation of hypothesis 1. To

validate the hypothesis 2, the mathematical relation between the two domains needs to be established and justified first. Then, subsumption in DL needs to be used and tested to realize the mapping. Hence, establishing and justifying the mathematical relation between the two domains contributes toward the theoretical validation of hypothesis 2.

As discussed in chapter 2, there has not been a research effort in formally relating the design and process planning domains especially in additive manufacturing. In this research, the manufacturing rules and their subsumption relations map the design and process planning domains. Through the subsumption relations in the condition part, the design domain is mapped to the process planning domain. Through the subsumption relations in the result part, the process planning domain is mapped to the design domain. The manufacturing rules are derived by representing the empirical models that relate accuracy measurements to process variables into tabular form (discretization discussed in section 3.3). Through this derivation, subsumption relations in design requirements and MPTs are identified as the mathematical relation that maps design and process planning domains. The established mapping relation forms a solid mathematical base that relates design and process planning domains correctly and consistently. Hence, the identified mathematical relation is a new contribution in the area of DFM. Table 4.4 summarizes this.

**Table 4.4 New knowledge discovered in DFM**

DFM
<ul style="list-style-type: none"> <li>• Identification and justification of mathematical relation between the design and the process planning domains</li> </ul>

## **CHAPTER 5: INFORMATION MODELING (USING DESCRIPTION LOGICS)**

In this chapter, the description logic implementation for representing design requirements and computing the manufacturing rules taxonomy is presented. More specifically, this chapter discusses building information models for design requirements and mapping design and process planning domains using description logic. First, the characteristics and requirements for representing design requirements and computing taxonomies are identified (section 5.1). Metrics for characteristics and requirements that need to be satisfied are developed to identify the appropriate representation formalism. Second, several representations and reasoning formalisms are discussed (section 5.2). The representation formalisms are critically analyzed against the metrics for requirements. Through this analysis description logic is selected and justified as an appropriate formalism. Finally, description logics' representation of design requirement and the manufacturing rules taxonomy computation are discussed (sections 5.3 and 5.4). Through identifying an appropriate description logic ( $\mathcal{AL}\mathcal{E}$ ) and being able to compute correct manufacturing rules taxonomy, hypotheses 1 and 2 are theoretically validated (section 5.5). Also, the new knowledge that is realized through validating the hypotheses is discussed (section 5.6).

### **5.1 Characteristics and requirements**

The design requirements identified in section 3.1.1 need to be represented in a computer processible form so that they can be related to relevant DFM problems in an automated procedure. More specifically, a computer processible representation is required to perform the tasks listed in Figure 5.1.

1. Represent the design requirement
2. Index manufacturing rules with their condition parts
3. Construct manufacturing rules taxonomy
4. Compare the given design requirements to the index of manufacturing rules

**Figure 5.1 Tasks to be performed using representation**

The tasks listed in Figure 5.1 are basic steps required for the proposed method shown in Figure 1.16 to function correctly. Step 1 corresponds to representing the design requirements so that they can be compared and used to index manufacturing rules. Step 2 is about indexing the manufacturing rules with the represented design requirements. Step 3 is about requiring a systematic method of constructing manufacturing rules taxonomy. As discussed in chapters 4, constructing the correct manufacturing rules taxonomy is critical in this research. Step 4 is about comparing the represented design requirements to the index of manufacturing rules to identify the appropriate manufacturing rules for the given design requirements. Those are steps that need to be carried out successfully for the proposed method to function correctly. However, there are some challenges in carrying out such tasks. The following paragraphs describe the challenges.

First, discovering all existing manufacturing rules for additive manufacturing processes is infeasible. Additive manufacturing is one of the fastest growing areas and new technology is introduced rapidly. Each technology uses different ways of fabricating parts[10, 109, 110]. Consequently, manufacturing rules vary from technology to technology although some rules are common in additive manufacturing. In this research, we confine the research scope to be in layer-based additive manufacturing. In other words, the method is developed to be applicable to manufacturing processes that fabricate parts by layer-by-layer material deposition. Hence, the manufacturing rules are confined

to those that handle design requirements expressible by concepts and relations in Figures 3.1 and 3.2. In short, determining how to represent and index undiscovered design requirements and manufacturing rules only by their characteristics is challenging.

Secondly, ensuring consistency in design requirements representation and manufacturing rule indexing is challenging. As new manufacturing rules are introduced, a new representation could be used to index new rules. The newly introduced representation should not conflict with or duplicate existing representations. However, ensuring such consistency is difficult when the new representations are introduced in distributed environment by multiple engineers. Hence, it is critical to have a systematic means for checking the consistency of information representations. Without any systematic checking method, design requirement representation and manufacturing rules indexing becomes chaotic. Consequently, the retrieval method could eventually fail. Therefore, the challenge is to check the representation consistency in distributed environment.

Third, the manufacturing rule taxonomy needs to be constructed and modified in a distributed environment. As discussed in chapter 4, a correct manufacturing rules taxonomy is critical in relating design requirements to relevant DFM problems in this research. However, ensuring correct manufacturing rules taxonomy construction is extremely difficult when the taxonomy is explicitly extended and modified in a distributed environment by multiple engineers. Hence, a systematic method of extending and managing manufacturing rules taxonomy is necessary.

The challenges and required steps discussed so far are summarized in Tables 5.1 and 5.2 respectively. In Table 5.1, three characteristics that summarize the above

challenges are presented. In Table 5.2, three requirements that summarize required steps shown in Figure 5.1 are presented. In these tables, the definitions of each characteristic and requirement are shown.

**Table 5.1 Characteristics of method operation environment**

Characteristics	Descriptions
Expansion	This refers to addition of new design requirement and manufacturing rules
Hierarchy	This refers to subsuming relation in condition and result parts of manufacturing rules
Dynamic	This refers to dynamic nature of design requirements and manufacturing rules in distributed environment

**Table 5.2 Requirements for successful functioning of method**

Requirement	Descriptions
Expressivity	This refers to the capability of providing enough expressivity for representing newly discovered and design requirements and manufacturing rules
Concept comparison	This refers to the capability of comparing represented design requirements to determine the relation between two different design requirements. This capability needs to be ensured in comparing undiscovered design requirements
Taxonomy computation	This refers to the capability of systematic construction of manufacturing rules taxonomy.

In Table 5.1, characteristics including expansion, hierarchy, and dynamic are shown with their definitions. Expansion represents the characteristics of extending design requirement representations for newly discovered design requirements and manufacturing rules. Hierarchy represents the nature of subsuming relations in condition and result parts of manufacturing rules. Dynamic characteristics represent dynamic nature of design requirement and manufacturing rule addition and modification in distributed environment. Those are identified as the important characteristics that summarize the challenges need to be addressed.

In Table 5.2, the requirements that need to be satisfied to address the characteristics in Table 5.1 are presented. Expressivity refers to the ability to express the design requirements that are to be discovered. Determination of the required expressivity is feasible because the design requirements information space for additive manufacturing is identified in section 3.1.1. Concept comparison represents the ability to compare concepts and determine their relations. Due to the hierarchical nature of design requirements and manufacturing rules, the relation to be determined is subsumption relation. The concept comparison capability needs to be ensured for comparing undiscovered design requirements. The taxonomy computation represents the ability to systematically construct the manufacturing rule taxonomy. As discussed previously, the management capability of manufacturing rules taxonomy is critical to construct and manage a correct taxonomy in a distributed environment.

The requirements in Table 5.2 must be satisfied under conditions imposed by characteristics in Table 5.1. To better satisfy requirements in Table 5.2, the relation between individual characteristics and requirements are investigated and shown in Figure 4.5.

	Expressivity	Concept comparison	Taxonomy computation
Expansion	●	●	●
Dynamic	○	●	●
Hierarchy	○	●	●
Key: ○ - Weakly related ● - Strongly related			

**Figure 5.2 Relation between characteristics and requirements**

In Figure 5.2, the relationships between characteristics and requirements are characterized by degree of strength. Expressivity is shown to be more strongly related to

expansion characteristics than any other characteristics. This is because expressivity is the capability that provides enough freedom to express newly introduced concepts. Concept comparison and taxonomy computation capability are shown to be strongly influenced by all three characteristics. This is because concept comparison and taxonomy computation capability are the essential components for systematically managing information models and rules taxonomy. Under expanding and dynamic conditions in distributed environments, a mechanism that enforces consistency in the information model and rules taxonomy is absolutely necessary. Concept comparison is basically a tool that can be used to compute the taxonomy. However, the two are separately treated because the capabilities of each are needed in the retrieval method.

In short, satisfying the requirements in Table 5.2 under conditions described in Table 5.1 is an enormous task that requires investigation of formalism utilization. The following section describes various formalisms and their ability to identify appropriate formalisms for this research.

## **5.2 Representation formalisms**

In this research, a formalism that can determine a subsumption relation between concepts by representing and reasoning with them is needed. This area is commonly known as knowledge representation and reasoning[111-118]. The majority of the formalisms in this area are rooted in first order logics. In first order logics, knowledge about a domain is explicitly represented and used to infer new knowledge. Figure 5.3 provides a brief example of how first order logics represent and reason with concepts[118].

### Knowledge base

- Jane is allergic to caffeine.
- Everyone who is allergic to caffeine does not drink coffee
- Either Jane or Jack must have drunken coffee

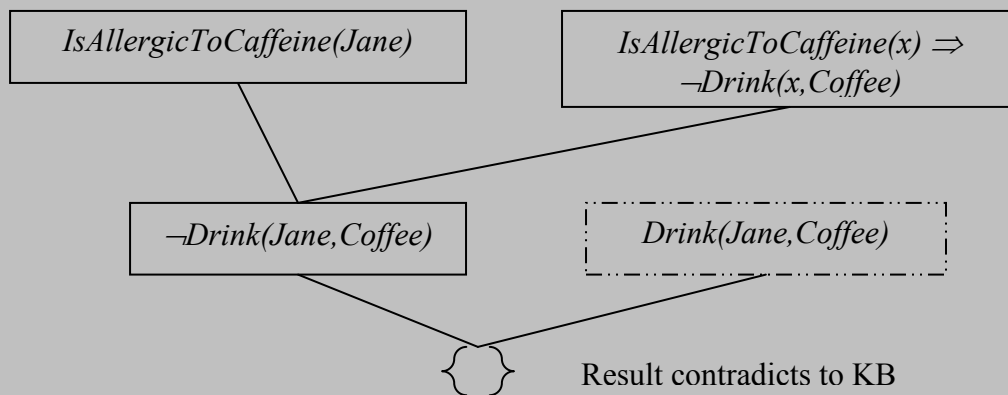
### Query

- Did Jack drink coffee?

### Representation

- Constants: Jane, Jack, Coffee
- Variable: x, y
- Predicates: IsAllergicToCaffeine(x)
- Function: Drink(x,y)
- Knowledge base representation:
  1. IsAllergicToCaffeine (Jane)
  2.  $\forall x (IsAllergicToCaffeine (x) \Rightarrow \neg Drink(x,Coffee))$
  3.  $Drink(Jane,Coffee) \vee Drink(Jack,Coffee) \Rightarrow True$

### Inference (resolution)



**Figure 5.3 Example of first order logic**

In Figure 5.3, the given knowledge is that Jane is allergic to caffeine and everyone who is allergic to caffeine does not drink coffee. Also, either Jane or Jack must have drunk coffee. The query is to find out if Jack drank coffee.

In first order logic, concepts for constants, variables, predicates and functions for representing knowledge are determined first. Constants represent objects that are unique in the world such as the name of a place or person. Predicates and functions determine types or relations between objects. Variables are used to hold objects in predicates and

functions[118]. Jane, Jack, and Coffee are defined to be constants in the example shown in Figure 5.3. Predicate ‘IsAllergicToCaffeine’ is used to instantiate a person who is allergic to caffeine. Function ‘Drink’ is used to describe the action of person drinking some liquid. Then the knowledge base is represented using those constants, predicates, and functions with logical operators such as conjunction ( $\wedge$ ), disjunction ( $\vee$ ), and implies ( $\Rightarrow$ ). In first order logics, inference is performed by a procedure called resolution. The key idea of resolution is to uncover implicit knowledge by a series of constants and terms substitution into existing knowledge base[114]. An example of the inference procedure is shown in Figure 5.3. In this example, the conclusion of Jack drank coffee is drawn by disproving the proposition that Jack did not drink coffee. As shown in knowledge base representation 3, either Jack or Jane must have drunk coffee. If Jack did not drink coffee then, Jane must have drunk coffee. Hence, our goal is to prove that Jane did not drink coffee. To prove this, knowledge representation 1 is substituted to left side of 2. Then the conclusion can be drawn such that Jane does not drink coffee. Therefore, Jack must have drunk coffee for knowledge representation 3 to be valid. The graphical illustration in Figure 5.3 shows the resolution procedure. Therefore, the implicit knowledge that Jack drank coffee is made explicit by the inference procedure in first order logic. This inference is the main focus in knowledge representation and reasoning.

In this research, such inference capability is crucial to update manufacturing rules and identify relevant DFM problems. More specifically, distributed engineers need to be able to update the manufacturing rules and the DFM problems repository by asking questions such as “*Are there equivalent manufacturing rules or DFM problems?*” or “*Where is the right place to insert new manufacturing rules or DFM problems into the*

*existing taxonomy?*” Therefore, consistent and correct answers for those questions are strongly desired from the utilization of formalisms. Numerous formalisms are developed in the area of knowledge representation and reasoning. Russell and Norvig classified those formalisms into four categories based on their utilization as shown in Table 5.3[118]. Those formalisms can be found in typical introductory artificial intelligence (AI) books. The majority of the formalisms in Table 5.3 are based on first order logics. The following sections provide brief descriptions of the formalisms in each category.

**Table 5.3 Representations and reasoning mechanisms**

Categories	Representation formalisms
Logical reasoning systems	<ul style="list-style-type: none"> <li>• Theorem provers and logic programming language</li> <li>• Production systems</li> <li>• Frame systems and semantic nets</li> <li>• Description Logics (DL)</li> </ul>
Probabilistic reasoning systems	<ul style="list-style-type: none"> <li>• Belief network</li> <li>• Fuzzy sets and fuzzy logics</li> </ul>
Reasoning systems for learning	<ul style="list-style-type: none"> <li>• Neural network</li> <li>• Belief network</li> </ul>
Natural language processing systems	<ul style="list-style-type: none"> <li>• Definite Clause Grammar (DCG)</li> </ul>

### 5.2.1 Logical reasoning system

The first category shown in Table 5.3 is logical reasoning systems. Formalisms belonging to this category are developed to support representations and inference procedure better for more specific purposes than first order logics. The following paragraphs describe some of the representative formalisms in this category.

- **Theorem provers:**

Theorem provers use resolution to prove sentences in full first order logics and are often used for mathematical and scientific reasoning tasks. This formalism is typically used to answer questions that use variables and whose answers instantiate those variables by constants. A logic programming language is similar to theorem provers but there is a difference in inclusion of non logical features of programming language such as input and output. Examples of theorem provers include: SAM, AURA, and OTTER. Examples of logic programming languages include Prolog, MRS, LIFE[114, 118].

- **Production Systems:**

Similar to logic programming language, these use implication as their primary representation. The consequences of implications are interpreted as action recommendations rather than logical conclusions. Actions include insertions and deletions from the knowledge base as well as input and output. Examples include OPS-5, CLIPS, and SOAR[113, 118].

- **Frame systems and semantic networks:**

In a semantic net, objects are represented in nodes in a graph; these nodes are organized in taxonomic structure, and the links between the nodes represent the binary relations between objects. In frame systems the binary relations are thought of as slots in one frame that are filled by another. These formalisms concentrate on representing categories of objects and relations between them. More specifically, they focus on representing the inheritance characteristics of categories and objects. Frame systems later adopted the object-oriented method. For inference, represented concepts using these

formalisms are typically translated to first order logics. Examples of semantic net include SNEPS, NETL, and Conceptual Graphs. Examples of frame systems include FRAIL and KODIAK[114, 117, 118].

- **Description logics systems (DL):**

These systems evolved from semantic nets. The key idea of this formalism is to express and reason with complex definitions of, and relations among, objects and classes. Those systems support inference services including satisfiability and subsumption. A subsumption algorithm determines the subsumption relation between objects and classes. Recent work has concentrated on the trade-off between expressivity in the language and the computational complexity of certain operations. Examples include KL-ONE, CLASSIC, LOOM, Protégé, and Racer[118-122].

### **5.2.2 Probabilistic reasoning system**

The second category in Table 5.3 is the probabilistic reasoning systems. The formalisms in logical systems infer new knowledge based on the assumption that the knowledge base is complete. However, there are cases where not enough information is collected to build complete knowledge base and an inference based on the existing knowledge base is desired. The formalisms that belong to this category focus on representation and inference under uncertainty. These formalisms are based on probability and utility theory. Such formalisms include belief network and fuzzy logics. The following paragraphs provide brief descriptions of them.

- **Belief network**

A belief network is a graph using nodes and directed arrows that link the nodes. Each node represents concepts and each arrow represents the causal relationships that link nodes. For example, “Burglary  $\rightarrow$  Alarm” shows that a burglary caused alarm. Then, each node is accompanied by a conditional probability that quantifies the effects that parents have on nodes. For example, “Burglary  $\rightarrow$  Alarm (0.9 by burglary)” shows that 90% of alarms are caused by burglary. The basic idea of inference in a belief network is to compute the posterior probability distribution for a set of queries, given exact values for some evidence[118].

- **Fuzzy sets and fuzzy logic**

This formalism is a means of specifying how well an object satisfies a vague description. For example, consider the proposition “Jane is tall” that can be described by the predicate TallPerson(Jane). Fuzzy set theory answers these types of queries by providing a truth value of between 0 and 1 rather than true or false. Then the fuzzy logic uses this truth value to determine the truth value of a more complex sentence such as TallPerson(Jane)  $\vee$  SmartPerson(Jane)[117, 118].

### **5.2.3 Learning**

The third category in Table 5.3 is reasoning systems for learning. The focus of the formalisms in this category is to support machine learning- programs that learn from experience. Representative formalisms in this category include neural network and belief network. The following paragraphs briefly describe them.

- **Neural network**

A neural network is composed of a number of nodes, or units, connected by links. Each link has a numeric weight associated with it. Weights are the primary means of long-term storage in a neural network, and learning usually takes place by updating the weights. Some of the units are connected to the external environment, and can be designated as input or output units. The weights are modified so as to try to bring the network's input/output behavior more into line with that of the environment providing the inputs. A neural network is a typical formalism for pattern recognition. Its application areas include pronunciation and handwritten character recognition[111, 115, 118].

- **Belief network**

Learning in a belief network is about updating numeric values in conditional probability tables for each node (please refer to the belief network description above). The learning procedure is analogous to that used by a neural network. The difference is that the updated values are a conditional probability table in a belief network rather than the weights during the training[118].

#### **5.2.4 Natural language processing system**

The final category in Table 5.3 is natural language processing systems (NLP). The formalisms belong to this category attempt to support human language processing. Definite clause grammar (DCG) is discussed as an example formalism in the following paragraph.

- **Definite Clause Grammar (DCG)**

In definite clause grammar, every sentence must be a definite clause. In other words, all the sentences must be in the form of an implication that has exactly one atom in its consequent ( $A_1 \wedge A_2 \wedge \dots \Rightarrow C_1$ ). Then the represented sentences are translated to first order logics to utilize the logic's inference capability to understand the represented sentences. Its practical application areas are database access, information retrieval, text categorization, etc. However, the major difficulty in natural language processing is in identifying the meaning of words in different contexts[117, 118].

So far, various formalisms are investigated and discussed. Those formalisms are by no means a complete list; rather they are representative and common formalisms for the domain where the formalism is known to be applicable.

### **5.2.5 Selection**

Based on the investigation of various formalisms, the appropriate formalism for this research is chosen by critically analyzing the formalisms. This analysis is performed by identifying the formalism that can satisfy the requirements list in Table 5.2. Figure 5.4 shows the capability of investigated formalisms in satisfying the requirements in Table 5.2.

	Logical reasoning systems				Probabilistic reasoning systems		Learning		Natural language processing
	Theorem provers and logical programming languages	Production systems	Semantic net and frame systems	Description logics	Belief network	Fuzzy set and fuzzy logic	Neural network	Belief network	Definite clause grammar
Expressivity	●	○	●	●	●	●	●	●	○
Concept comparison	○	○	○	●	○	○	○	○	○
Taxonomy computation	○	○	○	●	○	○	○	○	○
Computational feasibility	○	○	○	●	○	○	N/A	○	○
Key: ○ - Weak support ● - Strong support N/A – Not applicable									

**Figure 5.4 Support of the formalisms in satisfying requirements**

In Figure 5.4, the ability of a formalism to support the requirements is shown by three different levels of support; strong support, weak support, and not applicable. The formalisms in probabilistic reasoning systems are targeted toward reasoning under incomplete or uncertain knowledge. In this research, the concept comparison and taxonomy computations tasks are not performed under any uncertainty. In other words, those tasks are always performed based on the assumption that the knowledge base at any state is complete. Therefore, formalisms in probabilistic reasoning systems are ruled out.

The formalisms in the learning category are determined to be not appropriate either. The formalisms in this category are focused on supporting the reasoning process

for autonomous machine learning. Some formalism such as neural networks uses this learning capability for complex pattern recognition. However, such learning or pattern recognition capability is not desired in this research.

The formalisms in natural language processing attempt to process human natural language. The major challenge in this area is determining the context sensitive meaning of words and phrases. Currently available formalisms are severely restricted in their expressivity. The formalisms in this field are not developed for concept comparison and taxonomy computations. Therefore, formalisms in natural language processing are also ruled out.

Formalisms in logical reasoning systems are the most relevant to this research. However, not all of them are meaningful to this research. Specifically, what we need is a formalism that can be used to represent and compare concepts to determine their subsumption relation such that a taxonomy of concepts can be computed. Description logic is the closest formalism that supports such capability through its inference services such as satisfiability and subsumption algorithms.

Other formalisms in this category can be used to infer a subsumption relation between concepts; however inference services such as a subsumption algorithm need to be added to those formalisms to properly support the desired concept comparison. Frame systems are the closest formalisms in terms of taxonomy construction; however these formalisms enforce the user to explicitly construct the concept taxonomy rather than compute the taxonomy from represented concepts. Hence, a concept taxonomy constructed by frame systems could quickly become unmanageable if the taxonomy were modified by multiple persons in a distributed environment.

Most of the formalisms in logical reasoning systems are rooted in first order logics. First order logics provide strong expressive power but are undecidable[123]. Therefore, the formalisms that use first order logics' inference can be considered as undecidable. Description logics are decidable subsets of first order logic[100, 124]. Literature in description logics shows the trade off studies between expressive power and computational complexities[119].

In this research, formalisms are investigated to select one that provides the desired inference services. Hence, a better computational feasibility in the inference services is always desirable. As shown in Figure 5.4, description logics satisfy the selection criteria more strongly than any other formalism. Hence, we choose description logics as the appropriate formalism, and a detailed description of these is provided in the following section.

### **5.2.6 Description logics**

Description logics are knowledge representation formalisms that represent domain specific concepts and their relationships by first defining the relevant concepts of the domain (its terminology), and then using these concepts to specify the properties of objects and individuals occurring in the domain. The description logics can be viewed as formal languages for representing knowledge and reasoning about it. Among the many things that description logic provides, description language and inference algorithms are relevant to this research. The description language is used to define and manage concepts and their relationships. The inference algorithms are used to determine the similarities of concept descriptions. The following paragraph briefly explains basics, a utilization

example, computational complexities, and computational consistencies of the description logics.

### 5.2.6.1 Basics

In description language, elementary descriptions are atomic concepts and atomic roles. Complex descriptions can be built inductively from these by using concept constructors. Description logics provide the attributive language ( $\mathcal{AL}$ ) and other languages of this family are extensions of  $\mathcal{AL}$ . Concept descriptions in  $\mathcal{AL}$  are formed according to the following syntax rules:

$C, D \rightarrow A$		(atomic concept)
$T$		(universal concept)
$\perp$		(bottom concepts)
$\neg A$		(atomic negation)
$C \sqcap D$		(intersection)
$\forall R.C$		(value restriction)
$\exists R.T$		(limited existential quantification)

where  $A$  denotes atomic concepts,  $R$  denotes atomic roles, and  $C, D$  denotes concept descriptions. The expressive power can be further enhanced by the following constructors

$U \rightarrow C \sqcup D$	(union of atomic concepts)
$E \rightarrow \exists R.C$	(full existential quantification)
$N \rightarrow \geq nR, \leq nR$	(number restriction)
$C \rightarrow \neg C$	(negative for arbitrary concepts, “complement”)

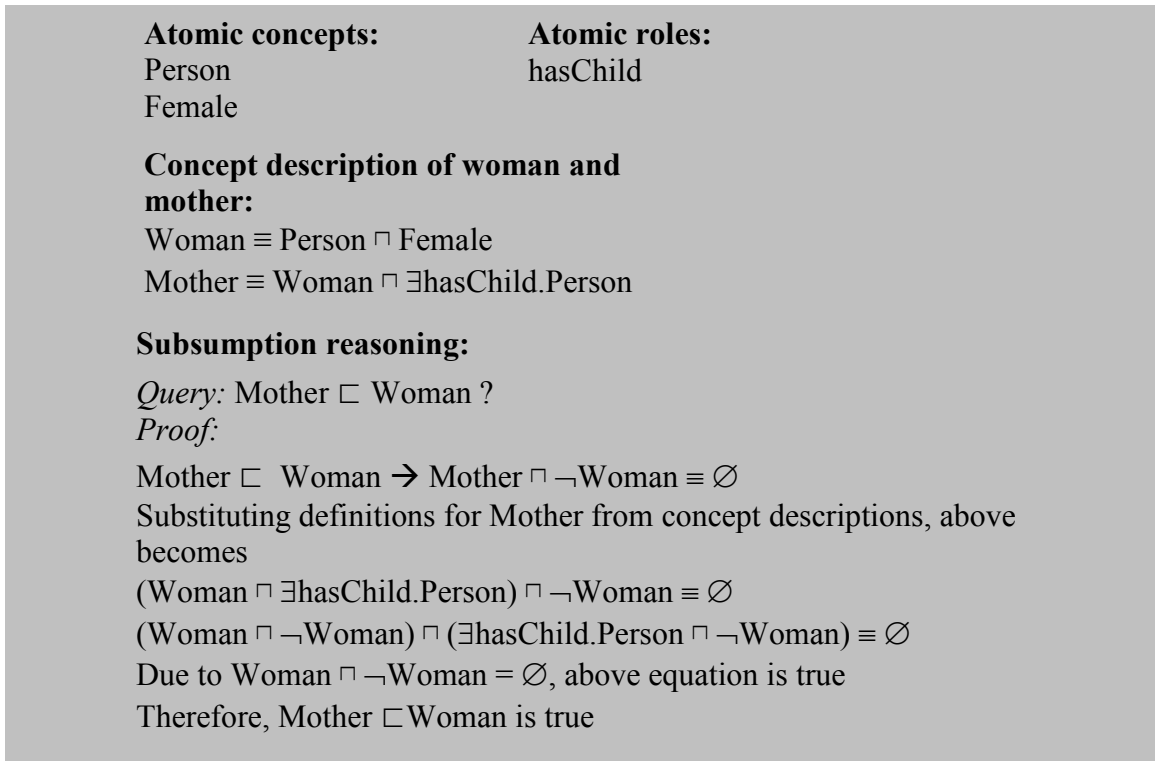
Extending  $\mathcal{AL}$  by any subset of the above constructors yields a particular  $\mathcal{AL}$  language[119]. Their names are  $\mathcal{AL}[\mathcal{U}][\mathcal{E}][\mathcal{N}][\mathcal{C}]$ . The concept descriptions using description logics are constructed by determining base symbols for atomic concepts and roles first. Then, the set theory constructors are used with atomic concepts and roles to describe more specific and complex concepts. The inference algorithms that are relevant to this research are satisfiability and subsumption.

*Satisfiability* algorithm determines the logical soundness of concepts with respect to terminologies. When the domain specific concepts are modeled, terminology is constructed by defining new concepts, possibly in terms of other concepts that have been defined before. During this process, a newly defined concept is checked to determine whether the concept makes sense or whether it contradicts existing concepts. The satisfiability algorithm tests the newly defined concept by determining whether there is some interpretation that satisfies the axioms of the terminology such that the newly defined concept denotes a nonempty set in that interpretation.

*Subsumption* is an algorithm that determines whether one concept or role can contain another concept or role. For example, a concept C subsumes concept D if every member of concept D is also a member of C[125].

#### 5.2.6.2 Utilization example

Figure 5.5 shows a simple example of description logics representation of concepts woman and mother[119]. Also, it presents subsumption reasoning procedures that determine their subsumption relations.



**Figure 5.5 Description logic representation example**

In Figure 5.5, the atomic concepts and roles are defined. For this example, Person and Female are chosen for atomic concepts. Also, hasChild is selected as atomic role. Then, mother and woman are defined using atomic concepts, role and set theory operators including full existential operator ( $\exists$ ) and intersection operator ( $\sqcap$ ). For example, Woman is defined as something that intersects Person and Female. Also, Mother is defined as something that intersects Woman and something that has person as its child. Then, the subsumption reasoning is presented. The set of presented procedures is called a tableau algorithm. It reduces subsumption to satisfiability. For example, the statement “Mother subsumed by Woman” is reduced to a statement “Mother intersect with not Woman is null”. Using the tableau algorithm, the subsumption relation between Mother and Woman is determined.

### 5.2.6.3 Computational complexities

Past years, trade-offs between expressive power and computational complexity in description logics is under intensive investigation. Table 5.4 shows the various description logics and their computational complexity[119].

**Table 5.4 Description logics and their computational complexity**

Language	Satisfiability performance
$\mathcal{AL}$	P(Polynomial time)
$\mathcal{ALE}$	NP(Non-deterministic polynomial time)
$\mathcal{ALC}$	PSPACE
$\mathcal{ALCN}$	PSPACE
$\mathcal{SHIQ}$	EXPTIME

In Table 5.4, P (polynomial time) represents the class of problems that require polynomial amount of computational time such as  $n^{100}$ ,  $n$ ,  $\log(n)$  where  $n$  is the problem size. NP (non-deterministic polynomial time) represents the class of nondeterministic polynomial problems. A problem is in this class if there is some algorithm that can guess a solution and then verify whether or not the guess is correct in polynomial time. PSPACE (polynomial space) represents the class of problems that require a polynomial amount of memory with unlimited time, even on a nondeterministic machine. EXPTIME (exponential time) represents class of problems that can be solved in  $O(2^{p(n)})$  time, where  $p(n)$  is a polynomial function of  $n$ [118, 126].

### 5.2.6.4 Consistency and correctness of subsumption in DL

In this research, subsumption in DL is used to computationally realize the forward and inverse mapping discussed in Figure 4.15. Specifically, subsumption in DL is used to compute the subsumption relations among the design requirements and among the MPTs to realize the mapping between the design and manufacturing domains. Hence,

successful mapping between the two domains is critically dependent on the consistent and correct performance of subsumption in DL. The following paragraphs discuss the mathematical proof of consistency and correctness of subsumption in DL.

Hierarchies created via computation of subsumption can be shown to be consistent and correct for acyclic terminologies by proving that subsumption imposes an order relation, or partial order on entities when the subsumption relationship between them is computed. Wille and Ganter list conditions for asserting that a binary relation  $R$  on a set  $M$  is a partial order relation[127]. They state that for all elements  $x, y, z \in M$ ,

- the relation is reflexive, i.e.,  $xRx$
- the relation is antisymmetric, i.e.,  $xRy$  and  $x \neq y \Rightarrow$  not  $yRx$
- the relation is transitive, i.e.,  $xRy$  and  $yRz \Rightarrow xRz$

These conditions can be shown to hold for subsumption by evaluating the condition for logical subsumption, which is a binary relation. The three conditions can be shown to hold by asserting their truth value when the condition for logical subsumption expressed in Equation 5.1 shown below is true.

$$C \sqsubseteq D \text{ iff } C \sqcap \neg D \rightarrow \emptyset \quad (5.1)$$

Equation 5.1 represents the tableau algorithm of subsumption in DL. Udoyen presents the mathematical proof for showing that Equation 5.1 satisfies above three conditions[128]. This proof is generic such that it is applicable to subsumption for any description logics.

In this research, the description logic  $\mathcal{AL}\mathcal{E}$  is used to represent and compare the concepts. In description logics, the subsumption is reduced to a satisfiability problem and a tableau algorithm is used to compute the subsumption as discussed in section 5.2.6.

Therefore, the mathematical proofs and properties discussed above are generically applicable to any description logics including  $\mathcal{AL}\mathcal{E}$ . Consequently, the subsumption computation in  $\mathcal{AL}\mathcal{E}$  is proven to be consistent and correct. The following sections discuss description logic implementations.

### 5.3 Description logics implementations

In this section, we present the description logics' implementation for representing design requirements and constructing manufacturing rule taxonomy. More specifically, description logic choice, design requirement representation and manufacturing rule taxonomy computation are discussed. Through these discussions, we study the applicability of description logics. First, atomic concepts and roles for describing design requirements are identified. The taxonomies for identifying atomic concepts are shown in Table 5.5.

**Table 5.5 Atomic concepts**

Accuracy Measurement (ANSI 14.5 1994 Standard)		Surface	Relative orientation	Accuracy measurement range
Surface finish		Flat	Opposite	< Value 1
Tolerance	Straightness	Cylindrical	Perpendicular	< Value 2
	Flatness	Conical	Angled	< Value 3
	Circularity	...	3D	< Value 4
	Cylindricity		Cross	< Value 5
	Perpendicularity		...	< Value 6
	Parallelism			< Value 7
	Angularity			< Value 8
	Position			...
	Symmetry			
	Concentricity			
	Circular runout			
	Total run out			
	Line profile			
Surface profile				

In Table 5.5, taxonomies of accuracy measurements, surfaces, relative orientations, and accuracy measurement ranges are shown. Those taxonomies are expansions of design requirements identified in Figure 3.1. The accuracy measurements taxonomy is constructed based on ANSI 14.5-1994 standard[23]. Among the many types of surfaces, only three are listed under surface taxonomy and more surface types can be added as needed. Three relative orientations are shown in the relative orientation taxonomy. Those are shown in Figure 4.2 and are derived by a simple meta rule: *If (accuracy measurement is specified on surface and its restriction is to be either “value 3” or “value 4” from Table 4.1) then (feasible part orientations are  $0^\circ$  or  $90^\circ$ ).* All other orientations in Figure 4.5 are basically the combinations of the three orientations shown in Table 5.5. There can be more relative orientations derived either by new simple meta-class rules or by combinations of those shown in orientation taxonomy. In such case, new orientations can be added as needed to Table 5.5. The accuracy measurement range taxonomy shows eight different ranges. The accuracy measurement ranges are classifications of numeric values into concepts as shown in Table 4.1. Therefore, more accuracy measurement value ranges can be added as needed. The key idea about the identified atomic concepts shown in Table 5.5 is that all the information we need to represent can be described by atomic concepts including numeric ranges. Also, all the information that has not been explored falls into one of the four categories in Table 5.5. Based on the taxonomy in Table 5.5, examples of description logic implementations are shown in Table 5.6.

**Table 5.6 Implementation of atomic concept in description logics**

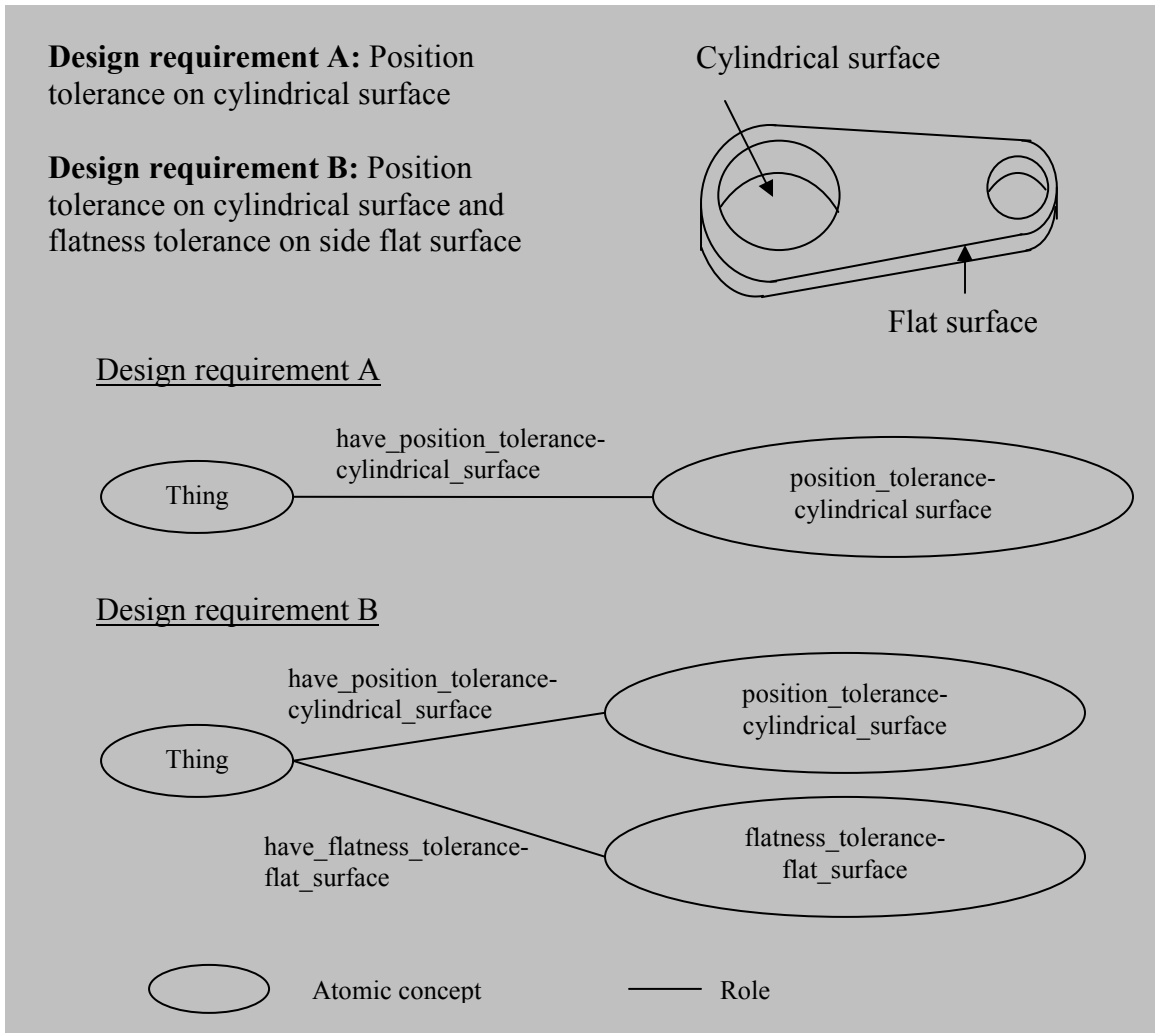
Accuracy measurement and surface combination	Relative orientation	Restriction level
Surface finish-Flat surface	Opposite	Level 1
Circularity tolerance-Cylindrical surface	Perpendicular	Level 2
Parallelism tolerance-Conical surface	Angled	Level 3
Position tolerance-Cylindrical Surface	3D	Level 4
Flatness tolerance-Flat Surface	Cross	Level 5
...	...	...

In Table 5.6, the first column represents the combinations of surface and accuracy measurement (accuracy measurement models). Models of accuracy measurement are developed for combinations of accuracy measurement and surface type. Hence, such combinations are implemented as shown in Table 5.6. Restriction level in the third column is the mapping of accuracy measurement value in Table 4.1 to classified atomic concepts. Such atomic concepts are used to describe the severity of restrictions on accuracy measurement. Then, the corresponding feasible space of process variables is identified by Table 4.1. The implemented atomic concepts in Table 5.6 basically represent the properties of design requirements. In other words, the design requirements are derived by addition of such atomic concepts. Hence, we need relations (roles) that can be used to link those atomic concepts to form design requirements. Such roles are identified and shown in Table 5.7.

**Table 5.7 Implementation of relation (role) in description logics**

Accuracy measurement and surface combination	Relative orientation	Restriction level
<ul style="list-style-type: none"> <li>• have_Surface_finish-Flat_Surface</li> <li>• have_Circularity_tolerance-Cylindrical_surface</li> <li>• have_Parallelism_tolerance-Conical_surface</li> <li>• ...</li> </ul>	<ul style="list-style-type: none"> <li>• have_OppositeOrientation</li> <li>• have_PerpendicularOrientation</li> <li>• have_AngledOrientation</li> <li>• have_3DOrientation</li> <li>• have_CrossOrientation</li> </ul>	<ul style="list-style-type: none"> <li>• have_RestrictionLevel</li> </ul>


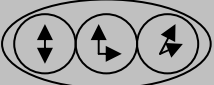
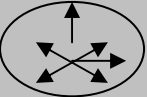
The first column in Table 5.7 shows the various roles that specify combinations of accuracy measurement and surface type. The second column shows the roles for specifying relative orientations and the third column shows the roles for specifying restriction level. All the roles start with the verb ‘have’. This is just a preference of forming design requirements. In this research, we treat a design requirement as a thing or object that becomes more specific and complex by adding atomic concepts as its properties. An example is shown in Figure 5.6.



**Figure 5.6 Example of simple design requirement formation**

In Figure 5.6, two examples of design requirements (A and B) are developed using the robot arm example. Requirement A can be explained as a thing that has a position tolerance on a cylindrical surface. Then, requirement B can be explained as a thing that has design requirement A and a flatness tolerance specified on a flat surface. Figure 5.6 is a simple example of how atomic concepts and roles are used to describe design requirements. More complex design requirements can also be described using a similar approach. Figure 5.7 shows such examples. The examples shown in Figure 5.7 are based on the meta rule hierarchy shown in Figure 4.5. Corresponding example rules

are also developed to show condition part representations (design requirements). The design requirements shown in Figure 5.7 assume the restriction level to be at value 4 in Table 4.1. To be more concise and precise, words are used rather than nodes and edges to describe design requirements.


