THESIS REPORT

Master's Degree

Integration of a Manufacturing Resource Planning System with a Manufacturing Information Repository

by D.W. Rush
Advisors: G.H. Harhalakis
I. Minis

M.S. 96 -3

INSTITUTE FOR SYSTEMS RESEARCH

Sponsored by
the National Science Foundation
Engineering Research Center Program,
the University of Maryland,
Harvard University,
and Industry

INTEGRATION OF A MANUFACTURING RESOURCE PLANNING SYSTEM

WITH A

MANUFACTURING INFORMATION REPOSITORY

by

David W. Rush

Thesis submitted to the Faculty of the Graduate School
of the University of Maryland in partial fulfillment
of the requirements for the degree of
Master of Science
1996

Advisory Committee:

Associate Professor George H. Harhalakis (Advisor)
Associate Professor Ioannis Minis (Co-Advisor)
Professor Michael O. Ball
Assistant Professor Jeffery W. Herrmann

ABSTRACT

Title of the Thesis: Integration of a Manufacturing Resource Planning System with a Manufacturing Information Repository

Name of degree candidate: David W. Rush

Degree and Year: Master of Science, 1996

Thesis directed by: Drs. George Harhalakis and Ioannis Minis
Departments of Mechanical Engineering and Institute for Systems Research
University of Maryland, College Park, MD 20742

This work employs a Systems Engineering approach to integrate two heterogeneous database systems in a chemical manufacturing facility. The first system is a Manufacturing Resource Planning system (MRPII) which supports production planning and control. The second system is a Manufacturing Information Repository (MIR) that manages and stores information concerning processes, equipment and materials. Phase I of this project compared the data structures of the two systems for common data fields. With very little commonality found, Phase II focused on the interrelationships and intra relationships of the data structures of the two systems and yielded the following results: 1) Detailed data models of the two systems that showed the MIR system to be hierarchical and the MRPII system to be relational; 2) A set of mapping conventions between the corresponding data fields of the two systems; 3) An algorithm and a computer program to upload information from the MIR to the MRPII system. To demonstrate the computer program, a case study was performed using sample MIR data.

INTEGRATION OF A MANUFACTURING RESOURCE PLANNING SYSTEM

WITH A

MANUFACTURING INFORMATION REPOSITORY

by

David W. Rush

Thesis submitted to the Faculty of the Graduate School
of the University of Maryland in partial fulfillment
of the requirements for the degree of
Master of Science
1996

Advisory Committee:

Associate Professor George H. Harhalakis (Advisor)
Associate Professor Ioannis Minis (Co-Advisor)
Professor Michael O. Ball
Assistant Professor Jeffery W. Herrmann

DEDICATION

To my wife and son for all of their love, support and patience.

ACKNOWLEDGMENTS

First and foremost I am ever grateful to the late Dr. George Harhalakis whose ingenuity and support were paramount throughout this project. When Dr. Harhalakis first approached me with this idea for a thesis, I had never met him nor knew of him and while I was ecstatic that he chose me among all the other Systems Engineering students, I was skeptical. I see now how this skepticism was totally unwarranted and I only regret that he is not here to see this thesis come to fruition.

I would also like to thank John Grillo and everyone else at Merck and Co. Inc. who assisted me with this thesis. Without their combined knowledge of database systems and the intricate details of the chemical manufacturing process, the integration of these two systems would not have been possible.

I would also like to thank Dr. Ioannis Minis who was kind enough to take over as my thesis advisor. His in-depth knowledge of MRPII systems was very helpful during the final stages of this research work.

Finally, I would like to thank everyone at the Computer Integrated Manufacturing lab, Dr. C. P. Lin, Dr. Rakesh Nagi, Howard and Amy and all my friends at the Department of Architecture, Engineering and Construction who have inspired me to pursue my educational goals.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. Introduction

Several information systems are employed in a manufacturing environment such as Computer Aided Design (CAD), Computer Aided Manufacturing (CAM), Computer Aided Process Planning (CAPP), and systems for purchasing, financial administration etc. In addition, the manufacturing industry typically uses Manufacturing Resource Planning systems (MRPII), [Harh86], [Voll88] to support production planning and control. These information systems are employed to improve operations of one or more individual departments and typically use self-contained architectures and incompatible heterogenous database systems. To access and manipulate this information across systems, a great deal of time and effort is needed to integrate the corresponding databases; this in practice is typically done on an ad-hoc basis. The objective of this project is to develop a systematic approach for integrating heterogeneous database systems within a chemical manufacturing facility. The facility under consideration produces bulk quantities of active ingredients used as raw materials in pharmaceutical manufacturing. Unlike discrete parts manufacturing plants, the operation of this chemical plant is continuous with processes operating in batches. Production of a batch can take from one day to several months.

Within this environment, two critical information systems were considered for integration. The first system is the Manufacturing Information Repository (MIR), which was developed internally by information systems experts, manufacturing and chemical processing engineers and data processing administrators. The second system is a

Manufacturing Resource Planning system (MRPII) proposed for the same facility. These two heterogeneous systems contain duplicate data and have some common functionality.

The object of the MIR system is to capture a data model of the actual bulk chemical manufacturing process in a common repository of information used by all levels of management and production personnel. MIR is comprehensive in its approach and contains detailed information about all equipment, materials and process operations associated with the production of bulk quantities of chemicals. It maintains an accurate account of the standard equipment used for a process (piping, pumps, flow meters, tanks, vessels, heat exchangers, centrifuges etc.) and actual equipment used during the production of each product.

MRPII systems are tools used by management to track dynamically the costs, equipment use, chemical processes (manufacturing operations and routings), chemical formulas (Bills of Materials or BOM's), inventory levels, purchase orders for raw materials, and quality control data. It will be employed for production planning and control of this chemical manufacturer. It is noted, however, that MRPII systems have typically been designed for use in discrete manufacturing facilities not chemical process plants where a routing is not so much a collection of steps but a continuous series of operations. While the MIR system was designed specifically for a chemical manufacturing facility, the proposed MRPII system is a discrete parts manufacturing resource planning system modified for the chemical manufacturing environment. Therein lies one of the challenges of this project: to integrate a process oriented database system with an MRPII system designed and developed primarily for use in a piece-parts manufacturing environment.

The MIR and MRPII systems store identical data associated with the equipment, materials, and basic process information. However, the MIR also contains myriads of additional technical information. For example, the MIR contains pressure ratings for vessels, material data safety sheets for all materials and detailed process control information such as control valve settings and valve positions. This level of information is far more detailed than what most MRPII systems need. The MRPII system on the other hand contains all of the necessary data and systems to monitor and control production costs and inventory of material. While these latter features could be added to the MIR system, they are not presently included. Similar data must be maintained within both systems. Therefore, care must be taken when determining what data is to be duplicated particularly considering the fundamental differences between the system functionality and data structures.

An iterative approach was chosen to solving the systems integration problem for which the outcome is known. The actual steps toward this outcome were determined along the way as each step of the iterative approach was evaluated. This approach is based on the Mills [Mill86] spiral method for software development where the results of each step are evaluated to decide if the problem has been solved or if more detailed analysis is required.

This thesis is structured as follows. Chapter 2 describes the research approach and literature survey. Chapter 3 addresses the first phase of the project - Analysis of Data Structures. Chapter 4 addresses the second phase of this project - Uploading Mandatory Static Information from MIR to MRPII. Chapter 5 presents a case study illustrating the uploading of sample data. And Chapter 6 includes discussion of the results and conclusions, and some discussion of future work.

2. Background and Research Approach

Substantial research has been done on the integration of database systems both within the Institute for Systems Research (ISR) of the University of Maryland and elsewhere. In this work a practical approach was developed to address the specific characteristics of the two systems being integrated.

2.1 Review of Previous and Current Research

Several research projects at the Computer Integrated Manufacturing laboratory of the ISR have addressed the use and integration of MRPII systems. Johri [Johr89] focused on the functions associated with the integration of CAD, CAPP and MRPII. Using an Artificial Intelligence data manipulation language, [1] this work dealt with maintaining data integrity between systems. While these systems are typically designed for complementary tasks within the manufacturing environment, our study deals with comparing and translating data between two heterogenous database systems that have similar functionality.

Lin [Lin91] dealt with the design and maintenance of a Knowledge Based System to control the functional relationships and information flow within a manufacturing facility. This work modeled a generic set of rules for flow of information between

[1] Update Dependencies Language developed at the University of Maryland College Park

manufacturing applications (CAD/CAPP/MRPII etc.) of a discrete parts facility using a special set of Petri nets called Updated Petri Nets (UPN). These nets were converted to general Petri Nets that were, in turn, analyzed to resolve conflicting company rules and to correct errors introduced during the modeling phase. Finally, the refined nets were translated to a knowledge base in Prolog that controls the system interactions. This work concentrated also on the dynamic control of information within the manufacturing facility assuming that the meaning of the data and semantics were consistent.

Garai [Gara91] presented a methodology for modeling established business rules for process changes within a chemical manufacturing facility. This work considered three scenarios: equipment changes, location changes, and chemistry changes, using the chemical manufacturing company's established rules for handling such changes. These rules were then converted into general Petri Nets for validation and verification. Garai's work was helpful in understanding the differences between discrete parts manufacturing facilities and the continuous nature of the processes associated with chemical manufacturing. However, it did not deal with the translation or validation of data equivalencies between database systems.

Considerable work has been reported in the literature on integration of heterogeneous database systems. While the methods of integration among these articles differ, a common theme is used throughout - the detection and resolution of semantic differences between systems are nearly impossible to achieve without manual input.

Sheth and Larson [Shet90] have conducted comprehensive research on integrating and managing the control of information between database systems. They have defined, in detail, heterogeneity due to semantics and differences in Relational Data Base Management System's (RDBMS). Semantic heterogeneity occurs when there is an incompatibility of the meaning, interpretation or intended use of the same or related data. Heterogeneity associated with differences in RDBMS's deals with differences in structures or data models, differences in constraints and differences in query languages. It is noted that the MIR and MRPII database systems have semantic differences as well as differences in data structures and data models.

Sheth and Larson specifically mentioned the difficulty in detecting semantic heterogeneity; typically Data Base Management System schemas do not provide enough semantics to interpret data correctly. The authors indicated that decoupling the heterogeneity is difficult due to differences in DBMS's from those resulting from semantic heterogeneity. For this project, we will show that the MIR and MRPII data structures differ drastically; furthermore there are several instances where the same or related data have different meanings within the two systems.

Sheth and Larson also indicated that a reference architecture is necessary to clarify the various issues within the respective DBMS's. They denoted a reference architecture to contain the following components: data, database, commands, processors, schemas and mappings. They propose the use of a Federated Database System that is a collection of cooperating but autonomous databases.

In our research we considered most of these components, laying the groundwork for a potential Federated DBMS. The data, database, schemas and mapping components were developed for both the MIR and MRPII systems. The commands (specific actions by a user) or the processors (software modules that manipulate commands and data) were not analyzed since the scope of work was limited to uploading mandatory static data from the MIR to the MRPII.

Thomas et al. [Thom90] provided insight into the types of heterogeneous distributed database capabilities available from off-the-shelf systems. While they described the fundamental aspects of schema integration, they concentrated more on query and transaction management than on the translation of data from one system to another.

Litwin et al. [Litw90] also provided valuable information about multiple database systems with concentration on dynamic activities between these systems. One important aspect of this work is the fact that they considered Oracle version 5 (the RDBMS used in this thesis) to be a multidatabase. The multidatabase approach assumes that the user needs to access multiple databases without the benefit of a global schema. This is clearly what our analysis for this project involves. As will be described in detail, the MIR and MRPII databases were both created in Oracle with three users defined, one for each respective system and the third, the author, for the analysis and integration of the data.

Chatterjee and Segev [Chat91] described in detail the structural and semantic incompatibilities of heterogeneous databases. This information was helpful when analyzing the MIR and MRPII data structures. In their work the definition of a join operator is broad, it assumes that there are no inconsistencies between common attributes, and that the actual data within the common attributes is consistent. For this project, we are interested in maintaining consistent data between the MIR and MRPII systems but first we must establish the relationships between the attributes of the respective systems.

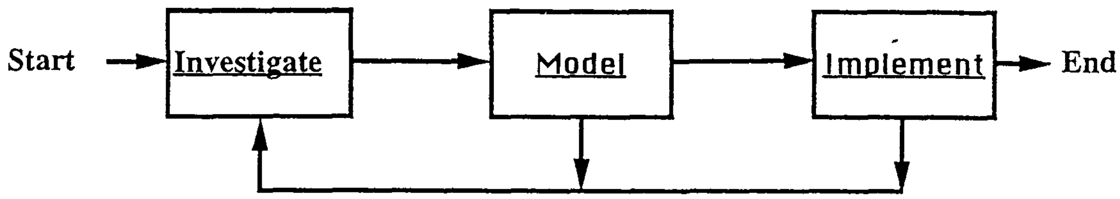
2.2 A Systems Engineering Approach

An iterative approach was chosen to address the integration of the MIR and MRPII database systems using the Mills spiral method for software development [Mill86]. Based on this method, software developers can initiate and manage the development process based on the outcome of previous activities. There are three types of loops or spirals - investigation, specification, and implementation - and each contains three sequential steps - plan, perform and evaluate. During an investigation loop, a project team gathers information and develops a policy or procedure to solve the problem under consideration. During the specification loop, the team determines the steps needed and develops a model based on the proposed solution. The project team designs and implements the proposed solution during the implementation loop. Figure 1 illustrates our interpretation of the spiral method as it applies to the integration of these two systems. The specification loop has been labeled "model" which better reflects this activity in the present work.