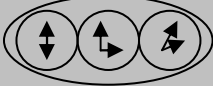
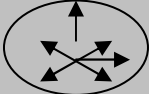
<p><b>MR1</b></p>  <p>A thing that (has opposite orientation)</p>	<p><b>R1</b></p> <p>A thing that (have opposite orientation) and (have surface finish on cylindrical surface) and (have flatness tolerance on flat surface)</p>
<p><b>MR2</b></p>  <p>A thing that (have opposite orientation) and (have perpendicular orientation) and (have angled orientation)</p>	<p><b>R2</b></p> <p>A thing that (have opposite orientation) and (have surface finish on cylindrical surface) and (have flatness tolerance on flat surface) and (have surface finish on cylindrical surface) and (have position tolerance on conical surface) and (have flatness tolerance on flat surface)</p>
<p><b>MR3</b></p>  <p>A thing that (have opposite orientation) and (have perpendicular orientation) and (have angled orientation) and (have 3D orientation) and (have cross orientation)</p>	<p><b>R3</b></p> <p>A thing that (have opposite orientation) and (have perpendicular orientation) and (have angled orientation) and (have 3D orientation) and (have cross orientation) and (have surface finish on cylindrical surface) and (have position tolerance on conical surface) and (have flatness tolerance on flat surface) and (have parallelism tolerance on conical surface)</p>

*Where MR: meta-class level manufacturing rule  
R: class level manufacturing rule*

**Figure 5.7 Meta rule representations by atomic concepts and roles**

The rules are derived from meta rules by adding arbitrary combinations of surface types and accuracy measurements. As shown in Figure 5.7, more complex

manufacturing rules are formed by adding more atomic concepts. More specifically, design requirements become more complex as more relative orientations are added to them. As described in Figures 4.5 and 4.6, increasing the complexity of design requirements decreases the feasible space of process variables. In the previous chapter, we have shown that addition of relative orientations of surfaces where accuracy measurements are specified decreases the feasible space of part orientations as shown in Figure 4.5. This is why more complex design requirements can be derived by adding more atomic concepts (relative orientations). This fact is important in making an appropriate description logic selection. In selecting description logic, we need to consider the least expressive one because of the intractability in most description logics. Consequently, the description logic  $\mathcal{AL}\mathcal{E}$  is chosen for representation. Description logic representation of Figure 5.7 is shown in Figure 5.8.

<p><b>MR1</b></p>  <p>Thing <math>\sqcap</math>  <math>\forall, \exists \text{haveOppositeOrientation.Opposite}</math></p>	<p><b>R1</b></p> <p>Thing <math>\sqcap</math>  <math>(\forall, \exists \text{haveOppositeOrientation.Opposite}) \sqcap</math>  <math>(\forall, \exists \text{haveSurface\_finish-Cylindrical\_surface.Surface\_finish-Cylindrical\_surface}) \sqcap</math>  <math>(\forall, \exists \text{haveFlatness\_tolerance-Flat\_surface.Flatness\_tolerance-Flat\_surface})</math></p>
<p><b>MR2</b></p>  <p>Thing <math>\sqcap</math>  <math>(\forall, \exists \text{haveOppositeOrientation.Opposite}) \sqcap</math>  <math>(\forall, \exists \text{havePerpendicularOrientation.Perpendicular}) \sqcap</math>  <math>(\forall, \exists \text{haveAngledOrientation.Angled})</math></p>	<p><b>R2</b></p> <p>Thing <math>\sqcap</math>  <math>(\forall, \exists \text{haveOppositeOrientation.Opposite}) \sqcap</math>  <math>(\forall, \exists \text{havePerpendicularOrientation.Perpendicular}) \sqcap</math>  <math>(\forall, \exists \text{haveAngledOrientation.Angled}) \sqcap</math>  <math>(\forall, \exists \text{haveSurface\_finish-Cylindrical\_surface.Surface\_finish-Cylindrical\_surface}) \sqcap</math>  <math>(\forall, \exists \text{haveSurface\_finish-Flat\_surface.Surface\_finish-Flat\_surface}) \sqcap</math>  <math>(\forall, \exists \text{haveFlatness\_tolerance-Flat\_surface.Flatness\_tolerance-Flat\_surface})</math></p>
<p><b>MR3</b></p>  <p>Thing <math>\sqcap</math>  <math>(\forall, \exists \text{haveOppositeOrientation.Opposite}) \sqcap</math>  <math>(\forall, \exists \text{havePerpendicularOrientation.Perpendicular}) \sqcap</math>  <math>(\forall, \exists \text{haveAngledOrientation.Angled}) \sqcap</math>  <math>(\forall, \exists \text{have3DOrientation.3D}) \sqcap</math>  <math>(\text{haveCrossOrientation.Cross})</math></p> <p><i>Where MR: meta-class level manufacturing rule</i>  <i>R: class level manufacturing rule, <math>\forall, \exists</math> represents <math>\forall \sqcap \exists</math> i.e.,</i>  <math>\forall, \exists \text{hasOppositeOrientation.Opposite} \Rightarrow</math>  <math>\forall \text{hasOppositeOrientation.Opposite} \sqcap</math>  <math>\exists \text{hasOppositeOrientation.Opposite}</math></p>	<p><b>R3</b></p> <p>Thing <math>\sqcap</math>  <math>(\forall, \exists \text{haveOppositeOrientation.Opposite}) \sqcap</math>  <math>(\forall, \exists \text{havePerpendicularOrientation.Perpendicular}) \sqcap</math>  <math>(\forall, \exists \text{haveAngledOrientation.Angled}) \sqcap</math>  <math>(\forall, \exists \text{have3DOrientation.3D}) \sqcap</math>  <math>(\text{haveCrossOrientation.Cross}) \sqcap</math>  <math>(\forall, \exists \text{haveSurface\_finish-Cylindrical\_surface.Surface\_finish-Cylindrical\_surface}) \sqcap</math>  <math>(\text{haveSurface\_finish-Flat\_surface.Surface\_finish-Flat\_surface}) \sqcap</math>  <math>(\forall, \exists \text{haveFlatness\_tolerance-Flat\_surface.Flatness\_tolerance-Flat\_surface}) \sqcap</math>  <math>(\forall, \exists \text{haveParallelism\_tolerance-Conical\_surface.Parallelism\_tolerance-Conical\_surface})</math></p>

**Figure 5.8 Description logic (ALE) representation of manufacturing rules**

As shown in Figure 5.8, all the design requirements are represented with  $\mathcal{ALE}$ .

More specifically, the intersection operator ( $\sqcap$ ), the universal restriction operator ( $\forall$ ), and

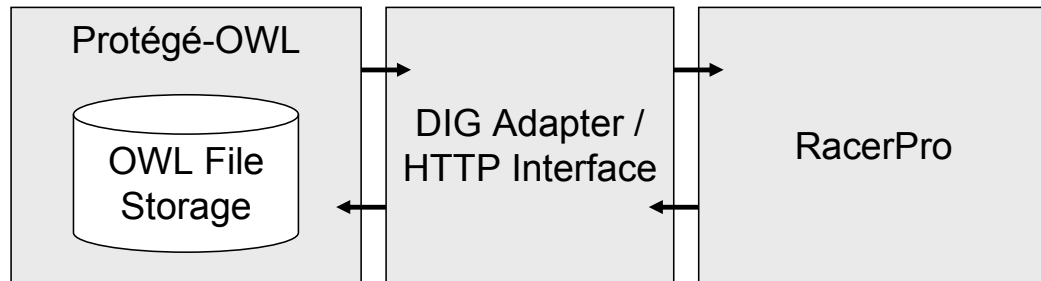
the full existential operator ( $\exists$ ) are sufficient to describe the entire design requirement. The existential operator is used to state that there is at least one particular property (role with role filler). For example, the condition part of MR1 with only full existential operator states that there is at least one property that is “ $\exists$ haveOppositeOrientation.Opposite”. Then, the universal restriction operator is used to further restrict the role filler such that the role “ $\exists$ haveOppositeOrientation” can only take role filler “Opposite”. Then, the description logic representation of MR1 is interpreted as “a thing that must have role “ $\exists$ haveOppositeOrientation” and its role filler can only be “Opposite”. In other words, the condition part of MR1 states that “a thing that has only opposite orientation”. In this research, the majority of the roles are developed such that they take unique role filler. Hence, the universal restriction operator is used to restrict the role fillers. Also in description logics, it is preferred to provide stronger concept definition by using universal restriction operator due to the open world reasoning characteristic [129]. The universal restriction operator is part of the basic attributive language [ $\mathcal{AL}$ ]. Therefore, usage of universal restriction operator does not cause additional expressivity or increase in computational complexity.

The three operators ( $\exists$ ,  $\forall$ ,  $\sqcap$ ) are sufficient because addition of atomic concepts can be fully satisfied by those operators[119]. Other description logics provide more operators such as full negation ( $\neg$ ), union ( $\sqcup$ ), role hierarchy, etc. However, such operators are not necessary for adding atomic concepts. Therefore,  $\mathcal{AL}\mathcal{E}$  is chosen as the minimum expressive description logic for this research. Using  $\mathcal{AL}\mathcal{E}$ , the design requirements are represented and a meta rules taxonomy is computed. The significance

of identifying appropriate description logic is that it ensures the applicability of the subsumption algorithm. As discussed in section 5.2.6, the subsumption performance is determined to be nondeterministic polynomial time for  $\mathcal{AL}\mathcal{E}$ . Therefore, we expect the performance of taxonomy computation in this research to be in nondeterministic polynomial time. The following section describes the software environment and manufacturing rules encoding.

#### 5.4 Software environment for encoding the manufacturing rules

The description logic implementation is performed in a software environment as shown in Figure 5.9[71].



**Figure 5.9 Implementation environments for design requirement and manufacturing rule taxonomy representation**

- **Protégé-OWL**

Protégé-OWL editor is an extension of Protégé that supports developing ontologies using the Web Ontology Language (OWL) [130-132]. Protégé-OWL is an open-source ontology development environment with functionality for editing OWL based ontologies[133].

- **RacerPro**

RacerPro is a knowledge representation system that implements a highly optimized tableau calculus for various DLs. RacerPro provides algorithms for reasoning at the TBox and ABox level. RacerPro is the back-end reasoner used within Protégé-OWL. RacerPro implements the HTTP interface called DIG for connecting with Protégé-OWL. RacerPro was initially developed at the University of Hamburg, Germany. RacerPro is actively supported and future releases are developed at Concordia University in Montreal, Canada, and at the University of Applied Sciences in Wedel near Hamburg, Germany. While RacerPro can be used for developing DLs knowledge bases, it is primarily used in this research for reasoning services at the TBox (concept terminology) level. RacerPro supports the SHIQ (equivalent to  $ALCQHI_{R+}$ ) representation, although less expressive languages can be used. The SHIQ language extends the basic ALC language through the inclusion of additional restrictions and axioms including qualifying number restrictions, role hierarchies, inverse relationships, and transitive roles. Similar to other knowledge-based systems, RacerPro is based on the open world assumption (OWA). The OWA states that *what is not explicitly stated in the knowledge base cannot be proven to be true or false*[134].

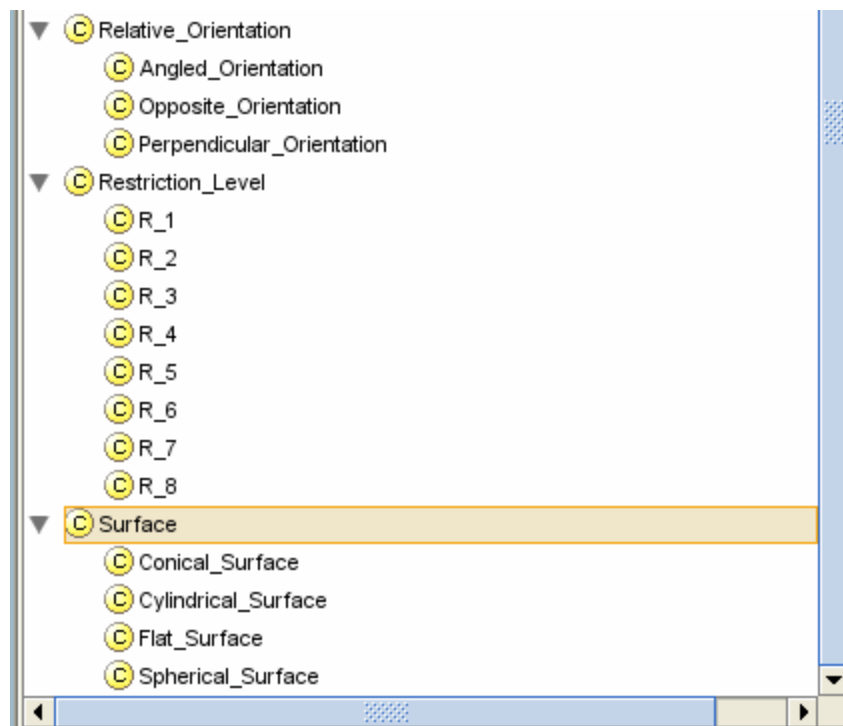
- **OWL DL File Storage**

OWL DL is a standard XML-based language that is used for explicitly representing the meaning of terms in vocabularies and the relationships between those terms. OWL DL provides support for developing ontologies using DLs representations. OWL is a standard ontology language by W3C. OWL is the markup language used to store DL ontologies[135].

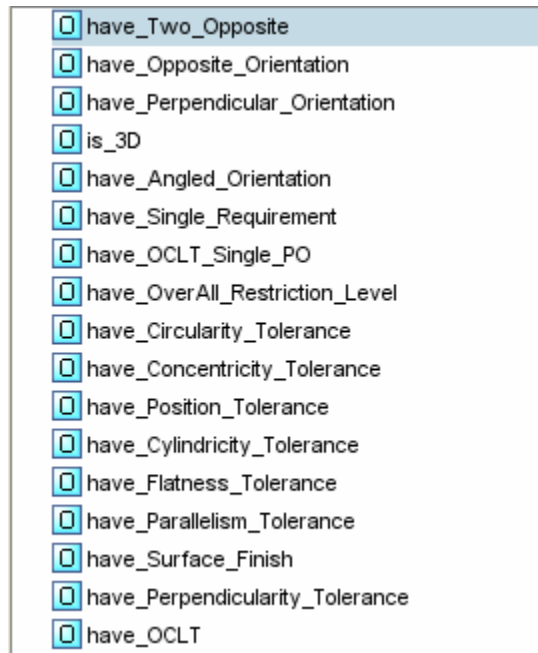
- **DIG Interface**

The DIG Interface is a standardized interface based on XML for DLs systems. The DIG interface is developed by the DL Implementation Group (DIG). The DIG interface is an emerging standard for providing access to description-logic reasoning via an HTTP-based interface to a separate reasoning process[136, 137].

Using the above environment, atomic concepts and roles are implemented in Protégé as shown in Figures 5.10 and 5.11.

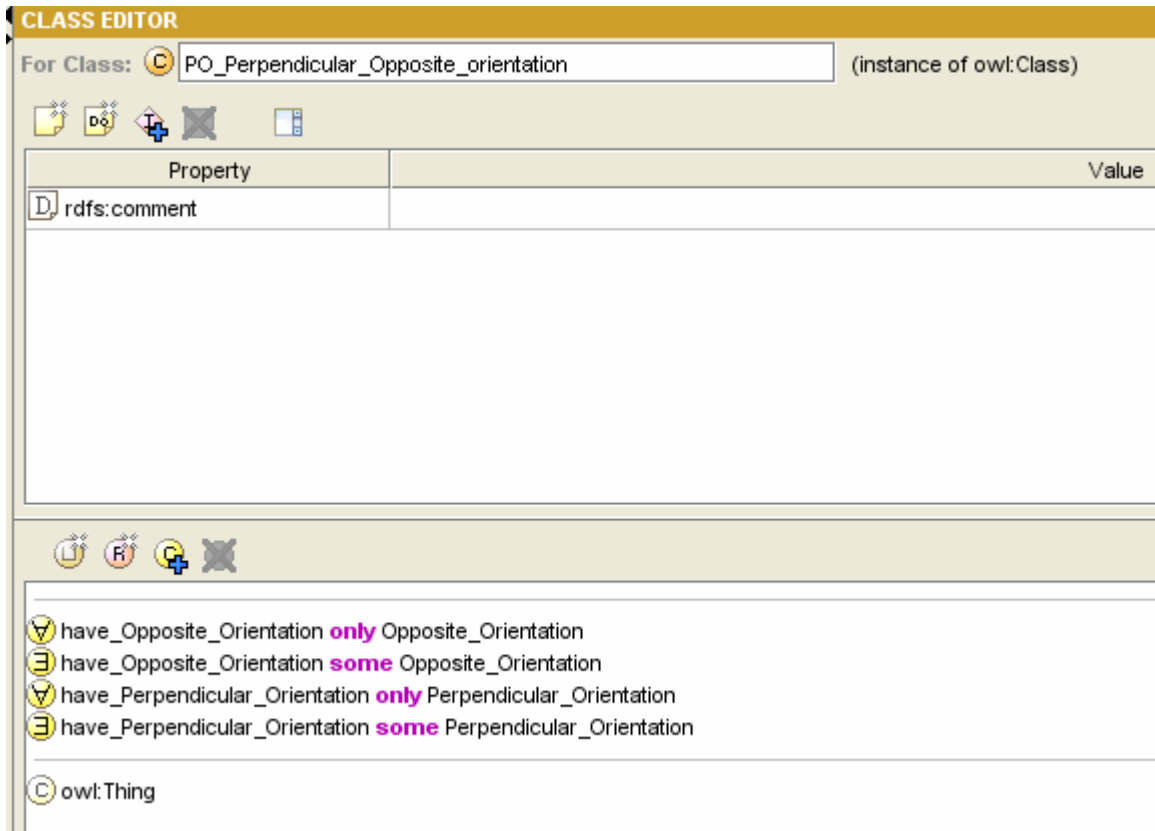


**Figure 5.10 Atomic concept implementation using Protégé**



**Figure 5.11 Role implementation using Protégé**

Figures 5.10 and 5.11 are part of the Protégé graphical user interface environment that allows the user to explicitly insert atomic concepts and roles. Then, the design requirements are implemented using such concepts and roles. An example is shown in Figure 5.12.



**Figure 5.12 Meta rule implementation using Protégé**

The design requirement shown in Figure 5.12 is the condition part of the meta rule shown in Figure 5.13.

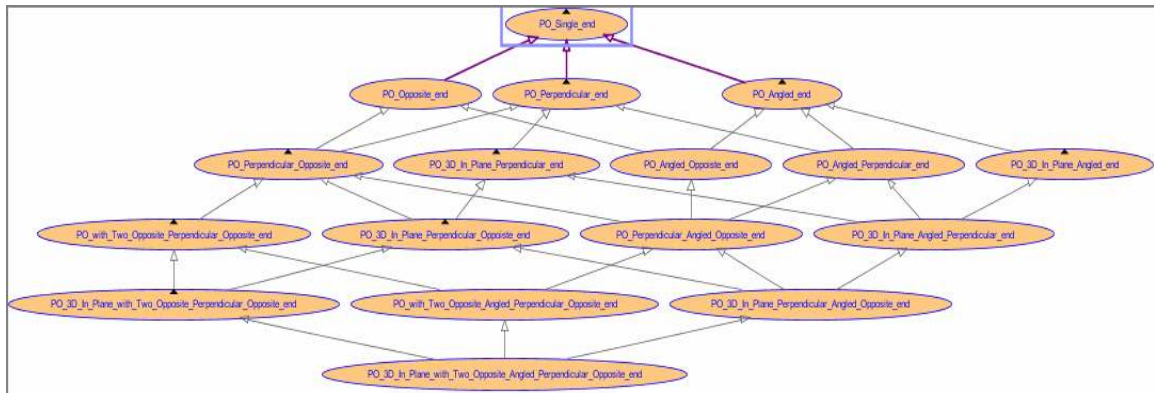
*If (accuracy measurement specified on surfaces and their relative orientation is opposite and perpendicular) then (number of feasible part orientations are 3)*

**Figure 5.13 Meta rule example**

In Figure 5.12, the meta rule in Figure 5.13 is modeled by addition of relative orientation characteristics: opposite and perpendicular. More specifically, the condition part of the meta rule in Figure 5.13 shows that the relative orientations are both perpendicular and opposite. Hence, the relative orientation characteristics are encoded and shown in the bottom window of Figure 5.12. Those relative orientations are

manually encoded directly in the bottom window. The modeling is done such that the role and corresponding atomic concept are paired and inserted as a property for the concept that is defined. For example, the role `have_Opposite_Orientation` is paired with concept `Opposite_Orientation` and inserted using the universal quantifier ( $\forall$ ) and the full existential operator ( $\exists$ ). Similarly, the properties regarding perpendicular orientation are inserted. Then, the represented meta rule in Figure 5.12 is interpreted as “a thing that has perpendicular and opposite orientation” as shown at the bottom window. The result part of the meta rule in Figure 5.13 is not used for representing the meta rules. However, the result parts are used to inspect the correctness of the meta rule modeling by forming the quantity Qs, introduced in Figure 4.18. The details are presented in a later paragraph.

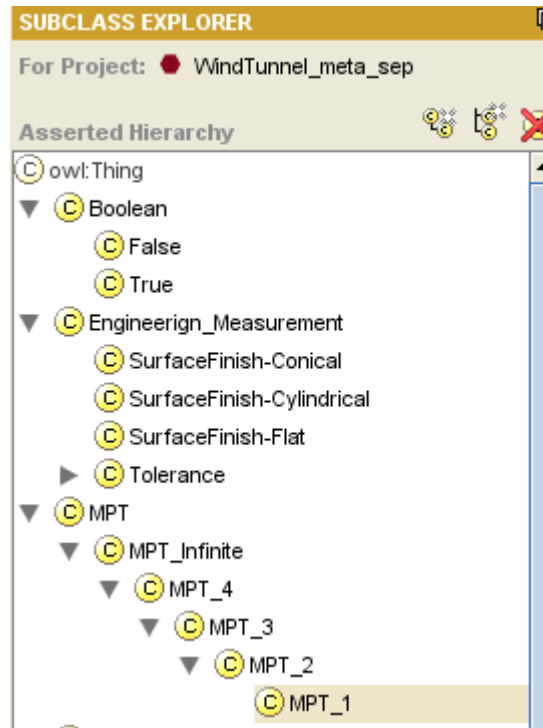
By using such techniques, all the 17 meta rules shown in Figure 4.5 are indexed and the taxonomy is computed as shown in Figure 5.14.



**Figure 5.14 Meta rule taxonomy computation**

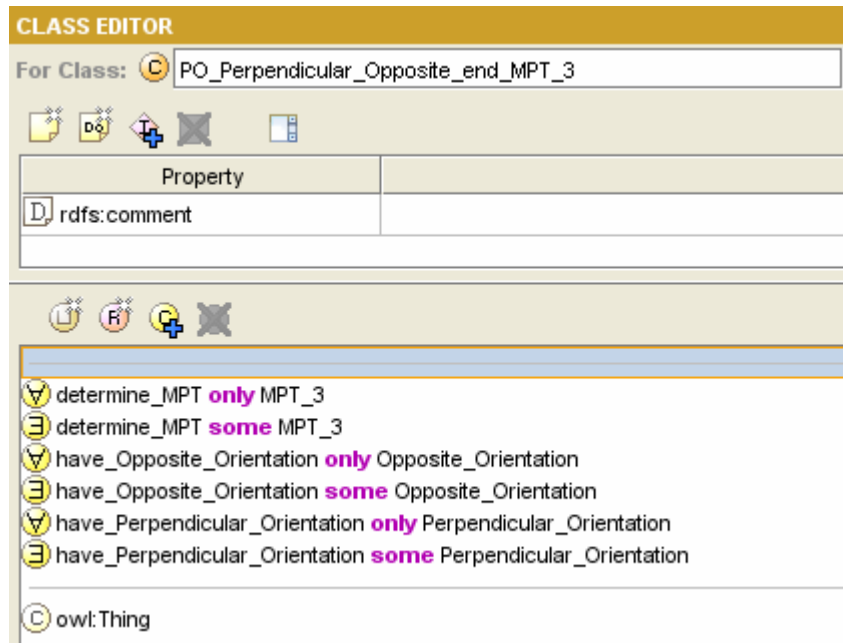
The taxonomy shown in Figure 5.14 exactly matches the one in Figure 4.5. In other words, using description logic  $\mathcal{AL}\mathcal{E}$ , it is possible to represent design requirements, index manufacturing rules, and compute the taxonomy.

In section 4.3, the Qs taxonomy is introduced to relate the MPTs to the corresponding design requirements. To represent and compute Qs taxonomy, the additional atomic concepts and their taxonomic relation are introduced and shown in Figure 5.15.



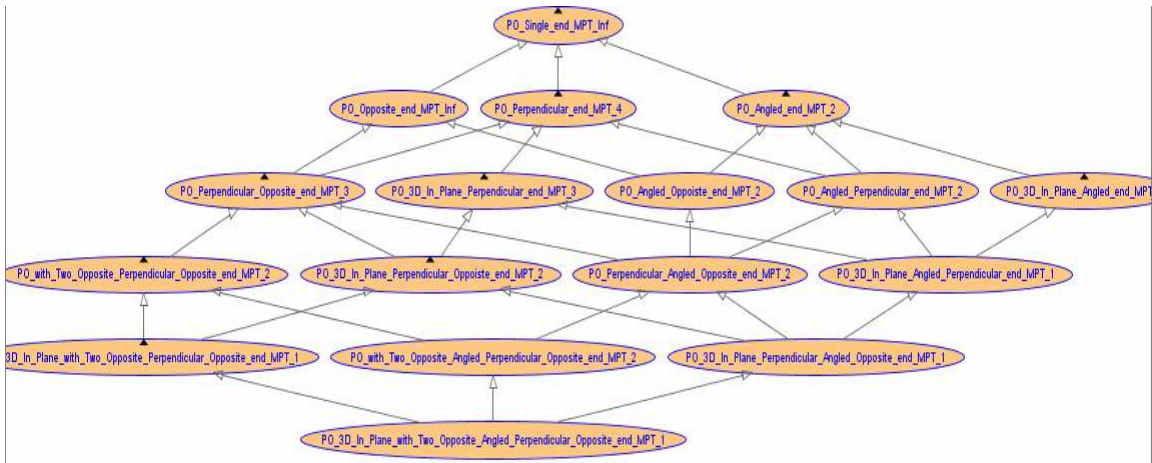
**Figure 5.15 Addition atomic concepts for representing Qs**

In Figure 5.15, atomic concepts called MPT are added. There are five MPTs. The MPTs correspond to the feasible spaces of part orientation. For example, MPT\_Infinite, MPT\_4, MPT\_3, MPT\_2, and MPT\_1 correspond to the number of feasible spaces of part orientation; infinite, 4, 3, 2, and 1 respectively. Their subsumption relation is manually enforced in Figure 5.15. The MPTs represent the result part of the meta rules in Figures 4.4 and 4.5. Then, the representation of Q is formed by combining the condition and result part of the meta rules. Figure, 5.16 shows an example.



**Figure 5.16 Example of Q representation**

In Figure 5.16, the Q representation for the meta rule in Figure 5.14 is shown. As discussed before, the condition and result part of the meta rules in Figure 4.5 are used to represent Q. For example, the bottom window in Figure 5.16 shows that relative orientations (opposite and perpendicular) are combined with MPT\_3 by the intersection operator. Therefore, the Q in Figure 5.16 is interpreted as “a thing that has opposite and perpendicular orientation and determines MPT\_3”. For representing Qs,  $\mathcal{AL}\mathcal{E}$  is sufficient because the additional concepts (MPTs) are represented and added as atomic concepts. By a similar approach, Q for each meta rule in Figure 4.5 is represented and the Qs taxonomy is produced. Figure 5.17 shows the Qs taxonomy.



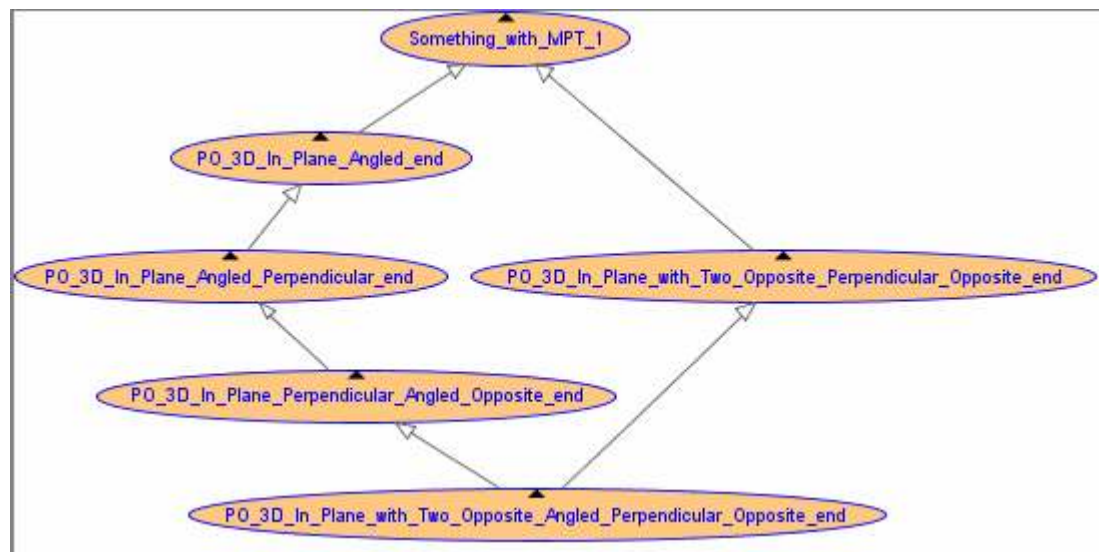
**Figure 5.17 Qs taxonomy**

After computing the Qs taxonomy, it is inspected to check if all the propositions in Figure 4.18 are satisfied. Also, the Qs taxonomy is compared to the meta rules taxonomy in Figure 5.14. It is found that the Qs taxonomy in Figure 5.17 satisfies all the propositions in Figure 4.18 and that it matches exactly to the meta rules taxonomy in Figure 5.14. This concludes that the meta rules shown in Figures 4.4, 4.5 and 5.14 satisfy properties shown in Figure 4.17 (structure preserving properties). Hence, subsumption relation in design and process planning can be used to map the design and process planning domains.

As discussed section 4.3, the manufacturing rules taxonomy relates the design requirements domain to the process planning domain. Also, the Qs taxonomy can relate the manufacturing domain to the design requirements domain. In both taxonomies, the subsumption can be used to identify appropriate MPTs or design requirements. This is because both taxonomies are computed using subsumption.

Figure 5.18 shows an example of determining all the design requirements that require MPT\_1 (from Figure 5.15) using subsumption. The focus of this example is to

demonstrate that the Qs taxonomy and subsumption in DL can map the given MPT to the appropriate design requirements. Figure 5.18 is constructed by following the two steps. First, a concept node that has single property “determine\_MPT.MPT\_1” is added to the Qs taxonomy. Second, the Qs taxonomy is re-computed. Figure 5.18 only shows the Qs that are subsumed by the added concept node (has single property “determine\_MPT.MPT\_1”).



**Figure 5.18 Example of mapping process planning to design requirements**

In Figure 5.18, the given process plan (MPT\_1) is used to identify all the design requirements that can be satisfied by the given process plan (MPT\_1) using subsumption. Basically, the design requirements of all the descendants of the top node (Something\_with\_MPT\_1) are the ones that can be satisfied by the given process plan (MPT\_1). In the next chapter, in depth discussion of relating design requirements to relevant DFM problems through mapping the design and process planning domains are discussed.

In short, the description logic  $\mathcal{AL}\mathcal{E}$  provides a minimum level of expressivity in representing design requirements. The subsumption in DL can be utilized to systematically compute manufacturing rules taxonomy. Also, subsumption provides mathematical means to map the design domain and the process planning domain. The complete information models for design requirements, manufacturing rules, and Qs are encoded in  $\mathcal{AL}\mathcal{E}$  and presented in Appendix A.

### 5.5 Hypotheses validation

In this chapter, hypotheses 1 and 2 in Table 5.8 are validated theoretically.

**Table 5.8 Research questions and hypotheses (1 and 2)**

Questions	Hypotheses
Q1. How should design requirement be represented?	H1. Description logics enable representation of design requirements
Q2. How should the design and process planning domains be mapped?	H2. Subsumption in DL enables mathematical mapping between design domain and process planning domain

In this chapter, an appropriate description logic is selected and justified for representing the design requirements. Through Figures 4.4 and 4.5, it is found that complex design requirements are derived by combining simpler design requirements. Hence, the subsumption relations among design requirements are caused by addition of simple design requirements as shown in Figures 5.6 and 5.7. In description logic, the simplest way to model this subsumption relation is by representing the simplest concepts as atomic concepts and deriving complex concepts by addition of atomic concepts. To accomplish this, a minimum expressive description logic is identified to be  $\mathcal{AL}\mathcal{E}$ . This investigation and justification serve as the theoretical validation of hypothesis 1.

Using  $\mathcal{AL}\mathcal{E}$ , the discovered manufacturing rules in Figure 4.4 are represented by their condition part and the corresponding manufacturing rules taxonomy is computed using subsumption (Figure 5.14). Also, the corresponding Qs taxonomy is computed (Figure 5.17). The structure of both taxonomies is the same and agrees with the expected taxonomy (Figure 4.5). The ability to compute both taxonomies using subsumption in DL enables subsumption in DL to map the two domains as shown in Figures 4.16, 4.20, and 5.18. Hence, this investigation and justification serve as the theoretical validation of hypothesis 2.

## 5.6 Summary

In this chapter, description logics' implementation for representing the design requirements and computing the manufacturing rules taxonomy is discussed. Specifically, the justification of selecting description logics over other formalism is presented in sections 5.1 and 5.2. Then, the appropriate description logics' selection by identifying the minimum expressivity for representing the design requirements is presented in section 5.3. Then, the manufacturing rules and the corresponding Qs taxonomy computations are presented in section 5.4.

As discussed previously, many of the description logics are known to be theoretically intractable in their inference services. There are trade offs in expressivity and computational complexities. Hence, methods or systems that are built using description logics typically need to identify the minimum expressivity required so that the corresponding computational complexity can be identified. Without such identification of computational complexity, the performance of the method or systems is largely unknown. In this research, in depth investigation for applying DL in design for additive

manufacturing is performed. Through this investigation, minimum expressivity and appropriate description logic ( $\mathcal{AL}\mathcal{E}$ ) for representing the design requirements are identified and justified. Also, the corresponding theoretical computational complexity is identified. Table 5.9 summarizes the new knowledge discovered through the investigation.

**Table 5.9 New knowledge discovered in EIM**

EIM
<ul style="list-style-type: none"> <li>• Identification and justification of description logic (<math>\mathcal{AL}\mathcal{E}</math>) for representing and computing manufacturing rules taxonomy <ul style="list-style-type: none"> <li>○ Information models for the design requirements</li> <li>○ Identification and justification of minimum expressivity required for representing design requirement information models</li> <li>○ Identification and justification of appropriate description logic (<math>\mathcal{AL}\mathcal{E}</math>)</li> </ul> </li> <li>• Identification of the corresponding computational complexity for subsumption (non deterministic polynomial time)</li> </ul>

## **CHAPTER 6: RETRIEVAL METHOD (RETRIEVAL AND RANKING)**

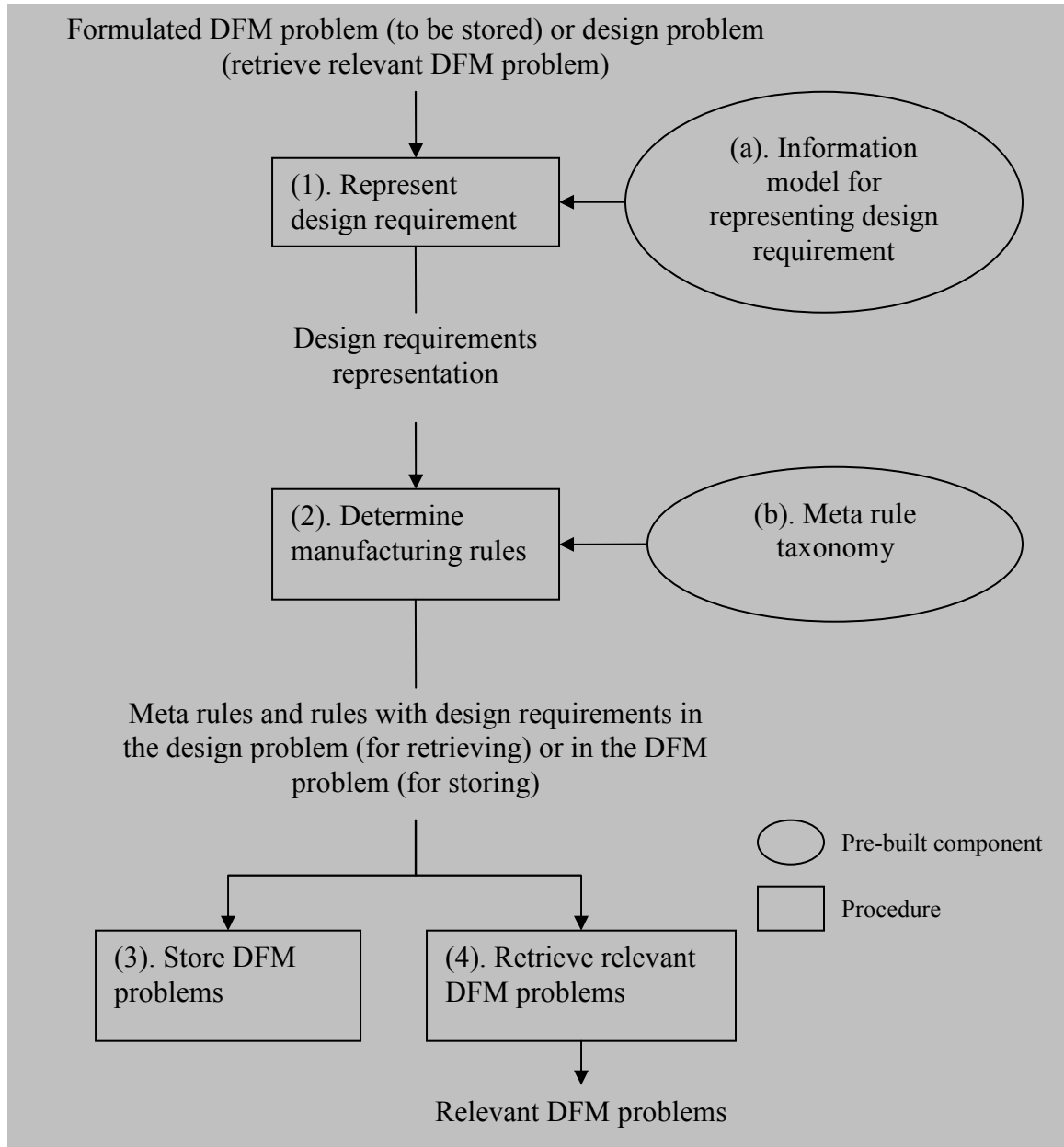
As discussed in section 1.1.4, the retrieval method consists of four components including an information model, a meta rule taxonomy, a storing algorithm, and a retrieval algorithm. An information model and meta rules taxonomy development are discussed in chapter 5. In this chapter, developments of storing and retrieval algorithms are discussed. Through this process, the retrieval method development is completed. First, the development of the storing and retrieval algorithms is discussed (section 6.1). The specific requirements that the storing and retrieval algorithms have to satisfy are presented. Then, the algorithms are developed such that the identified requirements are satisfied. Using a robot arm example, the storing and retrieval procedures are demonstrated (section 6.2). Through the completion of the retrieval method, the theoretical validation of hypotheses 3 and 4 is discussed (section 6.3). Then, the theoretical structure validation discussed in section 1.3.1 is concluded. Finally, the new knowledge contribution is discussed (section 6.4).

### **6.1 DFM framework: components**

In this section, the topics including overview and components of the retrieval method including information modeling, meta rules taxonomy, storing algorithm, and retrieval algorithm are discussed. Through the overview, we present the overall structure of the retrieval method and the needs that the retrieval method should satisfy. Then, storing algorithm and retrieval algorithm are discussed.

### 6.1.1 Overview

To realize the method described in Figure 1.16, a detailed description of the DFM framework is developed and shown in Figure 6.1.



**Figure 6.1 DFM framework components**

In short, the DFM framework consists of four process components and two pre-built components. The four process components are:

- (1) Representing design requirement (manual procedure)
- (2) Determining manufacturing rules from design requirements
- (3) Structuring the DFM problem repository
- (4) Retrieving and ranking the relevant DFM problems

And the two pre-built components are:

- (a). Information model for representing design requirements
- (b). Meta rule taxonomy

In Figure 6.2, the procedures for storing and retrieving DFM problems based on given design requirements are described.

Storing	Retrieving
<ol style="list-style-type: none"> <li>1. Using information model (a) for representing design requirements, represent the design requirements belong to DFM problem that need to be stored</li> <li>2. Using represented design requirements and meta rule taxonomy (b), determine the appropriate manufacturing rules (meta rule and rule) for the represented design requirements</li> <li>3. Based on the manufacturing rules (meta rule and rule) that are determined, store the DFM problem at appropriate position (structuring DFM problem repository)</li> </ol>	<ol style="list-style-type: none"> <li>1. Using information model (a) for representing design requirements, represent the design requirements belong to the given design problem</li> <li>2. Using represented design requirements and meta rule taxonomy (b), determine the appropriate manufacturing rules (meta rule and rule) for the represented design requirements</li> <li>3. Based on the manufacturing rules (meta rule and rule) that are determined, retrieve the relevant DFM problems and rank them</li> </ol>

**Figure 6.2 General procedures of storing and retrieving in DFM framework**

In Figure 6.2, the first step of the storing and retrieving procedure is to extract the design requirements from the design or DFM problem and to represent them by the pre-

built information models. In the retrieval method, this step is performed manually. Then, the represented design requirements are compared to the index (condition part) of the meta rules to identify an appropriate meta rule. This step is performed by finding the direct parent of the represented design requirements using subsumption. In this research, there will always be a meta rule whose condition part is the direct parent of the given design requirements. This is because the information models that represent the extracted design requirements are built such that they will always have a meta rule whose condition part is the direct parent of the represented design requirements. In other words, only the design requirements that match the condition part of the manufacturing rules are extracted. Then, the appropriate rule is formed by substituting specific surface types and accuracy measurements into the condition part of the meta rule. The specific surface types and accuracy measurements are in the given design requirements. Then the determined manufacturing rules are used to identify the relevant DFM problems (for retrieval) or position (for storing) for the new DFM problems in the repository. The following paragraphs describe how the determined manufacturing rules should be used to store and retrieve relevant DFM problems. Through which, the requirements that the storing and retrieving algorithms need to satisfy are discussed.