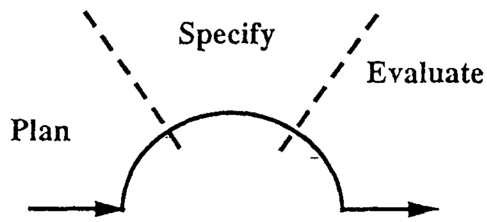
The research approach for this project incorporates two phases. Phase I involved detecting common and uncommon data elements within the data structures of the two systems. Several iterations of the investigation, modeling and finally implementation loops were done before we obtained a comprehensive comparison of the data structures. Phase II also required several iterations between modeling and implementation before the final uploading program was complete. With the MIR system assumed to be fully populated and the MRPII system void of any data, it was determined during the evaluation step of Phase I that the integration of data would be limited to only uploading mandatory static data from the MIR to the MRPII system.

It is noted that using the Mills spiral method allows a systems developer the flexibility of modifying the design and implementation process to ensure the correct solution to the correct problem. The disadvantage of using this method, however, is the potential for delay in the completion of the project, which ultimately equates to cost. Management can usually justify this added cost if it reflects in added value or improved quality.

Spiral Method:



Activities for each loop:



Phases:

1. detect common and uncommon data elements
2. establish algorithm for uploading of mandatory static information from MIR to MRPII

Figure 1 - Spiral Method for Software Development

The contributions of this project stem from the fact that it focuses on the integration between heterogeneous database systems used in a process manufacturing environment. A solution to this practical problem was developed such that the results of this project - the interrelationships inferred between these two data systems - would be essential in establishing the dynamic interaction between these systems.

The computer programs for this study were developed on a personal computer using the Pro*C interface to the Oracle Relational Database Management System. Pro*C is an extension of the C language that incorporates embedded SQL commands. The resulting source code was then precompiled into C commands that were then compiled and linked using a conventional C compiler.

3. Phase I: Analysis of Data Structures

The first step in the Mills process was to gather information about the two database systems and to compare their data structures, particularly the names of the corresponding fields or columns. This was accomplished by loading both systems into a multidatabase system. We selected Oracle version 5 (a personal computer version) for this task. To help in developing a more realistic environment, three users were defined: 'MIR', 'MRP' and 'RUSH' where the MIR and MRPII users loaded their respective data tables and RUSH developed several additional tables for comparison purposes.

3.1 Comparison of Field Names Between Systems

Figure 2 depicts the first iteration of the comparison process. Since both systems were dealing with the same type of chemical process data, it was assumed that several field names between these two systems would be identical or at least similar. Therefore, the first comparison simply divided the field names into one of three types: 1) those that matched exactly; 2) those that were exclusive to the MIR system; and 3) those that were exclusive to the MRPII system.

A computer program was developed to compare the data structures using the Oracle system table ACCESSIBLE_COLUMNS. The objective was to determine detailed information about the fields or columns, such as table name, data type and length of the

Initial Comparison

Begin with 3 catagories

- EXACT_MATCH - data fields that match exactly
- NOTMIR - those data fields found only in MRPII
- NOTMRP - those data fields found only in MIR

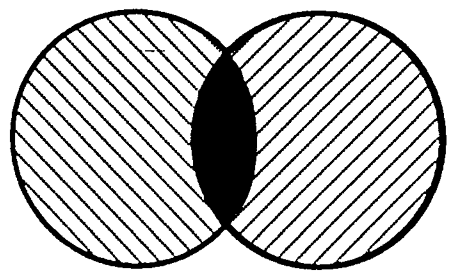


Figure 2 - Initial Comparison of Data Structures

column.

After dividing the column names into the three categories described above (exact match, exclusive to MIR or MRPII), of the seventeen hundred field names apparently used between the these two data structures, only six were exact matches (see Table I below).

At this point in our research the Mills Spiral method proved to be advantageous. The first version of the comparison software produced insignificant results and the next step to resolve the problem was not immediately known. During the evaluation step of the first implementation loop it was determined that a more refined model was needed to understand fully the data structures of the two systems. It was determined that this refined model should include two additional comparison types as indicated in Figure 3 (PARTIAL_MATCH and USER_MATCH). During the second implementation loop the comparison program was revised to include these additional comparison categories. The partial match routine prompts the user for a substring and subsequently attempts to match this substring with the column names of either system. The user match routine, on the other hand, prompts the user for two substrings: one substring to match column names from the MIR system and the other to match column names from the MRPII system. The partial match algorithm assumes that the corresponding columns have common fundamental components within the column name whereas the user match algorithm links elements that are uniquely defined. The user match algorithm also implies an in-depth knowledge of both systems. Under separate cover [Rush93] we have included the final version of the computer program that compares the column names for these two systems along with a program manual that describes in more detail the inner workings of the program.

Final Comparison

Ended up with 5 catagories

- EXACT_MATCH - data fields that match exactly
- NOTMIR - those data fields found only in MRPII
- NOTMRP - those data fields found only in MIR
- PARTIAL_MATCH - data items that match substrings
- USER_MATCH - user defined matches

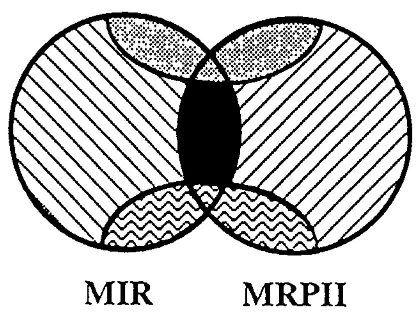


Figure 3 -Final Comparison of Data Structures

3.2 Results and Conclusion of Phase I

The results of this phase of the project are listed in Tables I, II, and III. Table I shows exact matches between field names from the two respective database systems. Of the seventeen hundred different field names within the two systems only six matched exactly. Table II shows the partial matches found between field names. With the percent character representing a wildcard for zero or more characters, the only significant partial match that was found was on "%batch%." Table III shows the results of the user defined matches. Rather than list the hundreds of corresponding field names, we have condensed the list into general categories that are coincident between the two systems. In other words, the MIR system uses process, equipment and material to describe the manufacturing of the bulk chemicals where the MRPII system uses routing, resource, items respectively. The most important issue to note is that nearly all the equivalencies are user defined.

Table I - Exact Matches of Field Names between MIR and MRPII Systems

date_added
op_code
start_date
table_name
vendor_id
(batch_no) - see partial matches below

Table II - Partial Matches of Field Names between MIR and MRPII Systems

%batch%

Table III - User Defined Matches of Field Names between MIR and MRPII Systems

MIR	MRPII
process	routing
operation	activity, operation
equipment	resource
material	item, formula

The final version of the comparison program was helpful in understanding the respective data structures. Most of the corresponding MIR and MRPII data fields were determined by user knowledge of both systems and respectively by user input. Also, comparing the column names was a practical first step when attempting to translate data from one system to another particularly when these systems appear to store similar information. Once again the Mills Spiral method proved to be very useful. During the evaluation step of the final implementation loop for the comparison software it was determined that in order to understand the database systems fully and successfully translate data from system to the other, the relationship of columns within each system and of columns and tables between systems needs to be closely examined. In order to implement the upload of data from one system to the other, a new modeling technique was needed that included these relationships. Chapter 4 describes relational aspects of these systems and the development of the algorithm to upload data from MIR to MRPII.

4. Phase II: Uploading Mandatory Static Information from MIR to MRPII

Using the comparison results from Phase I, data models were generated for the two systems and an algorithm was developed to upload information from one system to the other. Data or information can be divided into two fundamental types: static and dynamic. In the MIR system, equipment, materials, standard routings and operations are static data whereas dynamic data represents changes that occur during the actual chemical process. Additionally, static data can be further subdivided into two types; either the data is mandatory or it is not. Mandatory data is the minimum data required for proper operation of the system. Due to the complexities of the dynamic information within and between the two respective systems, in this work we limited the uploading of information from the MIR to the MRPII system to mandatory static information only. The results of Chapter 3 provided general guidelines concerning how the two respective database systems compared. Three issues needed to be resolved to make this translation work: (1) develop detailed data models for the MIR and MRPII systems; (2) establish mapping rules (numeric to character, character to numeric etc.) between column names and tables in the MIR and MRPII data structures; and (3) establish a procedure for uploading the data.

4.1 MIR Data Model

The MIR system has four major components: Process Standards, Equipment, Materials and Process Actuals. Figure 4 is an entity relationship diagram for the MIR that shows how these four components interact: Process Standards use Equipment, Process Standards use Materials, and Process Actuals are made from Process Standards. There is a m-n relationship between Process Standards and Equipment and Process Standards and Materials while there is a 1-m relationship between Process Standards and Process Actuals; i.e., there are multiple standard processes that use many different pieces of equipment and many materials; for each standard process there is one or more actual processes. Several tables from the Process Standards group and all the tables from the Process group were not relevant when uploading mandatory static data from the MIR to the MRPII, therefore, these tables were not included in this analysis.

Figure 5 presents a more detailed look of the MIR data structure that is hierarchical. This figure is a natural extension of the entity relationship diagram - it shows how the tables within the MIR structure are interconnected. A brief explanation of this interconnection follows:

1. For each standard process (SP) there are one or more standard process steps (SPS).
2. For each SPS there are one or more standard process step revisions (SPSR).

MIR E-R Diagram

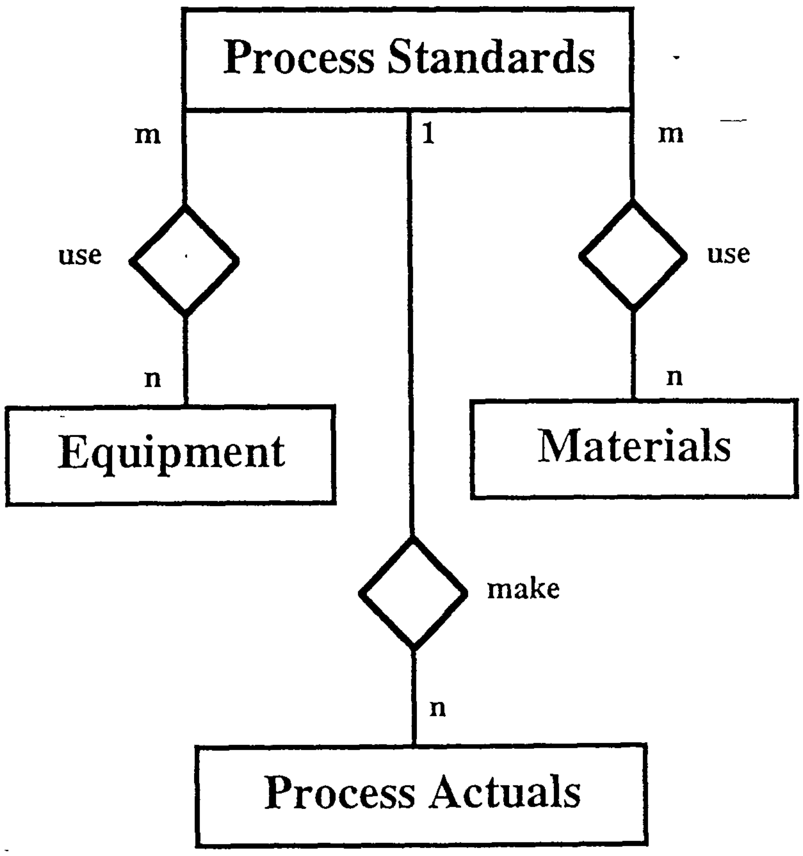


Figure 4 -MIR Entity Relationship Diagram

3. For each SPSR there are one or more standard operations (SO).

4. Each SO involves either:

    a. the movement of a product from one vessel to another via a stream (STRM) or

    b. a chemical reaction of one or more products within a vessel; the relevant information is stored in the STANDARD_OP_EQUIPMENT (SOE) and STANDARD_OP_MATERIALS (SOM) tables.

5. For each STRM there are one or more pieces of equipment and one or more materials associated with the movement of the product; the relevant information is stored in the SOE and SOM tables.

6. For each piece of equipment described in the SOE table relevant information or properties of the material within the vessel (temperature, pressure etc.) are stored in the STANDARD_OP_PROPERTIES (SOP) table.