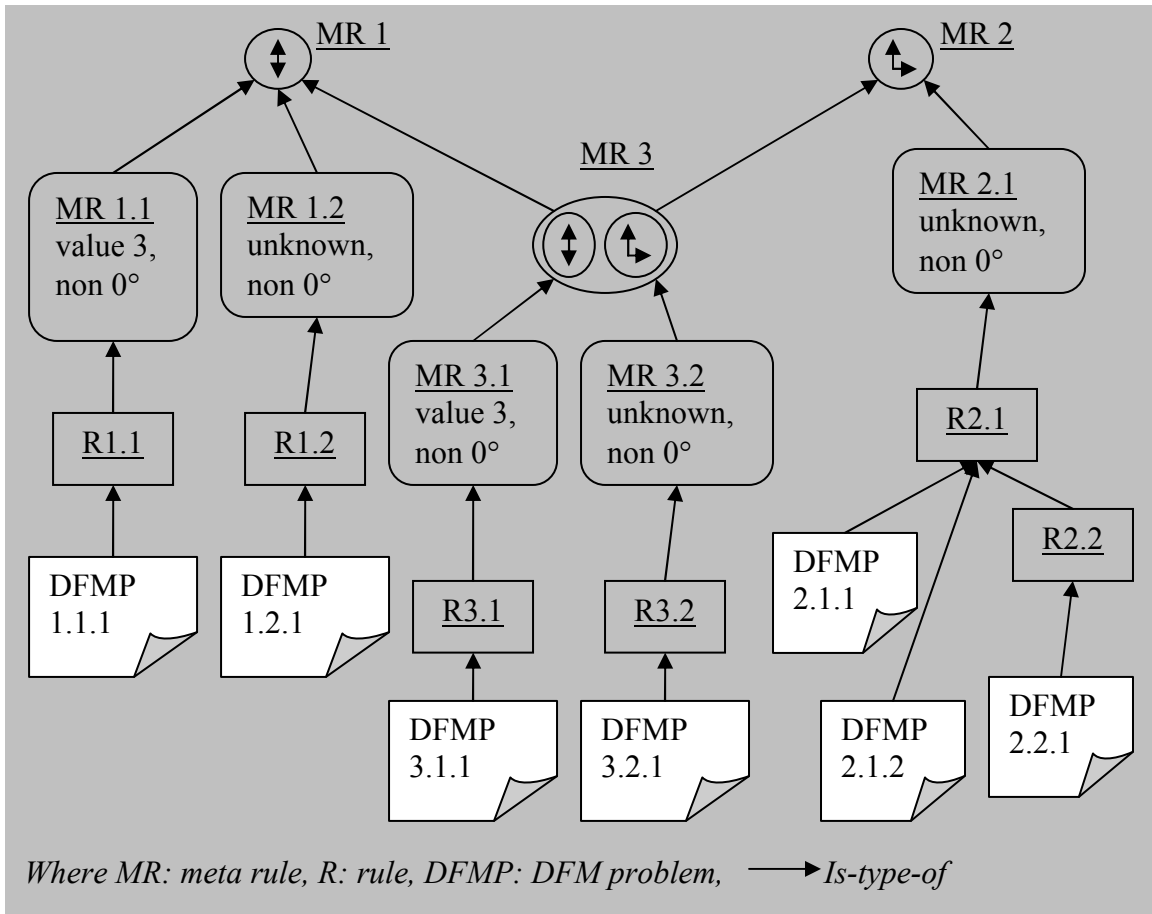
As defined in chapter 1, the relevant DFM problems provide appropriate solution strategy and accuracy models. Specifically, relevant DFM problems show feasible spaces of process variables and accuracy models to support formulating and solving a new DFM problem. However, due to the characteristics of open repository, there is no guarantee that the DFM problems that exactly match the needs are always available.

Hence, the DFM problems in the repository are classified such that the classification criteria always lead the given design requirements to the most relevant DFM problems.

In this research, the DFM problems are stored in the repository in the structure of manufacturing rule taxonomy. In other words, DFM problems are classified by the manufacturing rules. Then, the condition parts of the determined manufacturing rules are compared to the index of the manufacturing rules in the repository to locate the relevant DFM problem (retrieving) or to position the new DFM problem (storing). The condition part of the meta rules consists of the surface orientation and accuracy measurement values. The condition part of the rules consists of the specific accuracy measurements, values, surface types, and surface orientations. Therefore, the DFM problems under the same meta rule share the same feasible spaces of process variables due to having the same surface orientations and accuracy measurements values. The DFM problems under the same rules then share the same feasible spaces of process variables and models of accuracy measurements due to having the same accuracy measurements, values, surfaces types, and surface orientations. Hence the meta rules classify DFM problems based on the feasible spaces of process variables and the rules further classify the DFM problems based on the accuracy models. A detailed discussion of this is presented in chapters 3 and 4.

In Stereolithography, meta rules are discovered for three process variables: part orientation, layer thickness, and overcure. The meta rules for part orientation determine feasible spaces of part orientation for the given design requirements. The meta rules for layer thickness determine the minimum layer thickness that needs to be considered. The meta rules for overcure determine whether the overcure needs to be considered as a

process variable or not. As discussed in chapter 4, the meta rules are coupled due to the coupled nature of process variables in additive manufacturing. The complete meta rules are realized by supplementing the meta rules in Figure 4.5 by the meta rules in Figures 4.7 and 4.9. Hence the repository structure follows this pattern. In other words, DFM problems are classified based on meta rules for part orientation first. Then, the DFM problems are further classified by meta rules for layer thickness and overcure. Finally, DFM problems are classified by rules. To illustrate this, an example of repository structure is shown in Figure 6.3. Through the example in Figure 6.3, the specific needs from the storing and retrieving algorithms are discussed.



**Figure 6.3 DFM problem repository structure example**

In Figure 6.3, the examples of supplemented meta rules are presented (the meta rules in Figures 4.7 and 4.9 are supplemented to the ones in Figure 4.5). The three meta rules for the part orientation (MR x) are supplemented by the meta rules for layer thickness and overcure to derive supplemented meta rules (MR x.x). For example, the condition part of the MR 1 is “opposite orientation”. Then the condition part of the supplemented meta rule MR 1.1 is “opposite orientation, minimum accuracy measurement value at “value 3” from Table 4.1 on a surface that has to be in 90°, and surfaces in orientation that is other than flat in 0°”. Therefore, the supplemented meta rules inherit the relative orientations from the meta rules for part orientation. In the

condition part of the supplemented meta rules, “value 3” means that the accuracy measurement value that is specified on a surface that must be in  $90^\circ$  is “value 3” in Table 4.1. Hence the corresponding meta rule determines that the small value of layer thickness must be considered according to Table 4.1. The condition part “unknown” means everything other than the condition described above; hence the layer thickness is completely unknown. The condition part “non  $0^\circ$ ” means that there are surfaces in orientations other than flat in  $0^\circ$ . Hence, the overcure needs to be considered as process variable. In the example in Figure 6.3, all the relative orientations involve surfaces in  $90^\circ$  orientation. Therefore, all the meta rules determine that overcure must be considered as process variable. Under each supplemented meta rule (MR x.x), corresponding rules and DFM problems are attached. The condition part of the rules is all the condition part of directly subsuming meta rule and accuracy models. For example, the condition part of the rule R1.1 could be all the condition part of MR 1.1 and surface finish on cylindrical surface. In Figure 6.3, it is assumed that all the rules have different condition parts except R 2.1 and 2.2. The condition part of R 2.1 is assumed to subsume R 2.2 such that the condition part of R 2.2 has all the condition part of R 2.1 and more in terms of accuracy models. Then the design requirements in the DFM problems have the condition part of the subsuming rule. The taxonomic structure is assumed by assuming hierarchical relations in the condition parts of manufacturing rules and in the design requirements of the DFM problems. As discussed in Figures 6.1 and 6.2, the DFM problems repository is dynamically structured. In other words, the taxonomic structure of the repository expands as new DFM problems are introduced. Using the example in Figure 6.3, the requirements of storing and retrieving algorithms are discussed in the following sections.

#### 6.1.1.1 Specific requirements from storing algorithm

There are three general cases of expanding the DFM problem repository. The first case is where a newly introduced DFM problem introduces a new meta rule and a rule. The second case is where a newly introduced DFM problem introduces a new rule only. The third case is where a newly introduced DFM problem does not introduce any new manufacturing rules in the repository. For example, let's assume that DFM problems 1.1.1, 1.2.1, 3.1.1, and 2.1.1 are the only problems in the repository in Figure 6.3. Then, the corresponding meta rules (1, 2, 3, 1.1, 1.2, 2.1, 3.1) and rules (1.1, 1.2, 2.1, 3.1) are in the taxonomic structure in the repository. The additional introduction of DFM problem 3.2.1 introduces the meta rule MR 3.2 and rule R 2.1 into the repository. Therefore the repository taxonomy needs to be recomputed with the newly introduced meta rule and rule. In addition to that, introduction of DFM problem 2.2.1 introduces the new rule R 2.2 in the repository but not the new meta rule because DFM problem 2.2.1 shares a meta rule with DFM problem 2.1.1. Hence, the repository taxonomy needs to be recomputed with the newly introduced rule. Additional introduction of DFM problem 2.1.2 does not introduce any manufacturing rules because it shares manufacturing rules with DFM problem 2.1.1. Hence, no computation of manufacturing rules taxonomy is needed. This is because computation of taxonomy is only performed at the manufacturing rules level not at the DFM problems level. Storing of DFM problems is then assumed to be performed in a file systems or database system by tagging the problems with corresponding manufacturing rules information. In short, the storing algorithm needs to consider the three cases of DFM problems introduction.

### 6.1.1.2 Specific requirements from retrieval algorithm

The retrieval of relevant DFM problems supports process planning by providing feasible spaces of process variables and accuracy models. In Stereolithography or additive manufacturing in general, process planning is performed in the order of part orientation, layer thickness, and overcure[2, 10, 107]. To determine overcure, layer thickness needs to be determined and to determine layer thickness, part orientation needs to be determined. Therefore, DFM problems should be retrieved this order. In other words, DFM problems that support process planning for part orientation are identified first. Then, the identified DFM problems are further ranked based on the supporting capability of process planning for layer thickness and overcure. Finally, the retrieved DFM problems are further ranked based on the supporting capability of providing accuracy models. Table 6.1 shows an example of retrieval and ranking. In Table 6.1, the query is the condition part of rule R 3.1.

**Table 6.1 Retrieval example**

MR x	MR 3		MR 2		MR 1		
MR x.x	“at value 3 & non 0 <sup>o</sup> ”	“unknown & non 0 <sup>o</sup> ”	“at value 3 & non 0 <sup>o</sup> ”	“unknown & non 0 <sup>o</sup> ”	“at value 3 & non 0 <sup>o</sup> ”	“unknown & non 0 <sup>o</sup> ”	
R	3.1		3.2		2.1, 2.2	1.1	1.2
DFMP	3.1.1		3.2.1		2.1.1, 2.1.2, 2.2.1	1.1.1	1.2.1
Rank	1		2		3	4	5

The direct parent meta rule of the given query is MR 3.1. Then, the DFM problems that support process planning for part orientation need to be identified first. To accomplish this, the relative orientations from the condition part of MR 3.1 are extracted first. Then using the extracted relative orientations, all the equivalent and subsuming meta rules for part orientation (MR x) are identified. The DFM problems under

equivalent and subsuming meta rules are identified as the relevant DFM problems. The detailed justification of this procedure is discussed in chapters 3 and 4. In Figure 6.3, the equivalent meta rule for part orientation is MR 3 and the subsuming meta rules are MR 2 and 1. Hence, DFM problems under those meta rules are identified as relevant DFM problems. The identified DFM problems can further be ranked by their ability to support process planning for part orientation. Determining the feasible spaces of part orientation involves performing partial process planning on part orientation. Hence, DFM problems support process planning better when they provide closer feasible spaces to the determined feasible space (query) of part orientation. In the example above, the feasible spaces of part orientation for MR 3, 2, and 1 are 3, 4, and infinite. Hence, the identified DFM problems should be ranked in the order of MR 3, 2, and 1. The top row in Table 5.1 shows this.

Then, the DFM problems should be further ranked by ability to support process planning for layer thickness and overcure. The meta rule (MR 3.1) for layer thickness determines the minimum layer thickness that needs to be considered for process planning. The meta rule(MR 3.1) for overcure determines whether the overcure needs to be part of the process planning for the given design requirement or not. Hence, there are two ranks for layer thickness and overcure. For the layer thickness, the problems in the first rank are those that consider the determined minimum layer thickness (query) in the process planning. Then the problems in the second rank are those that do not consider the determined minimum layer thickness. For the overcure, further ranking is not performed because overcure should always be part of the process planning for the meta rules in Figure 6.3 based on the discussion in the section 4.1.4. Then, the problems in the second

rank are those that consider overcure to be completely unknown. From the give query (condition part of R 3.1), it is determined that there is an accuracy measurement specified on a surface that must be in  $90^\circ$  and its value is “value 3” in Table 4.1. Also, there are surfaces in orientations other than flat in  $0^\circ$ . Therefore, small value of layer thickness must be considered and overcure must be part of the process planning. The DFM problems are classified and ranked accordingly. This is shown in the second row of Table 6.1. The DFM problems for each MR x are further ranked such that the DFM problems that consider thin layer thickness and overcure in process planning are ranked higher.

Finally, the rules are used to further rank DFM problems. As discussed previously, rules are used to classify DFM problems based on accuracy models. In ranking based on accuracy models, the DFM problems that provide all the accuracy models should be definitely ranked higher than those that do not. Further ranking of DFM problems that provide partial accuracy models is not feasible because that involves subjective decisions. Therefore, there are two ranks for accuracy models. The DFM problems that contain all the required accuracy models are ranked higher than those that do not. In the example above, the query matches the condition part of R 3.1. In this example, it is assumed that the condition part of all the rules except R 2.1 and 2.2 are different in terms of accuracy models. Hence, all the DFM problems except DFM problem 3.1.1 do not have all the accuracy models required for the query. This is shown in the third and fourth row of Table 6.1. The rank is further divided into two in the third and fourth rows (R and DFMP) from the second row (MR x.x). Under each meta rule (MR x.x) in Table 6.1, the left column is for the rules and DFM problems that have all

the required accuracy models. Then, the right column is for the ones that do not. Hence, only R 3.1 and DFM problem 3.1.1 are on the left column and all other rules and problems are in the right column under corresponding meta rule (MR x.x).

In summary, DFM problems should be retrieved and ranked to support process planning of additive manufacturing. In additive manufacturing, process variables are coupled such that values are determined one by one. For example, process variables in Stereolithography including part orientation, layer thickness, and overcure are determined in the presented order. Hence, retrieval of DFM problems also needs to follow this order to properly support process planning. The DFM problems that support process planning of part orientation need to be identified first. Then, the identified DFM problems need to be classified and ranked by process planning of layer thickness and overcure. Finally, the DFM problems should further be ranked by the capability of supporting required accuracy models. To address the requirements discussed above, the components of DFM framework are developed. Those components include information models, meta rule taxonomy, storing algorithm, and retrieval algorithm. The following paragraphs describe each component in detail followed by the discussion of the theoretical validity of storing and retrieval algorithms.

### **6.1.2 Information models and meta rule taxonomy**

In chapters 3, 4, and 5, we have discussed how information models are developed and implemented using description logics ( $\mathcal{AL}\mathcal{E}$ ). Also, the implementation and computation of the meta rule taxonomy is discussed. The information models for representing design requirements are implemented in OWL. The meta rules are also represented with OWL. Using these, the meta rules taxonomy can be computed as

needed using software such as RacerPro. A complete listing of those is in the Appendix A.

### **6.1.3 Storing algorithm**

For developing a storing algorithm, three general cases need to be considered as discussed in section 6.1.1.1. The three general cases include introduction of a new meta rule and a rule, introduction of a new rule, and introduction of a DFM problem only. The repository taxonomy needs to be recomputed whenever new meta rules or rules are introduced. This is because the meta rules and rules capture all the necessary information for classifying the DFM problems. Hence, the subsumption reasoning is only performed at meta rule and rule level. Once DFM problems are tagged with the corresponding meta rule and rule information, actual storing of DFM problems can be accomplished by utilizing files systems or database applications. Based on this, a storing algorithm is developed and shown in Figure 6.4.

1. Given: MR (meta-rule) taxonomy, manufacturing rules taxonomy in the repository, GDR (given design requirement), and DFM problem to be stored
  - a. Extract GDR from the given DFM problem and represent it using predetermined information model
2. Find DMR (determined MR) that satisfies  $DMR \sqsupseteq GDR$  from the MR taxonomy
  - a. Form DR (determined rule) based on the DMR
3. Collect all MR(i) in the (dynamic) meta-rules taxonomy that are subsumed by DMR
  - a. Does any  $MR(i) = DMR$ ?
    - i. No: Insert DMR and DR into the repository and re-compute the taxonomy
      1. Tag DFM problem by DR and DMR then store the problem
    - ii. Yes: Is there R(i) under MR(i) that satisfies  $R(i) \equiv DR$ ?
      1. No: Insert DR into the repository and re-compute the taxonomy
        - a. Tag DFM problem with DR and DMR and store the problem
      2. Yes: Tag DFM problem with DR and DMR and store the problem

### **Figure 6.4 Storing algorithm**

The storing algorithm in Figure 6.4 is the systematic procedure that implements the three general cases for storing a DFM problem discussed above. First, the design requirements are extracted and represented with pre-built information models at step 1. Then, the represented design requirements are compared to the condition part of the pre-

built meta rules that are in the taxonomic structure to determine appropriate meta rule. The appropriate meta rule is determined by finding a meta rule such that its condition part directly subsumes the given design requirements (direct parent). This is performed at step 2. Once an appropriate meta rule is determined, a rule is formed by substituting specific surface types and accuracy measurements in the condition part of determined meta rule at step 2.a. The specific information regarding surface types and accuracy measurements is presented in the given design requirements. At step 3 and its sub steps, the determined meta rule and rule are used to identify the three general cases. If the determined meta rule and rule are not in the repository taxonomy, then those manufacturing rules are added and the manufacturing rule taxonomy is re-computed. If the determined rule is not presented in the manufacturing rule taxonomy in the repository, the rule is added and the manufacturing rule taxonomy is re-computed. If the determined manufacturing rules are in the repository, there is no need for computing the manufacturing rule taxonomy.

The subsumption algorithm operates on the manufacturing rules taxonomy in the repository to identify meta-rules and rules that best match. If the DFM problem matches an existing rule, the DFM problem is tagged with the subsuming meta-rule and rule and it is then stored in the repository. If no exact matches are found, a meta-rule and rule are created using the DFM problem to be stored and are added to the repository under the meta-rule that matches most closely. Then, the manufacturing rules taxonomy in the repository is recomputed. In all three cases, DFM problems are tagged with corresponding meta rule and rule information and stored in any type of storage (ex: database). Therefore, the storing algorithm satisfies the requirements for automated

storing procedure discussed previously by addressing the three general cases. The following paragraphs describe the details of the retrieval algorithm.

#### **6.1.4 Retrieval algorithm**

As discussed in section 6.1.1.2, the retrieval algorithm should retrieve and rank DFM problems to best support process planning in additive manufacturing. Hence, the DFM problems that support process planning for part orientation should be retrieved and ranked first. Then, those problems should be further ranked by the ability to support process planning for layer thickness and overcure. Finally, the DFM problems should be further ranked by ability to provide required accuracy models. To accomplish this, a retrieval algorithm is developed. The retrieval algorithm consists of an algorithm and two metrics; a metric for part orientation and a metric for layer thickness, overcure, and accuracy models. Figures 6.5, 6.6, and 6.7 present the algorithm, metric for part orientation, and metric for layer thickness, overcure, and accuracy models.

1. Given: GDR (given design requirement), MR (meta-rule) taxonomy, and manufacturing rules taxonomy in the repository
  - a. Extract GDR from the given DFM problem and represent it using predetermined information model
2. Find DMR (determined MR) that satisfies  $DMR \sqsupseteq GDR$  from the MR taxonomy
  - a. Form DR (determined rule) based on the DMR
3. Are there MR(i)s that satisfy  $MR(i) \sqsupseteq DMR$  in the repository?
  - a. No: There is no DFM problem to be ranked or retrieved
  - b. Yes: Rank MR(i)s based on the ranking metric for part orientation
    1. Rank MR(i)s further based on the ranking metric for layer thickness and overcure
    2. For each MR(i),
      1. Retrieve R(i)s and rank them based on ranking metric for accuracy models
        1. Rs that satisfy  $DR \sqsupseteq R(i)$  are ranked higher (details are in Fig.

12)

**Figure 6.5 Retrieval algorithm**

Among the DFM problems that satisfy the following condition:

$$\mathbf{DR(i)} \supseteq \mathbf{GDR}$$

where *GDR* : given design requirements (from designer)

*DR(i)* : design requirement in *i*th retrieved DFM problem

they should be ranked in ascending order of  $DS_{diff}(i)$ , computed as following:

$$\mathbf{DS_{diff}(i)} = | \mathbf{DDS} - \mathbf{DS(i)} | \quad (6.1)$$

where *DDS*: feasible spaces of part orientation determined by manufacturing rules

*DS(i)* :feasible space of part orientation that explored by *i*th retrieved DFM problem

### Figure 6.6 Metric for part orientation

The DFM problems that satisfy following condition:

$$\mathbf{EDR} \supset \mathbf{DR(i)} \quad (6.2)$$

where *EDR* = extracted design requirements from the given design requirements

*DR(i)* = design requirements from *i*th DFM problem

has to rank higher than the DFM problems that do not satisfy above condition

### Figure 6.7 Metric for layer thickness, overcure, and accuracy models

In the retrieval algorithm in Figure 6.5, the first two steps (step 1 and 2) are exactly the same as the first two steps in the storing algorithm. They are basically representing design requirements and identifying appropriate meta rules and rules. Once the meta rules and rules are determined, the relative orientations from the condition part of the meta rule are extracted and used to identify equivalent and subsuming meta rules in the repository at step 3. This task is to identify all the meta rules in the repository that address the entire or part of the given design requirements (relative orientation) that influence the feasible space of part orientation. Then, the DFM problems under each meta rule that are identified in step 3 are retrieved and ranked by metric for part orientation.

The metric for part orientation in Figure 6.6 first computes the difference between the determined feasible space of part orientation and the feasible space that is searched by the retrieved DFM problem. The computed quantity is called  $DS_{diff}$  and is computed by equation 6.1. Then, the DFM problems are ranked by the ascending order of the  $DS_{diff}$ . As discussed previously, determining the feasible spaces of part orientation involves performing partial process planning on part orientation. Hence, DFM problems that search closer feasible spaces to the determined feasible space of part orientation support process planning better. Then, the DFM problems that provide smaller difference between the feasible space of part orientation that they search and the feasible space that is determined support process planning of part orientation better. Hence, the DFM problems are ranked by the ascending order of quantity  $DS_{diff}$ . As discussed in chapter 4, the number of feasible values to be considered ranges from infinite to constant; infinite, 4, 3, 2, and 1. Also, there are subsumption relations among them; infinite  $\supseteq$  4  $\supseteq$  3  $\supseteq$  2  $\supseteq$  1. These feasible spaces of part orientation are used in equation 6.1 in Figure 6.6. For infinite part orientation, any large value such as 100 or 1000 can be assigned to use equation 6.1.

The metric for layer thickness, overcure, and accuracy models is shown in Figure 6.7. As discussed previously, the ranking results for those process variables and accuracy models should be binary. For layer thickness, the DFM problems that use the determined minimum layer thickness should be ranked higher than the ones that do not. For overcure, the DFM problems that consider overcure as a process variable should be ranked higher than the ones that do not if the given design requirements demand overcure to be a process variable. For the accuracy models, the DFM problems that provide all the

required accuracy models should be ranked higher than the ones that do not. Hence, the metric in Figure 6.7 uses equation 6.2 to rank DFM problems by testing the condition described above using equation 6.2. For the layer thickness, the surface orientation and the associated accuracy measurement value are extracted (EDR) from the given design requirement to be compared to the design requirement (DR(i)) of DFM problems. Then, the DFM problems that satisfy equation 6.2 are ranked higher than the ones that do not. For the overcure, the surface type and orientation are extracted (EDR) from the given design requirements to be compared to the design requirements (DR(i)) of the DFM problems. Then, the DFM problems that satisfy equation 6.2 are ranked higher than the ones that do not. For the accuracy models, accuracy measurement and associated surface type are extracted (EDR) from the given design requirements to be compared to the design requirements (DR(i)) of DFM problems. Then, the DFM problems that satisfy equation 6.2 are ranked higher than the ones that do not. This completes the retrieval and ranking of DFM problems. Therefore, the metrics in Figures 6.6 and 6.7 are developed to meet the requirements for the automated retrieval algorithm discussed previously. This completes the justification and theoretical validation of the metric. The complete theoretical validation of the storing and retrieving algorithms is discussed in the following paragraphs.

The storing and retrieval algorithms are developed to store and retrieve DFM problems using the subsumption and ranking metric. The performance of those algorithms is critically dependent on the following required capabilities:

1. Correct computation of manufacturing rule taxonomy
2. Correct determination of equivalent and subsuming manufacturing rules

### 3. Correct ranking metrics

As far as those capabilities are guaranteed, the algorithms will perform correctly because everything else in the algorithms is dependent on those three capabilities. Capabilities 1 and 2 are provided by the subsumption algorithm. Capability 3 is provided by the ranking metrics. Hence, the theoretical validity of hypotheses 3 and 4 in Table 1.7 is verified by testing the description logics applicability for representing the design requirements. This is because representation and reasoning in this research is performed on design requirements. Further discussion on validating hypotheses is provided in section 6.3.

The above algorithms and metrics are developed such that they conform to the mapping relation discussed in section 4.3. In section 4.3, the mathematical foundations for relating the design requirements and the MPTs are discussed. Those mathematical foundations are concisely presented in Figure 4.20. In Figure 4.20, the forward mapping (item 1) determines the MPTs for the given design requirements using the subsumption relations among the design requirements. Then, the identified MPTs are ranked (item 2) by their subsumption relations and used for inverse mapping. The inverse mapping determines the appropriate design requirements for a given MPT. For the inverse mapping, the appropriate design requirements are identified from the design requirements that are identified during the forward mapping.

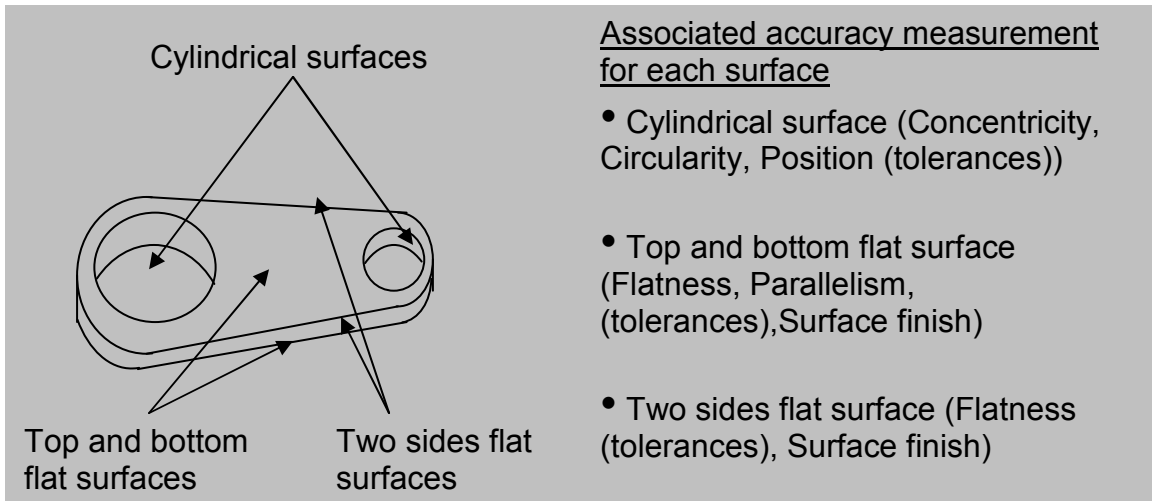
The forward mapping is realized by step 3 of the retrieval algorithm in Figure 6.5. At this step, the relevant meta rules (also rules under the relevant meta rules) are identified by finding equivalent and subsuming meta rules for the given design requirement. Through identifying those meta rules and rules, appropriate MPTs are

identified. Hence, this conforms to the forward mapping (item 1 in Figure 4.20). The ranking and inverse mapping is realized by metrics in Figures 6.6 and 6.7. In those metrics, the identified meta rules and rules from the forward mapping are ranked by the feasible spaces of the process variables (subsumption relations of MPTs). Through those identified meta rules and rules, appropriate design requirements are identified. Hence, this conforms to the ranking and inverse mapping (items 2 and 3 in Figure 4.20).

In the following section, the storing and retrieval performance of DFM framework is demonstrated using the Robot Arm example.

## **6.2 Retrieval demonstration using Robot Arm example**

In this section, a demonstration of the DFM framework is presented using the robot arm example. The objective of this example is to illustrate the functionality of the DFM framework through a simple example. More specifically, this example demonstrates how information models and algorithms are used to store and retrieve relevant DFM problems. Figure 6.8 shows the robot arm and its various surfaces with the associated accuracy measurements. Using the identified surfaces and associative accuracy measurements, three DFM problems and three design requirements are created to experimentally examine storing and retrieving DFM problems.



**Figure 6.8 Robot Arm example**

Stereolithography is chosen as the manufacturing process to study. In Stereolithography, there are process variables including part orientation, layer thickness and overcure (fill and hatch). The generalized relation between the part orientation and layer thickness to accuracy measurement ranges are shown in Table 6.2.

**Table 6.2 Relation between accuracy measurement value range and feasible space of process variables in Stereolithography**

Part orientation	Layer thickness	Accuracy measurement value range
0°	Thin	<value 1
	Thick	< value 2
90°	Thin	<value 3
	Thick	<value 4
Any	Thin	< value 5
	Thick	< value 6

For this example, we only consider part orientation and we assume that all the accuracy measurement values are in between value 3 and value 4 as shown in the high lighted part of Table 6.2. Therefore, the available part orientation values are 0° and 90°.

Through such simplification, the corresponding simple meta rule can be identified and shown in Figure 6.9.

*If (accuracy measurement is specified on a surface) then (feasible part orientations for satisfying such requirement are 0° or 90°)*

### **Figure 6.9 Simple meta rule for robot arm example**

The rule in Figure 6.9 is the simple meta rule that is used to derive the meta rule taxonomy in Figure 4.5. Therefore, the meta rule taxonomy in Figure 4.5 is used to identify the feasible space of part orientation in this example. Then, the rules are formed by adding accuracy measurements and the surface to the meta rules. Based on the above simplifications, three DFM problems and design requirements are developed as shown in Tables 6.3 and 6.4 respectively. The DFM problems and design requirements in Tables 6.3 and 6.4 are developed using robot arm in Figure 6.8 and meta rule in Figure 6.9. The DFM problems in Table 6.3 are used to demonstrate the storing procedure. The design requirements in Table 6.4 along with stored DFM problems are used to demonstrate retrieval.

**Table 6.3 Example DFM problems**

DFM problem name	Design requirements	Relative orientation	Solution strategy	Accuracy measurement model
DFM problem 1	(Flatness tolerance and surface finish on top and bottom surfaces) and (Concentricity and circularity tolerance on cylindrical surface)	Opposite	Continuous ( $\infty$ )	Flatness tolerance and surface finish on flat surface Concentricity and circularity tolerance on cylindrical surface
DFM problem 2	(Flatness tolerance and surface finish on top surfaces) and (Concentricity and circularity tolerance on cylindrical surface) and (Flatness tolerance and surface finish on one side surface)	Perpendicular	Discrete (4)	Flatness tolerance and surface finish on flat surface Concentricity and circularity tolerance on cylindrical surface
DFM problem 3	(Flatness tolerance on top and bottom surfaces) and (Concentricity tolerance on cylindrical surface) and (Flatness tolerance on one side surface)	Opposite and Perpendicular	Discrete (3)	Flatness tolerance on flat surface Concentricity tolerance on cylindrical surface

**Table 6.4 Example queries (design requirements)**

Query	Design requirement	Relative orientation	Solution strategy	Accuracy measurement model
Q1	(Flatness tolerance on top and bottom surfaces) and (circularity tolerance on cylindrical surface)	Opposite	Continuous ( $\infty$ )	Flatness tolerance on flat surface Circularity tolerance on cylindrical surface
Q2	(Parallelism tolerance on top flat surface) and (Flatness tolerance on one side surface)	Perpendicular	Discrete (4)	Parallelism tolerance on flat surface Flatness tolerance on flat surface
Q3	(Flatness tolerance on top and bottom surfaces) and (Flatness tolerance on one side surface)	Opposite and Perpendicular	Discrete (3)	Flatness tolerance on flat surfaces

By specifying the accuracy measurement on various surfaces in Figure 6.8, three DFM problems are created in Table 6.3. In each DFM problem, properties such as design requirements, relative orientations, solution strategy, and accuracy measurements are presented. The numbers in parenthesis represent the number of feasible part orientations

that can satisfy design requirements in the problem. Such properties are offered to the user by the DFM problems when they are retrieved. In Table 6.4, three design requirements are developed. For each design requirement, relative orientation, solutions strategy and models of accuracy measurement required for formulating and solving a DFM problem are identified. When the DFM problems are retrieved for design requirements in Table 6.4, the retrieval should be based on the closeness of the corresponding solution strategy and accuracy models. The following subsections discuss representation, storing, and retrieval.

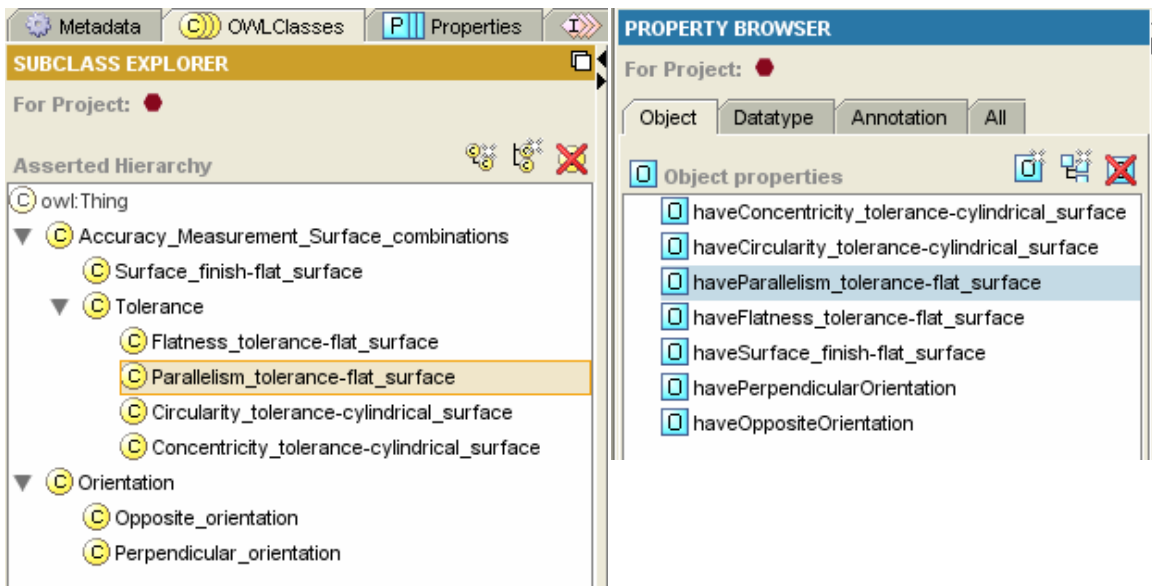
### 6.2.1 Representation and information modeling

To store and retrieve the DFM problems, the DFM problems and design requirements need to be represented (step 1 in Figures 6.4 and 6.5). First, the atomic concepts and roles are identified for the representation as shown in Table 6.5.

**Table 6.5 Atomic concepts and roles for robot arm example**

	Relative orientation	Accuracy measurement models
Atomic Concepts	<ul style="list-style-type: none"> <li>• Opposite</li> <li>• Perpendicular</li> </ul>	<ul style="list-style-type: none"> <li>• flat_tolerance-flat_surface</li> <li>• surface_finish-flat_surface</li> <li>• parallelism_tolerance-flat_surface</li> <li>• circularity_tolerance-cylindrical_surface</li> <li>• concentricity_tolerance-cylindrical surface</li> </ul>
Roles	<ul style="list-style-type: none"> <li>• haveOppositeOrientation</li> <li>• havePerpendicularOrientation</li> </ul>	<ul style="list-style-type: none"> <li>• haveFlat_tolerance-flat_surface</li> <li>• haveSurface_finish-flat_surface</li> <li>• haveParallelism_tolerance-flat_surface</li> <li>• haveCircularity_tolerance-cylindrical_surface</li> <li>• haveConcentricity_tolerance-cylindrical_surface</li> </ul>

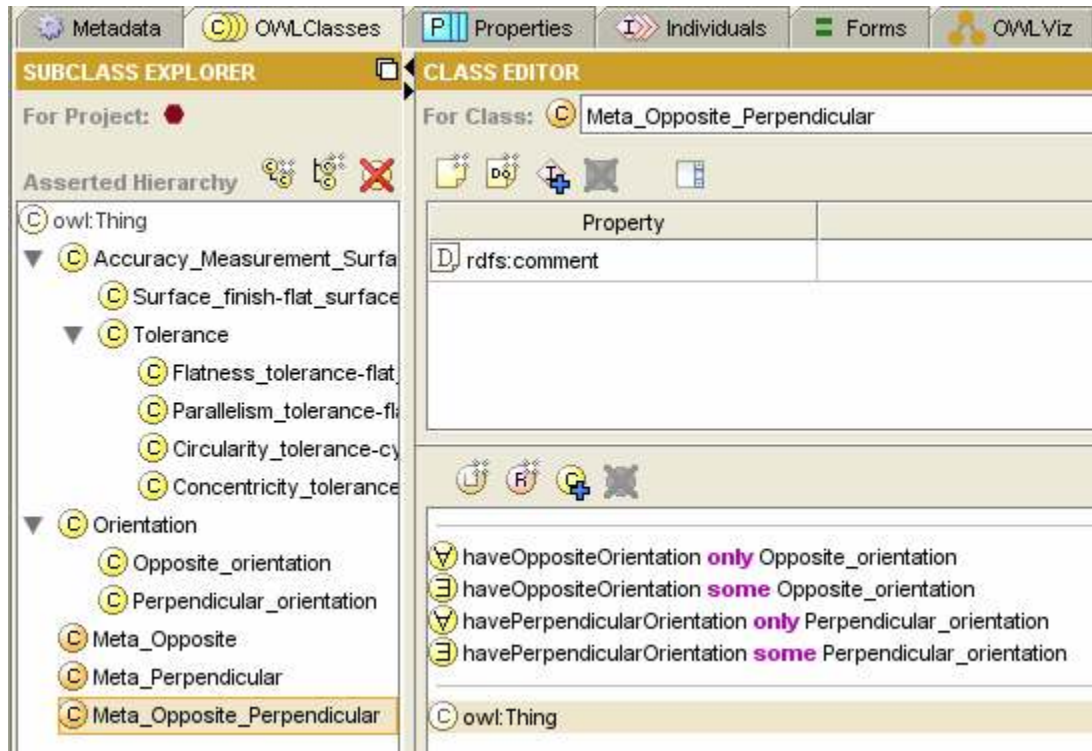
As discussed previously, the meta rule taxonomy in Figure 4.5 is used to determine the feasible space of part orientations in this example. Hence the relative orientations of surfaces where accuracy measurements are specified are used to determine the feasible part orientations. The second column in Table 6.5 shows the atomic concepts and roles for describing relative orientations. The third column in Table 6.5 shows the atomic concepts and roles for specifying combinations of accuracy measurement and surface. The concepts and roles in the third column are used to form rules by adding this information to meta rules. Using Protégé, the information in Table 6.5 is implemented as shown in Figure 6.10.



**Figure 6.10 Atomic concept and role implementation for robot arm example**

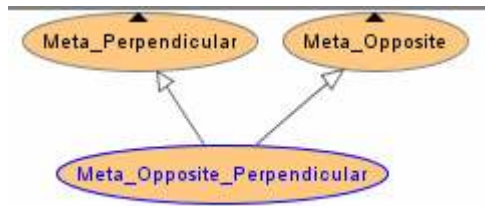
In Figure 6.10, the implementation of atomic concepts is on the left and the roles are on the right. In this example, two individual relative orientations are identified; Opposite and Perpendicular. Then, the combined orientation that is opposite and perpendicular can be derived from the two individual relative orientations. Hence, we introduce three meta rules based on relative orientations; opposite, perpendicular and

opposite-perpendicular. Using the atomic concepts and roles in Figure 6.10, the condition part of the meta rules are represented for indexing. Then, the indexes are used to compute the meta rule taxonomy. Figure 6.11 shows the example for implementing an index for the meta rule opposite-perpendicular.



**Figure 6.11 Example of implemented meta rule in robot arm example**

Figure 6.11 indexes the meta rule whose condition part has opposite and perpendicular orientations of surfaces where accuracy measurements are specified. The lower right part of Figure 6.11 shows the indexing and this can be interpreted as something that has opposite and perpendicular orientation. After indexing the other two meta rules, the taxonomy is computed using RacerPro and shown in Figure 6.12.



**Figure 6.12 Meta rule taxonomy computation for robot arm example**

As expected, the indexes of perpendicular and opposite are different. Also, the index of opposite-perpendicular is the intersection of the two. The DFM problems are also represented using the implemented concepts and roles. The example of DFM problem 3 from Table 6.3 is shown in Figure 6.13.

**CLASS EDITOR**

For Class:  (instance of owl:Class)

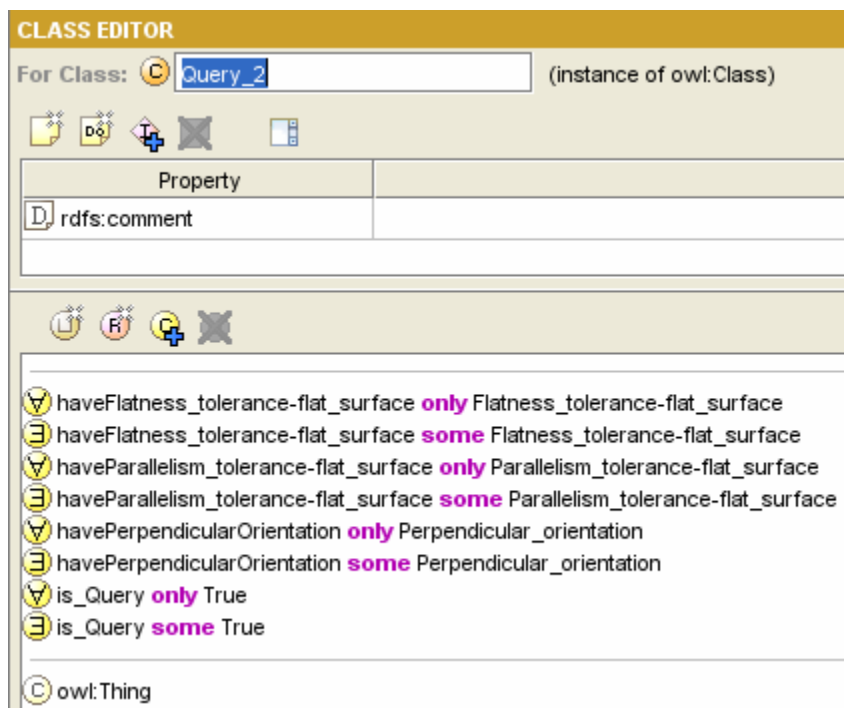
Property	Value
<input type="text" value="rdfs:comment"/>	

- haveConcentricity\_tolerance-cylindrical\_surface **only** Concentricity\_tolerance-cylindrical\_surface
- haveConcentricity\_tolerance-cylindrical\_surface **some** Concentricity\_tolerance-cylindrical\_surface
- haveFlatness\_tolerance-flat\_surface **only** Flatness\_tolerance-flat\_surface
- haveFlatness\_tolerance-flat\_surface **some** Flatness\_tolerance-flat\_surface
- haveOppositeOrientation **only** Opposite\_orientation
- haveOppositeOrientation **some** Opposite\_orientation
- havePerpendicularOrientation **only** Perpendicular\_orientation
- havePerpendicularOrientation **some** Perpendicular\_orientation
- is\_DFM\_Problem **only** True
- is\_DFM\_Problem **some** True

owl:Thing

**Figure 6.13 Example representation of DFM problem 3**

DFM problems are represented using design requirements. In Figure 6.13, the relative orientations (opposite and perpendicular) and accuracy measurement (flatness and concentricity tolerance) with associative surfaces are added to represent DFM problem 3. The purpose of the role “is\_DFM\_Problem” in Figure 6.13 is added to distinguish it from the rule representation shown later this section. Similarly, the design requirements in Table 6.4 are represented. Figure 6.14 shows the representation of query 2 from Table 6.4.



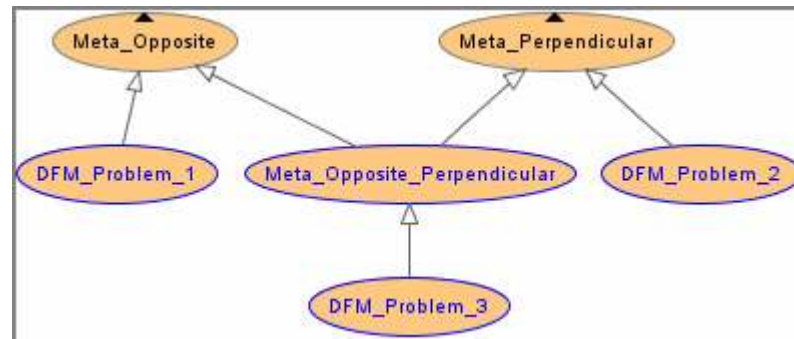
**Figure 6.14 Example representation of design requirement (query) 2**

In Figure 6.14, the relative orientation (perpendicular) and accuracy measurements (flatness and parallelism tolerance) with associated surfaces are used to represent query 2. The purpose of the role “is\_Query” in Figure 6.14 is added to distinguish the query from the rule representation shown later in this section. So far,

DFM problems, design requirements, and manufacturing rule taxonomy are represented and constructed. Storing and retrieving are discussed in the following paragraphs.

### 6.2.2 Storing

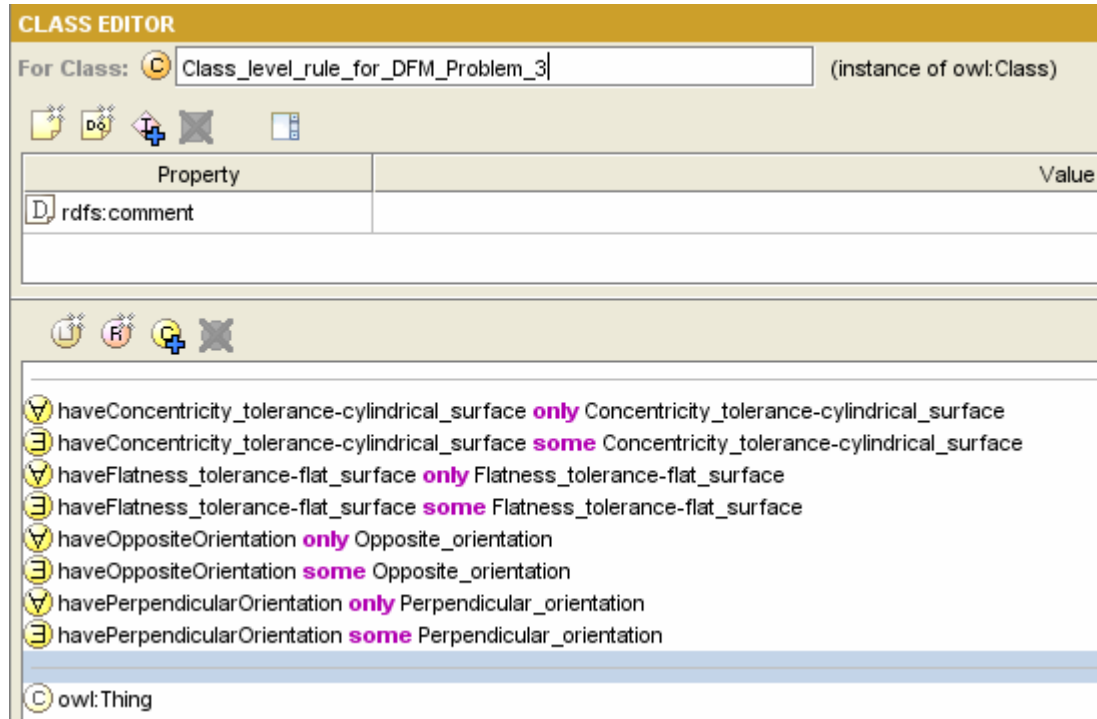
The first step in storing DFM problems is identifying the appropriate meta rule for the DFM problems (step 2 in Figure 6.4). This task is performed by finding the direct parent of represented DFM problems in the meta rule taxonomy using subsumption. Figure 6.15 shows the result of finding direct parent of DFM problems 1, 2 and 3 in the meta rule taxonomy using subsumption.



**Figure 6.15 Result of finding direct parent for each DFM problem in robot arm example**

The taxonomy in Figure 6.15 is computed by using the meta rules shown in Figure 6.12 along with represented DFM problems. In this example, the task of finding a parent is performed manually by observing the taxonomy. However, such tasks are fully automated in empirical validation example in the next chapter. When the taxonomy is computed, the DFM problems are expected to be under meta rules that have the same relative orientations. Hence, DFM problems 1, 2, and 3 are expected to be under meta rules that have opposite, perpendicular, and opposite-perpendicular orientations respectively. This is exactly what is shown in Figure 6.15. Once the meta rule for each

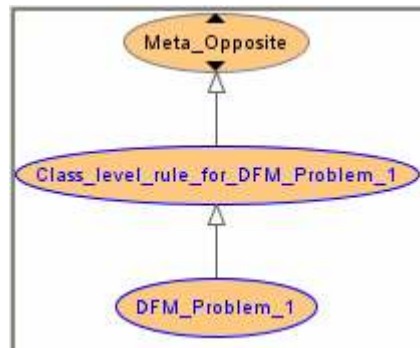
DFM problem is identified, the next step is forming rule. The rule is basically the represented DFM problem itself except for the role of “is\_DFM\_Problem” in Figure 6.13. By removing the role “is\_DFM\_Problem” from Figure 6.13, the represented DFM problem 3 becomes the rule for DFM problem 3. Figure 6.16 shows the rule for DFM problem 3.



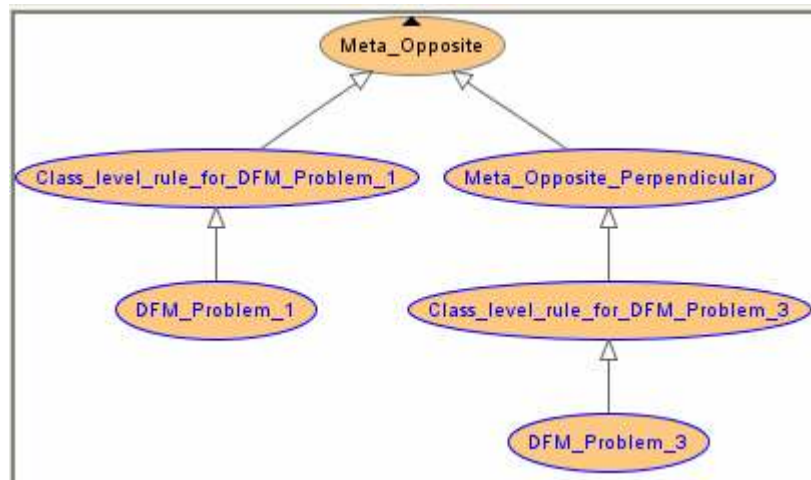
**Figure 6.16 Example of rule representation for DFM problem 3**

If we compare the rule in Figure 6.16 to the meta rule in Figure 6.11, we can observe that the rule is formed by addition of accuracy models to the meta rule. By similar procedures, we can form a rule for DFM problems 1 and 2. So far meta rules and rules are determined for each DFM problem. Using the storing algorithm in Figure 6.4, the represented DFM problems are stored. The repository is initially assumed to be empty. The problems are introduced in the order of DFM problems 1, 3 and 2. Figures

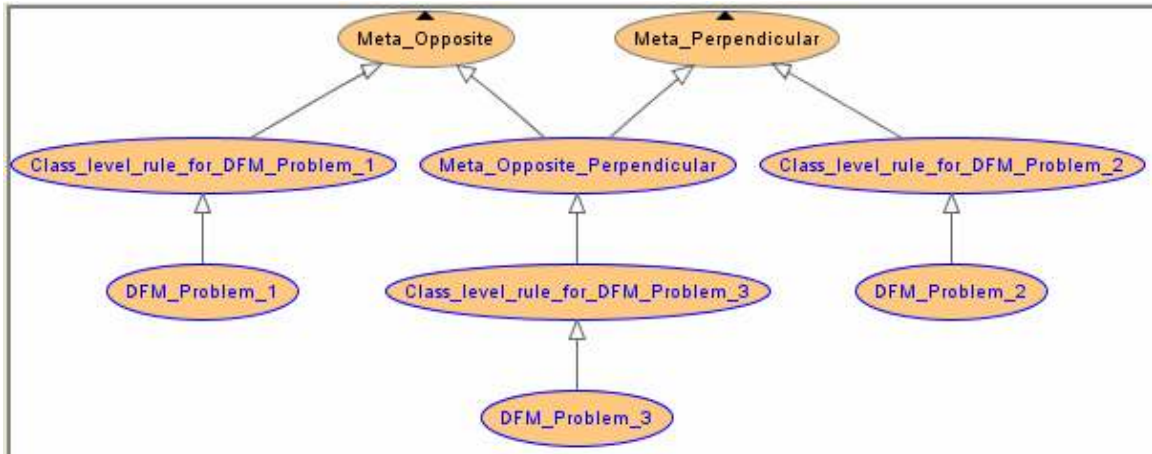
6.17, 6.18 and 6.19 show the repository structure as problems 1, 3, and 2 are introduced respectively.



**Figure 6.17 Introduction of DFM problem 1**



**Figure 6.18 Introduction of DFM problem 3**



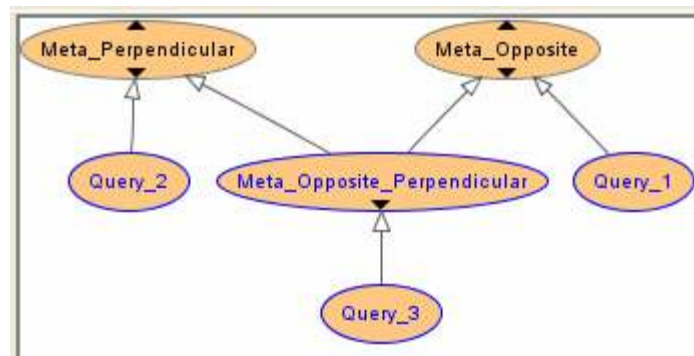
**Figure 6.19 Introduction of DFM problem 2**

Each time a new DFM problem is introduced, the existence of corresponding meta rules and rules in the repository is checked according to the storing algorithm in Figure 6.4. If meta rules or rules are not in the repository, the algorithm creates those manufacturing rules and restructures the taxonomy in the repository. In this example, the meta rules and rules are created in the repository each time a new DFM problem is introduced because meta rules and rules for each DFM problem are different. The rules should be subsumed by the meta rules if they share the same relative orientations. The rules are the abstractions of DFM problem instances. Therefore, the DFM problems are subsumed by the rules and rules are subsumed by meta rules as discussed in chapters 3 and 4. In this example, we expect DFM problems 1, 2, and 3 be under corresponding rules and the rules should be under meta rules that have the same relative orientations. This is shown in Figures 6.17, 6.18, and 6.19. Also, the subsumption relation among meta rules shown in Figure 6.19 makes sense because meta rules that have opposite orientation differ from the rules having perpendicular orientation in the condition part.

However, the one that has both perpendicular and opposite orientations in its condition part has to be the intersection of the two other meta rules. In short, the algorithm in Figure 6.4 structures the repository as expected.

### 6.2.3 Retrieval

The first step in retrieval is representing the query (design requirements) in Table 6.4. Then, an appropriate meta rule for each query needs to be identified (step 2 in Figure 6.5). This task is performed by using subsumption to find direct parent of the represented query in the meta rule taxonomy. Figure 6.20 shows the result of finding the direct parent of represented queries 1, 2 and 3 from the meta rule taxonomy in Figure 6.12.



**Figure 6.20 Finding meta rule for each query in robot arm example**

The taxonomy in Figure 6.20 is computed by using the meta rules taxonomy in Figure 6.12 along with the represented queries. When the taxonomy is computed, each given design requirement is expected to be under a meta rule that has the same relative orientation. Hence, queries 1, 2, and 3 are expected to be under meta rules that have opposite, perpendicular, and opposite-perpendicular orientations respectively. This is exactly what is shown in Figure 6.20. Once the meta rules are identified, the next task is to form rules. The process of forming rules is shown for the storing procedure hence, it is

omitted here. Using identified manufacturing rules for the given design requirements, steps 3, 4, and 5 in Figure 6.5 are carried out to identify and rank the relevant DFM problems. The retrieved and ranked DFM problems for each query are shown in Table 6.6.

**Table 6.6 Retrieval result in robot arm example**

	Rank 1	Rank 2	Rank 3
Query 1	DFM problem 1		
Query 2	DFM problem 2		
Query 3	DFM problem 3	DFM problem 2	DFM problem 1

For query 1, the appropriate meta rule is identified as the one that has opposite orientation in its condition part as shown in Figure 6.20. Using this determined meta rule, equivalent and subsuming meta rules in Figure 6.12 are identified (step 2 in Figure 6.5). In the case of query 1, there is no subsuming meta rule and there is only an equivalent one. Hence, all the DFM problems under equivalent meta rule are retrieved and ranked based on the metric in Figure 6.6 (step 5 in Figure 6.5). Since only one DFM problem is under the equivalent meta rule, a single problem (DFM problem 1) is retrieved as shown in Table 6.6. Similarly, DFM problem 2 is retrieved for query 2. For query 3, the appropriate meta rule is identified to be the one that has opposite and perpendicular (opposite-perpendicular) orientations in its condition part. Using this determined meta rule, equivalent and subsuming meta rules in Figure 6.12 are identified and ranked based on the metric in Figure 6.6. The result of the meta rule ranking is shown in Table 6.7.

**Table 6.7 Ranking result for meta rule (opposite-perpendicular) in robot arm example**

	Rank 1	Rank 2	Rank 3
Query 3	Opposite-Perpendicular (3)	Perpendicular (4)	Opposite ( $\infty$ )

The rank in Table 6.7 is created by the ascending order of difference between the feasible space of part orientation for query 3 and that for each meta rule. As discussed previously, DFM problems that search feasible spaces of part orientation closer to the determined feasible space (feasible space for query 3) support process planning better. The determined feasible space of part orientation for query 3 is 3. The feasible spaces of part orientation that the DFM problem 1, 2, and 3 search are infinite, 4, and 3 respectively. Therefore, the rank should be in the order of DFM problem 3, 2, and 1. Figure 6.21 demonstrates the computational procedure for ranking DFM problems.

### Utilization of ranking metric for part orientation

Determined feasible space of part orientation for query 3: 3

Feasible space of part orientation that each DFM problem searches:

- DFM problem (1) = infinite
- DFM problem (2) = 4
- DFM problem (3) = 3

Computation of  $DS_{diff}(i)$  in Figure 5.6

- $DS_{diff}(1) = |100-3| = 97$  assuming infinite = 100
- $DS_{diff}(2) = |4-3| = 1$
- $DS_{diff}(3) = |3-3| = 0$

Therefore, DFM problems should be ranked in the order of DFM problem 3, 2, and 1

### Utilization of ranking metric for accuracy models

Accuracy models for query 3:

Thing  $\sqcap \exists haveFlatnessTolerance-FlatSurface.FlatnessTolerance-FlatSurface$

Design requirements for DFM problem 3:

(Thing  $\sqcap \exists haveFlatnessTolerance-FlatSurface.FlatnessTolerance-FlatSurface$ )  $\sqcap$

$\exists haveConcentricityTolerance-CylindricalSurface.ConcentricityTolerance-CylindricalSurface$

Then, “Accuracy models for query 3  $\sqcap$  Design requirements for DFM problem 3” can be determined by subsumption. Therefore, this satisfy equation 6.2 in Figure 6.7

**Figure 6.21 Computation example of ranking metric for part orientation and accuracy models usage**

In Figure 6.21, the use of a ranking metric for part orientation and accuracy models is demonstrated. First, the ranking metric for part orientation is used to rank the DFM problems 1, 2, and 3. The quantity  $DS_{diff}(i)$  from Figure 6.6 is computed for each DFM problem. Then the rank is produced in the ascending order of quantity  $DS_{diff}$ . The resulted rank exactly matches the expected rank discussed in the previous paragraph. Then, the ranking metric for accuracy model is used to further rank DFM problem 3. By

extracting accuracy models from the given design requirements and comparing it to the design requirements for DFM problem 3, their subsumption relation is determined. Therefore, DFM problems 3 should be ranked higher than the DFM problems that do not satisfy equation 6.2 in Figure 6.7. A similar computation can be shown for DFM problems 1 and 2. In this example, there is only one problem under each rank in Table 6.7. Hence, further ranking is not quite appropriate. However, the purpose of the demonstration in Figure 6.21 is to show how metric that ranks for accuracy models can be utilized.

The resultant ranking in Figure 6.21 and Table 6.7 is also justified by the relation between the feasible space of the process variables and the solution strategy. The expected solution strategy for query 3 is a solution strategy for discrete process variable (multi solutions). The DFM problems in rank 1 and 2 utilize multi solutions strategy and the DFM problem in rank 3 utilize solution strategy for continuous variable (decomposition). A detailed discussion of the relation between the feasible space of the process variables and the solution strategy is provided in chapter 1.

In conclusion, the retrieval method is demonstrated using the robot arm example. More specifically, we have demonstrated how each component, including the information model, the manufacturing rule hierarchy, and the storing and retrieval algorithms is used to store and retrieve DFM problems. The following paragraphs describe the theoretical validation of hypotheses 3 and 4.

### **6.3 Hypotheses validation**

In this chapter, hypotheses 3 and 4 are validated. Table 6.8 shows the research questions and hypotheses (3 and 4).

**Table 6.8 Corresponding research questions and hypotheses**

Questions	Hypotheses
Q3. How should DFM problems be stored?	H3. Subsumption in DL enables systematic and consistent structuring of the repository
Q4. How should DFM problems be retrieved and ranked?	H4. Subsumption in DL and the ranking metric enables retrieval and ranking of the relevant DFM problems

Research question 3 and hypothesis 3 are about structuring the repository using subsumption. As discussed previously, the repository is structured by the manufacturing rules taxonomy. Hence, correct and consistent manufacturing rules taxonomy computation in the repository is crucial to correctly retrieve the relevant DFM problems. Hypothesis 3 states that subsumption in DL enables systematic and consistent structuring of the repository. The theoretical validity of hypothesis 3 can be achieved by identifying the appropriate DL to represent design requirements such that the manufacturing rules taxonomy can be computed using subsumption in DL. This has been theoretically validated in chapter 5 through selecting the appropriate DL ( $\mathcal{AL}\mathcal{E}$ ). Hence, hypothesis 3 is theoretically validated. To demonstrate the theoretical validity, a storing algorithm that uses subsumption in DL is developed and tested using a simple robot arm example.

Research question 4 and hypothesis 4 are about identifying the relevant DFM problems using subsumption in DL and ranking the identified DFM problems by metric. As discussed in Figures 4.15 and 4.16, the subsumption relation in the condition part of the manufacturing rules relates the given design requirements to the relevant DFM problems through appropriate MPTs. Therefore, identifying an appropriate description logic to represent the design requirements enables the relevant DFM problems to be retrieved using subsumption in DL. This is theoretically validated in chapter 5. Furthermore, two metrics are developed in Figures 6.6 and 6.7. Those metrics are developed

such that the DFM problems that support process planning better for the given design requirements are ranked higher. Hence, this meets the initial objective of supporting the process planning by retrieving the relevant DFM problems. The detailed justifications are presented in section 6.14. This completes the theoretical validation of hypothesis 4. To demonstrate the theoretical validity, a retrieval algorithm is realized and tested using a simple robot arm example.

Through theoretically validating the hypotheses 1~4 (1 and 2 in chapter 5 and 3 and 4 in chapter 6), the theoretical structure validation discussed in section 1.3.1 is completed. In this research, the internal consistency and theoretical soundness of the retrieval method is achieved by two steps. First step is establishing a theoretical foundation for relating design domain to process planning domain. Second step is realizing and justifying the retrieval method. By validating hypotheses 1 and 2, the theoretical foundation for mapping design domain and process planning domain is established. By validating hypotheses 3 and 4, a retrieval method that consistently and correctly stores and retrieves DFM problems is realized and justified. Through the robot arm example in section 6.2, the theoretical validity of hypotheses 1~4 are demonstrated. Hence, this completes the theoretical structure validation discussed in section 1.3.1.

## **6.4 Summary**

In this chapter, storing and retrieving algorithms are developed (section 6.1). By combining those algorithms with the information models and meta rules taxonomy realized in chapter 5, the retrieval method is completed. Then, the method's storing and retrieving performance is demonstrated using a simple robot arm example (section 6.2).

Through the development and justification of storing and retrieving algorithms, hypotheses 3 and 4 in Table 6.8 are theoretically validated (section 6.3).

In short, the retrieval method (DFM framework) is completed by developing four components including information model for representing the design requirements, the meta rules taxonomy, the storing algorithm, and the retrieval algorithm. The method is realized based on the formal mapping between design and process planning domains. The two domains are clearly defined. Hence, the applicability of the retrieval method is guaranteed within the scope of the research (Table 3.2). Using the retrieval method, the DFM knowledge (DFM problems) can be shared in a distributed environment to support process planning in geometric tailoring. Therefore, the retrieval method is new knowledge in DFM to enable sharing DFM knowledge in a distributed environment based on a formal mapping between design and additive manufacturing. Table 6.9 summarizes this contribution.

**Table 6.9 New knowledge discovered in DFM**

DFM
<ul style="list-style-type: none"> <li>• The retrieval method that allows sharing DFM knowledge (DFM problems) in a distributed environment based on formal mapping between design and additive manufacturing               <ul style="list-style-type: none"> <li>○ Information models for representing the design requirements</li> <li>○ Meta rules taxonomy</li> <li>○ Storing algorithm</li> <li>○ Retrieval algorithm</li> </ul> </li> </ul>

## **CHAPTER 7: VALIDATING DFM FRAMEWORK**

The objective of empirical performance validation is to demonstrate the utility of the retrieval method. Through this empirical validation, the hypotheses (1~4 in Table 1.7) are confirmed. First, an overview of empirical validation is presented (section 7.1). In the overview, the measurements that are taken in empirical validation are presented and justified. Second, the test case selection and samples of design requirements development are discussed (section 7.2). A wind tunnel design project is selected and justified for the empirical validation. Through this selection and justification, the empirical structure validation discussed in section 1.3.2 is completed. Third, the test case set up for collecting the measurements is presented (section 7.3). The test case set up discusses the variables and constants that are used to collect each measurement. This section only discusses the setting up of the test case not the results. Fourth, the test bed implementation and the collected results is presented (sections 7.4 and 7.5). From the collected results, the three experiments are concluded. The empirical validation of the hypotheses in Table 1.7 is concluded (section 7.6). Also, the empirical performance validation discussed in section 1.3.3 is completed. Based on the collected results and hypotheses validation, this research delivers a retrieval method (DFM framework) that satisfies the initial research objectives. Detailed discussions on how the retrieval method satisfies the research objective are presented (section 7.7).

### **7.1 Overview**

In this research, relevant DFM problems for the given design requirements are retrieved by successful mapping of the design and process planning domains. The manufacturing rules taxonomy is identified as a function or a mathematical operator that

maps the two domains in chapter 4. Then, the retrieval method structures the repository by computing a manufacturing rules taxonomy and retrieves relevant DFM problems using the computed taxonomy. Hence, a correct and consistent manufacturing rules taxonomy is crucial for the successful retrieval. To compute the manufacturing rules taxonomy consistently and correctly, subsumption in DL is used. In empirical performance validation, the entire method's performance in storing and retrieving the DFM problems is demonstrated. Therefore, two major test criteria are identified for empirical performance validation: 1. correct and consistent taxonomy computation capability and 2. retrieval performance. Additionally, the computational complexity of subsumption is a concern in this research. Most description logics are known to be theoretically intractable for inference. The description logic,  $\mathcal{AL}\mathcal{E}$  is selected in this research and the corresponding theoretical computational complexity is non deterministic polynomial time. Hence, computational complexity becomes an additional test criterion. For the three identified testing criteria, three testing strategies are developed. The following paragraphs briefly describe each testing strategy.

First, the DL applicability testing is to test the ability of subsumption in DL to compute a correct and consistent manufacturing rules taxonomy. In this research, it is expected that the set of manufacturing rules will expand as new manufacturing rules are discovered. Hence, the manufacturing rules taxonomy should be manageable systematically. In chapter 5, description logic is selected and justified for this purpose. Hence, the DL applicability testing demonstrates the ability of subsumption in DL to compute a correct and consistent manufacturing rules taxonomy by using the identified

repository expansion scenarios. Through this demonstration, the validity of hypotheses 1, 2, and 3 in Table 1.7 is demonstrated. Details are shown in section 7.3.1.

Second, retrieval performance testing evaluates the method's ability to retrieve relevant DFM problems. As discussed in chapter 4, the retrieval performance is dependent on the ability to correctly map between design and process planning domains. In this research, the manufacturing rules taxonomy maps the two domains and the DFM problems are classified by the rules. Hence, retrieval of the relevant DFM problems can be tested by retrieval of the relevant rules. Through this demonstration, the validity of hypotheses 1, 2, and 4 in Table 1.7 is demonstrated. The details are discussed in section 7.3.2.

Third, the computational feasibility testing studies the computational complexity of subsumption corresponding to the description logic  $\mathcal{AL}\mathcal{E}$ . The corresponding theoretical computational complexity is nondeterministic polynomial time. However, empirical computational complexity can be different than theoretical computational complexity depending on the problem size and the complexities in expressions [118, 119]. Hence, empirical computational complexity testing is performed to predict the retrieval method's performance in computing subsumption and taxonomy computation. Specifically, the time required to compute the subsumption of one concept by another is measured by varying the problem size (knowledge base size that describes the two concepts). Also, taxonomy computation time is measured by increasing problem size only and then by increasing both problem size and number of manufacturing rules. The details are discussed in section 7.3.3.

Table 7.1 summarizes the three testing strategies by presenting the variable and constants that are used in each test.

**Table 7.1 Summary of testing strategy**

measurements variables	DL applicability	Retrieval performance	Computational feasibility		
			Subsumption	Taxonomy computation (increasing problem size)	Taxonomy computation time (increasing problem size and number of manufacturing rules)
Query	N/A	Variable	N/A	N/A	N/A
Taxonomy structure	Variable	Constant	N/A	Constant	Variable
Introduction order	Variable	N/A	N/A	N/A	N/A
Problem size	N/A	N/A	Variable	Variable	Variable

In Table 7.1, four variables are related to the three measurements discussed above. The four variables include query, taxonomy structure, introduction order, and problem size. For DL applicability measure, the manufacturing rules taxonomy is computed and observed with variations of order in which the manufacturing rules are added to the taxonomy. Hence, the taxonomy structure and introduction order are variables for DL applicability testing. For the retrieval performance measure, the repository is fully loaded and various queries are performed. Hence, the taxonomy structure is constant and the query is variable. For the computational feasibility measure, there are three measurements including subsumption computation time, taxonomy computation time with increasing problem size, and taxonomy computation time with both increasing problem size and increasing number of manufacturing rules. For measuring subsumption computation time, two concepts are compared with increasing the problem size. The problem size means the knowledge base size that constitutes each

concept (design requirement). For example, the statements that define the concepts (design requirement) in Figures 5.12 and 5.16 are the knowledge base. Then, the size means the number of atomic concepts in the knowledge base. For the taxonomy computation time measure with increasing problem size, the taxonomy structure is constant but the problem size increases. For the taxonomy computation time measure with increasing problem size and increasing number of manufacturing rules, both the problem size and the taxonomy structure are variables. Repository structure is variable because the number of rules in the repository increases. Details are presented in section 7.3. In the following section, the test case selection and justification are presented.

## **7.2 Test case selection and samples of design requirements and DFM problems**

In this section, the test case selection criteria are discussed first. Then, the selected test case is presented. Finally, the selected test case is justified by developing the sample design requirements that satisfy the selection criteria.

### **7.2.1 Criteria**

To collect the measurements discussed in section 7.1, the test case needs to be selected such that it covers the entire research scope. The research scope is identified in chapter 3. Within the research scope, the manufacturing rules are collected in chapters 3 and 4. In this research, the discovered manufacturing rules and their subsumption relations relate the design requirements to the relevant DFM problems. Hence, the test case needs to be selected such that the entire set of manufacturing rules can be practiced. To test the entire set of manufacturing rules, the sample design requirements need to be developed such that they can be used to practice all the manufacturing rules. Table 7.2 shows detailed criteria

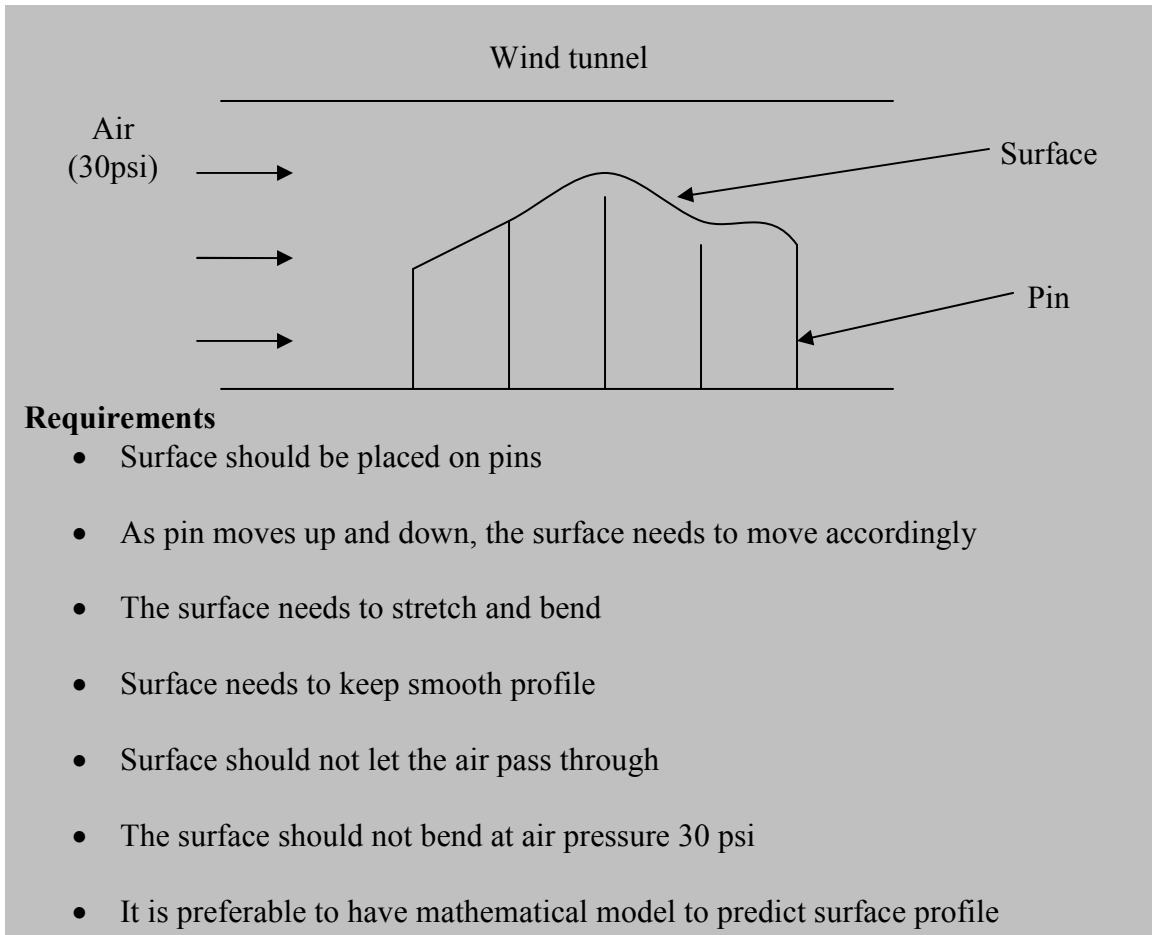
**Table 7.2 Testing criteria**

Criteria
<ul style="list-style-type: none"><li>• The test case should involve various surface orientations (to cover meta rules discovered in this research, Figures 4.5, 4.7, and 4.9)</li><li>• The test case should involve various types of surfaces (to cover the rules that involve various accuracy requirements)</li></ul>

The criteria in Table 7.2 basically state that a design project is needed that involves complex geometry that can satisfy the criteria in Table 7.2. More specifically, the geometry needs to be sufficiently complex to realize the condition part of all the manufacturing rules discovered in this research. A wind tunnel surface design project is selected. The following section discusses the details of the selected design project.

### **7.2.2 Wind tunnel surface design**

The objective of this design project is to design and fabricate a surface that is to be placed in the wind tunnel to guide the air. Figure 7.1 shows the details. The example shown in Figure 7.1 originates from the digital clay project that was carried out at the Georgia Institute of Technology[138-140].

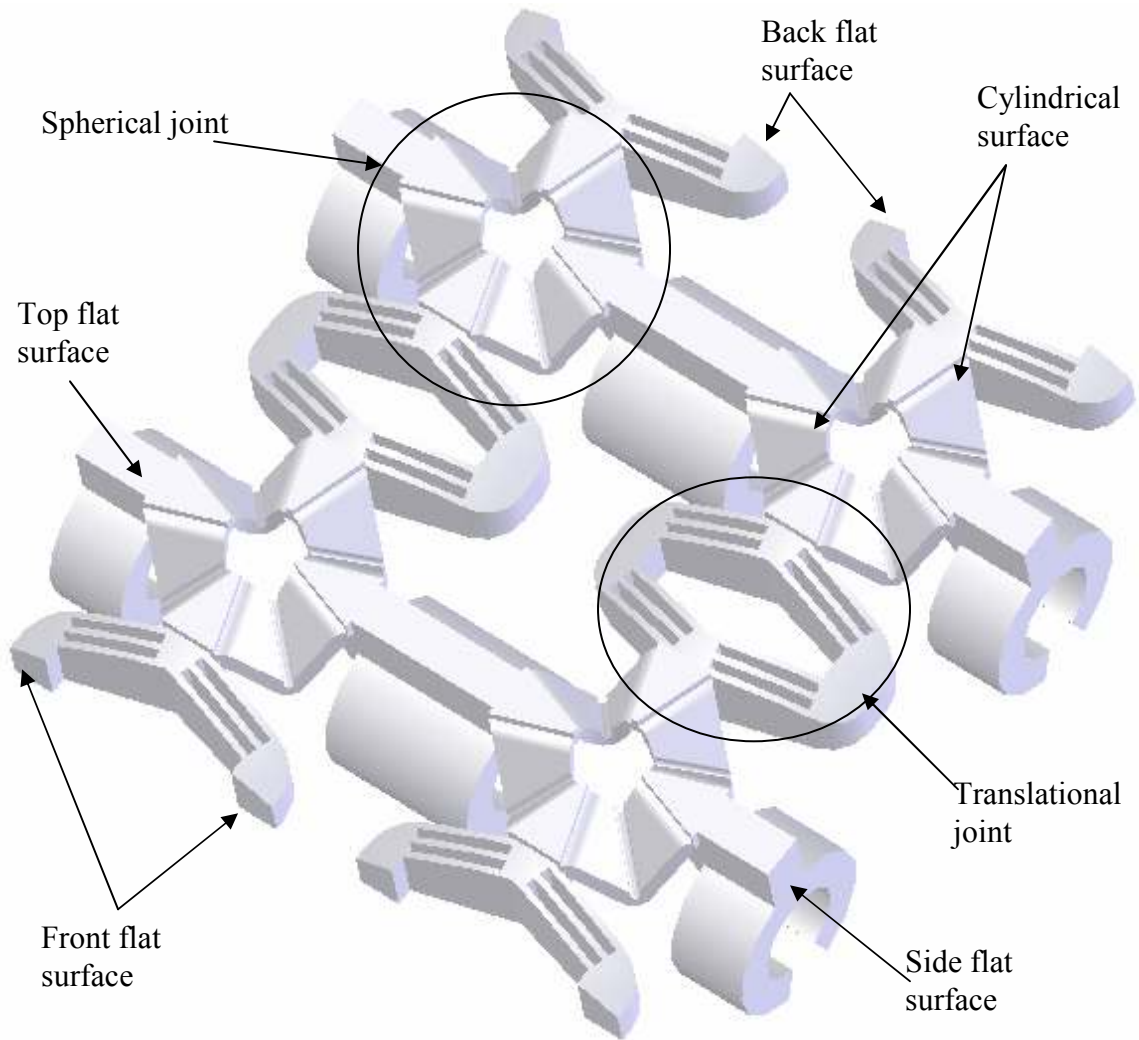


**Figure 7.1 Wind tunnel surface schematic**

In Figure 7.1, a side view of wind tunnel, surface, and pins is presented. As the requirements show, the surface needs to be placed on the pins and generate a smooth profile as pins move up and down. Also, the surface needs to stretch and bend to generate desired surface profile. The air is blowing at 30 psi and the surface should not let the air through it or bend due to the air pressure. Finally, it is desirable to generate the surface profile such that it is predictable by mathematical model[138, 139].

To meet the requirements, it is desired to develop a surface that is composed of structure and skin. The structure provides a spring mechanism that can bend and stretch. Then the skin provides the air tight capability. The compliant mechanism is used for the

structure and the rubber sheet is used for the skin. Figure 7.2 shows the compliant mechanism.



**Figure 7.2 Surface structure (compliant mechanism)**

The surfaces shown in Figure 7.2 are extensions of the surfaces that are designed previously in the digital clay project[139]. In Figure 7.2, the structure is composed of two sub mechanisms. Those are the spherical joint and the translational joint. The spherical and the translational joints provide bending and stretching mechanisms

respectively. The surface structure in Figure 7.2 has such a complex geometry that it is not easily producible with conventional manufacturing processes including injection molding, machining, etc. Therefore, the layer-based additive manufacturing process is chosen to take advantage of its capability in fabricating complex geometry. In this research, Stereolithography is used to fabricate the part.

In fabricating the part, some accuracy requirements are identified such that they could be specified to better achieve the initial design requirements. Firstly, there could be accuracy requirements specifications such as surface finish and position tolerance on the cylindrical surface in Figure 7.2. Bending takes place in the spherical joint. More specifically, it takes place around the cylindrical groove in the spherical joint. For the surface to bend predictably so that the profile can be predicted by a mathematical model, the cylindrical groove needs to be produced accurately. Therefore, some accuracy requirement can be specified on cylindrical surfaces. Secondly, there could be accuracy requirement specifications on the sides, front, and back flat surfaces in Figure 7.2. One of the design requirements is the air tightness. Placing the skin on the top surface can prevent the air passing through the surface. However, there could be some restrictions on sides, front, and back surfaces where they mate with other parts to not let the air pass through. In this case, accuracy requirements including surface finish and flatness tolerance can be specified on those surfaces. Finally, there could be some accuracy requirement specifications on the top surface. To place the skin on the top surface and provide a smooth surface profile, the top surface may need to be produced with some accuracy. Therefore, accuracy requirements including surface finish and flatness tolerance specifications can be imposed on the top surface.




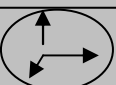


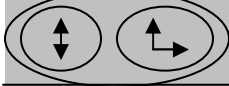
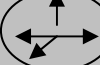


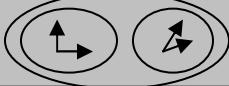
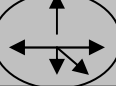




So far, various accuracy requirements specification on the various surfaces in Figure 7.2 is discussed. The following section discusses the realization of the condition part of the manufacturing rules using the surface structure.

### **7.2.3 Realization of the condition part of manufacturing rules**

In this section, the condition parts of the manufacturing rules are realized using the surface structure in Figure 7.2. Through this realization, the wind tunnel surface structure is justified as an appropriate test case. Also, the rules developed in this section are used for the retrieval performance testing. The accuracy measurements specifications, surface types, and their relative orientations of the surface structure in Figure 7.2 justify the wind tunnel surface design as an appropriate test case. With the geometry shown in Figure 7.2 and the possible accuracy requirements specifications described in section 7.2.2, the condition part of the meta rules in Figure 4.4 can be generated. Figure 7.3 shows the detail of how each condition part of meta rule in Figure 4.4 can be realized by specifying accuracy requirements on the surfaces in Figure 7.2.

In Figure 7.3, 16 meta rules (MR 2~17) from Figure 4.4 are selected. As discussed in Figure 4.10, there are two types of supplemented meta rules; Case 1 (MR1 in Figure 4.4 supplemented by meta rules in Figure 4.9) and Case 2 (MR2~17 supplemented by meta rules in Figure 4.7). Both supplementing meta rules determine binary feasible values for process variables (“thin” or “thick” for layer thickness in Figure 4.7 and “part of the process planning” or “not” for overcure in Figure 4.9) from simple design requirements (“surface at 90° with value 3” for layer thickness and “other than flat in 0°” for overcure). Therefore, the ways the design requirements are represented and values for the process variables are determined using subsumption are the same for the both

process variables (layer thickness and overcure). Consequently, demonstrating design requirements representation and determining feasible spaces for both process variables is redundant. Therefore, Case 2 meta rules are used for the empirical validation.