7. For each piece of equipment there is additional information found in the EQUIPMENT_BASIC_INFORMATION (EB) table.

8. For each material described in the SOM table there is material data found in the MATERIALS_BASIC (MB) table.

MIR - Data Structure

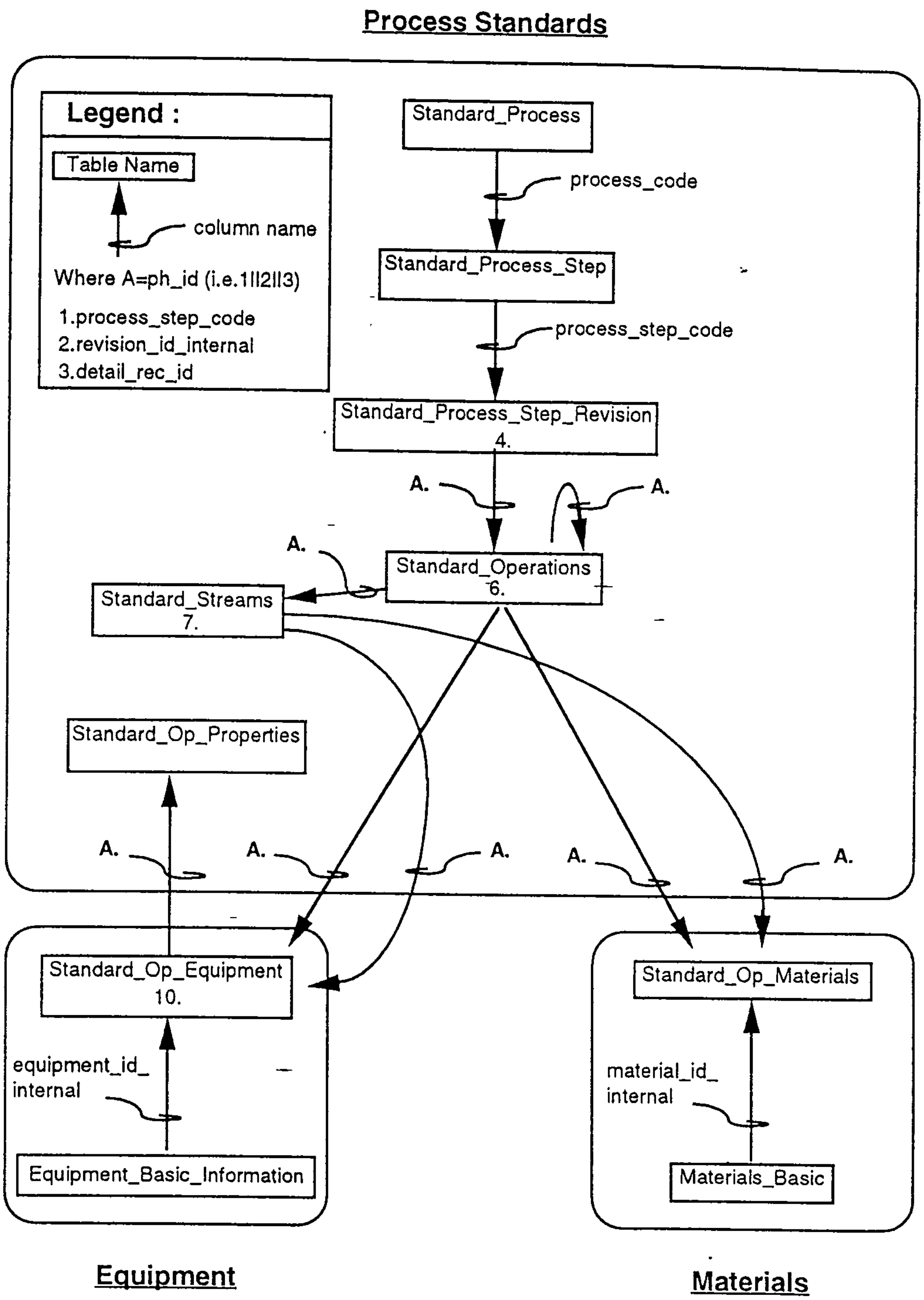


Figure 5 - MIR Data Structure

The relationship between Process Standards and Equipment is embodied in the SOE table while the relationship between Process Standards and Materials resides in the SOM table. There is no direct relationship between the equipment and material used during an operation - it is implied through the standard process.

Since this work deals with uploading mandatory static data, at first glance there is no interest in the Process Actuals tables. It is important to note, however, that the Process Actuals tables are duplicates of the Process Standards tables. Process Standards are intended to be continuously modified to reflect current requirements for the production of bulk quantities of active ingredients. These active ingredients are then used as raw materials in pharmaceutical manufacturing facilities. Process Actuals are not intended to reflect revisions and modifications made to the standards, their purpose is to show actual information associated with the batch manufacturing processes - actual equipment used, temperature and pressure of vessels, quantities of materials transferred, consumed and produced. One major role of the Process Actuals tables is the ability to record the actual production conditions should this information ever be needed. Since the structures of the Process Standards and Process Actuals tables are identical, the Process Actuals tables have identical relationships to the Equipment and Materials tables as the Process Standards tables do. A natural extension of this project would be to take the interrelationships inferred between the MIR and MRPII systems and the developed upload algorithm and use them to track the actual process information dynamically.

The hierarchy of the system is implemented through the use of several common fields or pointers within the data structure. The first such pointer is the field ph_id (which stands for process header id) and is the concatenation of the process_step_code (alphanumeric), the revision_id_internal (numeric) and the detail_rec_id (numeric). Figure 5 indicates that the field ph_id is the link between the tables from the SPSR table down to the SOE, SOM and SOP tables. Of the three fields that comprise ph_id, the detail_rec_id field is the only field that changes for all the operations within a process step; the process_step_code and revision_id_internal remain the same throughout the process step. Revisions are made to the process by modifying the SPSR table and rippling this change down through all associated tables.

The tables SPSR, SO, STRM, SOE, SOM, and SOP each have two other fields that contribute to the hierarchy - parent_id and parent_table_id. Parent_id is similar to the ph_id in that it is the concatenation of three fields, process_step_code, revision_id_internal and detail_rec_id_parent; the latter is the detail_rec_id of the parent table. The parent_table_id field is an internal numerical indication of the parent table and can be found in figure 5 just below the table name (i.e., SPSR=4, SO=6, STRM=7 etc.). All the numeric fields mentioned above effectively act as pointers and are used to navigate throughout the data structure. For example, consider a set of standard operations (SO) for a particular SPSR. For each SO, the parent_id field would contain the ph_id of the SPSR parent and the parent_table_id field would contain "4" indicating that the parent is an SPSR.

An interesting fact to note is that the MIR was developed to handle recursion on standard operations. This was introduced to accommodate modifications at various levels of detail within the process definition. For example outer levels of the SO table contain more macro instructions such as "prepare intermediates" while inner recursive levels describe micro operations such as changing valve settings. This recursion is accomplished by using the pointers as described above. While the MIR has the ability of defining and maintaining several levels of detailed operations, a typical MRPII system will simply have a sequential listing of the operations associated with a routing or process.

A sample of MIR data (used for the case study in Chapter 5) is included in Appendix B. The sample data was restructured according to a depth first search on the SO table to help understand the hierarchy of the MIR system. The results of this restructuring, included in Appendix C, illustrate the inherent recursion.

4.2 MRPII Data Model

The MRPII system used for this project is a conventional one such as those used in a discrete parts environment, and was modified slightly to adapt to a process manufacturing environment. The system has eight different modules as listed below:

1. CM - cost management
2. CR - capacity resource
3. FM - formula management

4. IC - inventory control
5. OP - operations
6. PM - production management
7. PO - purchase orders
8. QC - quality control

Analysis of the system shows that only three of these modules pertain to static data - CR, FM and IC - all others' deal with the dynamic exchange of information. The CR module handles all the information associated with the equipment or resources available. The IC module deals with the materials or Part Master records. The FM module contains the following: i) all the tables that define the routings; ii) operations that use the equipment and materials; and iii) the formulas or Bill of Materials (BOM) for the final products.

The MRPII system has a few more tables and interrelationships than the MIR system. A brief explanation of the entity relationship diagram (figure 6) is as follows:

1. Routings have operations and formulas;
2. Operations consist of activities and use resources;
3. formulas use items.

MRPII E-R Diagram

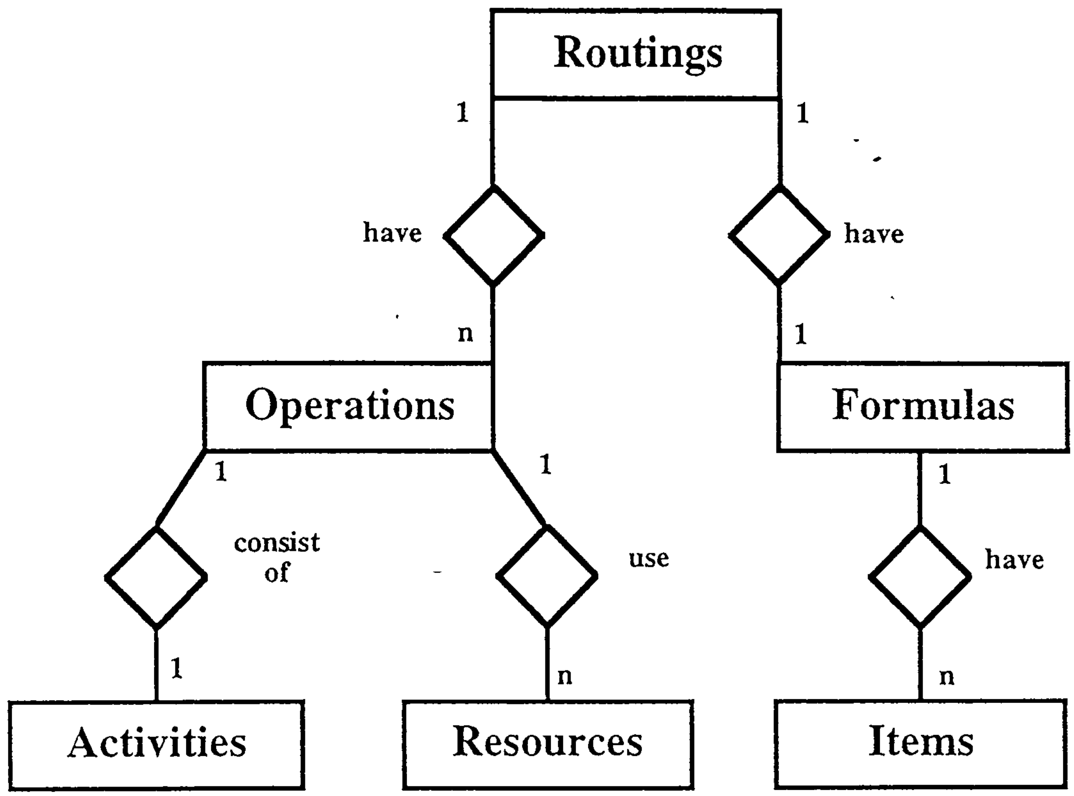


Figure 6 - MRPII Entity-Relationship Diagram

Figure 7 provides a more detailed look at the MRPII data structure. Here the data structure is not top down and hierarchical, but bottom up and relational. Figure 7 is a natural extension of the entity relationship diagram and shows, in more detail, the interconnection between the tables within the MRPII system. The table names for the MRPII system use the following convention: the first two characters represent the module where the table is found (as previously listed), the next four characters represent the sub-modules and the last three characters indicate the specific contents of the table. Listed below are some definitions associated with the abbreviations used for the table names.

ACTV	- activity	MATL	- material
DTL	- detail	MST	- master
EFF	- effective	MTL	- material
FORM	- formula	OPRN	- operation
HDR	- header	ROUT	- routing
ITEM	- item or material	RSCR	- resource or equipment

The Items module has only one table that is of immediate interest - inventory control item master, IC_ITEM_MST. The Resources module has two tables that are of interest - capacity resource master, CR_RSRC_MST, and capacity resource detail CR_RSRC_DTL. These two modules are fundamentally equivalent to the materials and equipment tables of the MIR data structure.