Relative orientations	Associated surfaces	Relative orientations	Associated surfaces
	Both sides flat		Side, front, back flat and Cylindrical
	Side flat, front flat		Top, front, and side flat
	Cylindrical		Top flat, cylindrical
	Side flat, Front and Back flat		Top, front, back, and side flat
	Side flat, cylindrical		Top, front, side flat and cylindrical
	Side flat, Front flat, Cylindrical		Top, front, back, side flat and cylindrical
	Side flat, Front flat, Cylindrical		Top, front, back, and both sides flat
	Side, front, and back flat		Top, front, back, both sides flat and cylindrical

**Figure 7.3 Relative orientations and the associated surfaces**

In Figure 7.3, the surfaces in Figure 7.2 are used to realize the condition part of the meta rules in Figures 4.4 and 4.5. The relative orientations in Figure 7.3 are the condition part of the meta rules in Figures 4.4 and 4.5. The associated surfaces are the surfaces in Figure 7.2. For example, specifying accuracy measurements on both side surfaces in Figure 7.2 realizes the opposite orientation shown in Figure 7.3 (First row and first column). Hence, the condition part of the meta rules in Figure 4.4 can be realized using the various surface orientations in Figure 7.2. The condition part of the

supplementary meta rules in Figures 4.7 and 4.9 can be realized by simply specifying accuracy requirements values on the surfaces discussed in Figure 7.3. Therefore, the geometric complexity in Figure 7.2 allows realization of the condition part of the entire case 2 meta rules. The following paragraphs discuss the realization of the condition part of rules and combination of meta rules and rules in a taxonomic structure.

To realize the condition part of the rules, accuracy measurements are specified on two types of surfaces. They are flat and cylindrical surfaces as shown in Figure 7.2. In Table 7.3, accuracy measurements that are applicable to each surface are shown.

**Table 7.3 Accuracy measurements specification on surfaces**

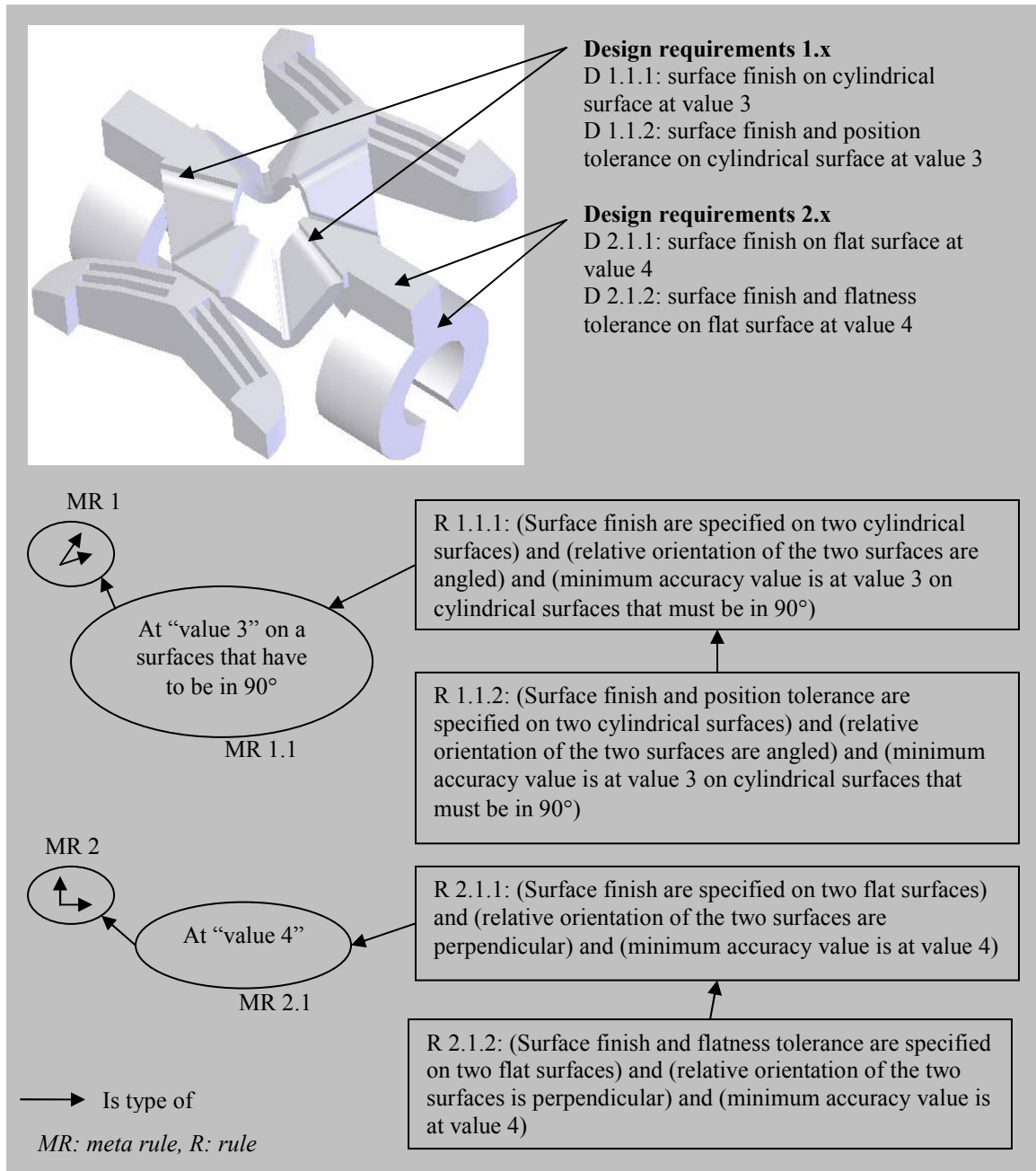
Surfaces	Accuracy measurements
Flat surface	<ul style="list-style-type: none"> <li>• Surface finish</li> <li>• Flatness tolerance</li> </ul>
Cylindrical surface	<ul style="list-style-type: none"> <li>• Surface finish</li> <li>• Position tolerance</li> </ul>

In Table 7.3, surface finish and flatness tolerance are associated with the flat surface. Also, surface finish and position tolerance are associated with the cylindrical surface. The purpose of introducing various types of accuracy measurement with various types of surface is to demonstrate retrieval of DFM problem based on accuracy models. For that purpose, having two types of surfaces and three different accuracy measurements is enough. This is because the way surfaces and accuracy measurements are represented and compared does not change with additional surfaces and accuracy measurements. Using the surfaces and accuracy measurements specified in Table 7.3 and surface orientations described in Figure 7.3, design requirements for the rules are developed. Table 7.4 illustrates the design requirements for the rules.

**Table 7.4 Additional design requirements for rules and DFM problems**

	Cylindrical surface	Flat surface
Rule 1	<ul style="list-style-type: none"><li>• Surface finish</li></ul>	<ul style="list-style-type: none"><li>• Surface finish</li></ul>
Rule 2	<ul style="list-style-type: none"><li>• Surface finish</li><li>• Position tolerance</li></ul>	<ul style="list-style-type: none"><li>• Surface finish</li><li>• Flatness tolerance</li></ul>

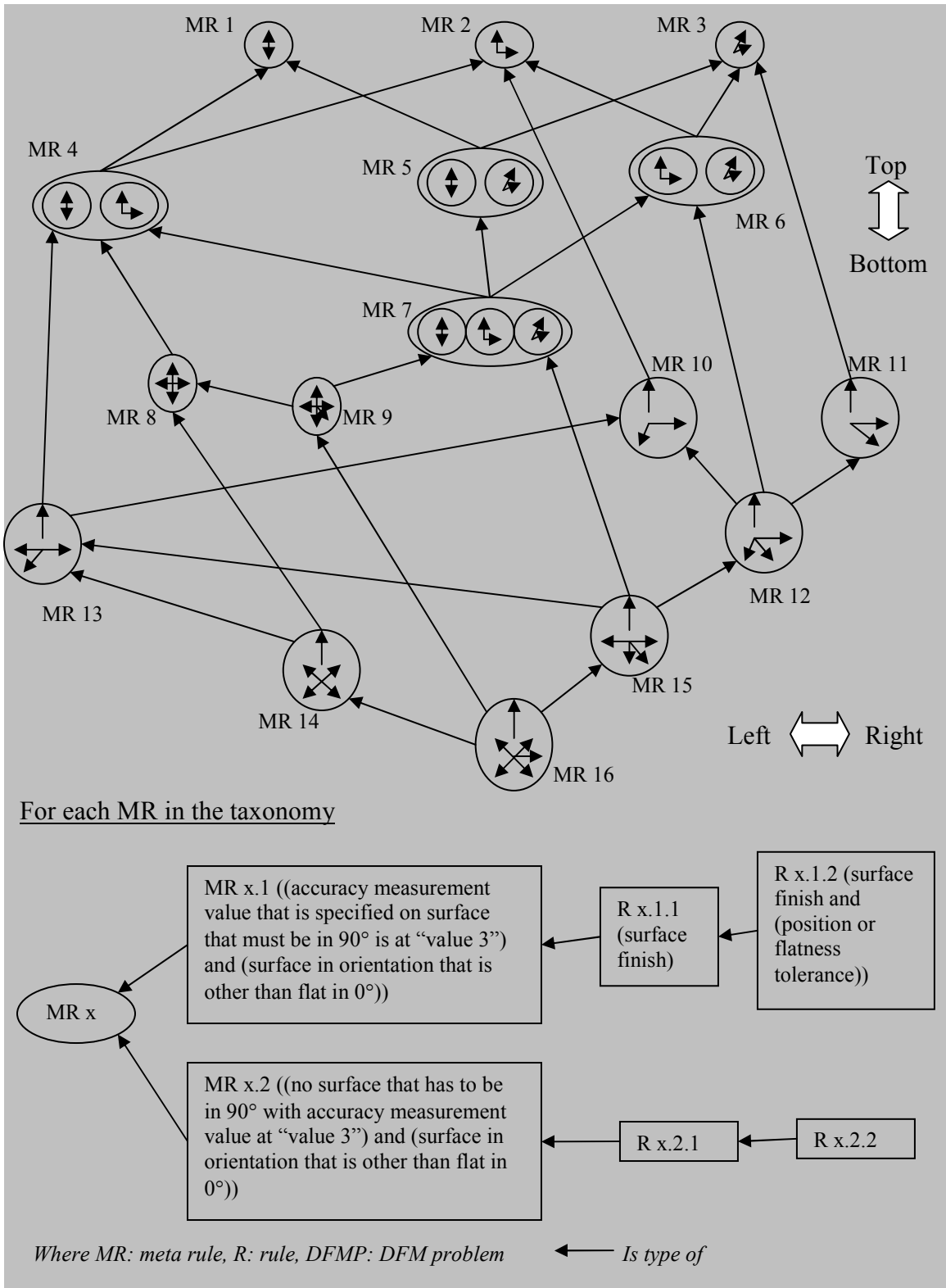
In Table 7.4, the additional design requirements for rule 1 and rule 2 are developed such that there is a subsumption relation between them (rule 1 subsume rule 2). Those design requirements are used to derive rules. Figure 7.4 shows the example of rules derivation.



**Figure 7.4 Development of rules**

In Figure 7.4, the condition part of the rules are developed using the surface structure. Then, the developed rules are added to the manufacturing rules taxonomy. For example, design requirement D 1.1.1 shows that surface finish is specified on cylindrical surfaces and their relative orientation is angled. Also, it shows that the accuracy

measurement value that is specified on surfaces that have to be in  $90^\circ$  is “value 3”. Then, the relative orientation of D 1.1.1 is used to identify the appropriate meta rule MR 1 (from Figure 4.5). Furthermore, those design requirements that describe the accuracy measurement value specified on surfaces that have to be at  $90^\circ$  orientation as well as relative orientation are used to identify meta rule MR1.1 (supplemented meta rule, Figures 4.5 and 4.7). Then, the design requirements for MR1.1 are augmented by surface finish, and the augmented design requirements become the condition part of the rule; R1.1.1. Finally, the manufacturing rule taxonomy is developed by manufacturing rules; MR1, MR 1.1, and R 1.1.1. A similar approach is used to develop manufacturing rule taxonomy for R 1.1.2, 2.1.1, and 2.1.2 by using D 1.1.2, 2.1.1, and 2.1.2 respectively. By following this procedure, two rules for each meta rule are developed. Therefore, total 32 meta rules (16 for part orientation and 2 for layer thickness and overcure,  $16 \times 2 = 32$ ) and 64 rules ( $32 \times 2 = 64$ ) are realized for empirical validation. Figure 7.5 shows the complete meta rules and rules taxonomy.



**Figure 7.5 Meta rules and rules taxonomy**

In Figure 7.5, the taxonomy of meta rules and rules that is realized by deriving design requirements using the surface structure shown in Figure 7.2 is presented. The taxonomy at the top of Figure 7.5 was originally shown in Figure 4.5; it is the meta rules taxonomy developed for relating design requirements (relative orientations of surfaces) to feasible space of part orientations. Also, the supplemented meta rules are shown below the taxonomy. These supplemented meta rules are developed by attaching the meta rules in Figures 4.7 and 4.9 to the meta rule taxonomy at the top of Figure 7.5. Two supplementary meta rules are appended for each meta rule in the taxonomy. Those meta rules determine the minimum layer thickness that needs to be considered and whether the overcure should be considered as a process variable for the given design requirements. Then, two rules are attached to each supplemented meta rule based on the discussions in Table 7.4 and Figure 7.4. In Figure 7.5, only design requirements are shown for meta rules and rules.

In short, the surface geometry in Figure 7.2 provides a sufficiently complex geometry that the meta rules in Figure 4.5 and the supplementary meta rules in Figures 4.7 and 4.9 can be practiced. Also, the geometry provides different types of surfaces such that the various accuracy measurements in Tables 7.3 and 7.4 can be specified as design requirements. Therefore, the wind tunnel surface design project satisfies the criteria shown in Table 7.2 and this completes the empirical structure validation discussed in section 1.3.2. For the empirical validation in this chapter, the meta rules and rules in Figure 7.5 are encoded in DL and used to construct the repository. The following section describes the test case set up.

### **7.3 Test case set up**

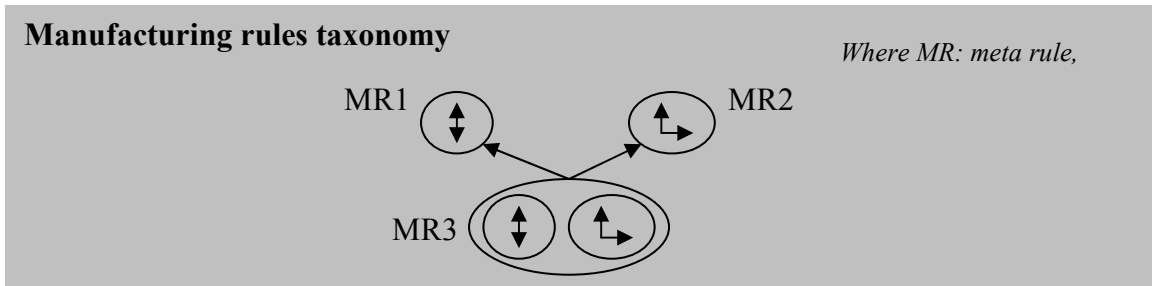
In this section, the details of the test case set up for taking the three measurements are discussed. The three measurements include DL applicability, retrieval performance, and computational complexity measure. The corresponding results are presented and discussed in section 7.5. The following paragraphs describe the test case set up for each measure in detail.

#### **7.3.1 DL applicability measure**

In this section, the case set up for testing DL applicability is discussed. First, the basic principle behind the testing is discussed followed by the details of the test case set up for DL applicability.

The reason for using description logic is to rely on its consistent and correct concept comparison capability in computing the manufacturing rule taxonomy. In description logics, the computed taxonomy is consistent and correct as far as the concept representations are unique for each node in the taxonomy[119]. In the retrieval method, it is expected that the manufacturing rule representations are unique and their interpretation to the real world is unique. Therefore, the computed manufacturing rule taxonomy is always expected to be consistent and correct.

As discussed with robot arm example in chapter 6, the repository is structured by the manufacturing rules taxonomy, and the repository structure is expected to expand as new manufacturing rules are introduced. Under this dynamic expansion, the repository structure is expected to be consistent and correct. To illustrate this, an example is presented in Figure 7.6.



**Figure 7.6 Illustration of testing strategy**

In Figure 7.6, the structure of the repository is shown. For simplicity, three meta rules (MR1, 2, and 3) are considered. Also, it is assumed that there are DFM problems under each meta rule. Based on these assumptions, the following paragraph discusses the basic idea of DL applicability testing.

If the repository is completely empty, the introduction of DFM problems under MR1 causes the introduction of MR1. Then, the introduction of DFM problems under MR2 introduces MR2. Finally, the introduction of DFM problems under MR3 introduces MR3. As discussed in the storing algorithm (Figure 6.4), the taxonomy in the repository is recomputed each time new manufacturing rules are introduced. Hence, the manufacturing rules taxonomy is computed after introduction of MR2 and MR3. The key idea of consistent and correct taxonomy computation is that the computed taxonomy should always be the same as Figure 7.6 no matter which meta rule is introduced first. In other words, introduction of meta rules in the order of 1, 2, 3 or 2, 1, 3, or 3, 1, 2 should not change the taxonomy in Figure 7.6. Therefore, the consistency and correctness of the manufacturing rules taxonomy computation is tested by observing the computed taxonomy with the varying order of manufacturing rules addition. The following paragraphs discuss the details of the test case set up for DL applicability testing.

For the demonstration of description logic’s applicability, 16 meta rules for the part orientation developed in section 7.2.3 are encoded into DL and used to construct the repository. The taxonomy size is increased by incrementing the meta rules gradually. Each time new meta rules are introduced, the taxonomy is computed and compared to the expected taxonomy. For DL applicability measure, supplementary meta rules in Figures 4.7 and 4.9 and rules are not considered because the ways those manufacturing rules are represented and the way the taxonomy is computed are the same as the 16 meta rules for the part orientation. Hence, the 16 meta rules are sufficient to demonstrate the consistent and correct manufacturing rules taxonomy computation. The detailed scenarios for varying the meta rules introduction to the repository are shown in Tables 7.5.

**Table 7.5 Scenarios and sets of DFM problems in the repository**

Scenario	Set 1	Set 2	Set 3	Set 4
Top → Bottom	MR 1, 2, 3, 4	MR 5, 6, 7, 8	MR 9, 10, 11, 12	MR 13, 14, 15, 16
Bottom → Top	MR 13,14,15,16	MR 9, 10, 11, 12	MR 5, 6, 7, 8	MR 1, 2, 3, 4
Left → Right	MR 1, 4, 8, 13	MR 5, 7, 9, 14	MR 2, 10, 15, 16	MR 3, 6, 11, 12
Right → Left	MR 3, 6, 11, 12	MR 2, 10, 15, 16	MR 5, 7, 9, 14	MR 1, 4, 8, 13
Random	MR 6, 11, 2, 9	MR 5, 10, 14, 13	MR 4, 15, 1, 12	MR 3, 7, 8, 16

Table 7.5 shows the various scenarios and corresponding sets of meta rules. The five scenarios are developed to simulate how the manufacturing rule taxonomy in the repository expands. In the repository, the manufacturing rule taxonomy can expand toward the top, left, bottom, right, and any random direction (the directions are shown in Figure 7.5). Consequently, there are five scenarios of expansion including top to bottom, bottom to top, left to right, right to left, and random. For each scenario, there are four sets of meta rules that are loaded incrementally. In other words, meta rules are added to the repository by the order of set numbers (ex: meta rules in set 1 first, then set 2, then set

3, finally set 4) in Table 7.5. Each time the meta rules are incremented, the taxonomy is computed and compared to the expected taxonomy. For DL applicability testing, 16 expected taxonomies for each set and scenario in Table 7.5 are developed and shown in Appendix C. There are 16 sets because set 4 is common for all scenarios. The meta rules are encoded using Protégé software and the demonstrations are shown in Figures 5.10, 5.11, 5.12, and 6.10. Hence, the demonstration of encoding is omitted here. In section 7.5.1, the collected results are discussed.

The successful completion of the above demonstration will show that the description logic can represent the design requirements and the subsumption algorithm can correctly and consistently structure the repository. Hence, this demonstrates the theoretical validity of hypotheses 1, 2, and 3 in Table 1.7. Hypotheses 1 and 2 are about computationally mapping the design and process planning domains using DL. By validating those hypotheses, the theoretical foundation for mapping design and process planning domains using DL is established. Hence, the empirical testing implicitly demonstrates the validity of hypotheses 1 and 2. Hypothesis 3 is validated by computing the manufacturing rules taxonomy in the repository consistently and correctly. Hence, demonstrating consistent and correct manufacturing rules taxonomy computation in the repository using description logic  $\mathcal{ALTE}$  demonstrates the validity of hypotheses 1, 2, and 3.

### **7.3.2 Retrieval performance measure**

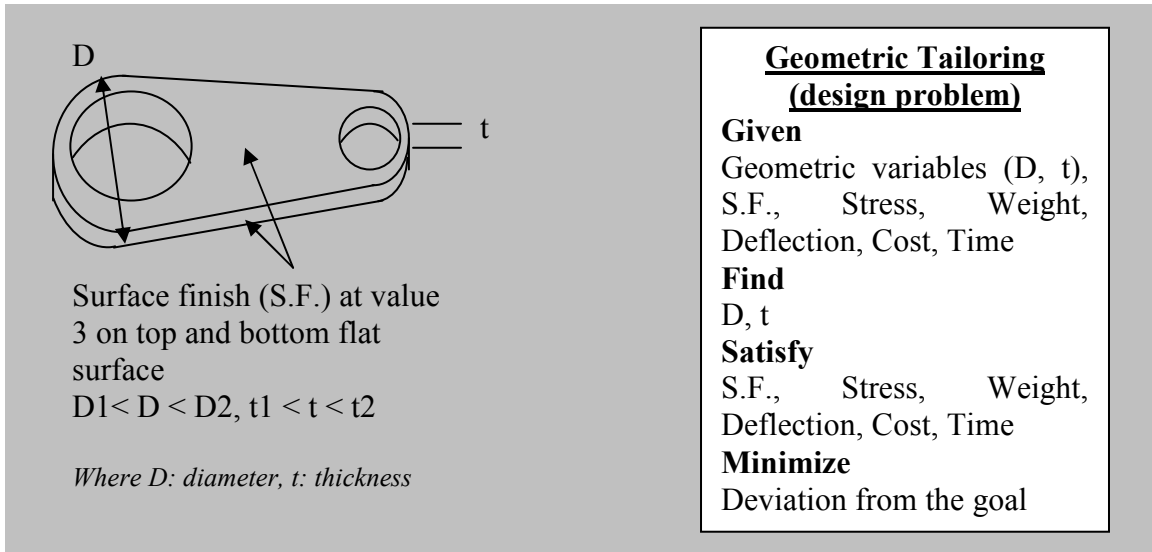
The next demonstration needs to show the method's ability to relate the given design requirements to the relevant DFM problems. As discussed previously, the retrieval method relates the given design requirements to the relevant DFM problems

through mapping between the design and process planning domains. Hence, the retrieval performance testing tests the ability of the method to determine appropriate MPTs for the given design requirements. Since determining appropriate MPTs is basically determining appropriate rules, the retrieval performance test addresses the ability to determine appropriate rules for the given design requirements. The following paragraphs discuss the details.

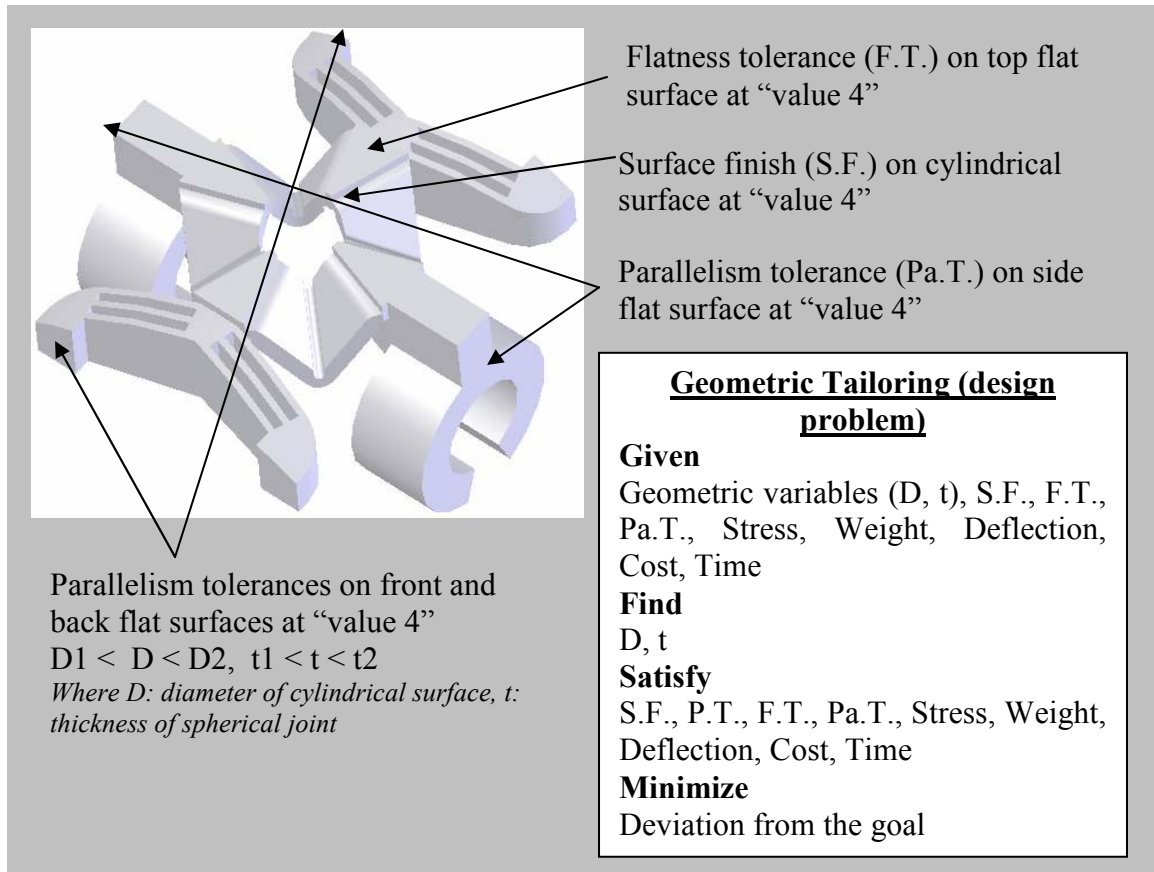
For this demonstration, 32 meta rules and 64 rules that are developed in Figure 7.5 are used to structure the repository and 16 new design requirements are developed for queries. Then, the manufacturing rules taxonomy is computed to become the repository and the 16 queries are used to test the retrieval method in determining the correct rules. The 16 design requirements are developed to be subsumed by the condition part of the 16 meta rules for part orientation in Figure 7.5. To further test the ranking capability for layer thickness and overcure, 8 of the 16 design requirements are developed to require thin layer thickness (with the accuracy measurement value specified at “value 3” on a surface that has to be in 90°). As discussed previously, the overcure always needs to be considered in the process planning for meta rules in Figure 7.5. Hence, rankings based on overcure are not performed. The accuracy requirements are assigned to the 16 design requirements such that 8 of the 16 are assigned with accuracy measurements shown in Tables 7.3 and 7.4. The other 8 design requirements are assigned with accuracy measurements that are different than the ones in Tables 7.3 and 7.4. This is to test the method’s ability to rank based on accuracy models.

The design requirements are derived from the robot arm and wind tunnel surface structure. This is to demonstrate manual extraction of the design requirements from the

simple and complex geometry. Figures 7.7 and 7.8 present example design problems and corresponding design requirements that are developed from the robot arm and wind tunnel surface respectively.



**Figure 7.7 Example design problem and design requirements using robot arm**



**Figure 7.8 Example design problem and design requirements using wind tunnel surface**

As shown in Figures 7.7, and 7.8, the two design problems are developed using the robot arm and the wind tunnel surface. In the design problem in Figure 7.7, surface finish is specified on the top and bottom flat surfaces and they are restricted at “value 3”. The relative orientation of the flat surfaces is opposite. Based on this, R 1.1.1 from Figure 7.5 is applicable to this design problem. The corresponding feasible space of part orientation is infinite. The minimum layer thickness that needs to be considered is thin and overcure needs to be considered in process planning. The required accuracy model for formulating the DFM problem is surface finish on flat surface.

In Figure 7.8, parallelism tolerance is specified on the sides, front and back flat surfaces at “value 4”. On the cylindrical surfaces, surface finish is specified at “value 4”. On the top flat surfaces, flatness tolerance is specified at “value 4”. According to this, the condition part of meta rule MR16.2 matches the given design requirements. However, there are no rules that exactly match the given design requirements because the design requirements contain parallelism tolerance on flat surfaces. The corresponding feasible space of the part orientation is constant. The layer thickness is unknown and overcure should be part of the process planning. The accuracy models required for formulating the DFM problem are surface finish, flatness tolerance, and parallelism tolerance. By following a similar procedure, 16 design requirements are developed with corresponding expected query results. In Table 7.6, the 16 design requirements are presented. In Table 7.7, an example query result is presented.

**Table 7.6 Design requirements for querying DFM problems**

Design requirement	Model used	Accuracy measurements, values, and surface types
d.r.1	Robot Arm	Surface finish on top and bottom surface at “value 4”
d.r.2	Robot Arm	Surface finish and flatness tolerance on top and side surface at “value 3”
d.r.3	Robot Arm	Flatness tolerance on side flat surfaces at “value 4”
d.r.4	Wind Tunnel surface	Flatness tolerance on top flat surface at “value 4” & Surface finish on side flat surfaces at “value 3”
d.r.5	Wind Tunnel surface	Surface finish and position tolerance on cylindrical surface at “value 4” & Parallelism tolerance on side flat surfaces at “value 3”
d.r.6	Wind Tunnel surface	Flatness tolerance on front and one side surface at “value 4” & Surface finish on cylindrical surface at “value 4”
d.r.7	Wind Tunnel surface	Parallelism tolerance on side flat surfaces at “value 3” & Surface finish on cylindrical surface at “value 4” & Surface finish on front flat surface at “value 3”
d.r.8	Wind Tunnel surface	Parallelism and perpendicular tolerance on sides, front, and back flat surface at “value 4”
d.r.9	Wind Tunnel surface	Surface finish on sides, front, and back flat surface at “value 4” & Surface finish on cylindrical surface at “value 3”
d.r.10	Wind Tunnel surface	Flatness tolerance on front and one side surface at “value 4” & Surface finish on top surface at “value 3”
d.r.11	Wind Tunnel surface	Surface finish on top surface at “value 3” & Surface finish on cylindrical surface at “value 4”
d.r.12	Wind Tunnel surface	Surface finish on top surface at “value 3” & Flatness tolerance on front and one side surface at “value 4” & Surface finish on cylindrical surface at “value 4”
d.r.13	Wind Tunnel surface	Flatness tolerance on side flat surfaces at “value 3” & Flatness tolerance on front flat surface at “value 3” & Surface finish on top surface at “value 3”
d.r.14	Wind Tunnel surface	Parallelism and perpendicular tolerance on sides, front, and back flat surface at “value 3” & Flatness tolerance on top surface at “value 3”
d.r.15	Wind Tunnel surface	Parallelism tolerance on side flat surfaces at “value 3” & Flatness tolerance on front flat surface at “value 3” & Surface finish on top surface at “value 3” & Surface finish on cylindrical surface at “value 4”
d.r.16	Wind Tunnel surface	Parallelism tolerance on sides, front, and back flat surface at “value 4” & Surface finish on cylindrical surface at “value 4” & Flatness tolerance on top flat surface at “value 4”
<p><i>Where &amp;: ‘and’ that joins two design requirements (combination of surface type and accuracy measurement)</i></p>		

**Table 7.7 Example of query result for d.r.4**

MR x	MR 4				MR 2				MR 1			
MR x.y	MR 4.2		MR 4.1		MR 2.2		MR 2.1		MR 1.2		MR 1.1	
MPT	MPT3.2		MPT3.1		MPT2.2		MPT2.1		MPT1.2		MPT1.1	
R	4.2.2	4.2.1	4.1.2	4.1.1	2.2.2	2.2.1	2.1.2	2.1.1	1.2.2	1.2.1	1.1.2	1.1.1
Rank	1	2	3	4	5	6	7	8	9	10	11	12

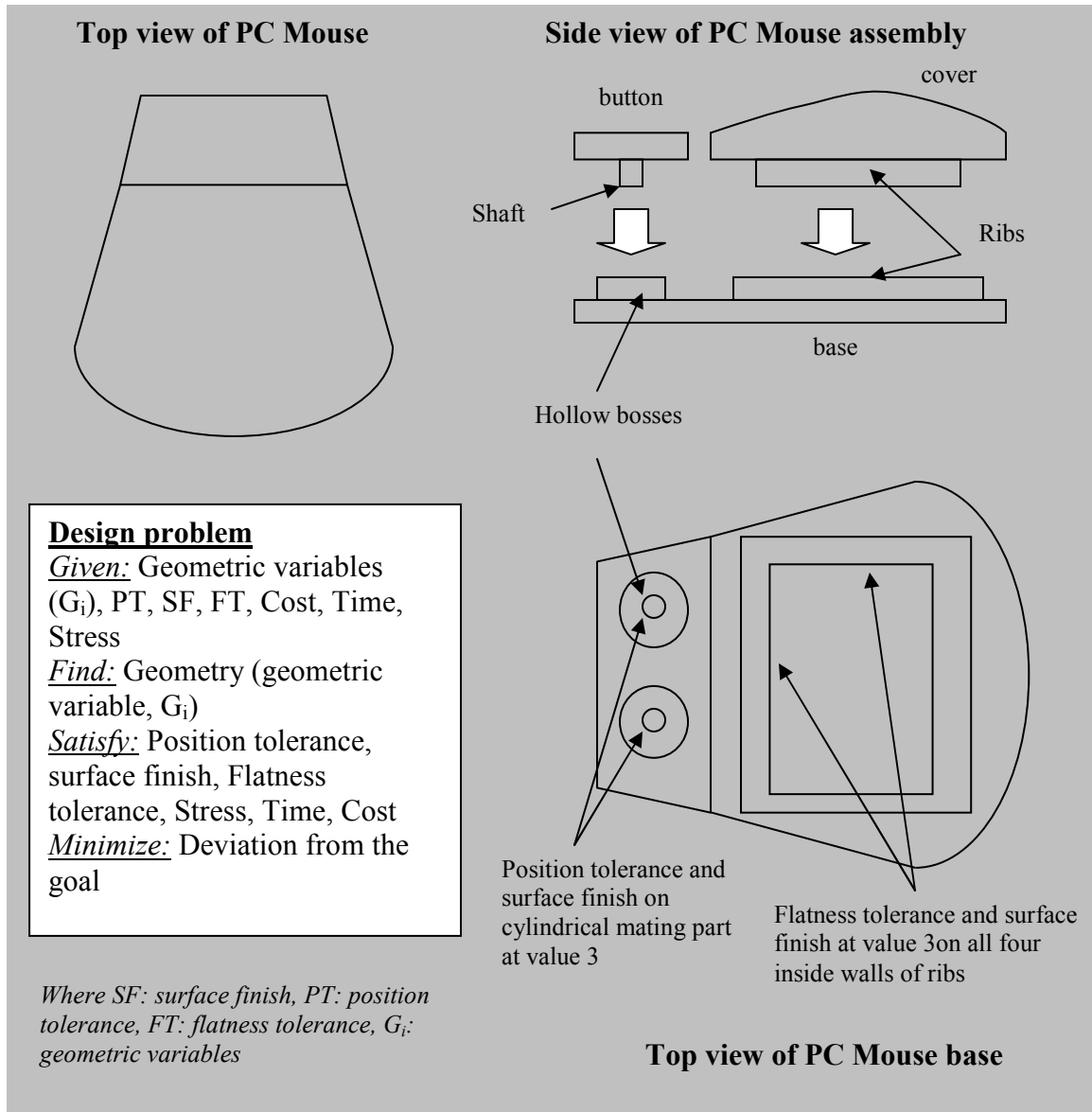
As discussed previously, the design requirements in Table 7.6 are developed such that its condition part matches the 16 meta rules (MR x) in Figure 7.5. Then, there are 8 design requirements (d.r.2, 4, 5, 6, 7, 9, 13, 14, 15) that require thin layer thickness. Also, there are 8 design requirements (d.r.3, 4, 5, 7, 8, 14, 15, 16) that require accuracy models that are not supported by the manufacturing rules in Figure 7.5. These are used to test the method's ability to rank based on layer thickness and accuracy models. For the design requirements developed in Table 7.6, the expected manufacturing rule retrieval and ranking are also developed. Table 7.7 shows an example. In Table 7.7, the expected manufacturing rules for d.r.4 in Table 7.6 and their ranks are presented. Design requirement d.r.4 is directly subsumed by the condition part of meta rule MR 4. Hence, MR 4 and the subsuming meta rules, including MR 2 and 1, are considered to be relevant for process planning part orientation. This is shown in the first row of Table 7.7. Then the meta rules (MR x.y) under each meta rule (MR x) are further ranked for layer thickness. Design requirement d.r.4 does not require thin layer thickness because there is no surface that has to be in 90° orientation and whose accuracy requirement is specified at "value 3". Therefore, the layer thickness is unknown and meta rules having layer thickness not influenced by accuracy requirements are ranked higher than the others. This is shown in the second row of Table 7.7. Then, the corresponding MPTs (from Table 4.2) for the MR x.y are shown in the third row. Finally, the accuracy models

required for d.r.4 are flatness tolerance on flat surface and surface finish on flat surface. Therefore, the rules that contain both accuracy models are ranked higher than the others. This is shown in the third row of Table 7.7 such that rules x.y.2 is ranked higher than x.y.1.

For the actual retrieval of DFM problems, the above procedure is carried out first, then the DFM problems under Rs are retrieved and ranked in the same way as the Rs in Table 7.7. The DFM problems under a rule (R) contain design requirements that match exactly to the condition part of a rule. In this way, the DFM problems are retrieved and ranked to support process planning as rules do. The expected results for each design requirement in Table 7.6 are developed and presented in Appendix C.

Successful completion of the above testing demonstrates the validity of hypotheses 1, 2, and 4. This is because the validity of hypothesis 4 is established by developing an algorithm that is able to retrieve relevant DFM problems for the given design requirements using subsumption. In this research, retrieval of the relevant DFM problems is basically retrieval of the relevant rules (R) because DFM problems are classified by the rules. Hence successfully retrieving relevant rules for the given design requirements demonstrates the validity of hypotheses 1, 2, and 4. As discussed previously, the validity of hypotheses 1 and 2 is implicitly demonstrated in empirical validation.

To illustrate the usefulness of the retrieved DFM problem, a demonstration of DFM problem formulation and solution generation is presented. The demonstration is performed using a design problem and a retrieved DFM problem. Figure 7.9 shows the example design problem.



**Figure 7.9 PC Mouse design problem**

In Figure 7.9, the problem of designing a base for a PC mouse is described. The PC mouse is assumed to consist of three pieces including a button, a cover, and a base. The objective is to design and fabricate the base such that it satisfies the given design requirements in Figure 7.9. The design requirements include geometric freedom ( $G_i$ ), position tolerance, flatness tolerance, surface finish, stress, cost, and time. Then,

formulation and solution generation of a DFM problem for the design problem in Figure 7.9 is demonstrated with a retrieved DFM problem.

First, the closest rule is retrieved from the repository. From the retrieved rule, required accuracy models and feasible spaces of process variables are determined. Then, a DFM problem is assumed to be under the retrieved rule and to provide the determined accuracy models and feasible spaces of process variables. Using the retrieved DFM problem, formulation and solution generation of a DFM problem for the design problem in Figure 7.9 is demonstrated. The closest rule is retrieved among the 64 rules that are stored in the repository (Figure 7.5).

### **7.3.3 Computational feasibility measure**

Besides the storing and retrieving performance, computational complexity needs to be investigated. Theoretically, the subsumption performance of  $\mathcal{AL}\mathcal{E}$  is non-deterministic polynomial time with respect to problem size. In this research, problem size is the number of atomic concepts that constitute the design requirements that represent the manufacturing rules. To represent the most complex manufacturing rules, around 45 atomic concepts are required. The simplest manufacturing rules require about 4~5 atomic concepts. This is because there are 2~6 atomic concepts that are used to represent the condition part of the meta rule for part orientation. There are two atomic concepts for representing the condition part of the meta rule for layer thickness and overcure. Finally, there are 30 additional atomic concepts for representing the condition part of the rules. The 30 additional atomic concepts are accuracy models that consist of surface and accuracy measurements. Table 7.8 shows the 30 accuracy models.

**Table 7.8 Accuracy models**

	Flat	Cylindrical	Conical	Spherical
Surface finish	V	V	V	
Flatness	V			
Perpendicularity	V	V	V	
Straightness	V	V	V	
Symmetry		V	V	V
Circularity		V	V	V
Cylindricity		V		
Concentricity		V	V	V
Position		V	V	V
Angularity	V	V	V	
Run out		V	V	
Total run out		V	V	

In Table 7.8, the accuracy models are shown by combinations of surface and accuracy measurements. Columns represent the types of surfaces and rows represent accuracy measurements. Then, the ‘v’ marks represent accuracy models that are combinations of surface type and accuracy measurement. For instance, the ‘v’ mark in the intersection of the second column and the third row represent flatness tolerance on flat surface. In short, the number of atomic concepts that represent design requirements ranges approximately from 2 to 45 in this research.

To determine the computational feasibility of subsumption, two design requirements are selected such that there is a subsumption relation between them. Then, the atomic concepts of the two design requirements are increased from 2 to 50 by 1 and subsumption computation is performed at each atomic concept addition. The subsumption computation time is measured and plotted (subsumption computation time versus number of atomic concepts). This test is performed to determine subsumption performance for the way  $\mathcal{AL}\mathcal{E}$  is used in this research.

In the method, subsumption is heavily used when the manufacturing rule taxonomy is computed. The manufacturing rule taxonomy is computed when the new manufacturing rule is inserted or new DFM problems are introduced with new rules or meta rules into the repository. In the future, the manufacturing rules taxonomy is expected to increase its size. Therefore, the speed of increase in taxonomy computation time with increasing number of manufacturing rules is of primary interest. The computation time of manufacturing rules taxonomy increases by two reasons. One is the increase of the atomic concepts that constitute the design requirements (increasing problem size). The other is increase of the number of the manufacturing rules in the taxonomy. Therefore, the worst case occurs when manufacturing rules taxonomy expands with increasing complexity of design requirements that represent manufacturing rules.

To investigate the feasibility of manufacturing rule taxonomy computation, two measurements are taken. One measures taxonomy computation time with increasing complexity in design requirements only. The other measures taxonomy computation time with increasing both complexity of design requirements and the number of manufacturing rules. For both measurements, the meta rules in Figure 7.5 are used. To measure the influence of increasing complexity of design requirements, 32 rules are developed in addition to the meta rules in Figure 7.5. The 32 rules are developed such that their condition part consists of accuracy models and the condition part of the corresponding meta rule. The manufacturing rules taxonomy is computed and its computation time is measured by incrementing the accuracy models in the condition part of the rules. The accuracy models in the condition part of the rules increased from 3 to 30 by 3 and the

manufacturing rules taxonomy is computed each time the accuracy models are incremented. Table 7.9 shows the increase of accuracy models for each manufacturing rule taxonomy computation.

**Table 7.9 Incremental accuracy models**

Accuracy model sets (number of accuracy models)	Additional accuracy models
Set 1 (3)	Surface finish on flat, Surface finish on cylindrical, Surface finish on conical
Set 2 (6)	Set 1 + Flatness on flat, Perpendicularity on flat, Perpendicularity on cylindrical
Set 3 (9)	Set 2 + Perpendicularity on conical, Straightness on flat, Straightness on cylindrical
Set 4 (12)	Set 3 + Straightness on conical, Symmetry on cylindrical, Symmetry on conical
Set 5 (15)	Set 4 + Symmetry on spherical, Circularity on cylindrical, Circularity on cylindrical
Set 6 (18)	Set 5 + Circularity on spherical, Cylindricity on cylindrical, Concentricity on cylindrical
Set 7 (21)	Set 6 + Concentricity on conical, Concentricity on spherical, Position on cylindrical
Set 8 (24)	Set 7 + Position on conical, Position on spherical, Angularity on flat
Set 9 (27)	Set 8 + Angularity on cylindrical, Angularity on conical, Run out on cylindrical
Set 10 (30)	Set 9 + Run out on conical, Total run out on cylindrical, Total run out on conical

In Table 7.9, the left column shows the set numbers and the right column shows corresponding accuracy models. The accuracy models in each set increase accumulatively as shown in Table 7.9. The manufacturing rule taxonomy is computed using each set in the condition part of the rules (32 rules). Each time the manufacturing rules taxonomy is computed, the computation time is collected. After the taxonomy computation time is collected, the plot that shows taxonomy computation time versus number of accuracy models is produced.

To measure the influence of both increasing complexity of design requirements and number of manufacturing rules, manufacturing rules taxonomy is computed with increasing complexity of design requirements and number of manufacturing rules. For

this measurement, the rules for each accuracy model set in Table 7.9 are accumulated each time manufacturing rules taxonomy is computed. Table 7.10 shows the strategy.

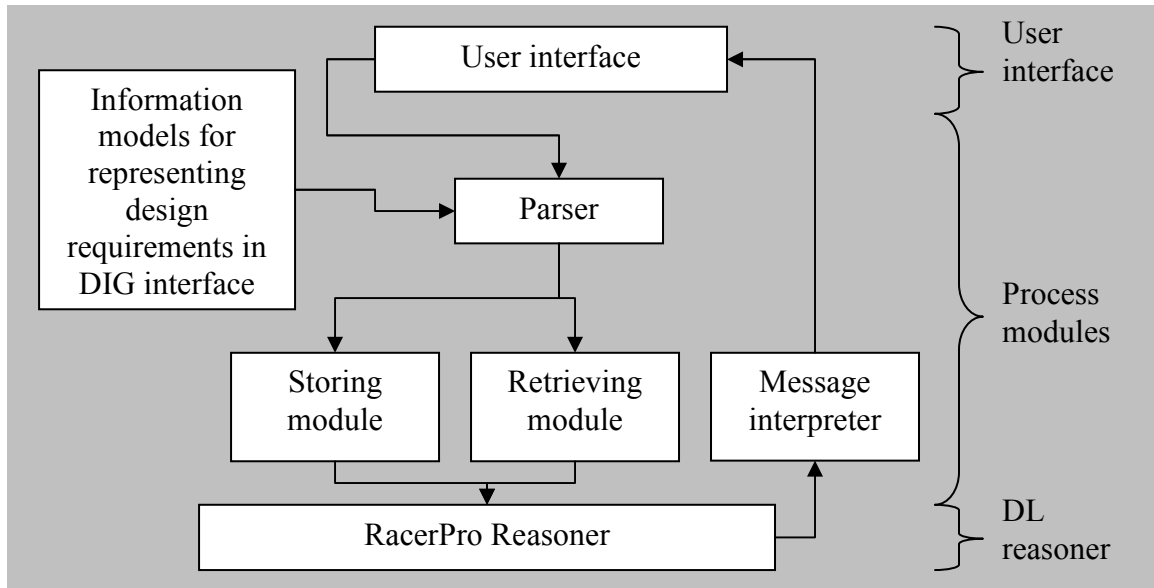
**Table 7.10 Incremental rules**

Rule sets (number of rules)	Rules for sets that are in Table 7.9
Rule set 1 (32)	Rules for set1
Rule set 2 (64)	Rules for set1, 2
Rule set 3 (96)	Rules for set1, 2, 3
Rule set 4 (128)	Rules for set1, 2, 3, 4
Rule set 5 (160)	Rules for set1, 2, 3, 4, 5
Rule set 6 (192)	Rules for set1, 2, 3, 4, 5, 6
Rule set 7 (224)	Rules for set1, 2, 3, 4, 5, 6, 7
Rule set 8 (256)	Rules for set1, 2, 3, 4, 5, 6, 7, 8
Rule set 9 (288)	Rules for set1, 2, 3, 4, 5, 6, 7, 8, 9
Rule set 10 (320)	Rules for set1, 2, 3, 4, 5, 6, 7, 8, 9, 10

In Table 7.10, 10 rule sets are developed by accumulating previous rule sets. For instance, rule set 2 contains rules for accuracy model sets 1 and 2 whereas rule set 1 only contains rules for accuracy model set 1. Hence, increasing rule set number in Table 7.10 means increasing the number of rules and complexity of design requirements that represent the rules. The number of rules is incremented by 32. The manufacturing rule taxonomy is computed using each rule set in Table 7.10. Each time the manufacturing rules taxonomy is computed, computation time is collected. Then, a plot that shows taxonomy computation time versus number of manufacturing rules is produced.

#### **7.4 Test bed implementation**

In this section, the test bed for empirical performance validation is discussed. A graphical illustration for DFM framework test bed is shown in Figure 7.10.



**Figure 7.10 Graphical illustration of DFM framework test bed**

In Figure 7.10, there are three major components in the test bed; reasoner, user interface, and processing modules. The user interface is used to collect information from the user. The computation modules are used to represent the collected information and either stores or retrieves from the repository. The information models, storing algorithm and retrieving algorithm are implemented in the computation modules. The reasoner performs subsumption computation and stores manufacturing rules taxonomy. The following paragraph describes details.

First, the collected design requirements from the user interface are parsed and represented by encoded information models. In this research, the DIG interface is used to encode the design requirements. The DIG interface is a standardized XML (extensible markup language) interface to description logic systems developed by Description logic Implementation Group[141]. Then, the represented design requirements are used to store and retrieve manufacturing rules using storing and retrieving modules respectively. Those modules are implementations of storing and retrieving algorithms and those



modules interact with reasoner. In this research, the reasoner is considered as the repository and used to store the manufacturing rules taxonomy. For the reasoner, an open source program called RacerPro is utilized. The detailed discussion on reasoner is presented in chapter 5. The interface to the reasoner is developed using the DIG interface. Most of the description logic systems that are currently available as open source systems implement the DIG interface. The information models for design requirements are initially implemented in OWL (web ontology language) using Protégé. Then, the information models are translated into the DIG interface and stored in the information models component in Figure 7.10. The retrieved manufacturing rules that are encoded in DIG interface are parsed and represented into a more human friendly message. The user interface and process modules are developed using Microsoft C# version 2.0 and XQuery (XML Query language) version 1.0[142-146]. In Figures 7.11 and 7.12, the front end of user interface is presented.



**DFM Framework Testbed**

Form Query | Retrieval and Ranking



**Orientation**

In Plane

Out of Plane

**Measurements, Restriction Level and Surface**


Measurements	Restriction Level	Surface
Concentricity Tolerance	None	None
Circularity Tolerance	None	None
Cylindricity Tolerance	None	None
Flatness Tolerance	None	None
Parallelism Tolerance	None	None
Perpendicularity Tolerance	None	None
Position Tolerance	None	None
Surface Finish	None	None

Reset      Insert      Form Query

---

**Formed Query**

Part Orientation

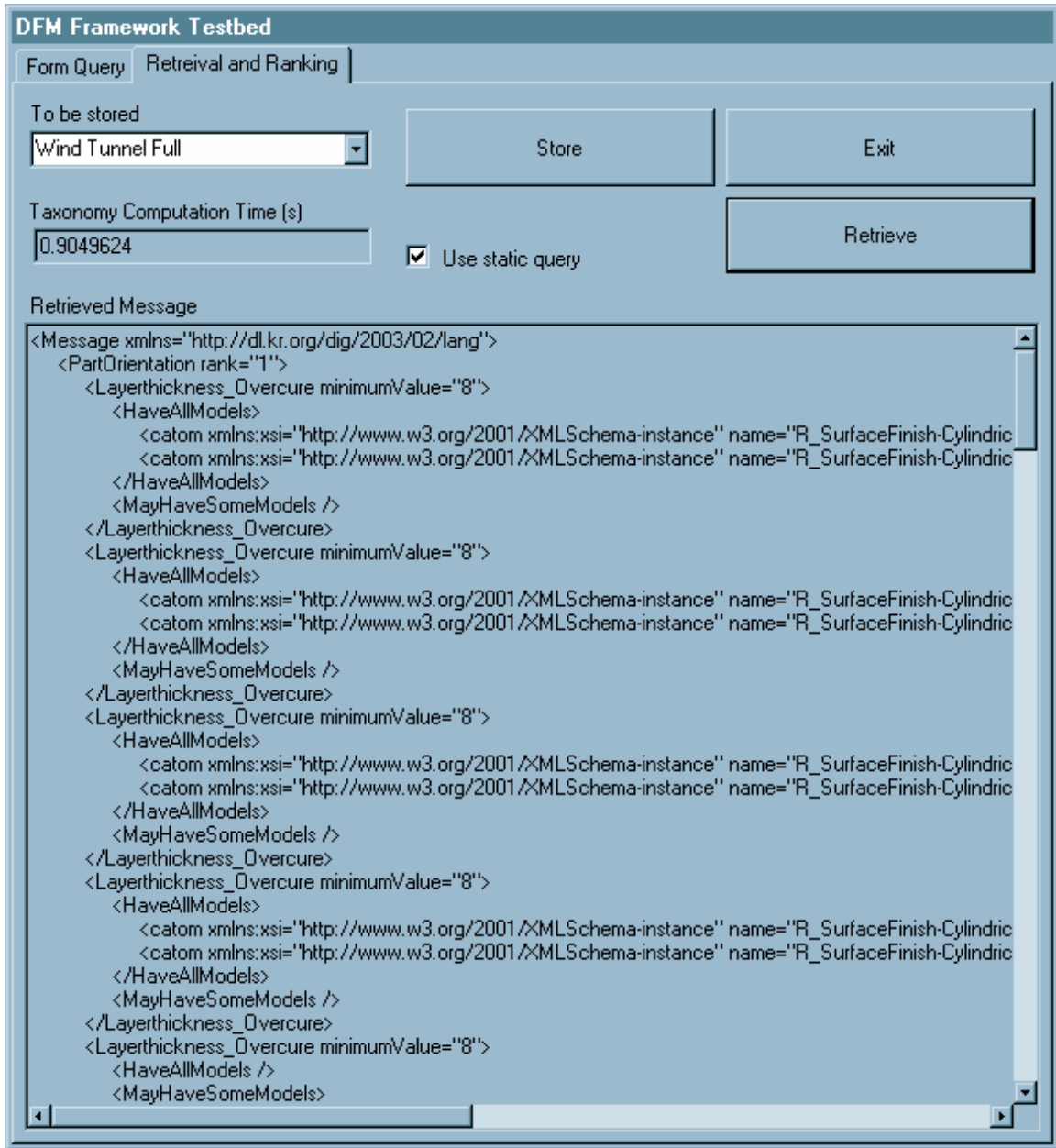


Layer thickness

```
<Message xmlns="http://dl.kr.org/dig/2003/02">
  <Message xmlns="http://dl.kr.org/dig/2003/02">
    <and>
      <some>
        <ratom name="is_3D" />
        <catom name="True" />
      </some>
      <all>
        <ratom name="is_3D" />
        <catom name="True" />
      </all>
      <some>
        <ratom name="have_Perpendicul
          <catom name="Perpendicular_Ori
        </some>
        <some>
          <ratom name="have_Angled_Orien
            <catom name="Angled_Orientati
          </some>
        </all>
      </some>
    </and>
  </Message>
</Message>
```

Overcure

Figure 7.11 Graphical user interface for forming query



**Figure 7.12 Graphical user interface for displaying retrieval results**

In Figure 7.11, the user interface for forming the design requirements is shown. Using this interface, the relative orientation of surfaces, surface types, associated accuracy requirements and corresponding values can be specified. The orientations of the individual surface can be selected by selecting the appropriate arrow in the top left. Then associated accuracy measurements, values and surface types can be selected by

using drop down boxes in the top right. The selected information is inserted as part of the design requirements by pressing the insert button. The above procedure is repeated until all the distinctive orientations are selected. Once all the information is selected and inserted, the design requirement is formed by pressing the form query button. Then, the appropriate solution strategy and required accuracy models are determined and displayed. Also, the feasible space of part orientation, minimum layer thickness value, and whether overcure is part of the process variable or not are determined and displayed.

In Figure 7.12, the user interface for displaying the query result is presented. In this interface, manufacturing rules to be stored can be selected by selecting the manufacturing rules sets in the drop down box in the top left corner. The manufacturing rules sets are named such as “top\_to\_bottom\_set1”. The manufacturing rules can be stored individually or together. The storing procedures are the same for both. Pressing the store button stores a manufacturing rule or a set of manufacturing rules. Then, the taxonomy computation time is measured and displayed. Once the desired manufacturing rules are stored, pressing retrieve button retrieves and ranks the rules. The text box at the bottom of the user interface displays the retrieval results in XML format. The message shows the retrieved rules with their ranks. The structure of the message can be explained as XML format of Table 7.7. For instance, the first child element (PartOrientation) of the root element (Message) in Figure 7.12 is used to rank retrieved rules based on meta rules of part orientation. Hence, the rules under this element share the same feasible value of part orientation. Then, the rules are further ranked by element “Layerthickness\_Overcure”. Using this child element, the rules are further ranked based on meta rules of layer thickness and overcure. Finally, the rules are further ranked by

child elements “HaveAllModels” and “MayHaveSomeModels”. Using these two elements, the rules that contain all the accuracy models are distinguished from the ones that do not. The rules are represented by element “catom” and its attribute “name” contains the name of the rule.

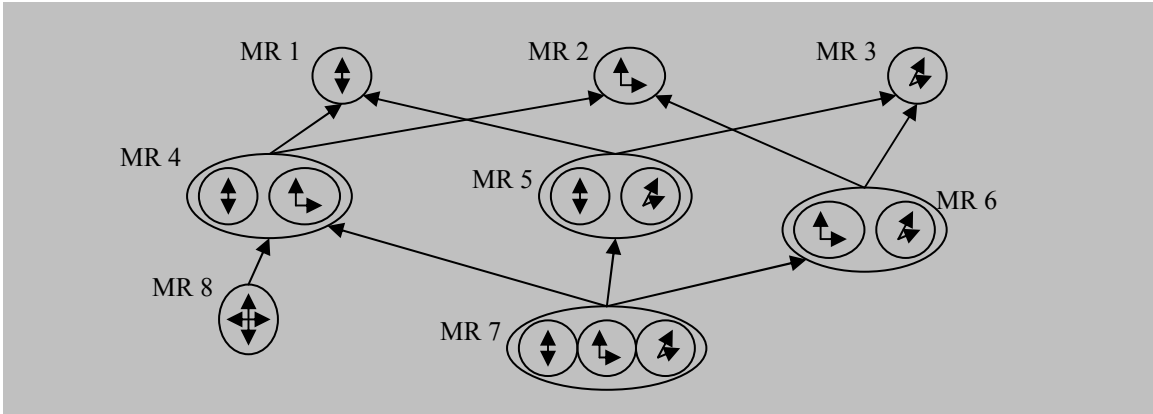
This software is implemented on a PC that has 2.67 Giga hertz of CPU speed and 1 Giga bytes of memory space. Using the developed test bed, the empirical performance is validated. The following section discusses the results.

## **7.5 Results**

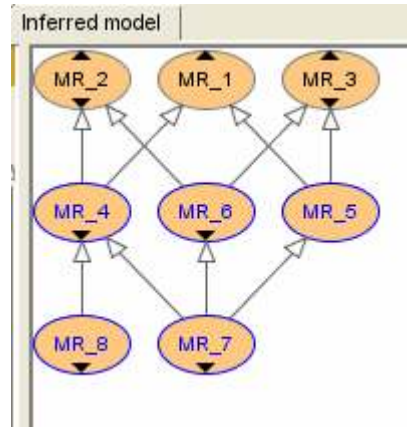
In this section, the results of empirical performance validation including description logic’s applicability, retrieval performance, and computational feasibility measure are discussed. This section reports detailed results of the experiments described in section 7.3.

### **7.5.1 Result for description logic’s applicability measure**

For the DL applicability test, the meta rules are introduced with variations of the order in which meta rules are introduced. Each time a set of meta rules is incremented, the meta rules taxonomy is computed and compared to the expected taxonomy. The test results show that the experimental results exactly match the expected results (Appendix C). Hence, this test result proves that description logic is capable of constructing and managing the manufacturing rules taxonomy such that it is consistent and correct. An example is shown in Figures 7.13 and 7.14.



**Figure 7.13 Example of expected taxonomy for set 2 of top to bottom scenario**



**Figure 7.14 Example of computed taxonomy for set 2 of top to bottom scenario**

In Figure 7.13, the expected meta rules taxonomy after storing sets 1 and 2 of the top to bottom scenario is shown. Figure 7.14 shows the corresponding computed result. Both taxonomies are the same. With the same procedure, it is confirmed that all the 16 taxonomies expected in Appendix C match the corresponding computed taxonomy. Hence, it is concluded that the subsumption algorithm in DL ( $\mathcal{AL}\mathcal{E}$ ) is able to compute the manufacturing rules taxonomy consistently and correctly.

This conclusion agrees with the expectations. As discussed in chapter 5, the design requirements are used to index the manufacturing rules. Hence, the representation

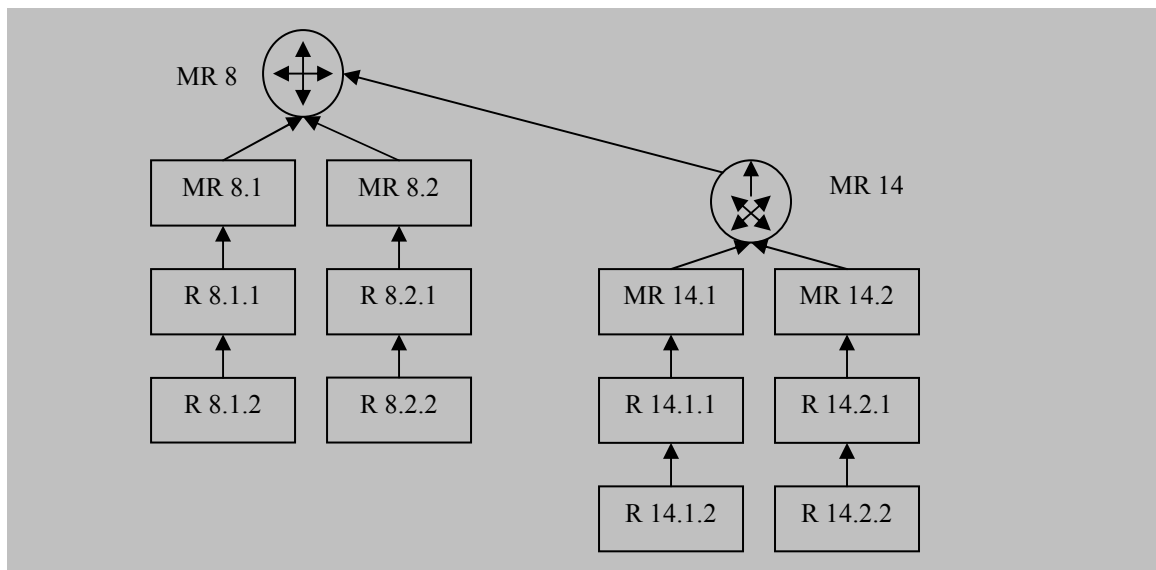
and reasoning are performed using design requirements. In chapter 5, appropriate description logic for representing design requirement in additive manufacturing is identified to be  $\mathcal{AL}\mathcal{E}$ . This is because the design requirements in additive manufacturing can be represented by addition of atomic concepts and roles. In this research, the subsuming relation between the two concepts is determined by the fact that one has more atomic concepts and roles than the other. Similarly, the difference between the two concepts is determined by the differences in the atomic concepts and roles that constitute the concepts. The manufacturing rules that are discovered in this research are represented by combining atomic concepts and roles of design requirements. Hence, their subsuming relations and differences are modeled by addition of atomic concepts and roles. A subsumption computation performed upon those information models should always be consistent and correct if the information models are correct and unique for each manufacturing rule. Hence, the consistency and correctness of the manufacturing rule taxonomy is expected and the experimental results conform to this.

### **7.5.2 Result for retrieval performance measure**

For the retrieval performance, 16 design requirements are developed such that they are used to test manufacturing rules for part orientation, layer thickness, overcure, and accuracy models. Then, the meta rules and rules are stored in the repository and expected query results are developed. The rules are retrieved for each design requirement and the experimental query results match the expected query results. The expected query results are presented in a tabular form in Appendix D. The successful completion of this test means that the algorithms and metrics perform correctly in retrieving and ranking rules. Hence, it is demonstrated that the retrieval method performs correctly in storing

and retrieving relevant DFM problems within the domain identified in Table 3.2. This is because the manufacturing rules are derived within the design requirements and process planning domains described in Table 3.2. The details of manufacturing rules derivation and justifications are presented in chapter 3 and 4. The process planning capability of retrieved manufacturing rules (DFM problems) for each design requirement is justified in chapter 4.

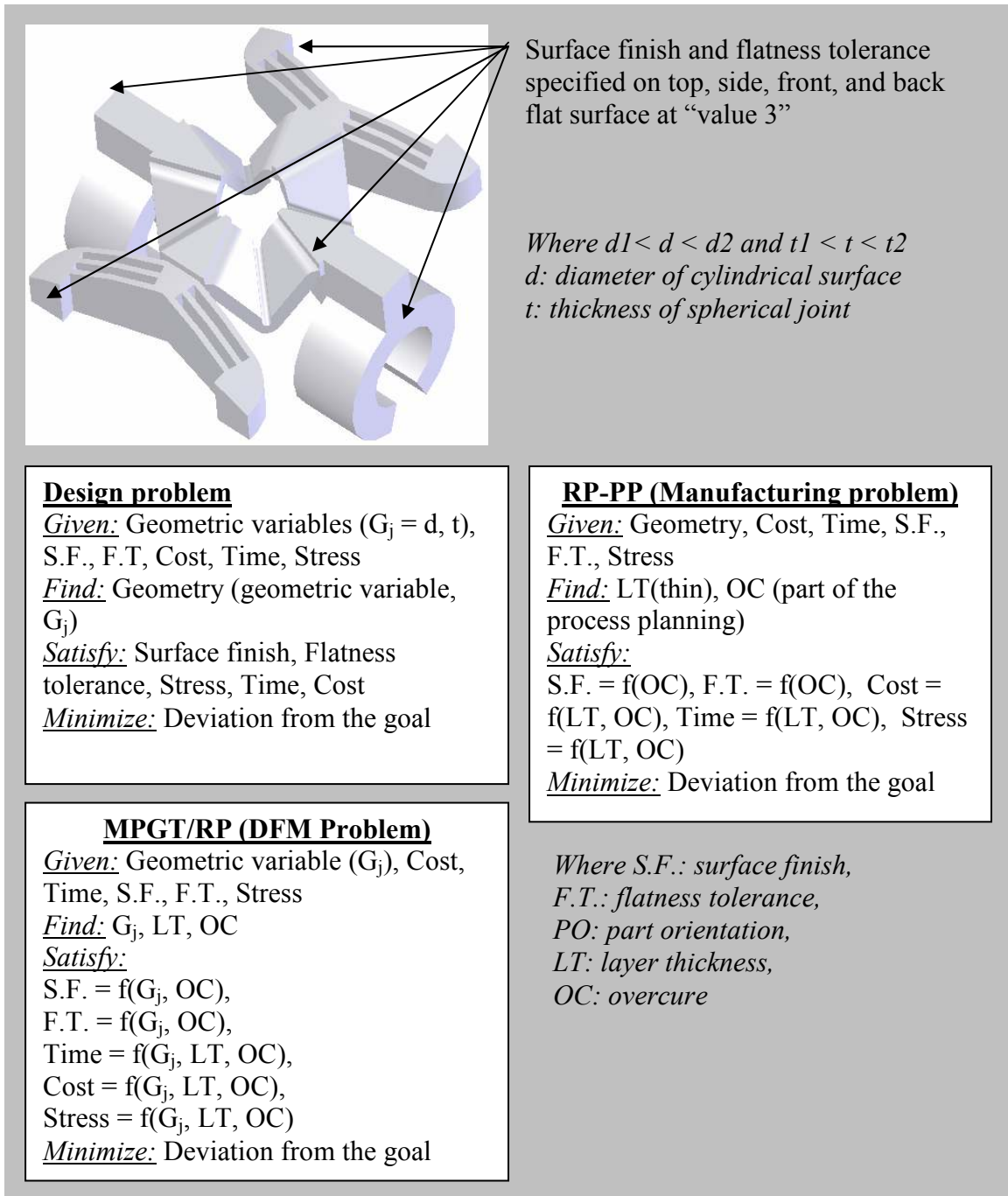
To illustrate how DFM problems are stored and retrieved and how retrieved DFM problems actually support process planning, an example is presented in Figures 7.15 and 7.16 and Table 7.11.



**Figure 7.15 Example of repository structure**

**Table 7.11 Retrieved manufacturing rules**

MR x	MR 14				MR 8			
MR x.y	MR 14.1		MR 14.2		MR 8.1		MR 8.2	
R		R14.1.1 R14.1.2		R14.2.1 R14.2.2		R8.1.1 R8.1.2		R8.2.1 R8.2.2
Rank		1		2		3		4



**Figure 7.16 Retrieved DFM problem**

In Figure 7.15, an assumed repository structure is presented. The meta rules and rules in Figure 7.15 are obtained from Figure 7.5. For this example, it is assumed that there are DFM problems under each rule in Figure 7.15. For instance, a DFM problem

14.2.1 is under rule R14.2.1. Storing the manufacturing rules and DFM problems and structuring the repository are demonstrated with the robot arm example in section 6.2.2 and DL applicability testing is demonstrated in section 7.3.1. Hence, it is omitted here. Once the DFM problems are stored and the repository is structured as shown in Figure 7.15, the query is performed.

The design requirements in Figure 7.9 are converted into a query to the repository as:

$$\begin{aligned} \text{Query} = & (\forall, \exists \text{hasOppositeOrientation.Opposite}) \sqcap \\ & (\forall, \exists \text{hasPerpendicularOrientation.Perpendicular}) \sqcap ((\forall, \exists \text{has3D.3D}) \sqcap \\ & (\forall, \exists \text{hasFlatness\_tolerance-Flat\_surface.Flatness\_tolerance-Flat\_surface}) \sqcap \\ & (\forall, \exists \text{hasPositionTolerance-Cylindrical\_surface.PositionTolerance-} \\ & \text{Cylindrical\_surface}) \sqcap (\forall, \exists \text{hasSurface\_finish-Flat\_surface.Surface\_finish-} \\ & \text{Flat\_surface}) \sqcap \\ & (\forall, \exists \text{hasSurface\_finish-Cylindrical\_surface.Surface\_finish-Cylindrical\_surface}) \\ & \sqcap (\forall, \exists \text{hasRestrictionLevel.Level3}) \end{aligned}$$

Also, the meta-rules and rules including MR8, 8.1, 8.2, 14, 14.1, 14.2, R8.1.1, 8.1.2, 8.2.1, 8.2.2, 14.1.1, 14.1.2, 14.2.1, and 14.2.2 are represented as:

$$\begin{aligned} \text{MR8} = & (\forall, \exists \text{hasOppositeOrientation.Opposite}) \sqcap \\ & (\forall, \exists \text{hasPerpendicularOrientation.Perpendicular}) \end{aligned}$$

MR8.1 = ( $\forall, \exists$ hasOppositeOrientation.Opposite)  $\sqcap$

( $\forall, \exists$ hasPerpendicularOrientation.Perpendicular)  $\sqcap$

( $\forall, \exists$ hasRestrictionLevel.Level3)

MR8.2 = ( $\forall, \exists$ hasOppositeOrientation.Opposite)  $\sqcap$

( $\forall, \exists$ hasPerpendicularOrientation.Perpendicular)  $\sqcap$

( $\forall, \exists$ hasRestrictionLevel.Level4)

R8.1.1 = ( $\forall, \exists$ hasOppositeOrientation.Opposite)  $\sqcap$

( $\forall, \exists$ hasPerpendicularOrientation.Perpendicular)  $\sqcap$

( $\forall, \exists$ hasRestrictionLevel.Level3)  $\sqcap$  ( $\forall, \exists$ hasSurface\_finish-Flat\_surface.Surface\_finish-Flat\_surface)

R8.1.2 = ( $\forall, \exists$ hasOppositeOrientation.Opposite)  $\sqcap$

( $\forall, \exists$ hasPerpendicularOrientation.Perpendicular)  $\sqcap$

( $\forall, \exists$ hasRestrictionLevel.Level3)  $\sqcap$  ( $\forall, \exists$ hasSurface\_finish-Flat\_surface.Surface\_finish-Flat\_surface)  $\sqcap$  ( $\forall, \exists$ hasFlatness\_tolerance-Flat\_surface.Flatness\_tolerance-Flat\_surface)

MR14 = ( $\forall, \exists$ hasOppositeOrientation.Opposite)  $\sqcap$

( $\forall, \exists$ hasPerpendicularOrientation.Perpendicular)  $\sqcap$  (( $\forall, \exists$ has3D.3D)

MR14.1 =  $(\forall, \exists \text{hasOppositeOrientation.Opposite}) \sqcap$

$(\forall, \exists \text{hasPerpendicularOrientation.Perpendicular}) \sqcap ((\forall, \exists \text{has3D.3D}) \sqcap$

$(\forall, \exists \text{hasRestrictionLevel.Level3})$

MR14.2 =  $(\forall, \exists \text{hasOppositeOrientation.Opposite}) \sqcap$

$(\forall, \exists \text{hasPerpendicularOrientation.Perpendicular}) \sqcap ((\forall, \exists \text{has3D.3D}) \sqcap$

$(\forall, \exists \text{hasRestrictionLevel.Level4})$

R14.1.1 =  $(\forall, \exists \text{hasOppositeOrientation.Opposite}) \sqcap$

$(\forall, \exists \text{hasPerpendicularOrientation.Perpendicular}) \sqcap ((\forall, \exists \text{has3D.3D}) \sqcap$

$(\forall, \exists \text{hasRestrictionLevel.Level3}) \sqcap (\forall, \exists \text{hasSurface\_finish-}$

$\text{Flat\_surface.Surface\_finish-Flat\_surface})$

R14.2.1 =  $(\forall, \exists \text{hasOppositeOrientation.Opposite}) \sqcap$

$(\forall, \exists \text{hasPerpendicularOrientation.Perpendicular}) \sqcap ((\forall, \exists \text{has3D.3D}) \sqcap$

$(\forall, \exists \text{hasRestrictionLevel.Level4}) \sqcap (\forall, \exists \text{hasSurface\_finish-}$

$\text{Flat\_surface.Surface\_finish-Flat\_surface}) \sqcap (\forall, \exists \text{hasFlatness\_tolerance-}$

$\text{Flat\_surface.Flatness\_tolerance-Flat\_surface})$

Using subsumption in DL, the following computation results can be shown:

1.  $MR8 \sqsupset MR14 \sqsupset \text{Query}$
2.  $MR8.1 \sqsupset MR14.1 \sqsupset \text{Query}$
3.  $MR8.2 \not\sqsupset \text{Query}$  and  $MR14.2 \not\sqsupset \text{Query}$
4.  $R8.1.1 \sqsupset R14.1.1 \sqsupset \text{Query}$
5.  $R8.1.2 \sqsupset R14.1.2 \sqsupset \text{Query}$
6.  $R8.2.1 \not\sqsupset \text{Query}$  and  $R14.2.1 \not\sqsupset \text{Query}$
7.  $R8.2.2 \not\sqsupset \text{Query}$  and  $R14.2.2 \not\sqsupset \text{Query}$

Based on item 1, the retrieving algorithm in Figure 6.5 (steps 1~3.b) and the ranking metric in Figure 6.6, rules under MR14 are ranked higher than rules under 8. This makes sense because MR14 specifies that part orientation (PO) is constant, while MR8 indicates that two part orientations are possible. The design requirements in Figure 7.9 require constant PO. This is shown in the first row of Table 7.11. Then, the rules under those meta-rules are further ranked based on 2~7 above and the ranking metric in Figure 6.7. This is shown in the second row of Table 7.11. At this stage, the rules that determine thin layer thickness are ranked higher than those that do not. Finally, the design requirements in Figure 7.9 contain position tolerance on cylindrical surfaces which none of the condition parts of the rules  $Rx.y.1$  or  $2$  contain. Hence, all the rules under the  $MRx.y$  are ranked the same. This is shown in the third row of Table 7.11. The corresponding ranks are shown in the fourth row. Therefore, the relevant rules are retrieved and ranked. These steps are demonstrated with the robot arm example in section 6.2.2 and with the retrieval performance testing in section 7.3.2.

Figure 7.16 shows a retrieved DFM problem that is assumed to be under R14.1.2. Its design requirements match the condition part of R14.1.2. Using this retrieved DFM problem, the formulation and solution generation for the design problem in Figure 7.9 are demonstrated. The following paragraphs describe the benefit of retrieving a DFM problem that supports process planning for the design requirements in Figure 7.9.

First, the method shows the design requirements that influence the process planning. For the design problem in Figure 7.9, those are the condition part of manufacturing rule MR 14.2. At this step, the appropriate MPT is identified (MPT 5.1 from Table 4.2). Then the method shows the feasible spaces of the process variables, the accuracy models, and the solution strategy required for formulating and solving a new DFM problem by retrieving an appropriate DFM problem. Figure 7.17 shows the retrieved and newly formulated manufacturing and DFM problems. The retrieved manufacturing and DFM problems in Figure 7.17 are from Figure 7.16.

### Retrieved manufacturing and DFM problem

#### **RP-PP (Manufacturing problem)**

Given: Geometry, Cost, Time, S.F., F.T., Stress  
Find: LT(thin), OC (part of the process planning)  
Satisfy:  
 S.F. =  $f(\text{OC})$ , F.T. =  $f(\text{OC})$ , Cost =  $f(\text{LT}, \text{OC})$ , Time =  $f(\text{LT}, \text{OC})$ , Stress =  $f(\text{LT}, \text{OC})$   
Minimize: Deviation from the goal

#### **MPGT/RP (DFM Problem)**

Given: Geometric variable ( $G_j$ ), Cost, Time, S.F., F.T., Stress  
Find:  $G_j$ , LT, OC  
Satisfy:  
 S.F. =  $f(G_j, \text{OC})$ ,  
 F.T. =  $f(G_j, \text{OC})$ ,  
 Time =  $f(G_j, \text{LT}, \text{OC})$ ,  
 Cost =  $f(G_j, \text{LT}, \text{OC})$ ,  
 Stress =  $f(G_j, \text{LT}, \text{OC})$   
Minimize: Deviation from the goal

### Newly formulated manufacturing and DFM problem

#### **RP-PP (Manufacturing problem)**

Given: Geometry, Cost, Time, S.F., F.T., P.T., Stress  
Find: LT(thin), OC (part of the process planning)  
Satisfy:  
 S.F. =  $f(\text{OC})$ , F.T. =  $f(\text{OC})$ , P.T. =  $f(\text{OC})$ , Cost =  $f(\text{LT}, \text{OC})$ , Time =  $f(\text{LT}, \text{OC})$ , Stress =  $f(\text{LT}, \text{OC})$   
Minimize: Deviation from the goal

#### **MPGT/RP (DFM Problem)**

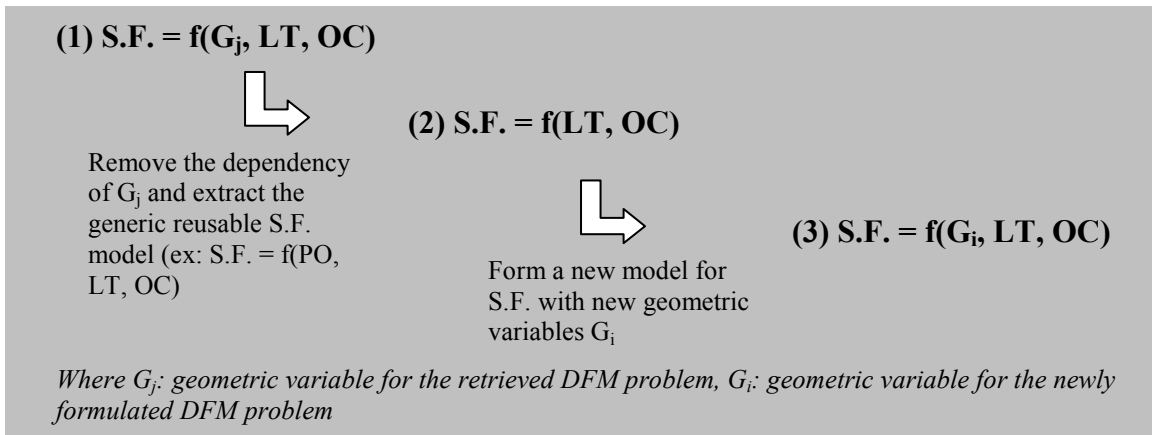
Given: Geometric variable ( $G_i$ ), Cost, Time, S.F., P.T., F.T., Stress  
Find:  $G_i$ , LT, OC  
Satisfy:  
 S.F. =  $f(G_i, \text{OC})$ , F.T. =  $f(G_i, \text{OC})$ ,  
 P.T. =  $f(G_i, \text{OC})$ ,  
 Time =  $f(G_i, \text{LT}, \text{OC})$ ,  
 Cost =  $f(G_i, \text{LT}, \text{OC})$ ,  
 Stress =  $f(G_i, \text{LT}, \text{OC})$   
Minimize: Deviation from the goal

**Figure 7.17 Retrieved and formulated DFM problem**

The retrieved manufacturing problem in Figure 7.17 shows the feasible spaces of process variables for the given design problem in Figure 7.9. The retrieved manufacturing problem in Figure 7.17 shows that the part orientation is constant, the minimum layer thickness is thin, and the overcure needs to be considered for process planning to satisfy the given accuracy requirements. Also, it shows the relation between the feasible space of process variables and accuracy models including S.F. and F.T.

Then, the manufacturing problem for the design problem in Figure 7.9 can be formulated by utilizing the retrieved manufacturing problem. First, the feasible spaces of the process variables for the newly formulated manufacturing problem are determined to be the same as those in the retrieved manufacturing problem. Also, the relation between the feasible spaces of the process variables and accuracy requirements (S.F. and F.T.) is determined from the retrieved manufacturing problem. The relation between the other design requirements (P.T., Cost, Time and Stress) and the feasible spaces of the process variables need to be determined for the new manufacturing problem. The newly formulated manufacturing problem is shown at the bottom left in Figure 7.17.

Then, the new DFM problem is formulated first by combining the design problem (Figure 7.9) and the newly formulated manufacturing problem (Figure 7.17). During the DFM problem formulation, the accuracy models are reused from the retrieved DFM problem. Figure 7.18 shows the details of the procedure for reusing accuracy models.

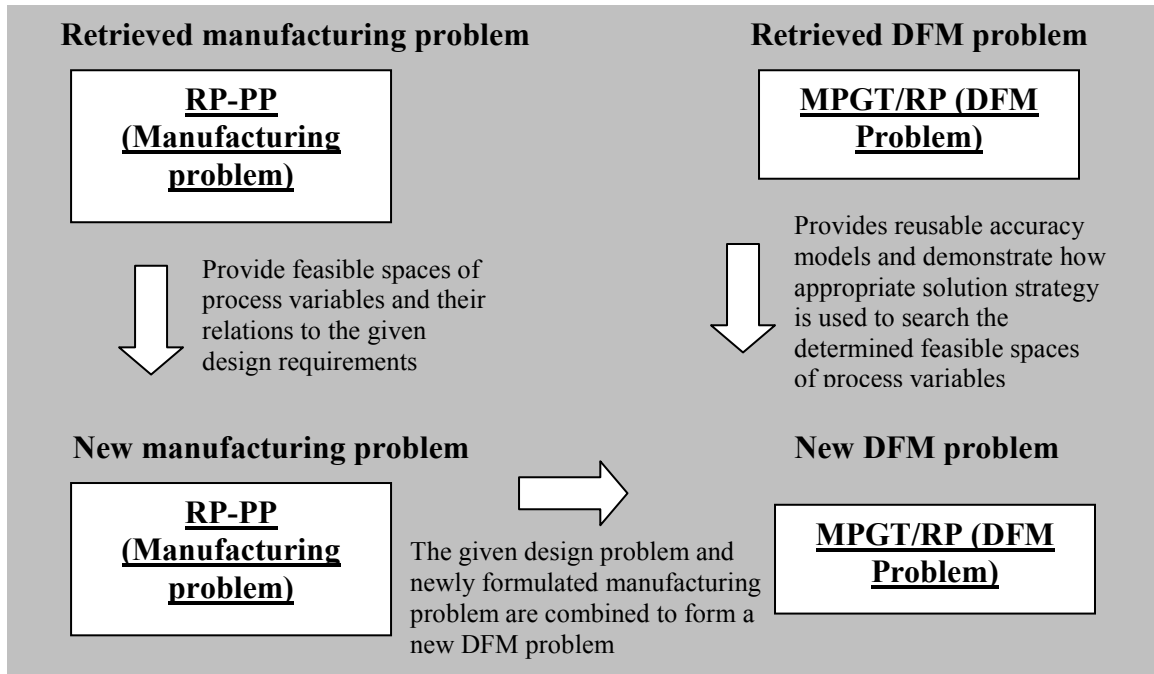


**Figure 7.18 Reusing process of accuracy model**

In Figure 7.18 the process of reusing part of the S.F. model is presented. Often, the accuracy models (1) in the design problem are used to evaluate the accuracy measurement value with varying the values of geometric variables ( $G_j$ ). Hence, the

generic and reusable accuracy models (2) need to be manually extracted from the retrieved DFM problem. The extracted generic accuracy models (2) are in the form of empirical equations shown in Table 1.5. Then, the appropriate new accuracy model (3) that accommodates the new geometric variables ( $G_i$ ) is developed. In Figure 7.17, the retrieved DFM problem supports the accuracy models for S.F. and F.T. Therefore, the accuracy model for P.T. needs to be developed from scratch.