MRPII - Data Structure

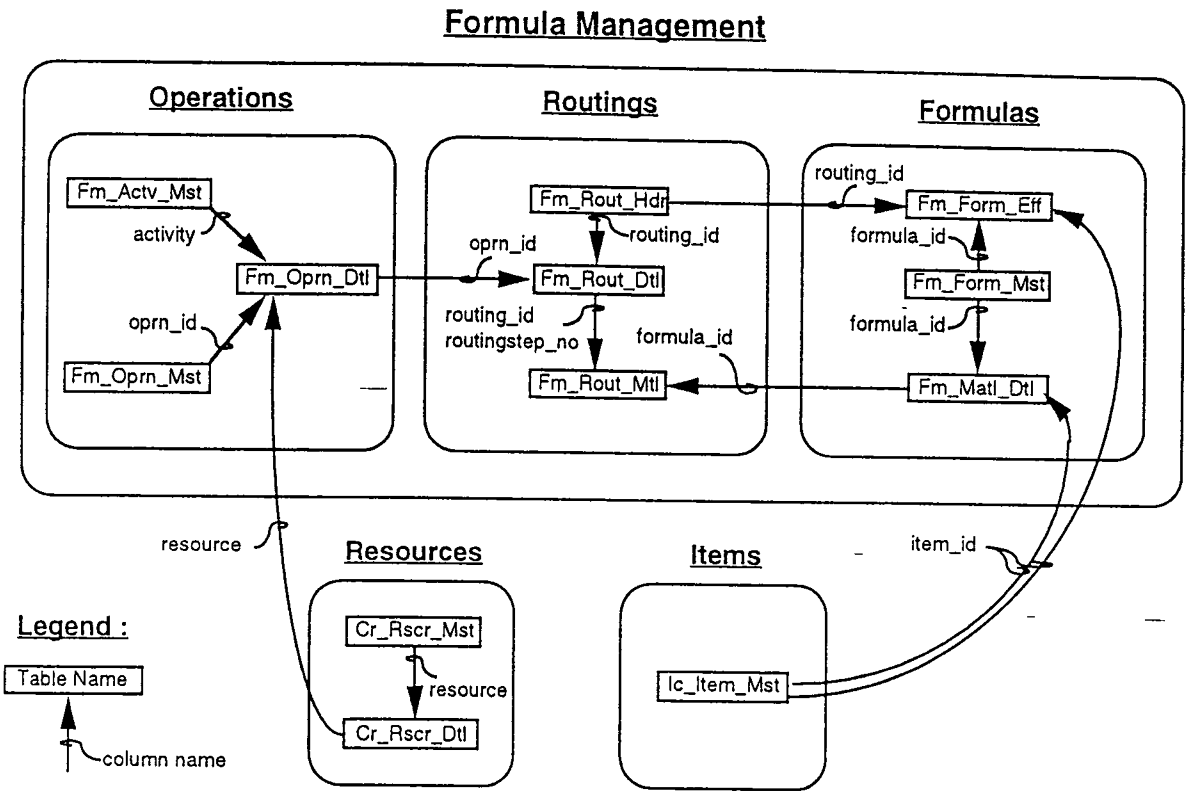


Figure 7 - MRPII Data Structure

The Formula Management module is more complicated. It has three sub-modules Operations, Routings and Formulas. The Operations sub-module links activities with resources. The Formulas sub-module associates the items used in a formula. Finally, the Routings sub-module links the operations and the formulas. A brief explanation of this interconnection follows.

From Routings to Resources:

1. For each routing there are one or more routing steps. In addition, each routing has one or more versions.
2. For each routing step there is one operation. In addition, each operation has one or more versions.
3. For each operation there is an activity and a resource number.

From Routings to Item:

1. For each routing there is one formula and one item (the first product). In addition, each formula has one or more versions.
2. For each formula there are one or more items (ingredients, products, by-products).
3. For each formula there are one or more associated routing steps.

Unlike the MIR data structure, these tables do not use pointers to link the respective components of this system. Figure 7 shows how common attributes are used to link these tables as explained below.

From Routings to Resources:

1. The routings table FM_ROUT_DTL (formula management routing detail) links the routings with the operations using the field oprn_id.
2. The operations table FM_OPRN_DTL (formula management operation detail) links the activities, operations and resources using the fields activity, oprn_id and resource respectively.

From Routings to Items:

1. The formula table FM_FORM_EFF (formula management formula effective) links the routing with a formula and an item using the fields routing_id, formula_id and item_id, respectively.
2. The formula table FM_MATL_DTL (formula management material detail) links the formula to the items using the fields formula_id and item_id.
3. The routings table FM_ROUT_MTL (formula management routing material) links the routing and routing steps to the formula using the fields routing_id, routing_step_no and formula_id.

Similar to the MIR there is no direct relationship between the resource (equipment) and item (material) used. Here, four tables are used to imply this relationship: FM_OPRN_DTL, FM_ROUT_DTL, FM_ROUT_MTL and FM_MATL_DTL.

There are several tables within the MRPII system that contain revision information; the system calls this information versions. For purposes of uploading data from MIR to MRPII this difference does not pose an immediate problem. However, when modifications are made to the chemical process, each system would handle these modifications differently. Consider a change to a routing, operation, or formula. To update the MIR structure, all the tables linked by the field ph_id have to be updated; on the other hand, only the respective module (operations, routings or formulas) within the MRPII system would need to be changed.

The MRPII system is driven by the items within the formula or bill of materials (BOM). Each intermediate product and the final product in the BOM has a routing that describes the operations necessary for production. The BOM resides in the FM_MATL_DTL table where there are ingredients, products or by-products for each formula. The relationship between the different elements of the BOM is established with the field formulaline_id which shows if the specific item listed was a product of an earlier formula. For example, if formula_id 100 produced item C, and formula_id 200 uses item C as an ingredient, then a row in the table FM_MATL_DTL would contain the following information: formula_id = 200, item C is an ingredient and formulaline_id = 100.

Similarly, each routing has a set of routing steps, and for each step there is an operation. The routing steps are sequential and are based on the manufacturing steps necessary to make the product. The table FM_ROUT_DTL, which contains the routing steps (routingstep_no) and the operations (oprn_id), also has a field called routingstep_id. This field is similar to the formulaline_id described above; it represents the routing_id of the preceding routing. For example, if routing_id 100, routingstep 20 produced item C, and routing_id 200, routingstep 10 used C as an ingredient, then a row in the table FM_ROUT_DTL would contain the following information: routing_id = 200, routingstep_no = 10 and routingstep_id = 100.

This method also applies to the operations module. The table FM_OPRN_DTL contains the activity and resource associated with the operation. It also has a field called oprnline_id. This field contains the oprn_id of a preceding operation that produced the product being used in this operation. For example, if operation 10, which produces item C, was "COOK X IN TANK Y" and operation 20 was "MOVE ITEM C TO TANK Z," then a row in the table FM_OPRN_DTL would contain the following information: oprn_id = 20, activity = move, resource = TANK Z, and oprnline_id = 10.

If this MRPII system were to accommodate the recursion aspect of the MIR system, the necessary procedures would use the information in the routings module simply because of the 1-1 relationship between routing steps and operations. However, the source code of the MRPII system would have to be modified for the system to handle recursion.

In conclusion, the MRPII system is bottom-up in nature: resources and items are established first; resources are used within operations; items are parts of formulas; and finally routings contain operations and formulas, both of which use items. The MRPII seems similar to the MIR in that it is process-oriented. However, the MRPII is not hierarchical but relational in nature.

4.3 Mapping Conventions

To load mandatory static information from the MIR to the MRPII, only three data types are of concern; numeric (num), character (char) and floating point (float). The mapping conventions listed below are appropriate for the uploading of data in one direction (MIR->MRPII). If it were necessary to download data from the MRPII to the MIR system a new set of rules would have to be established.

The mapping conventions are as follows:

1. For num, char and float data types for which the MRPII data field is of equal or greater length than the MIR data field, no modification is required.

2. If the MIR num or float (i.e., num(10,3)) field were larger than the MRPII num field, the least significant digits of the MIR field were selected for uploading purposes. There is one major disadvantage to this rule - if by some chance two mutually exclusive chemical processes contain the same

least significant digits within the MIR field there exists a possibility of conflict. For the sample data given this did not present a problem but would have to be examined carefully for future development of the translation of information.

The following example illustrates this rule:

> The value of the MIR field MB.material_id_internal is to be entered into the MRPII field routing_id within the table FM_ROUT_HDR. Material_id_internal is a number field of length nine and routing_id is a number field of length four. The value of the sample data set from the MB table is 122000755. Here, routing_id is assigned the value 755 in the table FM_ROUT_HDR and in all other tables that refer to this routing.

3. If the MIR character fields are longer than the MRPII character fields some translation or abbreviation must take place. In our study, most of the corresponding character fields did not fall into this case; for those that did, the actual data was of sufficient size not to cause a problem in the MRPII character fields. However, future work would be needed to either convert text and somehow shorten the contents without losing any of the meaning

of the data within this data field, or prompt the user for input where this mismatch occurred.

Example of character data conversion:

> Assume that the field amount_units in the MIR SOM table had a value of "POUNDS" or "GRAMS." The contents of this field are to be entered into the MRPII field item_um within the table FM_ROUT_HDR. The item_um field is a character field of length four. The equivalence rules would need to contain "POUNDS" -> "LBS" and "GRAMS" -> "GMS" to ensure data integrity.

4.4 Upload Algorithm

After several iterations of modeling and implementation loops, a model of what data to upload and where to place this data was developed as depicted in figure 8. The MRPII data fields that are listed on the right-hand side of the figure are the mandatory/static fields necessary for the MRPII system. The MIR data fields shown on the left side are the fields that contain the appropriate data to be uploaded to the MRPII system along with the data fields that were necessary to analyze the MIR data structure. The lines with arrows on the ends indicate what MIR data gets uploaded and where in the

MRPII it is to be placed. In the case that a line is broken, a number within circles has been placed at both ends across the break to signify a continuation of the line. For example, the MRPII field routing_id from the FM_ROUT_HDR table is to be populated by the data from the MIR field material_id_internal from the MATERIALS_BASIC table. The line that indicates this connection is broken with the number 1 in circles at both ends.

For the MRPII fields process_qty from the FM_OPRN_DTL table and process_qty_um from the FM_OPRN_MST table (process quantity and process quantity unit of measure respectively), the broken line has two numbers shown side by side. This indicates that the MIR data will come from one of two possible sources. The MRPII data field process_qty_um will be populated with either the MIR data from the SOM.amount_units field (as shown with the circle 4) or the SOP.value_units field (as shown with the circle 6). In the same fashion, the MRPII data field process_qty will either be populated with the MIR data from the SOM.amount_standard field or the SOP.value_standard (as shown with the circles three and seven respectively). The source of the corresponding data for these MRPII data fields is determined as follows: if an operation is anything other than a CHARGE or TRANSFER, then the process quantity and unit of measure are found in the SOP table; otherwise this information is found in the SOM table.