Finally, the retrieved DFM problem further shows how the formulated DFM problem needs to be solved (decomposition or multi solution). From the determined feasible space of process variables, the appropriate solution strategy for the new DFM problem is identified to be a multi solution strategy from Figure 1.14. Then, the retrieved DFM problem shows an example of utilizing a multi solution strategy that searches the determined feasible spaces of process variables. Through solving the formulated DFM problem, the values for geometric variable ( $G_i$ ) and process variables are determined such that they satisfy the given design requirements in Figure 7.9. Figure 7.19 summarizes the formulation and solution generation that are supported by DFM problem retrieval.



**Figure 7.19 Summary of support in DFM problem formulation and solution generation**

In Figure 7.19, the retrieved manufacturing problem supports formulation of the new manufacturing problem by providing the feasible spaces of the process variables and their relation to the given design requirements. Then, the newly formulated manufacturing problem is combined with the given design problem to formulate the new DFM problem. During this procedure, the reusable accuracy models are extracted from the retrieved DFM problem to be used in a newly formulated DFM problem. The retrieved DFM problem then also demonstrates how an appropriate solution strategy is used to determine values for the geometric and process variables that will satisfy the given design requirements.

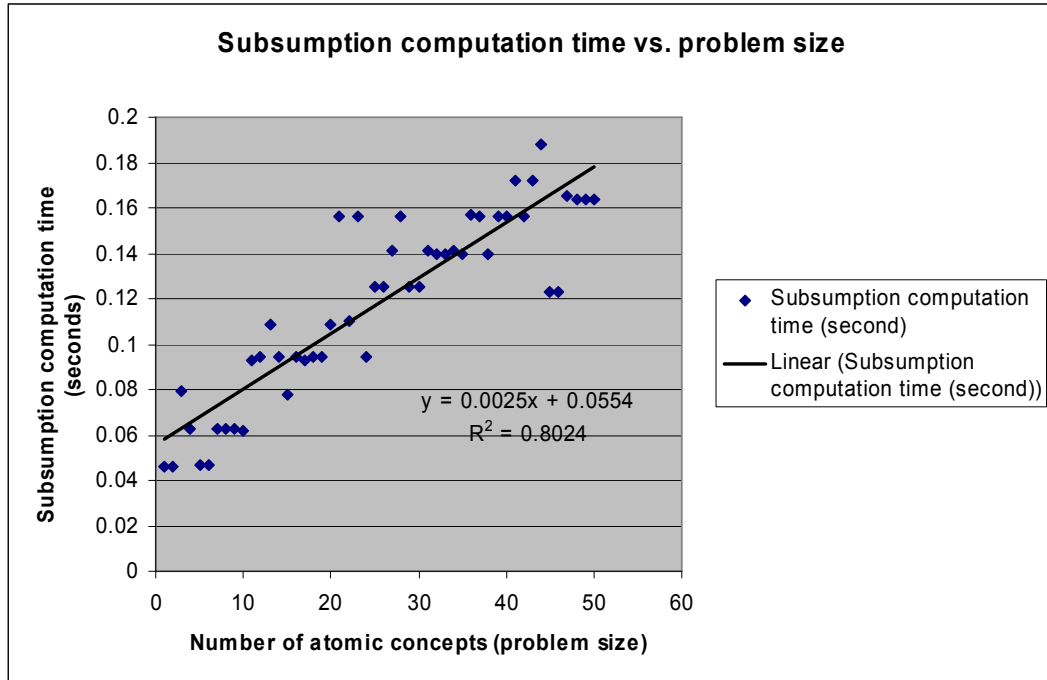
The DFM problems under MR8 in Figure 7.15 determine the feasible spaces of the process variables to be larger than the appropriate feasible spaces of the process

variables for the design problem in Figure 7.9. However, the method determines appropriate feasible spaces for process variables by identifying meta rule MR14.1 as an appropriate meta rule. Hence, the appropriate feasible spaces of process variables are informed before DFM problem retrieval. Then, the retrieved DFM problem (under MR8) should be appropriately used by understanding that the feasible spaces of process variables that are provided by the retrieved DFM problems are larger than needed. For example, the solution strategy used by the retrieved DFM problems (under MR8) searches larger spaces of process variables that include determined feasible spaces (by MR14.1).

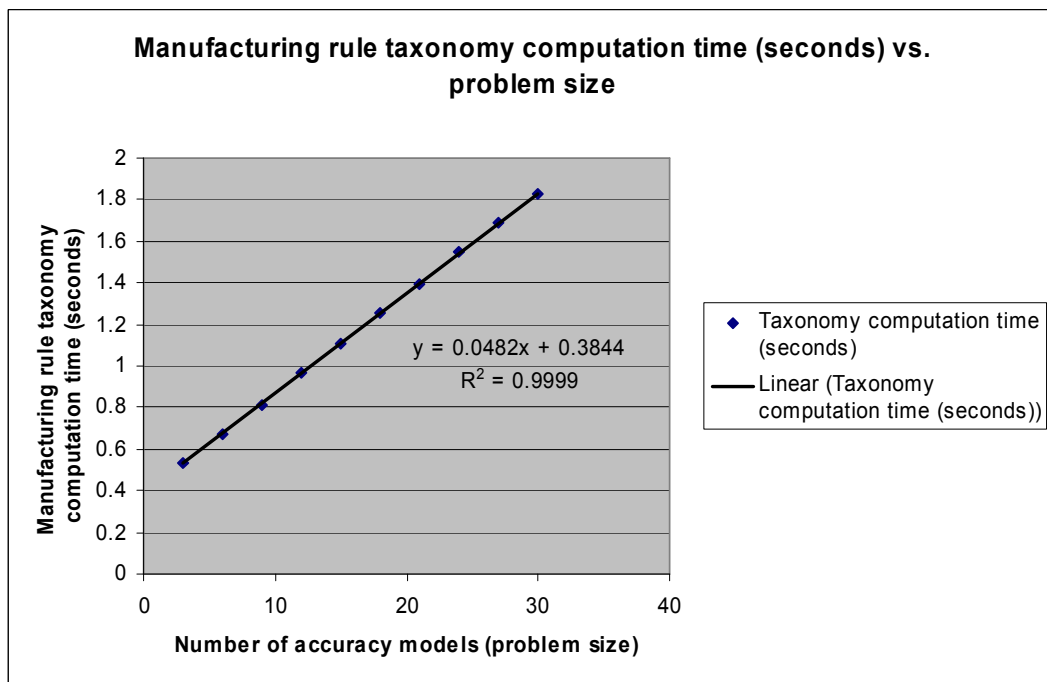
In short, the retrieved DFM problem supports process planning by providing feasible spaces of process variables and accuracy models to formulate and solve a new DFM problem. More specific details in DFM problem formulation and solution generation can be found in literature[4, 106].

### **7.5.3 Result for computational feasibility measure**

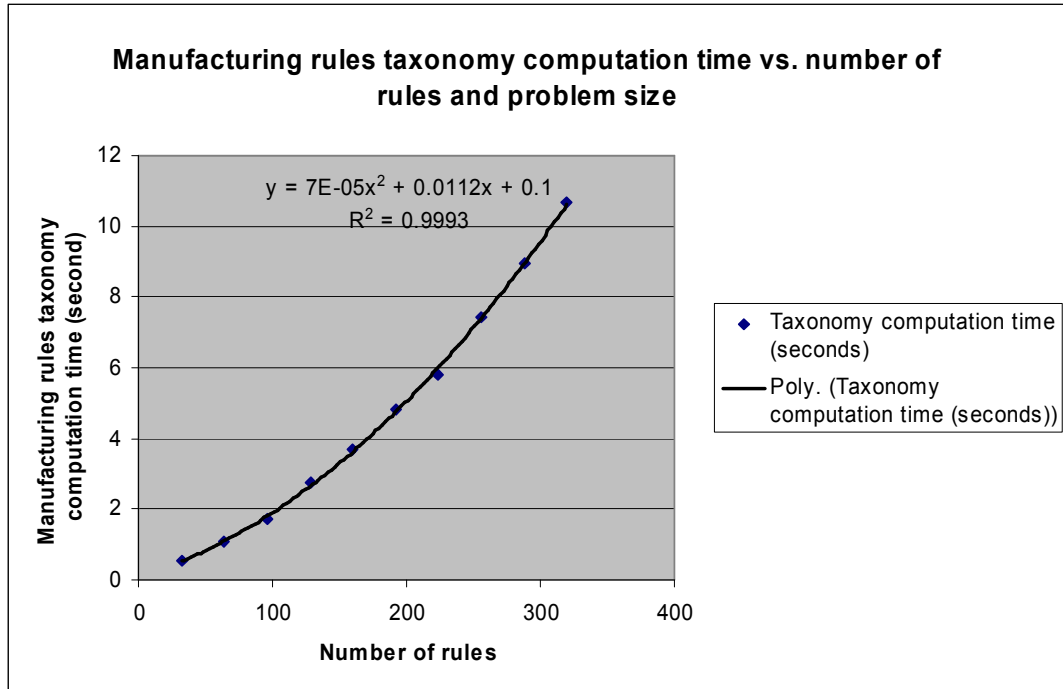
For the computational feasibility discussion, three plots are produced. The first plot shows subsumption computation time with increasing problem size. The second plot shows the manufacturing rule taxonomy computation time with increasing complexity of design requirements. The third plot shows the manufacturing rules taxonomy computation capability with increasing complexity of design requirements and number of manufacturing rules. Figures 7.20, 7.21, and 7.22 present each plot respectively. The plots are generated using Microsoft Excel software.



**Figure 7.20 Computational feasibility of subsumption using ALE**



**Figure 7.21 Manufacturing rule taxonomy computation capability with increasing complexity in design requirements**



**Figure 7.22 Manufacturing rule taxonomy computation time with increasing complexity of design requirements and number of rules**

In Figure 7.20, the subsumption computation time is demonstrated by increasing the number of atomic concepts that constitute the two design requirements that are compared. The problems size (number of atomic concepts) of design requirements is increased from 2 to 50. The plot shows that the subsumption computation time roughly increases linearly with increasing problem size. A linear line is fit with root mean square value of 0.8204. Theoretically, the subsumption performance is expected to be non-deterministic polynomial time. However, a linear behavior is observed as shown in Figure 7.20. The reasons for this are the problem size and the way design requirements are represented using  $\mathcal{AL}\mathcal{E}$ . In this research, the design requirements are expressed by addition of atomic concepts. None of the design requirements are represented by nested concepts. The nested concepts are the concepts that are defined by non-atomic concepts.

To perform subsumption computation on nested concepts, all the concepts that are used to define the nested concepts need to be normalized to atomic concepts. The intractability of subsumption is mostly caused by this normalization procedure. This normalization is heavily influenced by the complexity of the expression and the problem size. Hence, the theoretical subsumption performance (non-deterministic polynomial time) of  $\mathcal{AL}\mathcal{E}$  is determined by considering the possible expression complexity and problem size. In this research, no normalization procedure is needed because all the design requirements are represented with atomic concepts. Hence, this explains the linear behavior of the subsumption computation in Figure 7.20.

In Figure 7.21, the manufacturing rules taxonomy computation time is measured by increasing the complexity of the design requirements that represent rules. The complexity of design requirements is increased by increasing the number of accuracy models in the design requirements as discussed in Table 7.9. In Figure 7.21, the taxonomy computation time increases linearly with increasing the complexity of design requirements. A linear line is fit with root mean square value of 0.9999. This agrees with the result in Figure 7.20. In Figure 7.20, the subsumption computation time increases linearly with increasing numbers of atomic concepts. In Figure 7.21, increasing the complexity of design requirements is basically increasing the number of accuracy models which are represented by the atomic concepts. Therefore, the taxonomy computation in Figure 7.21 which involves many subsumption computations shows a linear increase in computation time with increasing complexity of design requirements.

In Figure 7.22, the taxonomy computation time is measured by increasing both the complexity of design requirements and the number of rules. The number of problems

is increased from 32 to 320. The plot shows that the taxonomy computation time increases nonlinearly with increase in design requirement complexity and number of rules. A second order polynomial line is fit with root mean square value of 0.9993. The reason for this nonlinear behavior is that the taxonomy computation time in Figure 7.22 is the accumulative taxonomy computation time (computation time for increase of problem size and number of manufacturing rules). The rule sets in Table 7.10 are developed such that the higher rule sets are created by accumulating all the lower rule sets (ex: rule set 3 contains rule sets 1 and 2). Hence, the taxonomy computation effort required to compute the taxonomy for each rule set is accumulative. Consequently, the taxonomy computation time required should be accumulative. This is exactly what is shown in Figure 7.22. The taxonomy computation performance in Figure 7.22 is the worst case performance that is expected when the complexity of design requirements and the number of manufacturing rules increase simultaneously.

As a conclusion, the method's ability to store and retrieve relevant DFM problems that support process planning is empirically demonstrated. Also, the computational feasibility investigation allows future performance prediction of DFM framework in storing and retrieving DFM problems. By demonstrating the consistency and correctness of manufacturing rule taxonomy computation, the theoretical validity of hypotheses 1, 2, and 3 in Table 1.7 is demonstrated. By demonstrating the retrieval of the relevant DFM problems for the given design requirements, the theoretical validity of hypotheses 1, 2, and 4 in Table 1.7 is demonstrated.

## 7.6 Hypotheses validation

In this chapter, empirical performance validation is discussed. For the empirical performance validation, three tests are performed: DL applicability testing, retrieval performance testing, and computational feasibility testing. Through performing the three tests, the theoretical validity of the hypotheses in Table 1.7 is demonstrated. The relation between the hypotheses and the three tests is shown in Table 7.12.

**Table 7.12 Relation between hypotheses and testing**

Hypotheses	Testing
H1. Among the various expressive description logics, there is one description logic that provides minimum expressivity for representing design requirements	DL applicability and retrieval performance testing
H2. Subsumption in DL enables mathematical mapping between design domain and process planning domain	
H3. Subsumption in DL enables systematic and consistent structuring of the repository	DL applicability
H4. Subsumption in DL and the ranking metric enables retrieval and ranking of the relevant DFM problems	Retrieval performance

As discussed previously, the empirical testing implicitly demonstrates the validity of hypotheses 1 and 2.

Hypothesis 3 is about computing a consistent and correct manufacturing rules taxonomy in the repository. One of the reasons for using a formalism for computationally mapping the design and process planning domain is to ensure consistent and correct mapping. In this research, the manufacturing rules taxonomy is the entity that maps the two domains. To ensure such consistent and correct mapping, the manufacturing rules taxonomy must be computed consistently and correctly. Hence, the DL applicability testing demonstrates the theoretical validity of hypotheses 1, 2, and 3. The results in Appendix C show that subsumption in DL computes consistent and correct manufacturing rules taxonomy under a dynamic condition (expansion of manufacturing

rules taxonomy). Therefore, the validity of hypotheses 1, 2, and 3 is successfully demonstrated.

Hypothesis 4 is about achieving desirable retrieval performance using subsumption in DL. As discussed in chapter 4, the manufacturing rules taxonomy can be used to determine the relevant DFM problems for the given design requirements. In this research, the repository is structured by the manufacturing rules taxonomy and it is found that subsumption in DL can consistently and correctly compute the manufacturing rules taxonomy. Then, it should be possible to automate the relevant DFM problem retrieval using subsumption in DL. Therefore, the retrieval performance testing demonstrates the validity of hypotheses 1, 2, and 4. The results that match Appendix D show that the algorithm developed using the subsumption and ranking metric correctly retrieves and ranks the relevant DFM problems. Therefore, validity of the hypotheses 1, 2, and 4 is demonstrated.

In short, the validity of the hypotheses is demonstrated through successful testing of the retrieval method. By successfully completing the empirical validation, the usefulness of the retrieval method in supporting process planning during geometric tailoring is demonstrated. Also, the demonstration of consistent and correct manufacturing rules taxonomy computation enables the user to share and expand the repository in a systematic manner. Hence, this completes the empirical performance validation discussed in section 1.3.3.

## **7.7 Summary**

By demonstrating the performance of the retrieval method in retrieving the relevant DFM problems, we have satisfied our initial objective. The initial objectives of

this research are to enable retrieval of the relevant DFM problems to support process planning and to share DFM problems in a distributed environment. Table 7.13 summarizes the initial research objectives and corresponding accomplishments that satisfied the objectives.

**Table 7.13 The research objectives and the corresponding accomplishment**

Objective	Satisfied by
<ul style="list-style-type: none"> <li>• Retrieval of relevant DFM problem to support process planning               <ul style="list-style-type: none"> <li>○ Provide feasible spaces of process variables</li> <li>○ Provide accuracy models</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>• Establishment of mathematical relation between the design domain and process planning domain               <ul style="list-style-type: none"> <li>○ Design requirements can be related to the relevant DFM problems through subsumption relations in the manufacturing rules taxonomy</li> <li>○ Enabled relating the design requirements to the relevant DFM problems through mathematical mapping between the two domains</li> </ul> </li> </ul>
<ul style="list-style-type: none"> <li>• Sharing DFM problems in a distributed environment               <ul style="list-style-type: none"> <li>○ Systematic expansions and modification of repository structure (manufacturing rules taxonomy)</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>• Identification and justification of the DL applicability               <ul style="list-style-type: none"> <li>○ Enable the computation of manufacturing rules taxonomy that is used to structure the repository</li> <li>○ Realized computational mapping between the design and process planning domains</li> <li>○ Provide means to systematically expand and modify the repository structure</li> <li>○ Enabled sharing the DFM problems in a distributed environment in a consistent and correct manner</li> </ul> </li> </ul>

In Table 7.13, the first research objective is satisfied by realizing the mathematical mapping between the design domain and the process planning domain. Through this formalized mapping, systematic identification of the relevant DFM problems for the given design problems is realized. The second research objective is satisfied by proving appropriate selection of description logic ( $\mathcal{AL}\mathcal{E}$ ) for representing

and computing manufacturing rules taxonomy. Through support of description logic's consistent and correct performance of manufacturing rules taxonomy computation, the DFM problem can be shared in a distributed environment in a consistent and systematic manner. Therefore, this research contributes toward enabling the support of process planning during the geometric tailoring through sharing the DFM problems in a distributed environment.

## **CHAPTER 8: ACHIEVEMENT AND RECOMMENDATIONS**

This chapter summarizes the hypotheses validation, research contributions, theoretical performance validations, and future work. First, summary of the hypotheses are presented (section 8.1). Second, the specific contributions in DFM and engineering information managements are discussed (section 8.2). Third, theoretical performance validation is discussed (section 8.3). Finally, the future works are discussed based on the identified limitations of the current research (section 8.4).

### **8.1 Summary of hypotheses validation**

The objective of this research is to support process planning in geometric tailoring by case-based retrieval of previously formulated and solved DFM problems. To automate DFM problem retrieval, a formalism is needed and the description logic is selected as an appropriate formalism. During the realization of automated retrieval method using description logic, several research issues are identified. Those research issues are identified such that addressing those issues ensures the applicability of the retrieval method in design for additive manufacturing. For each research issue, the corresponding research question and hypothesis are established as shown in Table 1.7 and repeated in Table 8.1.

**Table 8.1 Research questions and hypotheses**

Research Questions		Hypotheses
Representation	Q1. How should design requirements be represented?	H1. Among the various expressive description logics, there is one description logic that provides minimum expressivity for representing design requirements
	Q2. How should the design and process planning domains be mapped?	H2. Subsumption in DL enables mathematical mapping between design domain and process planning domain
Retrieval	Q3. How should DFM problems be stored?	H3. Subsumption in DL enables systematic and consistent structuring of the repository
	Q4. How should DFM problems be retrieved and ranked?	H4. Subsumption in DL and the ranking metric enables retrieval and ranking of the relevant DFM problems

In Table 8.1, the research questions and hypotheses are divided into two groups; representation and retrieval. The research questions and hypotheses are established such that validating hypotheses in the representation group provides theoretical proof and foundations for description logic applicability in design for additive manufacturing. Validating hypotheses in the retrieval group proves the description logic usage of automating the retrieval procedure. The following paragraphs briefly discuss the validation of each hypothesis.

### **8.1.1 Question 1 and Hypothesis 1**

Validating hypothesis 1 (H1 in Table 8.1) requires investigation, selection, and justification of an appropriate description logic for representing the design requirements. To validate hypothesis H1, the information models for design requirements are developed and analyzed to determine the minimum expressivity for representing the design requirements. To achieve this, the information space for the design and process planning are identified and justified first. Then, the manufacturing rules are collected within the identified information space to discover design requirements. Finally, the discovered

design requirements are carefully analyzed to realize information models that are used to determine the minimum expressivity. From such analysis and development of information models for the design requirements, description logic  $\mathcal{AL}\mathcal{E}$  is identified to provide minimum expressivity.

### **8.1.2 Question 2 and Hypothesis 2**

Validating hypothesis 2 (H2 in Table 8.1) requires investigation and proof of the mathematical relation between the design and the process planning domains. To map the two domains correctly, accurate relation between the two domains needs to be identified, and justified. In this research, the manufacturing rules and their subsumption relations are used to map the two domains. Through the investigation, the mathematical relation (subsumption relation) among the manufacturing rules are identified and justified. Then, the manufacturing rules taxonomy is computed using the subsumption in DL so that the subsumption in DL can be used to map the design and process planning domains. Hence, hypothesis 2 is validated.

### **8.1.3 Question 3 and Hypothesis 3**

Validating hypothesis 3 (H3 in Table 8.1) requires development and justification of a storing algorithm that uses subsumption to store DFM problems such that the stored DFM problems enable proper retrieval of relevant DFM problems. An algorithm that uses subsumption to structure the repository by manufacturing rules taxonomy is developed. The storing algorithm is developed such that the algorithm performs correctly as far as manufacturing rules taxonomy is computed correctly. In other words, the successful performance of the storing algorithm is dependent on the availability of

subsumption algorithm because subsumption in DL computes taxonomy. The availability of a subsumption algorithm is determined by validating hypothesis 1. Therefore, hypothesis 3 is validated by validating hypotheses 1.

#### **8.1.4 Question 4 and Hypothesis 4**

Validating hypothesis 4 (H4 in Table 8.1) requires development and justification of the retrieval algorithm that uses subsumption. An algorithm that uses subsumption and ranking metric for identifying and ranking the relevant DFM problems is developed. More specifically, the retrieval algorithm is developed such that the subsumption algorithm is used to identify relevant DFM problems and the metrics are used to rank the identified problems. Hence, successful performance of the retrieval algorithm is dependent on the availability of the subsumption and correctness of the metric. The metric is theoretically justified and validated. The availability of subsumption is determined by validating hypothesis 1. Therefore, hypothesis 4 is validated by validating hypotheses 1.

So far the theoretical validity of the hypotheses in Table 8.1 is summarized. The theoretical validity of those hypotheses is demonstrated by empirical performance validation of the retrieval method in chapter 7. Based on the validity of the hypotheses, the theoretical performance validation is discussed in the next section.

### **8.2 Theoretical performance validation**

Through the theoretical performance validation, we claim the method's applicability in design for manufacturing. The retrieval method in this research is developed and is validated in the domain of design for additive manufacturing. To extend the applicability of the retrieval method to design for manufacturing domain,

several conditions that need to be satisfied by the new domain are identified. Table 8.2 shows the conditions.

**Table 8.2 Conditions that need to be satisfied for the retrieval method to perform correctly**

Conditions	
1.	Empirical models for accuracy measurements or any design requirements can be developed in terms of process variables
2.	The empirical models in 1 can be represented in a tabular form such that the tabular form can be represented in the form of rules (manufacturing rules)
3.	There are process variables that the corresponding manufacturing rules show subsumption relations in their condition and result part (in this research, it is part orientation)
4.	The manufacturing rules and their subsumption relations contains the characteristics in Figure 4.17
5.	Subsumption enables mapping design and process planning domains due to 4 above
6.	There needs to be expressive enough description logic for representing the design requirements

Briefly, the conditions shown in Table 8.2 are the necessary conditions that need to be satisfied for validating hypotheses 1 and 2 in this research. Conditions 1~5 need to be satisfied in order to relate design and process planning through subsumption. Condition 6 needs to be satisfied in order to represent design requirements by DL and utilize subsumption in DL. Hence, satisfying condition 6 validates hypothesis 1 and satisfying conditions 1~5 validates hypothesis 2. As discussed previously, satisfying hypotheses 1 and 2 basically ensures the theoretical validity of the retrieval method. Therefore, satisfying the conditions in Table 8.2 ensures the applicability of the retrieval method.

In general, the conditions in Table 8.2 are believed to be true in design for manufacturing. This is because complex design requirements typically reduce the

feasible spaces of process variables in manufacturing. Also, the increase of the complexity of design requirements and the decrease of feasible spaces of process variables can be mathematically represented as a subsumption relation. Therefore, we believe that the conditions in Table 8.2 are generally true in design for manufacturing. Consequently, we claim that the retrieval method is applicable in the design for manufacturing domain with moderate modification. The following section discusses the research contributions.

### **8.3 Research contributions**

This research contributes in two domains: The design for manufacture and engineering information management. The following paragraphs describe each contribution in detail.

#### **8.3.1 Design for manufacture (DFM)**

In DFM, the gap identified in chapter 2 is the formal approach in representing and relating the design information to DFM knowledge such as the rules and algorithms that evaluate the design. One possible explanation for this gap is that the focus of the research in DFM has been in evaluation of the design to determine technical and economic feasibility. Therefore, the DFM knowledge discovered so far is mostly localized and difficult to share or reuse.

In this research, previously formulated and solved DFM problems are identified to be useful in support of formulating and solving a new DFM problem (process planning) during geometric tailoring (DFM method). Hence, the DFM knowledge to be shared is a set of previously formulated and solved DFM problems. To share the DFM problems, a retrieval method called the DFM framework is developed. Using the method, previously

formulated and solved DFM problems are stored and retrieved using an open repository in a distributed environment. The major challenge in developing this retrieval method is consistently and correctly mapping the design and process planning domain.

While addressing the challenge, research issues are identified and listed as research questions and hypotheses in Table 8.1. By validating those hypotheses, the research issues are addressed. During the validation of the hypotheses, the retrieval method is also realized. In DFM, the specific contributions are listed below.

1. A retrieval method that consists of:
  - a. Information model for representing the design requirements
  - b. Manufacturing rules taxonomy
  - c. Storing and retrieval algorithms
2. Identified information space for design requirements and process planning domain
3. Manufacturing rules and their mathematical properties that are used to map design and manufacturing domain (mathematical relation between design and process planning domains)

Listed item 1 is the retrieval method and its components. Listed item 2 is the identification of two domains that the retrieval method relates. Finally, item 3 is the identification of mathematical relation between two engineering domains (design and process planning). In short, items 2 and 3 provide theoretical foundation for realizing the retrieval method in item 1.

### **8.3.2 Engineering information management**

In engineering information management, the gap that is identified in chapter 2 is the lack of critical analysis of description logic's applicability in engineering domain. As discussed in chapter 2, a semantic web application in the engineering domain for building various types of design or knowledge repositories is gaining popularity. However, the studies reported in chapter 2 failed to provide critical analysis of description logic's applicability. More specifically, the analysis that identifies required expressivity and corresponding computational complexity of inference services is not presented.

As discussed in chapter 5, there are several description logics with varying expressivity. The majority of the description logics are known to be theoretically intractable in inference services. The computational complexities increase with increasing expressivity in description logics. Hence, the applicability study of description logics identifies the description logic that best supports minimum expressivity and determines the corresponding computational complexity in inference services. Without this study, the performances of systems or methods that rely on inference services of description logics are largely unknown due to computational complexities. Therefore, an applicability study of description logic is crucial to ensure the performance of systems that rely on inference services of description logics.

In this research, DFM problems are classified by the manufacturing rules and manufacturing rules are indexed by design requirements. Therefore, design requirements are represented and compared to determine subsumption relations among manufacturing rules. To select an appropriate description logic, the minimum expressivity for representing the design requirements needs to be determined. The minimum expressivity

is determined by developing information models for the design requirements and identifying the characteristics of those information models.

During this research, manufacturing rules that cover the scope of the project are discovered as discussed in chapters 3 and 4. Through discovering these manufacturing rules, the information models for design requirements are developed. The characteristics of the information models are identified such that all design requirements are formed by combinations of concepts. More complex design requirements can be formed by addition of more concepts. Therefore, appropriate description logics for representing design requirements are identified to be  $\mathcal{AL}\mathcal{E}$ . The corresponding theoretical computational complexity for inference services including satisfiability and subsumption is non-deterministic polynomial time.

To determine the empirical computational feasibility, three measurements are taken. The three measurements are listed below:

1. Subsumption computation time measure between two named concepts with increasing problem size (number of atomic concepts that are used to define the named concepts, 2~50)
2. Taxonomy computation time measure with increasing problem size
3. Taxonomy computation time measure with increasing problem size and number of manufacturing rules

From measure 1 above, it is found that the subsumption computation time increases linearly with increasing problem size. Hence, the taxonomy computation (using subsumption) time with increasing problem size is expected to be linear and the result from measurement 2 confirmed this. The worst case in computation complexity in

this research is expected when problem size and number of manufacturing rules increase simultaneously. Hence, measurement 3 above is collected and the results show that the manufacturing rules taxonomy computation time increases as second order polynomial.

Table 8.3 summarizes the contributions in each domain including DFM and engineering information management.

**Table 8.3 Specific contributions of the research in DFM and engineering information management**

DFM	Engineering Information Management
<ul style="list-style-type: none"> <li>• Retrieval method that relate design requirements to relevant DFM problems               <ul style="list-style-type: none"> <li>○ Information models for design requirements</li> <li>○ Meta rule taxonomy</li> <li>○ Storing and retrieving algorithms</li> </ul> </li> <li>• Identification of information space of design requirements and process planning domain</li> <li>• Manufacturing rules and their mathematical properties that are used to map design and manufacturing domain (mathematical relation between the design and process planning domains)</li> </ul>	<ul style="list-style-type: none"> <li>• Identification and justification of description logic (<math>\mathcal{AL}\mathcal{E}</math>) for representing and computing manufacturing rules taxonomy               <ul style="list-style-type: none"> <li>○ Information models for the design requirements</li> <li>○ Identification and justification of minimum expressivity required for representing design requirement information models</li> <li>○ Identification and justification of appropriate description logic (<math>\mathcal{AL}\mathcal{E}</math>)</li> </ul> </li> <li>• Identification of the corresponding computational complexity for subsumption (non deterministic polynomial time)</li> <li>• Prediction of subsumption performance with increasing the problem size and the number of manufacturing rules</li> </ul>

## 8.4 Limitations and future works

### 8.4.1 Limitations

In this section, the limitations of the method are discussed. The retrieval method retrieves relevant DFM problems that support process planning for the given design

requirements. As discussed in chapters 3 and 4, the input to the method is the set of design requirements, which consist of the information shown in Figures 3.1 and 3.2. The design requirements are expected to be manually extracted from the design problem. Then, the retrieval method retrieves DFM problems such that the retrieved DFM problems support process planning of the extracted design requirements. The retrieval method is limited by the design requirements, the manufacturing processes, and the DFM knowledge that are shared. Table 8.4 summarizes the limitations of the retrieval method.

**Table 8.4 Limitations**

Limitations
<ul style="list-style-type: none"> <li>• Design requirements               <ul style="list-style-type: none"> <li>○ Accuracy measurements (surface finish, flatness tolerance, etc)</li> <li>○ Values specified for accuracy measurements</li> <li>○ Surface types</li> <li>○ Surface orientations including opposite, perpendicular, angled, etc</li> </ul> </li> <li>• Layer-based additive manufacturing processes               <ul style="list-style-type: none"> <li>○ Stereolithography</li> <li>○ Selective laser sintering</li> <li>○ Fused deposition modeling</li> <li>○ Etc</li> </ul> </li> <li>• DFM knowledge               <ul style="list-style-type: none"> <li>○ DFM problem formulation</li> </ul> </li> </ul>

In Table 8.4, the design requirements are limited to accuracy measurements, values, surface types, and orientations of surfaces. The manufacturing processes are limited to layer-based additive manufacturing that includes Stereolithography, Selective Laser Sintering, Fused Deposition Modeling, etc. The DFM knowledge that is shared is

DFM problem formulation. The retrieval method in this research is developed and tested within the limitations in Table 8.4. Therefore, the performance of the method is not guaranteed outside of the limitation in Table 8.4. The following section discusses the corresponding future work.

## **8.4.2 Future work**

In this section, future steps for extending the retrieval method are discussed. First, the research issues and systematic guidelines for extending the retrieval method are discussed. Then, the specific research opportunities in the domains including design, manufacturing, and other DFM knowledge are discussed.

### **8.4.2.1 Research issues and systematic guidelines for extending the retrieval method**

Table 8.5 shows the list of research issues for extending the retrieval method.

**Table 8.5 Research issues for extending the retrieval method in other domains of design and manufacturing**

Research issues
1. Collection of the manufacturing rules
2. Identification and justification of the information models for design requirements and process plans that are derived from the collected manufacturing rules
3. Investigation to determine if there are subsumption relations among the design requirements and process plans
4. Investigation to determine if there are structure preserving properties (Figure 4.16) among the collected manufacturing rules
5. Analysis of the design requirements to determine if there is a description logic that provides minimum expressivity
6. Modification of the metrics according to the manufacturing processes
a. The metrics in this research are developed to support process planning for additive manufacturing (SLA, SLS, FDM, 3D Printing, etc)
b. The metrics are developed to rank DFM problems based on the subsumption relations among the MPTs (result part of the manufacturing rules)
c. Hence, the existence of subsumption relations among the result parts of the newly collected manufacturing rules enable generalization of the developed metrics

Table 8.5 shows the research issues that arise when the method is expanded into other domains of manufacturing or other design phases such as conceptual design. The first task in expanding the retrieval method is collecting the manufacturing rules. Through accomplishing this task, the information models for design requirements and process plans are discovered. Then, the discovered design requirements and process plans are analyzed to check for the subsumption relations among the design requirements and among the process plans. If there are subsumption relations among the condition and result parts of the collected manufacturing rules, the manufacturing rules are further

analyzed to check for the structure preserving properties. Finally, the metric may need to be modified for the new manufacturing rules.

- **Simplified and systematic approach in testing the applicability of DFM framework in other design and manufacturing domains**

The listed items and their orders in Table 8.5 serve as the systematic guidelines for extending the method into other domains of design and manufacturing. For the retrieval method to be extendable to other domains of design and manufacturing, items 1~4 in Table 8.5 should be satisfied. The items 1~4 are basically the domain properties that are tested to conclude if the retrieval method can be extended into the new domain. If items 1~4 cannot be satisfied in a new domain, one can definitely conclude that the retrieval method (DFM framework) cannot be extended in the new domain.

Item 5 determines the appropriate description logic for representing the design requirements. If item 5 cannot be satisfied, a different formalism or mathematical theory need to be investigated for representing design requirements. Then, the corresponding computational complexities also need to be investigated. Item 6 most likely needs to be redeveloped. If ranking is desired by the subsumption relations among the design requirements and MPTs, the existing metric is reusable with moderate modification. However if the ranking is performed not by the subsumption relations, the metric need to be developed accordingly.

- **Developing a retrieval method from scratch**

In case items 1~5 are not satisfiable in extending the retrieval method, it can be concluded that the retrieval method (DFM framework) cannot be extended in the new domain. In such case, the recommended systematic procedure is described in Table 8.6.

**Table 8.6 Systematic procedure of extending the retrieval method in a domain  
where subsumption relation is not supported**

Steps for constructing a retrieval method from scratch
1. Identify the domains that need to be mapped (design and manufacturing domains)
2. Identify the contents (detailed information models) of the information that need to be mapped
3. Identify consistent relation between them <ul style="list-style-type: none"> <li>a. Preferably mathematical relation</li> </ul>
4. Identify any mathematical theory or a formalism that allows represent and computationally map the two domains
5. Analyze the computational feasibility if computational complexities is expected to be an issue
6. Investigate how such mapping can be expanded into other domains (design and manufacturing)

In Table 8.6, the systematic procedure for developing the retrieval method in the new design and manufacturing domains where DFM framework cannot be extended is presented. First, the scope of the design and manufacturing domains needs to be identified (item 1). At this stage, the scope can be identified approximately based on the approximate relation between design and manufacturing. Second, the detailed information models in design and manufacturing need to be identified (item 2). At this stage, the detailed information models that will bridge the gap between design and manufacturing are identified. Third, a consistent and correct relation between the two domains is identified (item 3). At this stage, the scope in item 1 is clarified and a consistent and correct relation between the two domains is identified and realized. Fourth, appropriate mathematical theories or formalisms are identified to computationally map the two domains (item 4). At this stage, metrics are desired to systematically sort and select an appropriate formalism. Finally, the computation feasibility and future work

need to be identified. If there is concern regarding computational complexities in the formalism that is used, proper investigation is desired to ensure computational feasibility.

The following sections discuss the specific research opportunities for extending the retrieval method in design, manufacturing, and other DFM knowledge domains.

#### 8.4.2.2 Extending the design requirements

In this research, only the accuracy requirements are considered to influence the process planning. This is because the other requirements such as cost and time influence the process planning in a subjective way. In other words, the decision on process variables can vary with the same cost or time depending on other conditions such as geometry, size, height, allowable margin in cost or time, etc. Therefore, it is not a trivial task to model such design requirements so that they can be related to relevant DFM problems. However, mapping such design requirements to relevant DFM problems is an invaluable capability. Consequently, there is a huge research opportunity in modeling and relating such design requirements to relevant DFM problems.

#### 8.4.2.3 Extending the retrieval method for other manufacturing processes

In this research, only the layer-based additive manufacturing process is considered. However, the retrieval method should be extendable to other manufacturing processes including injection molding, milling, grinding, casting, etc. Table 8.6 shows the summary of other manufacturing processes[3].

**Table 8.7 Manufacturing processes**

Manufacturing processes
<ul style="list-style-type: none"><li>• Metal casting</li><li>• Forming and shaping<ul style="list-style-type: none"><li>○ Rolling, forging, extrusion, drawing, powder metallurgy, etc</li></ul></li><li>• Material removal<ul style="list-style-type: none"><li>○ Cutting, boring, drilling, milling, grinding, etc</li></ul></li><li>• Joining<ul style="list-style-type: none"><li>○ Welding, brazing, soldering, adhesive bonding, etc</li></ul></li></ul>

The retrieval method (DFM framework) can be systematically extended to the other manufacturing processes in Table 8.6 by following the guidelines in Table 8.5.

#### 8.4.2.4 Extending the retrieval method for other DFM knowledge

In this research, the retrieved DFM knowledge is DFM problem formulation. However, problem formulation is not the only knowledge that can be shared. Other DFM knowledge includes algorithms, rules, technical data, process planning guide lines, etc. To extend the retrieval method to retrieve other such DFM knowledge, information models for design requirements and manufacturing rules needs to be extended appropriately.

## APPENDICES

### Appendix A: Atomic concepts, roles, manufacturing rules, and Qs encoding in ALE

#### Atomic concepts:

Relative\_Orientation

Single\_Orientation  $\sqsubset$  Relative\_Orientation, Angled\_Orientation  $\sqsubset$  Relative\_Orientation,

Opposite\_Orientation  $\sqsubset$  Relative\_Orientation, Perpendicular\_Orientation  $\sqsubset$

Relative\_Orientation

Surface

Flat\_Surface  $\sqsubset$  Surface, Cylindrical\_Surface  $\sqsubset$  Surface, Conical\_Surface  $\sqsubset$  Surface

MPT

MPT\_Infinite  $\sqsubset$  MPT, MPT\_4  $\sqsubset$  MPT\_Infinite, MPT\_3  $\sqsubset$  MPT\_4, MPT\_2  $\sqsubset$  MPT\_3,

MPT\_1  $\sqsubset$  MPT\_2

Engineering\_Measurement

SurfaceFinish-Conical  $\sqsubset$  Engineering\_Measurement, SurfaceFinish-Cylindrical  $\sqsubset$

Engineering\_Measurement, SurfaceFinish-Flat  $\sqsubset$  Engineering\_Measurement, Tolerance  $\sqsubset$

Engineering\_Measurement

Tolerance

Circularity-Cylindrical  $\square$  Tolerance, Cylindricity-Cylindrical  $\square$  Tolerance, Flatness-Flat  $\square$  Tolerance, Parallelism-Cylindrical  $\square$  Tolerance, Parallelism-Flat  $\square$  Tolerance, Perpendicularity-Cylindrical  $\square$  Tolerance, Perpendicularity-Flat  $\square$  Tolerance, Position-Cylindrical  $\square$  Tolerance, Position-Flat  $\square$  Tolerance

OCLT (Represents the values in Table 4.1, ex: OCLT\_5 is equivalent to value 1 in Table 4.1)

OCLT\_5  $\square$  OCLT, OCLT\_6  $\square$  OCLT, OCLT\_7  $\square$  OCLT, OCLT\_8  $\square$  OCLT

**Roles:**

have\_Two\_Opposite, have\_Opposite\_Orientation, have\_Perpendicular\_Orientation, have\_Single\_Orientation, have\_Angled\_Orientation, have\_Circularity-Cylindrical, have\_Cylindricity-Cylindrical, have\_Flatness-Flat, have\_Parallelism-Cylindrical, have\_Parallelism-Flat, have\_Perpendicularity-Cylindrical, have\_Perpendicularity-Flat, have\_Position-Cylindrical, have\_Position-Flat, have\_OCLT\_5, have\_OCLT\_6, have\_OCLT\_7, have\_OCLT\_8, determineMPT, is\_3D, have\_Two\_Opposite

**Meta rules:**

All meta rules are based on have single orientation. In other words, condition part of MR\_x where  $x \geq 2$  are based on MR\_x. This is because MR\_x ( $x \geq 2$ ) are considered as having multiple single orientations in this research. The roles is\_3D and have\_Two\_Opposite takes true or false as the role fillers.