MANDATORY STATIC DATA to UPLOAD from MIR to MRPII

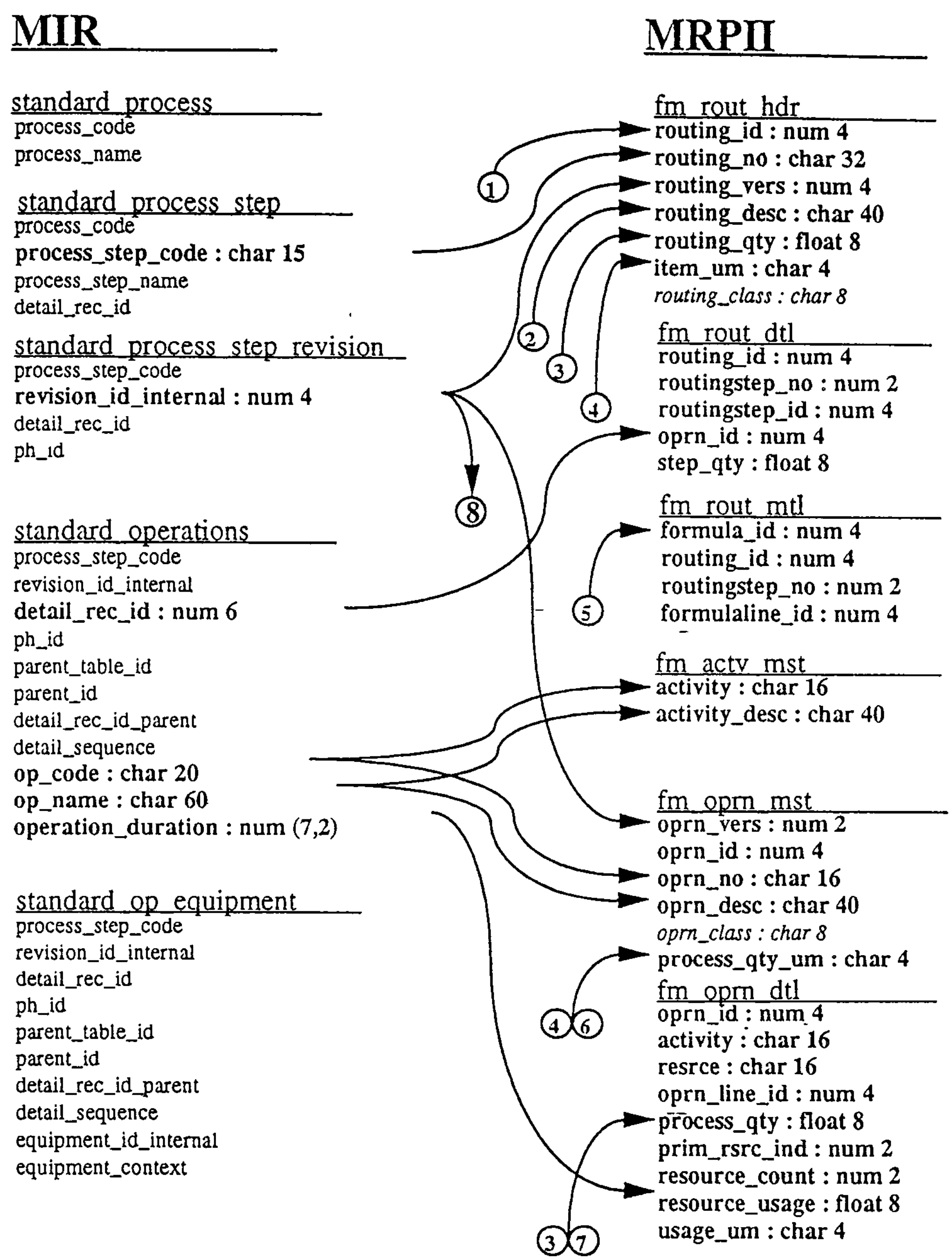


Figure 8 - Mandatory Static Data to Upload from MIR to MRPII

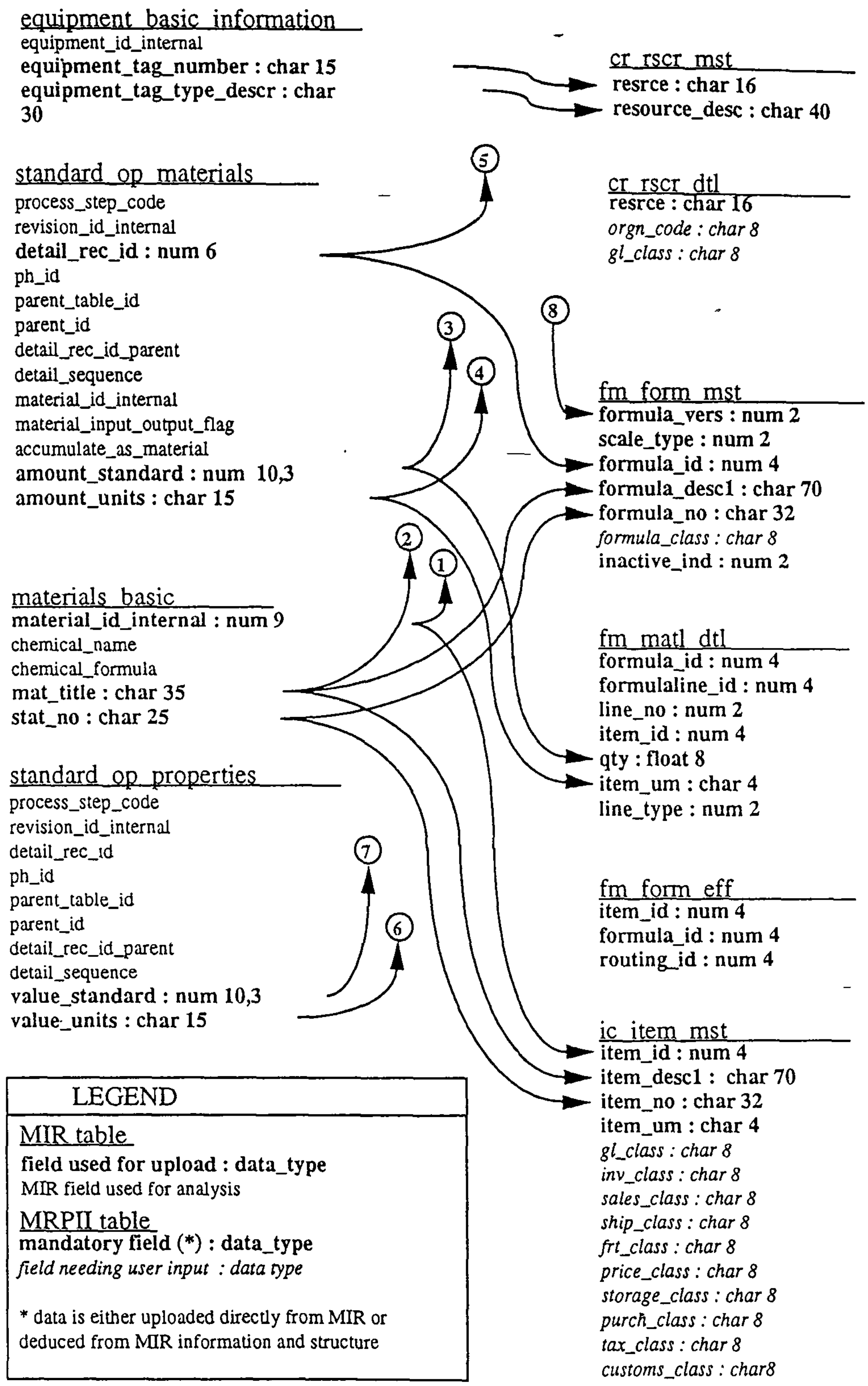


Figure 8 - (cont.)

As indicated in figure 7, the MRPII tables are linked by common fields that must contain identical data to preserve the relational characteristics of the system. In figure 8, for each of the common MRPII data fields, only one of them has an arrow drawn from the corresponding MIR field. To maintain the integrity of the MRPII system, all other common fields of the MRPII system will receive the same appropriate data according to the upload algorithm developed. The appropriate arrows to the other common MRPII data fields have not been shown to avoid unnecessary confusion. For example, according to figure 8, the routing_id field in the FM_ROUT_HDR table is to be populated with the data from the MB.material_id_internal field; the same data will be loaded into the routing_id fields of the FM_ROUT_DTL and FM_ROUT_MTL tables; however, figure 8 only shows the arrow to the FM_ROUT_HDR table. As a second example, the oprn_id field for the FM_ROUT_DTL table is to be populated with the data from the SO.detail_rec_id field; the same data will be loaded into the oprn_id fields of the FM_OPRN_MST and FM_OPRN_DTL tables with figure 8 only indicating the arrow to the FM_ROUT_DTL table. It is important to note that the links between MIR and MRPII data fields were developed concurrently with the upload algorithm described below. Refer to the case study in Chapter 5 for details of the application of the upload algorithm.

Using all the above information, the final implementation loop produced an algorithm and computer program that uploads data from the MIR to the MRPII. This algorithm is listed in Appendix A and performs the upload of information in one pass; it reads the MIR data once, stores all the relevant data in program variables and inserts this data into the appropriate MRPII table(s).

Using the Mills Spiral method it was determined during the evaluation step of the final implementation loop that the best way to illustrate how this algorithm and computer program worked was to apply it to a sample set of data. Investigation, modeling and implementation loops were done to respectively get the sample data, model how the two systems interpreted the data and apply the algorithm and computer program to upload the data.

A series of SQLPLUS macros was written to query the respective data systems. SQLPLUS is Oracle's interactive interpreter that interprets and executes SQL commands entered manually by the user. Depending on the privileges that the user has, SQLPLUS is an excellent tool for developing SQL commands to be used in Pro*C programs. The interpreter provides instantaneous results whereas the effect of the same statements within a Pro*C program would not be known until the program is successfully compiled and run.

Besides the SQL commands described above, several SQLPLUS macros were created to traverse the MIR data structure and investigate the interrelationships of the MIR data structure. The representative macro, a depth first search on the recursion of the STANDARD_OPERATIONS (SO) table, is listed in Table IV. This macro selects the SO information by recursively searching the MIR data structure for connections from parent to child by using the "CONNECT BY PRIOR" clause. This clause matches the detail_rec_id of one level in the SO table with the detail_rec_id_parent of the previous level. This macro was essential to both the understanding and the verification of the MIR structure.

All SQL statements included in the final program were first developed in the SQLPLUS environment. One such example is included in Table V that shows how the hierarchical link is made between the SPSR and SO tables. The final SQL upload program is included in [RUSH93].

Table IV - SQL Statement for Standard_Operation Hierarchy

```
SELECT              DETAIL_REC_ID, DETAIL_REC_ID_PARENT,
                    OP_CODE, OP_NAME
FROM                DR_STANDARD_OPERATIONS
CONNECT BY PRIOR    DETAIL_REC_ID = DETAIL_REC_ID_PARENT
START WITH          DETAIL_REC_ID = 3
;
```

Table V - Typical SQL Statement that Links MIR Tables

```
SELECT     OP_CODE,OP_NAME,DETAIL_REC_ID
FROM       DR_STANDARD_OPERATIONS SO
WHERE      DETAIL_REC_ID_PARENT IN
               (SELECT   DETAIL_REC_ID
               FROM      DR_STANDARD_PROCESS_STEP_REV SPSR
               WHERE     SO.DETAIL_REC_ID_PARENT=
                              SPSR.DETAIL_REC_ID
               )
;
```

5. Case Study

To illustrate how the program described in Chapter 4 uploads data from the MIR to the MRPII system, an example of a typical fermentation process is used. The corresponding MIR data had to be modified slightly to conform to the mapping conventions described in Chapter 4; the modified data was then used to populate the corresponding MRPII tables.

5.1 MIR and MRPII Models of a Fermentation Process

Figure 9 shows the sample fermentation process as described in the MIR system. MIR considers the fermentation process from the equipment point of view; i.e., it describes the process by indicating what pieces of equipment are used. For example, figure 9 shows the five storage tanks that provide the raw materials used for the fermentation which takes place in tanks 1000 and 2000, respectively. The intermediate products from tanks 1000 and 2000 are sent to tank 3000 and proceed to centrifuge 4000. The final products and by-products are sent to tank 4010 and hopper 5000.

Figure 10 illustrates the same process as described by the MRPII system. Unlike MIR, the MRPII system considers this process from a materials point of view using the bill of materials (BOM). The reactants R-A and R-B are mixed to produce intermediates I-K and I-L. Similarly, intermediates I-K and I-L in turn combine to make intermediate

SAMPLE DATA
MIR PROCESS

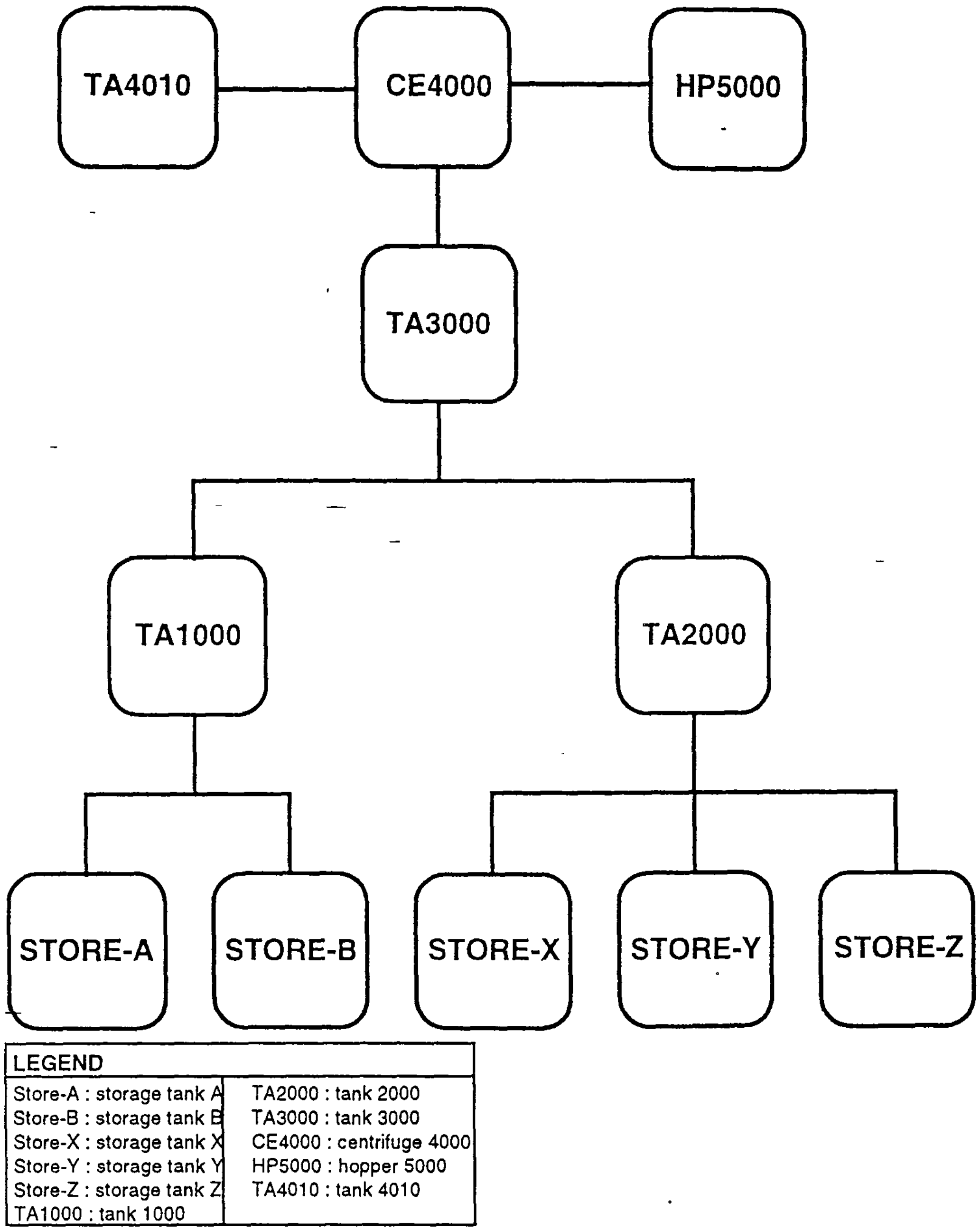
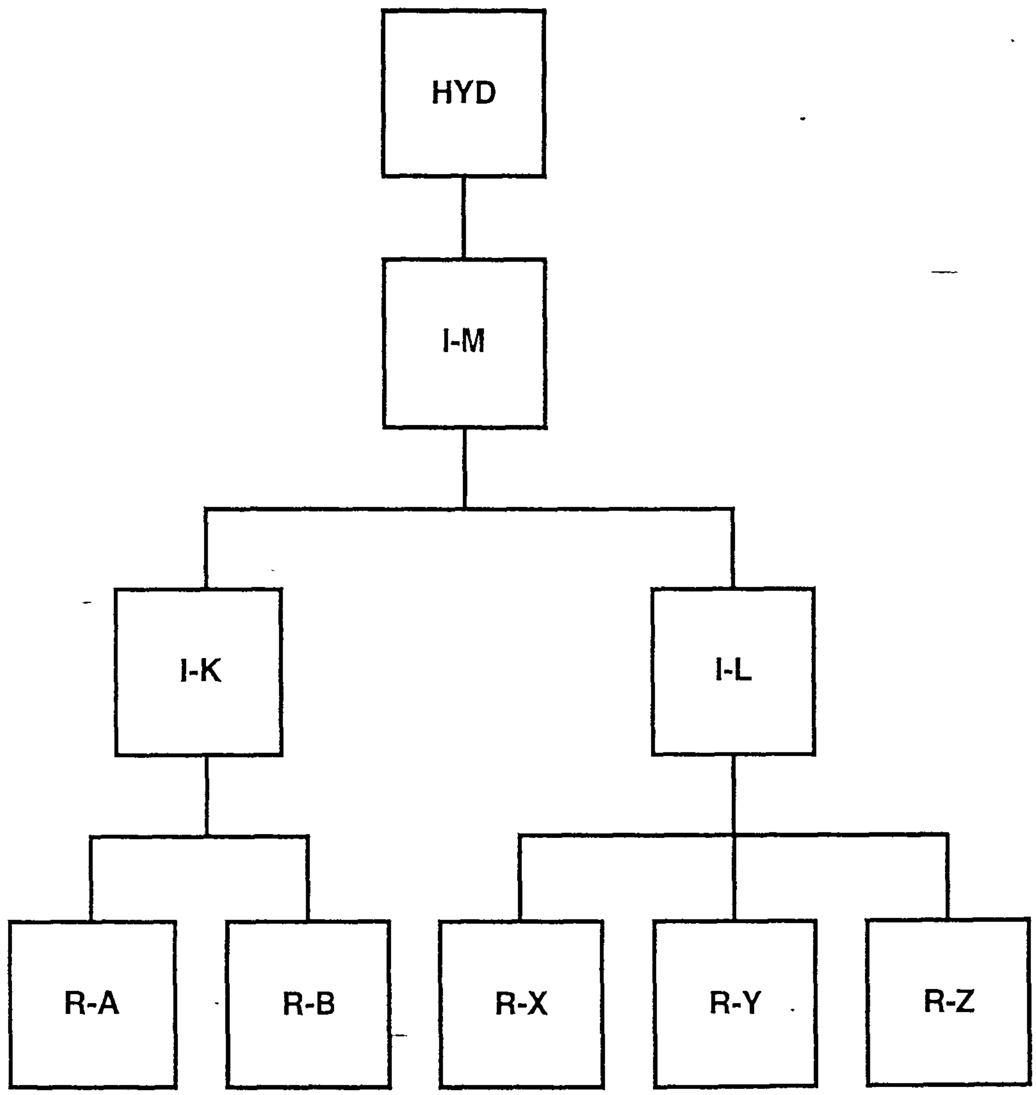


Figure 9 - Sample Data - Fermentation Process Captured By MIR

SAMPLE DATA

MRPII PROCESS



LEGEND	
R-A : reactant A	I-K : intermediate K
R-B : reactant B	I-L : intermediate L
R-X : reactant X	I-M : intermediate M
R-Y : reactant Y	HYD : hydrazime
R-Z : reactant Z	

Figure 10 - Sample Data - Fermentation Process Captured By MRPII

I-M and the final product hydrazime.

These illustrations clearly indicate that MIR is equipment / process / materials oriented and the MRPII is materials / process / equipment oriented.

5.2 Uploading of Information

Raw Data

The sample data used in this example are contained in Appendices B, C and D. Appendix B contains the MIR tables with the raw data. The latter is restructured in Appendix C to more clearly reflect the MIR data structure. The results of the upload algorithm (the populated MRPII tables) are shown Appendix D. The information displayed in Appendices B, C and D follows a nomenclature used throughout this thesis: table names are in UPPERCASE (and typically abbreviated), field names are in lowercase and are shaded, and field contents are in UPPERCASE. A detailed explanation of the data of the case study follows.

After the raw data was loaded into the respective MIR tables, each table was queried for its contents; the results of these queries are shown in Appendix B. Showing the raw data in this form is useful when trying to find individual pieces of information. However, it is very difficult and time consuming to page back and forth between the tables using the pointers as described in Chapter 4. Therefore, to make it easier to understand the MIR structure, this raw data was restructured with the results included in Appendix C. This restructuring was done manually and was not necessary to upload the data from the MIR to the MRPII. However, it was necessary to understand the interrelationships of the MIR data structure.

Restructured Data

Figure 11 is a duplication of the first page of Appendix C. It illustrates how the MIR data was restructured by traversing the MIR data structure using the pointers mentioned in Chapter 4. MIR was first queried in a top down manner from STANDARD_PROCESS (SP) down to STANDARD_PROCESS_STEP_REVISION (SPSR) (see also figure 5). The results of the SPS and SPSR tables are included in the top five rows of figure 11.

The 6th, 7th and 8th rows of figure 11 describe how to read this table and are repeated on each subsequent page of Appendix C. The columns of the "TABLE (Recur level --->)" contain the abbreviated names of the MIR tables and indicate, by indentation, the level of recursion. The "a", "b", and "c" columns describe the hierarchy of the system establishing a link between parent and child entities while the "d" column indicates the sequence of the process steps. The detail_rec_id field (column "a") is a unique internal identifier used throughout the MIR data structure. The "b" column contains (as its name depicts) the detail_rec_id of its parent. The "c" column (parent_table_id) indicates the internal numbering system associated with each table (see

STANDARD_PROCESS_STEP		STANDARD_PROCESS_STEP_REVISION	
process_code	Z-543	revision_id_internal	1
proc_step_code	Z-543C	detail_rec_id	2
proc_step_name	HYDRAZIME	pti_id	Z-543C00001000002
detail_rec_id	1		
a - detail_rec_id		c - parent_table_id	
b - detail_rec_id_parent		d - detail_sequence	

TABLE (Recur Level --->)					a	b	c	d	other	value
SO					3	2	4	1	op_code	OPERATION
									op_name	CRUDE HYDRAZIME
	SO				4	3	6	3	op_code	UNIT OPERATION
									op_name	PREPARE INTERMEDIATE
		SO			29	4	6	1	op_code	CHARGE
									op-name	CHARGE R-A TO TA1000
									duration	10
			STRM		30	29	6	1		
				SOM	31	30	7	1	matl_id	755
									amt_std	1000
									amt_units	LBS
									acc_as_matl	I
									matl_I_O_flag	I
				M_BAS	—	—	—	—	chemical_name	REACTANT-A
				SOP	32	30	7	1	val_std	1000
									val_units	LBS
				SOE	33	30	7	1	eq_id	132
									eq_context	TO
				E_BAS	—	—	—	—	eq_tag_num	TA1000
				SOP	35	30	7	1	prop_typ	C
				SOE	41	30	7	1	eq_id	145
									eq_context	FROM
				E_BAS	—	—	—	—	eq_tag_num	STORE-A
		SO			42	4	6	2	op_code	CHARGE
									op_name	CHARGE R-B to TA1000
									duration	6
			STRM		43	42	6	1		
				SOM	44	43	7	1	matl_id	756
									amt_std	600
									amt_units	LBS
									acc_as_matl	I
									matl_I_O_flag	I
				M_BAS	—	—	—	—	chemical_name	REACTANT-B
				SOP	45	43	7	1	val_std	600
									val_unts	LBS

Figure 11 - Example of Restructured Data (see Appendix C)

figure 5 - SPSR=4, SO=6 etc.). Finally, the "other" column represents the field name within the table listed to the left, and the "value" column contains the current value for the respective field.

Consider the following example. The first level of recursion in the SO table has two rows associated with it: op_code = OPERATION and op_name = CRUDE HYDRAZIME. Each abbreviated table name is listed once on the left-hand side followed by one or more rows of "other" field names and respective "values" from that table. The first SO table has a=3, b=2 and c=4 which indicates that the parent is a SPSR table (c=4) with a detail_rec_id=2. The second level of recursion on the SO table has a=4, b=3 and c=6 which indicates that the parent is an SO table (c=6) with a=3, i.e., the previous SO table. The third SO has b=4, c=6 linking it to the previous SO table. The STREAM table (a=30, b=29, c=6) is a child of the third SO table and so on. The "d" column is the detail_sequence field which enumerates the sequential operations of a process step.

The raw data was restructured by first doing a depth first recursive search on the SO table, and then searching the other MIR tables using the pointers as illustrated in figure 5 (i.e., SO->STRM, STRM->SOE, STRM->SOM etc.). The text in the last column on the right in figure 11("value") describes each process. For example, listed below is the process to produce Intermediate K.

I. For the first OPERATION "CRUDE HYDRAZIME" (a,b,c,d=3,2,4,1)

A. for the first UNIT OPERATION "PREPARE INTERMEDIATES", (a,b,c,d=4,3,6,3)

1. the first sub operation (a,b,c,d=29,4,6,1) is to "CHARGE" 1000 lbs. of REACTANT-A to tank TA1000 from storage tank STORE-A. This operation takes 10 minutes. REACTANT-A is to be accumulated in inventory (SOM.acc_as-Matl="I") and is considered an input to the process (SOM.matl_io_flag="I").

2. The second sub operation (a,b,c,d=42,4,6,2) is to "CHARGE" 600 lbs. of REACTANT-B to tank TA1000 from STORE-B. This operation takes 6 minutes and R-B is to accumulated in inventory and considered an input to the process.

3. The next sub operation (a,b,c,d=50,4,6,3) is to HEAT tank TA1000 to from 25□C to 69 □C. This operation takes 30 minutes.

4. The next sub operation (a,b,c,d=56,4,6,4) is to AGE TA1000 for 60 minutes to produce 1600 lbs. of INTERMEDIATE-K. I-K is to be accumulated in inventory and is considered to be an output for this process (SOM.matl_io_flag="O").

Upload Algorithm

Using the algorithm described in Chapter 4, the sample data was uploaded from MIR to MRPII. The detailed upload algorithm is included in Appendix A whereas a brief outline of the algorithm is shown in figure 12. Through a depth first recursive search (figure 12 item I), the upload program retrieves all of the appropriate MIR data and populates the MRPII equipment and material tables as the information is found. After retrieving all of the MIR data for the lowest level SO (figure 12 item II) the MRPII activity, operation and routing tables are populated with the appropriate data. One fundamental assumption of the upload algorithm is that there is one routing for each BOM; i.e., one routing for each process step where materials are used as input, activities or operations are done on these materials and products (and possibly by-products) are produced.

Populated MRPII Data Tables

The implementation of this case study is best illustrated by examining the populated MRPII data tables included in Appendix D. For example, consider the data associated with producing Intermediate K (I-K). Per the procedure in figure 12 all of the equipment and material data for this BOM was retrieved with the appropriate MRPII equipment and material tables populated and then the MRPII activity, operation and routing tables were populated. As described earlier, table names are in UPPERCASE, field names are in lowercase and are shaded, and field contents are in UPPERCASE. Referring to Appendix D, the FM_ROUT_HDR (formula management routing header) table indicates that the routing for I-K has a routing_id of 761. From the FM_ROUT_DTL (routing detail) table

OUTLINE of UPLOAD ALGORITHM

I. RETRIEVE MIR DATA:
Retrieve MIR data using depth first recursive search (from SP, SPS, SPSR, recursive on SO to STRM, SOE, SOP, EQ_BAS, SOM, and MATL_BAS - see figure 5 for MIR Data Structure).