MR\_1  $\equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\forall \text{have\_Single\_Orientation.Single\_Orientation}$

MR\_2  $\equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\forall \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\exists \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$

$\forall \text{have\_Opposite\_Orientation.Opposite\_Orientation}$

MR\_3  $\equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\forall \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\exists \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap$

$\forall \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation}$

MR\_4  $\equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\forall \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\exists \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap$

$\forall \text{have\_Angled\_Orientation.Angled\_Orientation}$

MR\_5  $\equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\forall \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\exists \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$

$\forall \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$

$\exists \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap$

$\forall \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation}$

MR\_6  $\equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\forall \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\exists \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap$

$\forall \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap$

$\exists \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$

$\forall \text{have\_Opposite\_Orientation.Opposite\_Orientation}$

MR\_7  $\equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\forall \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\exists \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap$

$\forall \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap$

$\exists \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap$

$\forall \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation}$

MR\_8  $\equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\forall \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\exists \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap$

$\forall \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap$

$\exists \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$

$\forall \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$

$\exists \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap$

$\forall \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation}$

MR\_9  $\equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\forall \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\exists \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$

$\forall \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$

$\exists \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap$

$\forall \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap$

$\exists \text{have\_Two\_Opposite.True} \sqcap \forall \text{have\_Two\_Opposite.True}$

MR\_10  $\equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\forall \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\exists \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap$

$\forall \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap$

$\exists \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$

$\forall \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$

$\exists \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap$

$\forall \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap$

$\exists \text{have\_Two\_Opposite.True} \sqcap \forall \text{have\_Two\_Opposite.True}$

MR\_11  $\equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\forall \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\exists \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap$

$\forall \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap \exists \text{is\_3D.True} \sqcap$

$\forall \text{is\_3D.True}$

MR\_12  $\equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\forall \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\exists \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap$

$\forall \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap \exists \text{is\_3D.True} \sqcap \forall \text{is\_3D.True}$

MR\_13  $\equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\forall \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\exists \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap$

$\forall \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap$

$\exists \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap$

$\forall \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap \exists \text{is\_3D.True} \sqcap$

$\forall \text{is\_3D.True}$

MR\_14  $\equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\forall \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\exists \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$

$\forall \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$

$\exists \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap$

$\forall \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap \exists \text{is\_3D.True} \sqcap$

$\forall \text{is\_3D.True}$

MR\_15  $\equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\forall \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$

$\exists \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$

$\forall \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$   
 $\exists \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap$   
 $\forall \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap$   
 $\exists \text{have\_Two\_Opposite.True} \sqcap \forall \text{have\_Two\_Opposite.True} \sqcap \exists \text{is\_3D.True} \sqcap$   
 $\forall \text{is\_3D.True}$   
 $\text{MR}_{16} \equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$   
 $\forall \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$   
 $\exists \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap$   
 $\forall \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap$   
 $\exists \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$   
 $\forall \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$   
 $\exists \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap$   
 $\forall \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap \exists \text{is\_3D.True} \sqcap$   
 $\forall \text{is\_3D.True}$   
 $\text{MR}_{17} \equiv \exists \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$   
 $\forall \text{have\_Single\_Orientation.Single\_Orientation} \sqcap$   
 $\exists \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap$   
 $\forall \text{have\_Angled\_Orientation.Angled\_Orientation} \sqcap$   
 $\exists \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$   
 $\forall \text{have\_Opposite\_Orientation.Opposite\_Orientation} \sqcap$   
 $\exists \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap$

$\forall \text{have\_Perpendicular\_Orientation.Perpendicular\_Orientation} \sqcap$

$\exists \text{have\_Two\_Opposite.True} \sqcap \forall \text{have\_Two\_Opposite.True} \sqcap \exists \text{is\_3D.True} \sqcap$

$\forall \text{is\_3D.True}$

The MRs for layer thickness and overcure are derived by combining the condition part of MR\_x above and OCLT. The OCLT\_7 represent the design requirements that have surface that has to be in 90° and accuracy requirement value is value 3 from Table 4.1. The OCLT\_8 then, represent the design requirements that do not have surface in 90° where accuracy requirement values is value 3.

$\text{MR\_x\_1} \equiv (\text{condition part of MR\_x}) \sqcap \exists \text{haveOCLT\_7.OCLT\_7} \sqcap$

$\forall \text{haveOCLT\_7.OCLT\_7}$

$\text{MR\_x\_2} \equiv (\text{condition part of MR\_x}) \sqcap \exists \text{haveOCLT\_8.OCLT\_8} \sqcap$

$\forall \text{haveOCLT\_8.OCLT\_8}$

The Qs are derived by adding MPT as property to the condition part of MRs.

$\text{Q\_2} \equiv (\text{condition part of MR\_2}) \sqcap \exists \text{determineMPT.MPT\_Infinite} \sqcap \forall$

$\text{determineMPT.MPT\_Infinite}$

## Appendix B: Expected resulted for description logics' applicability measure

Notation:  $(x, y_1, y_2, \dots, y_n)$ :  $x$  is direct child of  $y_1, y_2, \dots, y_n$

MR is omitted and only the numbers of meta rules in Figure 7.5 are shown.

### Top $\rightarrow$ Bottom set1:

1, 2, and 3 don't have parent, (4, 1, 2)

### Top $\rightarrow$ Bottom set2:

1, 2, and 3 don't have parent, (4, 1, 2), (5, 1, 3), (6, 2, 3), (7, 4, 5, 6), (8, 4)

### Top $\rightarrow$ Bottom set3:

1, 2, and 3 don't have parent, (4, 1, 2), (5, 1, 3), (6, 2, 3), (7, 4, 5, 6), (8, 4), (9, 7, 8), (10, 2), (11, 3), (12, 6, 10, 11)

### Top $\rightarrow$ Bottom set4:

1, 2, and 3 don't have parent, (4, 1, 2), (5, 1, 3), (6, 2, 3), (7, 4, 5, 6), (8, 4), (9, 7, 8), (10, 2), (11, 3), (12, 6, 10, 11), (13, 4, 10), (14, 8, 13), (15, 7, 13), (16, 9, 14, 15)

### Bottom $\rightarrow$ Top set1:

13 doesn't have parent, (14, 13), (15, 13), (16, 14, 15)

### Bottom $\rightarrow$ Top set2:

9, 10, and 11 don't have parent, (12, 10, 11), (13, 10), (14, 13), (15, 12, 13), (16, 9, 14, 15)

### Bottom $\rightarrow$ Top set3:

5, 6, 8, 10 and 11 don't have parent, (7, 5, 6), (9, 7, 8), (12, 6, 10, 11), (13, 10), (14, 8, 13), (15, 7, 12, 13), (16, 9, 14, 15)

### Bottom $\rightarrow$ Top set4:

Same as Top  $\rightarrow$  Bottom set 4

**Left → Right set1:**

1 doesn't have parent, (4, 1), (8, 4), (13, 4)

**Left → Right set2:**

1 doesn't have parent, (5, 1), (4, 1), (8, 4), (13, 4), (7, 4, 5), (9, 7, 8), (14, 8, 13)

**Left → Right set3:**

1 and 2 don't have parent, (5, 1), (4, 1, 2), (8, 4), (13, 4, 10), (7, 4, 5), (9, 7, 8), (14, 8, 13), (10, 2), (15, 7, 13), (16, 9, 14, 15)

**Left → Right set4:**

Same as Top → Bottom set 4

**Right → Left set1:**

3 doesn't have parent, (6, 3), (11, 3), (12, 6, 11)

**Right → Left set2:**

2 and 3 don't have parent, (6, 2, 3), (11, 3), (12, 6, 10, 11), (10, 2), (15, 12), (16, 15)

**Right → Left set3:**

2 and 3 don't have parent, (6, 2, 3), (11, 3), (12, 6, 10, 11), (10, 2), (15, 7, 12), (16, 9, 14, 15), (5, 3), (7, 5, 6), (9, 7), (14, 2)

**Right → Left set4:**

Same as Top → Bottom set 4

**Random set1:**

2 and 11 don't have parent, (6, 2), (9, 6)

**Random set2:**

2, 5, and 11 don't have parent, (6, 2), (9, 5, 6), (10, 2), (13, 2), (14, 13)

**Random set3:**

1, 2, and 11 don't have parent, (6, 2), (9, 4, 5, 6), (10, 2), (13, 4), (14, 13), (4, 1, 2), (15, 5, 12, 13), (12, 6, 10, 11)

**Random set4:**

Same as Top → Bottom set 4

**Appendix C: Expected query results for the retrieval performance measure**

**C-1 Rank for d.r.1~4**

	d.r.1	d.r.2	d.r.3	d.r.4
1	1.2.1, 1.2.2	2.1.1, 2.1.2	3.2.1, 3.2.2	4.1.1, 4.1.2
2	1.1.1, 1.1.2	2.2.1, 2.2.2	3.1.1, 3.1.2	4.2.1, 4.2.2
3				2.1.1, 2.1.2
4				2.2.1, 2.2.2
5				1.1.1, 1.1.2
6				1.2.1, 1.2.2

**C-2 Rank for d.r.5~8**

	d.r.5	d.r.6	d.r.7	d.r.8
1	3.1.1, 3.1.2 5.1.1, 5.1.2	3.2.1, 3.2.2 6.2.1, 6.2.2	3.1.1, 3.1.2 5.1.1, 5.1.2 6.1.1, 6.1.2 7.1.1, 7.1.2	3.2.1, 3.2.2 5.2.1, 5.2.2 6.2.1, 6.2.2 7.2.1, 7.2.2 8.2.1, 8.2.2
2	3.2.1, 3.2.2 5.2.1, 5.2.2	3.1.1, 3.1.2 6.1.1, 6.1.2	3.2.1, 3.2.2 5.2.1, 5.2.2 6.2.1, 6.2.2 7.2.1, 7.2.2	3.1.1, 3.1.2 5.1.1, 5.1.2 6.1.1, 6.1.2 7.1.1, 7.1.2 8.1.1, 8.1.2
3	1.1.1, 1.1.2	2.2.1, 2.2.2	4.1.1, 4.1.2	4.2.1, 4.2.2
4	1.2.1, 1.2.2	2.1.1, 2.1.2	4.2.1, 4.2.2	4.1.1, 4.1.2
5			2.1.1, 2.1.2	2.2.1, 2.2.2
6			2.2.1, 2.2.2	2.1.1, 2.1.2
7			1.1.1, 1.1.2	1.2.1, 1.2.2
8			1.2.1, 1.2.2	1.1.1, 1.1.2

**C-3 Rank for d.r.9~12**

	d.r.9	d.r.10	d.r.11	d.r.12
1	3.1.1, 3.1.2 5.1.1, 5.1.2 6.1.1, 6.1.2 7.1.1, 7.1.2 8.1.1, 8.1.2 9.1.1, 9.1.2	3.2.1, 3.2.2 5.2.1, 5.2.2 6.2.1, 6.2.2 7.2.1, 7.2.2 8.2.1, 8.2.2	11.2.1, 11.2.2	11.2.1, 11.2.2 12.2.1, 12.2.2
2	3.2.1, 3.2.2 5.2.1, 5.2.2 6.2.1, 6.2.2 7.2.1, 7.2.2 8.2.1, 8.2.2 9.2.1, 9.2.2	3.1.1, 3.1.2 5.1.1, 5.1.2 6.1.1, 6.1.2 7.1.1, 7.1.2 8.1.1, 8.1.2	11.1.1, 11.1.2	11.1.1, 11.1.2 12.1.1, 12.1.2
3	4.1.1, 4.1.2	4.2.1, 4.2.2 10.2.1, 10.2.2	3.2.1, 3.2.2 5.2.1, 5.2.2 6.2.1, 6.2.2 7.2.1, 7.2.2 8.2.1, 8.2.2	3.2.1, 3.2.2 5.2.1, 5.2.2 6.2.1, 6.2.2 7.2.1, 7.2.2 8.2.1, 8.2.2
4	4.2.1, 4.2.2	4.1.1, 4.1.2 10.1.1, 10.1.2	3.1.1, 3.1.2 5.1.1, 5.1.2 6.1.1, 6.1.2 7.1.1, 7.1.2 8.1.1, 8.1.2	3.1.1, 3.1.2 5.1.1, 5.1.2 6.1.1, 6.1.2 7.1.1, 7.1.2 8.1.1, 8.1.2
5	2.1.1, 2.1.2	2.2.1, 2.2.2	4.2.1, 4.2.2 10.2.1, 10.2.2	4.2.1, 4.2.2 10.2.1, 10.2.2
6	2.2.1, 2.2.2	2.1.1, 2.1.2	4.1.1, 4.1.2 10.1.1, 10.1.2	4.1.1, 4.1.2 10.1.1, 10.1.2
7	1.1.1, 1.1.2	1.2.1, 1.2.2	2.2.1, 2.2.2	2.2.1, 2.2.2
8	1.2.1, 1.2.2	1.1.1, 1.1.2	2.1.1, 2.1.2	2.1.1, 2.1.2
9			1.2.1, 1.2.2	1.2.1, 1.2.2
10			1.1.1, 1.1.2	1.1.1, 1.1.2

**C-4 Rank for d.r.13~16**

	d.r.13	d.r.14	d.r.15	d.r.16
1	11.1.1, 11.1.2 12.1.1, 12.1.2 13.1.1, 13.1.2	11.1.1, 11.1.2 12.1.1, 12.1.2 13.1.1, 13.1.2 14.1.1, 14.1.2	11.1.1, 11.1.2 12.1.1, 12.1.2 13.1.1, 13.1.2 14.1.1, 14.1.2 15.1.1, 15.1.2	11.2.1, 11.2.2 12.2.1, 12.2.2 13.2.1, 13.2.2 14.2.1, 14.2.2 15.2.1, 15.2.2 16.2.1, 16.2.2
2	11.2.1, 11.2.2 12.2.1, 12.2.2 13.2.1, 13.2.2	11.2.1, 11.2.2 12.2.1, 12.2.2 13.2.1, 13.2.2 14.2.1, 14.2.2	11.2.1, 11.2.2 12.2.1, 12.2.2 13.2.1, 13.2.2 14.2.1, 14.2.2 15.2.1, 15.2.2	11.1.1, 11.1.2 12.1.1, 12.1.2 13.1.1, 13.1.2 14.1.1, 14.1.2 15.1.1, 15.1.2 16.1.1, 16.1.2
3	3.1.1, 3.1.2 5.1.1, 5.1.2 6.1.1, 6.1.2 7.1.1, 7.1.2 8.1.1, 8.1.2 9.1.1, 9.1.2	3.1.1, 3.1.2 5.1.1, 5.1.2 6.1.1, 6.1.2 7.1.1, 7.1.2 8.1.1, 8.1.2 9.1.1, 9.1.2	3.1.1, 3.1.2 5.1.1, 5.1.2 6.1.1, 6.1.2 7.1.1, 7.1.2 8.1.1, 8.1.2 9.1.1, 9.1.2	3.2.1, 3.2.2 5.2.1, 5.2.2 6.2.1, 6.2.2 7.2.1, 7.2.2 8.2.1, 8.2.2 9.2.1, 9.2.2
4	3.2.1, 3.2.2 5.2.1, 5.2.2 6.2.1, 6.2.2 7.2.1, 7.2.2 8.2.1, 8.2.2 9.2.1, 9.2.2	3.2.1, 3.2.2 5.2.1, 5.2.2 6.2.1, 6.2.2 7.2.1, 7.2.2 8.2.1, 8.2.2 9.2.1, 9.2.2	3.2.1, 3.2.2 5.2.1, 5.2.2 6.2.1, 6.2.2 7.2.1, 7.2.2 8.2.1, 8.2.2 9.2.1, 9.2.2	3.1.1, 3.1.2 5.1.1, 5.1.2 6.1.1, 6.1.2 7.1.1, 7.1.2 8.1.1, 8.1.2 9.1.1, 9.1.2
5	4.1.1, 4.1.2 10.1.1, 10.1.2	4.1.1, 4.1.2 10.1.1, 10.1.2	4.1.1, 4.1.2 10.1.1, 10.1.2	4.2.1, 4.2.2 10.2.1, 10.2.2
6	4.2.1, 4.2.2 10.2.1, 10.2.2	4.2.1, 4.2.2 10.2.1, 10.2.2	4.2.1, 4.2.2 10.2.1, 10.2.2	4.1.1, 4.1.2 10.1.1, 10.1.2
7	2.1.1, 2.1.2	2.1.1, 2.1.2	2.1.1, 2.1.2	2.2.1, 2.2.2
8	2.2.1, 2.2.2	2.2.1, 2.2.2	2.2.1, 2.2.2	2.1.1, 2.1.2
9	1.1.1, 1.1.2	1.1.1, 1.1.2	1.1.1, 1.1.2	1.2.1, 1.2.2
10	1.2.1, 1.2.2	1.2.1, 1.2.2	1.2.1, 1.2.2	1.1.1, 1.1.2

## REFERENCES:

1. Groover, M.P., *Fundamentals of Modern Manufacturing: Materials, Processes, and Systems*. First ed. 1996, Upper Saddle River: Prentice Hall.
2. Jacobs, P.F., *Rapid Prototyping and Manufacturing: Fundamentals of Stereolithography*. Fifth edition ed. 1992, Dearborn: Society of Manufacturing Engineers.
3. Kalpakjian and Serape, *Manufacturing Engineering and Technology*. Second ed. 1992, New York: Addison-Wesley. 1258.
4. Chen, Y., *Computer-Aided Design for Rapid Tooling: Method for Mold Design and Design-for-Manufacturing*, in *Mechanical Engineering*. 2001, Georgia Institute of Technology: Atlanta.
5. Sambu, S., Y. Chen, and D.W. Rosen, *Gometric Tailoring: A Design for Manufacturing Method for Rapid Prototyping and Rapid Tooling*. *Journal of Mechanical Design*, 2004. **126**: p. 1-10.
6. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F., *The Description Logic Handbook*. Vol. . 2002: Cambridge University Press. .
7. Cohen, W.W., Borgida, A., Hirsh, H., *Computing Least Common Subsumers in Description Logics*, in *Tenth National Conference on Artificial Intelligence*. 1992: San Jose CA. p. .
8. Guangchun, W., et al., *A rapid design and manufacturing system for product development application*. *Rapid Prototyping*, 2004. **10**: p. 200-206.
9. Hopkinson, N. and P. Dickens, *Rapid prototyping for direct manufacture*. *Rapid Prototyping*, 2001. **7**: p. 197-202.
10. Kulkarni, P., A. Marsan, and D. Dutta, *A review of process planning techniques in layered manufacturing*. *Rapid Prototyping*, 2000. **6**: p. 18-35.
11. Rehman, S. and M.D. Guenov, *A Methodology for Modelling Manufacturing Costs at Conceptual Design*. *Computers Integrated Engineering*, 1998. **35**: p. 623-626.
12. McClurkin, J.E. and D.W. Rosen, *Computer-aided build style decision support for stereolithography*. *Rapid Prototyping*, 1998. **4**: p. 4-13.
13. Merz, R., et al. *Shape Deposition Manufacturing*. in *Proceedings of the Soplid Freeform Fabrication Symposium*. 1994. Austin, TX, University of Texas Austin.

14. Onuh, S.O., *Rapid prototyping integrated systems*. Rapid Prototyping, 2001. **7**: p. 220-223.
15. Chen, L. and S. Li, *A computerized team approach for concurrent product and process design optimization*. Computer-Aided Design, 2002. **34**: p. 57-69.
16. Cormier, D., K. Unnanon, and E. Sani, *Specifying non-uniform cusp heights as a potential aid for adaptive slicing*. Rapid Prototyping, 2000. **6**: p. 204-211.
17. Kumar, M. and A.R. Choudhury, *Adaptive slicing with cubic patch approximation*. Rapid Prototyping, 2003. **8**: p. 224-232.
18. Lin, F., W. Sun, and Y. Yan, *Optimization with minimum process error for layered manufacturing fabrication*. Rapid Prototyping, 2001. **7**: p. 73-81.
19. Lin, L. and S.W. Changchien, *A knowledge-based design critique system for manufacture and assembly of rotational machined parts in concurrent engineering*. Computers in Industry, 1996. **32**: p. 117-140.
20. West, A.P., S. Sambu, and D.W. Rosen, *A process planning method for improving build performance in stereolithography*. Computer-Aided Design, 2001. **33**: p. 65-79.
21. Mistree, F., O.F. Hughes, and B.A. Bras, *The Compromise Decision Support Problem and the Adaptive Linear Programming Algorithm*. Structural Optimization: Status and Promise, 1992: p. 249-289.
22. Lynn-Charney, C., *Accuracy Models for SLA Build Style Decision Support*, in *School of Mechanical Engineering*. 1998, Georgia Institute of Technology: Atlanta. p. 250.
23. Lynn-Charney, C. and D.W. Rosen, *Usage of accuracy models in stereolithography process planning*. Rapid Prototyping, 2000. **6**: p. 77-86.
24. Mistree, F., et al. *The Validation Square*. in *2000 ASME Design Theory and Methodology Conference*. 2000.
25. O'Driscoll, M., *Design for Manufacture*. Journal of Materials Processing Technology, 2002. **122**: p. 318-321.
26. Shah, J. and P.K. Wright. *DEVELOPING THEORETICAL FOUNDATIONS OF DFM*. in *ASME 2000 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2000. Baltimore, Maryland, USA.

27. Korngold, J. and T. Luscher, *Product Design for Ease of Assembly DFA/DFM Manual*. 2000, Rensselaer Polytechnic Institute. p. 27.
28. Wood, W.H., et all. *NEW DIRECTIONS IN DESIGN FOR MANUFACTURING*. in *ASME 2004 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2004. Salt Lake City, Utah, USA.
29. Susman, G.I., *Integrating Design and Manufacturing for Competitive Advantage*. 1992, New York: Oxford University Press.
30. Magrab, E.B. and D.T. E., *Geometric reasoning for manufacturability evaluation application to powder metallurgy*. *Computer-Aided Design*, 1996. **28**: p. 783-794.
31. McAdams, D.A. and R.A. Bidkar. *AUTOMATED MANUFACTURABILITY ANALYSIS FOR INJECTION-MOLDED AND DIE-CAST PARTS*. in *ASME 2003 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2003. Chicago, Illinois, USA.
32. Miao, H.K., J. Shah, and N. Sridharan. *INTEGRATION OF COMMERCIAL CAD WITH COMMERCIAL NC PROGRAMMING PACKAGES*. in *ASME 2000 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2000. Baltimore, Maryland, USA.
33. Shah, J., Davidson, Joseph K., Ramaswamy, Sanjay. *COMPUTER AIDED GD&T ADVISOR BASED ON Y14.5 CONFORMANCE & GOOD PRACTICE*. in *ASME 2001 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2001. Pittsburgh, Pennsylvania, USA.
34. Shah, J., Shen, Zhengshu, Davidson, Joseph K. *SIMULATION-BASED TOLERANCE AND ASSEMBLABILITY ANALYSES OF ASSEMBLIES WITH MULTIPLE PIN/HOLE FLOATING MATING CONDITIONS*. in *ASME 2005 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2005. Long Beach, California USA.
35. Li, X., S. Kambhampati, and J. Shah. *ASSUPA: A FRAMEWORK FOR INTERACTIVE AND ITERATIVE SYNTHESIS AND IMPROVEMENT OF PROCESS PLANS*. in *ASME 2000 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2000. Baltimore, Maryland, USA.
36. Roberts, C., M. Henderson, and R. Stage, *Generating resource based flexible form manufacturing features through objective driven clustering*. *Computer-Aided Design*, 1999. **31**: p. 119-130.

37. Midha, P.S. and L.N. Smith, *An Interactive System for Optimum and Concurrent Design of Components for Manufacture by Powder Metallurgy Technology*. Journal of Materials Processing Technology, 1996. **61**: p. 187-192.
38. Lu, S.C., et al., *A Simple Visualization Tool to Support Concurrent Engineering Design*. Computer-Aided Design, 1997. **29**: p. 727-735.
39. Dilts, D.M. and T.S. Geiger, *Automated design-to-cost: integrating costing into the design decision*. Computer-Aided Design, 1994. **28**: p. 423-438.
40. Gao, J.X. and R. Sharma, *A Progressive design and manufacturing evaluation system incorporation STEP AP224*. Computers in Industry, 2002. **47**: p. 155-167.
41. Golob, B. and G. Brunetti, *A feature-based approach towards an integrated product model including conceptual design information*. Computer-Aided Design, 2000. **32**: p. 877-887.
42. Ramani, K. and H. Patwardhn. *MANUFACTURING FEATURE BASED DYNAMIC COST ESTIMATION FOR DESIGN*. in *ASME 2004 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2004. Salt Lake City, Utah, USA.
43. Welp, E.G. *KNOWLEDGE-BASED COST ESTIMATION OF PRODUCT CONCEPTS*. in *ASME 2004 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2004. Salt Lake City, Utah, USA.
44. Winner, R.I., et al., *The Role of Concurrent Engineering in Weapons System Acquisition*. 1988, Institute for Defence Analysis: Alexandria, Virginia.
45. Eversheim, W., et al., *Simultaneous Engineering Approach to an Integrated Design and Process Planning*. European Journal of Operation Research, 1997. **100**: p. 327-337.
46. Jin, Y., R. Ganeshan, and P. Li. *Agent-Supported Collaborative Design*. in *CIRP 1997 International Design Seminar on Multimedia Technologies for Collaborative Design and Manufacturing*. 1997. Los Angeles, CA, USA.
47. Ma, Y.-S. and T. Tong, *Associative feature modeling for concurrent engineering integration*. Computers in Industry, 2003. **51**: p. 51-71.
48. Shah, J.J., R. D'Souza, and M.S. Medichalam. *N-REP: A NEUTRAL FEATURE REPRESENTATION TO SUPPORT FEATURE MAPPING AND DATA EXCHANGE ACROSS APPLICATION*. in *ASME 2004 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2004. Salt Lake City, Utah, USA.

49. Shah, J.J. and N.S. Joshi. *ON THE VIABILITY OF DEVELOPING CAD DATA EXCHANGE STANDARD FOR FORM FEATURES*. in *ASME 2005 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2005. Long Beach, California USA.
50. Shah, J.J. and Z. Zhao. *Modeling and Representation of Manufacturing Knowledge for DFM Systems*. in *ASME 2004 Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. 2004. Salt Lake City, Utah, USA.
51. Choi, H.-J., et al. *TOWARDS A STANDARDIZED ENGINEERING FRAMEWORK FOR DISTRIBUTED, COLLABORATIVE PRODUCT REALIZATION*. in *ASME 2003 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2003. Chicago, Illinois, USA.
52. Cutkosky, M.R. and T. Mori. *AGENT-BASED COLLABORATIVE DESIGN OF PARTS IN ASSEMBLY*. in *ASME 1998 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 1998. Atlanta, GA, USA.
53. Edwards, K.L., *Towards more strategic product design for manufacture and assembly: priorities for concurrent engineering*. *Computers in Industry*, 2002. **23**: p. 651-656.
54. Fischer, G.W., B.N. Yetukuri, and N.V. Yetukuri, *SPAW: A design tool for planning a manufacturing process in a concurrent engineering environment*. *Computers in Industry*, 1996. **32**: p. 79-93.
55. Gerhard, J., et al. *A DISTRIBUTED PRODUCT REALIZATION ENVIRONMENT FOR DESIGN AND MANUFACTURING*. in *ASME 2000 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2000. Baltimore, Maryland, USA.
56. Karne, R.K., et al. *WEB-IT-MAN: A WEB-BASED INTEGRATED TOOL FOR MANUFACTURING ENVIRONMENT*. in *ASME 1998 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 1998. Atlanta, GA, USA.
57. Kotak, D., et al., *Agent-based holonic design and operations environment for distributed manufacturing*. *Computers in Industry*, 2003. **52**: p. 95-108.
58. Mak, K.L. and G.Q. Huang, *Design for manufacture and assembly on the Internet*. *Computers in Industry*, 1999. **38**: p. 17-30.

59. Rajagopalan, S., et al. *INTEGRATED DESIGN AND RAPID MANUFACTURING OVER INTERNET*. in *ASME 1998 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 1998. Atlanta, GA, USA.
60. Rosen, D.W., et al. *DESIGN DECISION TEMPLATES AND THEIR IMPLEMENTATION FOR DISTRIBUTED DESIGN AND SOLID FREEFORM FABRICATION*. in *ASME 2000 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2000. Baltimore, Maryland, USA.
61. Sormaz, D.N. and N. Neerukonda. *Distributed Integration of Knowledge-based Process Planning with CAM Software*. in *ASME 2004 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2004. Salt Lake City, Utah, USA.
62. Wang, F.-C.F. and P.K. Wright. *WEB-BASED CAD TOOLS FOR A NETWORKED MANUFACTURING SERVICE*. in *ASME 1998 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 1998. Atlanta, GA, USA.
63. Brissaud, D. and S. Tichkiewitch, *Innovation and manufacturability analysis in an integrated design context*. *Computers in Industry*, 2000. **43**: p. 111-121.
64. Smith, J.I., et al. *DESIGN, IMPLEMENTATION AND USE OF A KNOWLEDGE ACQUISITION TOOL FOR SHEET METAL FORMING*. in *ASME 2004 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2004. Salt Lake City, Utah, USA.
65. Stauffer, L., H. Ren, and R. Rule. *A TEMPLATE FOR DESIGN FOR MANUFACTURING GUIDELINES*. in *ASME 2003 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2003. Chicago, Illinois, USA.
66. Tharakan, P.V., Z. Zhao, and J. Shah. *MANUFACTURABILITY EVALUATION SHELL: A RE-CONFIGURABLE ENVIRONMENT FOR TECHNICAL AND ECONOMIC MANUFACTURABILITY EVALUATION*. in *ASME 2003 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2003. Chicago, Illinois, USA.
67. Rosen, D.W., et al. *DESIGN DECISION TEMPLATES AND THEIR IMPLEMENTATION FOR DISTRIBUTED DESIGN AND SOLID FREEFORM FABRICATION*. in *ASME 2000 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2000. Baltimore, Maryland, USA.

68. Szykman, S., et al., *Design Repositories: Engineering Design's New Knowledge Base*. IEEE Intelligent Systems, 2000. **15**: p. 48-55.
69. Lewis, K. and F. Mistree. *Collaborative, Sequential, and Isolated Decisions in Design*. in *9th ASME International Conference on Design Theory and Methodology (DTM)*. 1997. Sacramento, California.
70. Brunnermeier, S.B. and S.A. Martin, *Interoperability Cost Analysis of the U.S. Automotive Supply Chain*. 1999, National Institute of Standards and Technology: Gaithersburg, Maryland.
71. Mocko, G.M., *A KNOWLEDGE FRAMEWORK FOR INTEGRATING MULTIPLE PERSPECTIVES IN DECISION-CENTRIC DESIGN*, in *School of Mechanical Engineering*. 2006, Georgia Institute of Technology: Atlanta. p. 441.
72. Ciocoiu, M., Nau, D. S. , Gruninger, M. , *Ontologies for Integrating Engineering Applications*. Journal of Computing and Information Science in Engineering, 2001. **1**: p. 12-22.
73. Hunten, K.A., *CAD/FEA Integration with STEP AP209 Technology and Implementation*. 1997, Lockheed Martin Corporation.
74. Kemmerer, S.J., *STEP: The Grand Experience: Special Publication*. 1999, Gaithersburg, Maryland USA.: National Institute of Standards and Technology. 939.
75. Zeiny, A. *COMPUTABLE DYNAMIC DESIGN REPOSITORY FOR PRODUCT DATA REPRESENTATION*. in *ASME 2004 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2004. Salt Lake City, Utah, USA.
76. Bespalov, D., W.C. Regli, and A. Shokoufandeh. *REEB GRAPH BASED SHAPE RETRIEVAL FOR CAD*. in *ASME 2003 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2003. Chicago, Illinois, USA.
77. Chang, H.C., et al., *Indexing and retrieval in machining process planning using case-based reasoning*. Artificial Intelligence in Engineering, 2000. **14**: p. 1-13.
78. Lou, K., et al. *A RECONFIGURABLE 3D ENGINEERING SHAPE SEARCH SYSTEM PART II: DATABASE INDEXING, RETRIEVAL, AND CLUSTERING*. in *ASME 2003 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2003. Chicago, Illinois, USA.

79. Lyer, N., et al. *A RECONFIGURABLE 3D ENGINEERING SHAPE SEARCH SYSTEM PART I: SHAPE REPRESENTATION*. in *ASME 2003 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2003. Chicago, Illinois, USA.
80. McWherter, D., et al., *Solid Model Databases: Techniques and Empirical Results*. *Journal of Computing and Information Science in Engineering*, 2001. **1**: p. 300-310.
81. Neelamkavil, J. and M. Kernahan. *A FRAMEWORK FOR DESIGN KNOWLEDGE REUSE*. in *ASME 2003 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2003. Chicago, Illinois, USA.
82. Prasad, B. and J. Rogers. *A KNOWLEDGE-BASED SYSTEM ENGINEERING PROCESS FOR OBTAINING ENGINEERING DESIGN SOLUTION*. in *ASME 2005 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conference*. 2005. Long Beach, CA. USA.
83. Qin, X. and W.C. Regli, *A study in applying case-based reasoning to engineering design: Mechanical bearing design*. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 2003. **17**: p. 235-252.
84. Lee, J.-y. and K.-h. Lee, *Agent-based collaborative design system and conflict resolution based on a case-based reasoning approach*. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 2001. **16**: p. 93-102.
85. Wood III, W.H. and A.M. Agogino, *Case-based conceptual design information server for concurrent engineering*. *Computer Aided Design*, 1995. **28**: p. 361-369.
86. Kim, J., et al., *Knowledge-rich catalog services for engineering design*. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 2003. **17**: p. 349-366.
87. Kopena, J.B., C.B. Cera, and W.C. Regli. *CONCEPTUAL DESIGN KNOWLEDGE MANAGEMENT AND THE SEMANTIC WEB*. in *ASME 2005 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2005. Long Beach, California USA.
88. Kopena, J.B. and W.C. Regli, *DESIGN REPOSITORIES ON THE SEMANTIC WEB WITH DESCRIPTION-LOGIC ENABLED SERVICES*. 2003, Drexel University.

89. Li, Z., et al. *SEMANTICS-BASED DESIGN KNOWLEDGE ANNOTATION AND RETRIEVAL*. in *ASME 2005 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2005. Long Beach, California USA.
90. MacGregor, R., *LOOM User Manual*, in *Working Paper ISI/WP-22*. 1990, University of Southern California: Marina del Ray, CA.
91. MacGregor, R. *A Description Classifier for the Predicate calculus*. in *Proc. Twelfth National Conf. Artificial Intelligence*. 1994.
92. Channa, N., et al. *A CAN-BASED P2P INFRASTRUCTURE FOR SEMANTIC WEB SERVICES*. in *ASME 2005 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2005. Long Beach, California USA.
93. Mohan, K. and B. Ramesh. *Ontology-based Support for Variability Management in Product and Service Families*. in *IEEE Proceedings of the 36th Hawaii Internal Conference on System Science*. 2002. Hawaii USA.
94. Yang, Q. and C.Y. Miao. *A SERVICE-ORIENTED APPROACH TO ENGINEERING ONTOLOGY DEVELOPMENT FOR PRODUCT LIFE CYCLE PROCESSES*. in *ASME 2005 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2005. Long Beach, California USA.
95. AbdelSalam, H.M. and H.P. Bao. *TOWARDS A COLLABORATIVE ENGINEERING-COMPUTATION ENVIRONMENT: AN APPLICATION OF AN OBJECT-ORIENTED DATABASE TO PROJECT MANAGEMENT*. in *ASME 2000 Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2000. Baltimore, Maryland, USA.
96. Al-Khudair, A., W.A. Gray, and J.C. Miles. *DYNAMIC CONFIGURATION EVOLUTION OF DISTRIBUTED CE DESIGN IN OBJECT-ORIENTED DATABASES*. in *ASME 2001 Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2001. Pittsburgh, Pennsylvania, USA.
97. Kerr, C.I., R. Roy, and P.J. Sackett. *A PRODUCT ONTOLOGY FOR AUTOMOTIVE SEAT SPECIFICATION*. in *ASME 2004 Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2004. Salt Lake City, Utah, USA.

98. Lee, J.-H. and H.-W. Suh. *OWL-BASED HYBRID PRODUCT KNOWLEDGE MODEL FOR COLLABORATIVE ENGINEERING ENVIRONMENT*. in *ASME 2005 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2005. Long Beach, California USA.
99. Nanda, J., et al. *EXPLOERING SEMANTIC WEB TECHNOLOGIES FOR PRODUCT FAMILY MODELING*. in *ASME 2004 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2005. Salt Lake City, Utah, USA.
100. Patil, L., D. Dutta, and R. Sriram. *ONTOLOGY FORMALIZATION OF PRODUCT SEMANTICS FOR PRODUCT LIFECYCLE MANAGEMENT*. in *ASME 2005 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2005. Long Beach, California USA.
101. Silva, J.P.M.A., et al. *PRODUCT LIFECYCLE MANAGEMENT ENHANCEMENT WITH AN ONTOLOGICAL APPROACH*. in *ASME 2005 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2005. Long Beach, California USA.
102. Aziz, H., et al., *Open standard, open source, and peer-to-peer tools and method for collaborative product development*. *Computers in Industry*, 2005. **56**: p. 260271.
103. Szykman, S., R. Sriram, and W.C. Regli, *The Role of Knowledge in Next-generation Product Development Systems*. *Journal of Computing and Information Science in Engineering*, 2001. **1**: p. 3-11.
104. Wijnker, T.C., et al. *INTEGRATION OF DATA MANAGEMENT SYSTEMS WITH AN ONTOLOGY BASED INFORMATION MANAGEMENT SYSTEM*. in *ASME 2001 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2001. Pittsburgh, PA, USA.
105. Zhan, P., U. Jayaram, and S. Jayaram. *ONTOLOGY-BASED SEMANTIC APPROACH FOR COMMUNICATION IN ENGINEERING DESIGN EVALUATION*. in *ASME 2005 Internal Design Engineering Technical Conferences & Computers and Information in Engineering Conferences*. 2005. Long Beach, California USA.
106. Sambu, S., *A Design for Manufacture Method for Rapid Prototyping and Rapid Tooling*, in *School of Mechanical Engineering*. 2001, Georgia Institute of Technology: Atlanta, Georgia.
107. Rosen, D.W., *RP in Engineering*. 2004, Georgia Institute of Technology: Atlanta, Georgia.

108. Pahl, G. and W. Beitz, *Engineering Design: a systematic approach*. Second ed, ed. K. Wallace. 1995, New York: Springer.
109. Furrens, K.K., *Standards for the rapid prototyping industry*. Rapid Prototyping, 1999. **5**: p. 169-178.
110. Wai, H.W., *RP in art and conceptual design*. Rapid Prototyping, 2001. **7**: p. 217-219.
111. Bourbakis, N.G., *Artificial Intelligence and Automation*. 1998, New York: World Science Publishing Co.
112. Brachman, R.J. and H.J. Levesque, *Knowledge representation and reasoning*. 2004, Toronto: Addison Wesley.
113. Giarratano, J. and G. Riley, *EXPERT SYSTEMS: Principles and Programming*. Third ed. 1998, Boston: PWS Publishing Company.
114. Jackson, P.C., *Introduction to Artificial Intelligence*. First ed. 1974, New York: Mason & Lipscomb Publishers.
115. Mirzai, A.R., *ARTIFICIAL INTELLIGENCE: Concepts and Applications in Engineering*. 1990, Cambridge, Massachusetts: MIT Press.
116. Poole, D., A. Mackworth, and R. Goebel, *Computational Intelligence: a logical approach*. 1998, Oxford: Oxford University Press.
117. Rich, E. and K. Knight, *Artificial Intelligence*. Second ed. 1991, Boston, Massachusetts: McGraw Hill.
118. Russell, S.J. and P. Norvig, *Artificial Intelligence: A Modern Approach*. 2nd ed. Prentice Hall Series in Artificial Intelligence. 1995, New York: Alan Apt. 932.
119. Baader, F., et al., *The Description Logic Handbook*. 2002.
120. Cohen, W.W., A. Borgida, and H. Hirsh, *Computing Least Common Subsumers in Description Logics*. Representation and Reasoning, 1994. **5**: p. 754-760.
121. Gil, Y., *Plan Representation and Reasoning with Description Logics*. 2003, USC/Information Science Institute: Marina del Rey.
122. Sanner, S.P., *Towards practical taxonomic classification for description logics on the semantic web*. 2002, University of Toronto: Toronto.
123. Wikipedia, t.f.e. *First-order logic*. 2006 [cited; Available from: [http://en.wikipedia.org/wiki/First\\_order\\_logic](http://en.wikipedia.org/wiki/First_order_logic).

124. Borgida, A., *On the relative expressiveness of description logics and predicate logics*. Artificial Intelligence, 1996. **82(1-2)**: p. 353-367.
125. Pan, J.Z., *Description Logics: Reasoning Support for the Semantic Web*, in *School of Computer Science*,. 2004, University of Manchester: Manchester.
126. Zolin, E. *Complexity of reasoning in description logics*. [cited; Available from: <http://www.cs.man.ac.uk/~ezolin/logic/complexity.html>].
127. Ganter, B. and R. Wille, *Formal Concept Analysis: Mathematical Foundations*. 1999, New York: Springer.
128. Udoyen, N., *Information Modeling for Intent-Based Retrieval of Parametric Finite Element Analysis Models*, in *GWW Woodruff school of Mechanical Engineering*. 2006, Georgia Institute of Technology: Atlanta.
129. Horridge, M., et al., *A Practical Guide To Building OWL Ontologies Using The: Prot'eg'e-OWL Plugin and CO-ODE Tools*. 2004, The University Of Manchester, Stanford University: Manchester, U.K.
130. Daconta, M.C., L.J. Obrst, and K.T. Smith, *The Semantic Web*. 2003, Indianapolis, Indiana: Wiley.
131. Herman, I. and J. Handler. *Web Ontology Language (OWL)*. 2004 [cited; Available from: <http://www.w3.org/2004/OWL/>].
132. Powers, S., *Practical RDF*. 2003, Sebastopol, CA: O'Reilly & Associates.
133. *Protégé*. [cited; Available from: <http://protege.stanford.edu/>].
134. *RacerPro*. [cited; Available from: <http://www.racer-systems.com/products/index.phtml>].
135. *Protégé-OWL*. [cited; Available from: <http://protege.stanford.edu/plugins/owl/>].
136. Bechhofer, S., *The DIG Description Logic Interface: DIG/1.0*. 2002, University of Manchester: Manchester, U.K. p. 16.
137. Bechhofer, S., *The DIG Description Logic Interface: DIG/1.1*. 2003, University of Manchester: Manchester, U.K. p. 16.
138. Allen, M., et al., *DIGITAL CLAY FOR SHAPE INPUT AND DISPLAY*. 2001, Georgia Institute of Technology: Atlanta, Georgia. p. 24.

139. Becker, G., *Continued Development of the 'Formable Crust' Design and its Manufacturability*. 2005, Georgia Institute of Technology: Atlanta, Georgia. p. 131.
140. Nguyen, A.N., *Designing, Manufacturing, and Predicting Deformation of a Formable Crux Matrix*, in *GWW School of Mechanical Engineering*. 2004, Georgia Institute of Technology: Atlanta, GA.
141. Bechhofer, S. *DL Implementation Group (DIG)*. 2006 [cited; Available from: <http://dl.kr.org/dig/index.html>].
142. Brundage, M., *XQuery: The XML Query Language*. 2004, Boston, Massachusetts: Addison-Wesley.
143. Griffiths, I., J. Flanders, and C. Sells, *Mastering Visual Studio .NET*. 2003, Sebastopol, CA: O'Reilly and Associates Inc.
144. Key, M., *XPath 2.0*. Third ed. 2004, Indianapolis, Indiana: Wiley Publishing, Inc.
145. Liberty, J. and D. Hurwitz, *Programming ASP.NET*. 2003, Sebastopol, CA: O'Reilly and Associates Inc.
146. Robinson, S., et al., *Professional C#*. Second ed. 2003, Indianapolis, Indiana: Wiley Publishing Co.