   A. EQUIPMENT:
   As MIR equipment data is retrieved, insert data into CR_RSRC_MST and CR_RSRC_DTL tables.

   B. MATERIAL:
   As MIR material data is retrieved:

      1. insert material data into IC_ITEM_MST and FM_MATL_DTL tables;

      2. if material is 1st product, insert appropriate data into FM_FORM_MST;

      3. look for FM_MATL_DTL.formulaline_id and if found update FM_MATL_DTL table.

II. MRP.ACTIVITY, OPERATION, ROUTING:
After retrieving all MIR data for lowest level SO:

   A. insert activity data into FM_ACTV_MST;

   B. insert operation data into FM_OPRN_MST and FM_OPRN_DTL tables;

   C. look for FM_OPRN_DTL.oprn_line_id and update FM_OPRN_DTL table;

   D. insert routing data into FM_ROUT_DTL table;

   E. look for FM_ROUT_DTL.routingstep_id and update FM_ROUT_DTL table;

   F. insert routing/material data into FM_ROUT_MTL table;

   G. look for FM_ROUT_MTL.formulaline_id and update FM_ROUT_MTL table;

   H. if routing produces 1st product, insert appropriate data into FM_ROUT_HDR and FM_FORM_EFF tables.

Figure 12 - Outline of Upload Algorithm

there are four entries for this routing with oprn_id's 29, 42, 50 and 56 respectively. The FM_OPRN_DTL (operation detail) table indicates that the activities for these four operations are CHARGE, CHARGE, HEAT and AGE. The FM_OPRN_MST (operation master) table further defines each of these operations including a detailed description (oprn_desc) with no explicit reference to any materials used in or produced by each operation.

The previous example illustrates one fundamental difference between the MIR and MRPII systems concerning the relationship between BOM's and routings. For each step in the process, the MIR system explicitly references both the equipment and the materials used. In other words, MIR combines both the routing and bill of materials into the Standard_Process. On the other hand, the MRPII system implicitly refers to the materials used in each operation via the FM_OPRN_MST.oprn_desc field. While the FM_OPRN_DTL and FM_OPRN_MST tables describe the operations within a routing, the FM_FORM_EFF, FM_FORM_MST and FM_MATL_DTL tables describe the BOM. The link between the BOM and the operations is found between the FM_ROUT_DTL table, which contains the fields routing_id and oprn_id, and the FM_ROUT_MTL table, which contains the fields routing_id and formula_id.

One of the more challenging aspects of the implementation process was the determination of the formulaline_id, oprn_line_id, and routing_step_id fields, since each required an in-depth knowledge of the interrelationships between the MRPII tables and the entire MIR structure. Following the Mills Spiral methodology, each of these routines were developed in an iterative fashion using the previous routines as templates or models. These items are shown in figure 12 as I.B.3, II.3, II.5 and II.7. Step I.B.3 shows the procedure for finding the formulaline_id in the FM_MATL_DTL table that is found at the inner most depths of the recursive search. This formulaline_id is used to associate the materials of the BOM within the MRPII structure. For each ingredient of a formula (or item_id shown in the FM_MATL_DTL table), if this ingredient is a product of a previous formula (and not a raw material), the formulaline_id field is assigned the formula_id of that previous formula. In other words, the formulaline_id field is a pointer down the BOM.

Referring to the FM_MATL_DTL table in Appendix D, the line_type field indicates whether the material is an ingredient, product or by-product (line_type=1, 2 and 3 respectively). To find the formulaline_id for a given formula item, we search the FM_MATL_DTL table looking for a match on item_id where line_type=2. The formula_id of the row where the match occurred is inserted into the formulaline_id field for the current item_id. For example, as shown in figure 10, Intermediate M (I-M, formula_id=134) has two ingredients, I-K and I-L. Searching the FM_MATL_DTL table looking for a match for item_id=761 (I-K) and line_type=2, the program found formula_id=59. The FM_MATL_DTL table shows the formulaline_id's for these two ingredients as follows. I-K, which was produced from formula_id=59, was used twice each time for 800 pounds - that is formula_id=134, formulaline_id=59, line_no=1 and

3, item_id=761, qty=800, item_um=LBS and line_type=1. I-L, which was produced from formula_id=90, was used once for 2400 pounds - that is formula_id=134, formulaline_id=90, line_no=2, item_id=762, etc. This process was continued successfully finding the formulaline_id's for all of the formulas. Table VI illustrates the SQL statement that was developed to retrieve the appropriate information for the formulaline_id field.

Table VI - SQL Statement to Find FM_MATL_DTL.formulaline_id

```
SELECT          FORMULA_ID
FROM            MRP.FM_MATL_DTL
WHERE           ITEM_ID=:matl_id_int AND
                LINE_TYPE=2;
```

The oprnline_id field in the FM_OPRN_DTL table (figure 12 item II.C) is similar to the FM_MATL_DTL formulaline_id except that it is used to link both sequential operations and materials. For each oprn_id, the oprnline_id field is assigned the oprn_id of a previous operation, if the ingredient for the current operation was a product of this previous operation. In other words, the oprnline_id field is a pointer to the previous operation. To find the oprnline_id for a given oprn_id we search the MIR data structure looking for the previous operation where the current material was produced. For example, from the FM_OPRN_DTL table in Appendix D, for oprn_id=92, the activity is CHARGE and the resource used is TA3000. Searching the MIR data structure we find that the product being charged was produced in the aging step of oprn_id 56. TABLE VII shows the SQL statement developed to retrieve the appropriate information for the oprnline_id field. This SQL statement finds the DETAIL_REC_ID (or oprn_id) of the operation that produced the material that is used during the current operation. In other words, the oprnline_id links sequential operations by way of the BOM.

Table VII - SQL Statement to Find Oprnline_id

```
SELECT      DETAIL_REC_ID
FROM        DR_STANDARD_OPERATIONS SO4
WHERE       DETAIL_REC_ID IN (
     SELECT      DETAIL_REC_ID_PARENT
     FROM        DR_STANDARD_OP_MATERIALS SOM2
     WHERE       SOM2.MATERIAL_ID_INTERNAL=:matl_id_int
                 AND
                 SOM2.MATERIAL_INPUT_OUTPUT_FLAG='O'
                 AND
                 SO4.DETAIL_REC_ID!=:so_det_rec_id
     );
```

Once the oprnline_id has been established the routingstep_id field for the FM_ROUT_DTL table can be determined (figure 12 item II.E). The routingstep_id field is a pointer to the previous routing, establishing a link between sequential routings. To find the routingstep_id for a given routing_id, we search the already populated MRPII data structure looking for a match between the current oprn_id from the routing table and the oprnline_id found in the FM_OPRN_DTL table. TABLE VIII shows the SQL statement developed to retrieve the appropriate information for the routingstep_id field.

Table VIII - SQL Statement to Find Routingstep_id

```
SELECT        ROUTING_ID
FROM          MRP.FM_ROUT_DTL FM1
WHERE         FM1.OPRN_ID =
     (SELECT       OPRN_LINE_ID
     FROM          MRP.FM_OPRN_DTL FM2
     WHERE         FM2.OPRN_ID=:so_det_rec_id)
;
```

For example, from the FM_ROUT_DTL table in Appendix D, for routing_id=763 and the oprn_id=92, we find the oprnline_id=56 from the FM_OPRN_DTL table that gives us routingstep_id=761. This indicates that the routing that preceded this routing was routing_id=761. This same process was repeated successfully finding all appropriate routingstep_id's.

Once the routingstep_id has been established the formulaline_id field for the FM_ROUT_MTL table can be determined (figure 12 item II.G). This formulaline_id is similar to FM_MATL_DTL.formulaline_id field except that this field is used to link both the formulas and routings. In other words, the formulaline_id for the FM_ROUT_MTL table is a pointer to a previous sequential formula. TABLE IX shows the SQL statement developed to retrieve the appropriate information for the FM_ROUT_MTL.formulaline_id field.

Table IX - SQL Statement to Find FM_ROUT_MTL.formulaline_id

```
SELECT    FORMULA_ID
FROM      MRP.FM_ROUT_MTL FM3
WHERE     FM3.ROUTING_ID IN (
     SELECT    ROUTINGSTEP_ID
     FROM      MRP.FM_ROUT_DTL FM4
     WHERE     FM3.ROUTING_ID
                    =FM4.ROUTINGSTEP_ID AND
          FM3.ROUTINGSTEP_NO=FM4.ROUTINGSTEP_NO AND
          FM4.OPRN_ID=:so_det_rec_id
          )
     );
```

The upload program will successfully transfer mandatory static data from MIR to the MRPII system if the user of the program is not only very familiar with both data structures but also with the process steps contained within the MIR system. This program also lays the ground work for dynamic integration between these systems.

6. Discussion and Conclusions

6.1 Discussion

Phase I - Detect Common and Uncommon Data Elements

Comparing data structures for common field names and data types can be beneficial when the data elements in both systems are very similar. However, there may be a distinct difference in the nomenclature between the MIR and MRPII systems examined in the work (see Tables I, II and III). The results of the first phase of the project indicated that further study and development was necessary before data could be successfully uploaded from one system to the other.

Phase II - Uploading Mandatory Static Information from MIR to MRPII

The modeling effort of the second phase showed that the data structures of each system are unique; each has its own idiosyncracies. Some are almost impossible to replicate in the other system. For example, without modification to the MRPII source code, duplicating the recursive nature of MIR would be impossible. Fundamental reasons to include this recursion within the MRPII system should be evaluated very carefully before making any modifications to the MRPII source code.

The equivalence rules established during Phase II of the project were beneficial when uploading data from MIR to MRPII. However, a more detailed analysis would be required if this translation of data were to be done in both directions. Here, comprehensive understanding of both systems is required before the dynamic equivalences could be developed.

Perhaps the most important development made during Phase II of the project was the mapping of data from the MIR to the MRPII system (as shown in figure 8). This mapping represents the fundamental interrelationships needed to translate data from one system to another. In order for this translation to take place, however, the mapping conventions established (see 4.3 Mapping Conventions) had to be taken into account.

It is also interesting that at first glance the BOM's of the two systems look almost identical (figures 9 and 10). However, closer analysis illustrates that the MIR is equipment-oriented and the MRPII is materials-oriented. A chemical manufacturing plant could benefit from these two different points of view in their information infrastructure. However, for integration purposes and for common understanding, a single view point would best be served. This single view point could potentially be developed using a unified model or data structure that was capable of reading and writing to each respective system. However, for this to happen, all of the software features within each system would have to be duplicated in the unified system and access to each respective system would have to be controlled through the unified system. If control was not restricted and each system was allowed to operate independently, the unified model would not be capable of maintaining data integrity between systems.

6.2 Conclusions

The objective of this project was to develop a systematic approach for integrating heterogeneous database systems within a chemical manufacturing facility. The Mills spiral method for software development worked quite well for this purpose. We learned that not only does the Mills method work well for software development but it works well when modeling systems particularly heterogenous database systems that are each unique but have similar functionalities. The simple analysis of Phase I determined some commonalities between the two systems with respect to naming conventions. Although this may be a practical first step when attempting to translate data from one system to another, it is not sufficient due to semantic differences between the systems, particularly for systems whose functionalities are very similar.

Phase II - Uploading Mandatory Static Information from MIR to MRPII, was the second iteration of the Mills Spiral Method where a detailed data model for each system was developed as well as the corresponding upload algorithm. This spiral model was effective in the data translation since information about both systems was readily available. At the end of each investigation, modeling and implementation loop, the outcome was evaluated to determine the next appropriate step.

The MIR system effectively models the processes of the chemical manufacturing plant but it does not address a few of the fundamental functions associated with an MRPII system. With concentration on the equipment used during a process, the MIR assumes that process changes effect both equipment and materials. The MRPII system on the other hand, as a modified discreet parts manufacturing system, treats changes to chemical processes as if equipment or materials could be changed without effecting each other; this is not so in a process chemical manufacturing facility. Additionally, the costing, "what-if" capabilities, inventory control and master planning features of the MRPII system are not present in the MIR system. Clearly, a combination of these two systems is needed which would require that the two systems be integrated.

Integration of two heterogeneous database systems is a complicated task where most facets must be taken into account. A comprehensive analysis of each database system is required to be assured that the integration will be successful. Analysis of the data fields of the two systems, the structural and relational aspects within each system and how each system is designed to handle their respective data must be taken into account particularly if both systems have some duplicity of functionality and each must maintain the same information. The mandatory / static data that we have transferred reflects the minimum data that is necessary to ensure that these two data models match as far as chemical formulas and manufacturing processes are concerned at a given point in time. To maintain these respective systems without inconsistencies between them, the dynamic aspects must be taken into account.

Several issues surfaced during this study that would be excellent topics for further research. The most challenging one would be to use the results of this project and extend them such that these two systems could exchange information dynamically. The first step towards dynamic interaction would be the development of a common abstract data model. This data model would illustrate conceptually what data is needed to properly operate the manufacturing facility and it would immediately show what information is either redundant or insignificant.

To develop the common abstract data model, we would have to analyze the programmatic aspects of both systems including how each of them handles changes and modifications to formulas, processes, equipment and how each system reflects actual results from each manufacturing process. This would require complete access to all programming documentation, hardware and software to run both systems independently and extensive modeling on how each system handles changes to their respective databases. This common data model would potentially have some of the characteristics of both systems in it. The architecture of the integrated system would most likely resemble the MRPII system with it's costing functions, inventory control, production planning and overall control of the chemical manufacturing process while it could benefit from a stronger relationship between the equipment used and the process steps simply because of the continuous nature of a chemical manufacturing facility. As an ultimate Systems Engineering challenge, one could develop an Expert System that uses this common abstract data model and integrates the interaction between the two systems using an Artificial Intelligence program that would automatically update each respective system should either of them modify any data within the system.

Appendix A: Algorithm to Upload Data from the MIR to the MRPII

This appendix contains the Algorithm developed to upload data from the MIR to the MRPII system. This algorithm is in a pseudo code format. The main routine is a series of indented "for loops" which illustrate the retrieval of the MIR data through an iterative process for most of the MIR data tables and a recursive process on calls to the Standard Operation (SO) table. Case statements were also used to call respective subroutines that either continued the MIR data retrieval process and/or input MIR data into the MRPII data tables. The algorithm is as follows:

```
for each SP.process_code
{
    for each SPS.process_step_code where SPS.process_code=SP.process_code
    {
        for the latest SPSR.revision_id_internal where
            SPSR.process_step_code=SPS.process_step_code
        {
            /** first level of recursion on SO **/
            for each SO.op_code where SO.detail_rec_id_parent=SPSR.detail_rec_id
            {
                /** 2nd level of recursion on SO **/
                for each SO.op_code='UNIT OPERATION' where
                    SO2.detail_rec_id_parent=SO1.detail_rec_id
                        (SO.op_name = 'PREPARE INTERMEDIATES','MAIN
                        REACTION','CENTRIFUGE')
                {
                    /** 3rd level of recursion on SO **/
                    get all SO info
                    switch SO.op_code
                    {
```

```
                    case ='CHARGE' or 'TRANSFER'
                    {
                        call    strm()    where
                            STREAM.detail_rec_id_parent=SO.detail_rec_id
                    }
                    case = 'HEAT' or 'COOL'
                        call soe() where
                            STREAM.detail_rec_id_parent=SOE.detail_rec_id
                    case = 'AGE'
                    {
                        call soe()
                        call som()
                            where STREAM.detail_rec_id_parent=SOE or
                            SOM.detail_rec_id
                    }
                } /* end switch */
                call input_fm_actv()
                call input_fm_oprn(op_type)
                if acc_as_matl = 'I'
                {
                    if matl_io_flag = 'I'
                        call input_fm_rout(1)
                    else if matl_io_flag = 'O'
                        call input_fm_rout(2)
                }
                else if acc_as_matl = 'S'
                    call input_fm_rout(3)
                /* end sub operation */
            } /* end so.unit_operation */
        } /* end so.operation */
    } /* end spsr */
  } /* end sps */
} /* end sp */

strm()
{
    call soe(strm_det_rec_id)
    call som(strm_det_rec_id)
} /* end strm */

soe()
{
    get SOE info
```

```
    get EQ_BAS info where
        EQUIPMENT_BASIC_INFORMATION.equipment_id_internal =
        SOE.equipment_id_internal
    get SOP info where SOP.detail_rec_id_parent = SOE.detail_rec_id
    if SOE.equipment_context='TO' or 'FROM'
    {
        input_cr()
    } /* end SOE eq_context='TO' or 'FROM' */
} /* end soe */

som()
{
    get SOM info
    get matl_bas info
    call input_ic()
    if SOM.acc_as_matl = 'I'
    {
        if matl_io_flag = 'I'
            call input_matl_dtl(1)
        else if matl_io_flag = 'O'
            call input_matl_dtl(2)
    }
    else if acc_as_matl = 'S'
        call input_matl_dtl(3)
} /* end som */

input_cr()
{
    CR_RSRC_MST.resrce = eq_tag_num
    CR_RSRC_MST.resource_desc = eq_tag_desc
    CR_RSRC_DTL.resrce = eq_tag_num
    CR_RSRC_DTL.orgn_code = orgn_code
    CR_RSRC_DTL.gl_class = gl_class
} /* end input_cr */

input_ic()
{
    IC_ITEM_MST gets all info
} /* end input ic */

input_fm_actv()
{
    FM_ACTV_MST.activity=SO.op_code
```

```
   FM_ACTV_MST.activity_desc=SO.op_name
} /* end input_fm_actv */

input_fm_matl_dtl(line_type)
{
   FM_MATL_DTL.formula_id = 9999;

         /** find formulaline_id for FM_MATL_DTL **/
   FM_MATL_DTL.formulaline_id = FM_MATL_DTL.formula_id where
      SELECT  FORMULA_ID
      FROM    MRP.FM_MATL_DTL FM1
      WHERE   FM1.ITEM_ID=:SOM.material_id_internal     AND
              FM1.LINE_TYPE=2;
   FM_MATL_DTL.line_no = cur_line_no++;
   FM_MATL_DTL.item_id = SOM.material_id_internal;
   FM_MATL_DTL.qty = SOM.amount_standard;
   FM_MATL_DTL.item_um = SOM.amount_units;
   FM_MATL_DTL.line_type = line_type;
   if line_type==2 {
      UPDATE MRP.FM_MATL_DTL
      SET FORMULA_ID = SOM.DETAIL_REC_ID
      WHERE FORMULA_ID=9999;
      FM_FORM_MST.formula_no = SOM.stat_no;
      FM_FORM_MST.formula_vers = SPSR.revision_id_internal;
      FM_FORM_MST.formula_type = 1;        /* default */
      FM_FORM_MST.scale_type = 1;          /* default */
      FM_FORM_MST.formula_id = SOM.detail_rec_id;
      FM_FORM_MST.formula_desc1 = SPS.process_step_name;
      FM_FORM_MST.formula_class = 'FORM1';
      FM_FORM_MST.inactive_ind = 0;        /* default */
   }
} /* end input_fm_matl_dtl */

input_fm_oprn()
{
   FM_OPRN_MST.oprn_id = SO.detail_rec_id;
   FM_OPRN_MST.oprn_no = SO.op_code;
   FM_OPRN_MST.oprn_desc = SO.op_name;
   FM_OPRN_MST.oprn_vers = SPSR.revision_id_internal;
   FM_OPRN_MST.oprn_class = 'OP1';
   FM_OPRN_MST.process_qty_um = SOP.value_units;

   FM_OPRN_DTL.oprn_id = SO.detail_rec_id;
```

```
    FM_OPRN_DTL.activity = SO.op_code;
    FM_OPRN_DTL.resrce = EB.equipment_tag_number;

        /** find oprnline_id for FM_OPRN_DTL **/
    FM_OPRN_DTL.oprn_line_id = SO.detail_rec-id
      SELECT  DETAIL_REC_ID
      FROM    STANDARD_OPERATIONS SO4
      WHERE   SO4.DETAIL_REC_ID IN (
        SELECT  DETAIL_REC_ID_PARENT
        FROM    SOM
        WHERE   SOM.MATERIAL_ID_INTERNAL  =  SOM.som_matl_id  and
                SOM.matl_io_flag = 'O' and
                SO4.DETAIL_REC_ID != current detail_rec_id
    FM_OPRN_DTL.process_qty = SOP.value_standard;
    FM_OPRN_DTL.prim_rsrc_ind = 1;
    FM_OPRN_DTL.resource_count = 1;
    FM_OPRN_DTL.resource_usage = SO.operation_duration;
    FM_OPRN_DTL.usage_um =  'MIN';
} /* end input fm_oprn */

input_fm_rout(line_type)
line_type (1-ingr,2-prod,3-by-prod)
{
      /** find routingstep_id for FM_ROUT_DTL **/
    FM_ROUT_DTL.routingstep_no = rstep_no;
    FM_ROUT_DTL.routingstep_id =
      SELECT  ROUTING_ID
      FROM    FM_ROUT_DTL FM1
      WHERE   FM1.OPRN_ID = (
        SELECT  OPRNLINE_ID
        FROM    FM_OPRN_DTL FM2
        WHERE   FM2.OPRN_ID = current so_det_rec_id
        );
    FM_ROUT_DTL.routing_id = SOM.material_id_internal;
    FM_ROUT_DTL.oprn_id = :so_det_rec_id;
    FM_ROUT_DTL.step_qty = SOM.amount_standard;
    FM_ROUT_MTL.formula_id = 9999;
    FM_ROUT_MTL.routing_id = 9999;
    FM_ROUT_MTL.routingstep_no = rstep_no+5;

      /** find formulaline_id for FM_ROUT_MTL **/
    FM_ROUT_MTL.formulaline_id = FM_ROUT_MTL.formula_id
      SELECT  FORMULA_ID
```

```
        FROM      FM_ROUT_MTL FM3
        WHERE   FM3.ROUTING_ID IN (
            SELECT  ROUTINGSTEP_ID
            FROM     FM_ROUT_DTL FM4
            WHERE   FM3.ROUTING_ID     =     FM4.ROUTINGSTEP_ID     AND
                    FM3.ROUTINGSTEP_NO = FM4.ROUTINGSTEP_NO AND
                    FM4.OPRN_ID = current so_det_rec_id );
    if direction==2 {
        UPDATE MRP.FM_ROUT_DTL
        SET ROUTING_ID = SOM.MATERIAL_ID_INTERNAL
        WHERE FORMULA_ID = 9999
        UPDATE MRP.FM_ROUT_MTL
        SET FORMULA_ID = :so_det_rec_id;
        WHERE FORMULA_ID = 9999
    }
} /* end input_fm_rout */
```

Appendix B: Sample MIR Data

This appendix contains the sample MIR data used for the upload of mandatory / static information from the MIR to the MRPII system. This data is shown in tabular format with only the appropriate fields for each table listed. The format for these tables is as follows: the table name is in uppercase at the top of each respective block; the field names as in lowercase in the shaded sections and the field contents are listed below the field names are in uppercase.

STANDARD_PROCESS		
process	process_ name	plant_ name
Z-543	SCYLLAMYCIN	ZANZIBAR

STANDARD_PROCESS_STEP			
process	process_ step_code	process_ step_name	detail_ rec_id
Z-543	Z-543C	HYDRAZIME	1

STANDARD_PROCESS_STEP (cont.)				
revision_id internal_next	detail_ rec_id_next	batch_no_ internal_next	plant_ name	factory_ code
1	164	0	ZANZIBAR—	V22

STANDARD_PROCESS_STEP_REV				
process step_code	revision_ id_internal	detail_ rec_id	ph_id	process_step effective_date
Z-543C	1	2	Z-543C0001000002	30-Mar-93

STANDARD_OPERATIONS						
proc_step_code	rev_id_int	del_rec_id	del_rec_id_par	par_table	detail_sequenc	op_code
Z-543C	1	3	2	6	1	OPERATION
Z-543C	1	4	3	6	3	UNIT OPERATION
Z-543C	1	5	3	6	5	UNIT OPERATION
Z-543C	1	29	4	6	1	CHARGE
Z-543C	1	42	4	6	2	CHARGE
Z-543C	1	50	4	6	3	HEAT
Z-543C	1	56	4	6	4	AGE
Z-543C	1	61	4	6	5	CHARGE
Z-543C	1	68	4	6	6	CHARGE
Z-543C	1	75	4	6	8	CHARGE
Z-543C	1	82	4	6	7	HEAT
Z-543C	1	87	4	6	9	AGE
Z-543C	1	92	5	6	10	CHARGE
Z-543C	1	103	5	6	11	CHARGE
Z-543C	1	112	5	6	12	HEAT
Z-543C	1	116	5	6	13	CHARGE
Z-543C	1	125	5	6	14	COOL
Z-543C	1	131	5	6	15	AGE
Z-543C	1	144	3	6	6	UNIT OPERATION
Z-543C	1	147	144	6	16	TRANSFER
Z-543C	1	153	144	6	17	TRANSFER
Z-543C	1	159	144	6	18	TRANSFER

STANDARD_OPERATIONS (cont.)	
op_name	oper_dura
CRUDE HYDRAZIME	
PREPARE INTERMEDIATES	
MAIN REACTION	
CHARGE REACTANT-A to TA1000	10.00
CHARGE REACTANT-B to TA1000	6.00
HEAT TA1000 to 69 DEGC	30.00
AGE TA1000 for 60 minutes at 69 DEGC	60.00
CHARGE REACTANT-X to TA2000	9.00
CHARGE REACTANT-Y to TA2000	8.00
CHARGE REACTANT-Z to TA2000	7.00
HEAT TA2000 to 50 DEGC	10.00
AGE TA2000 for 60 minutes at 50 DEGC	60.00
CHARGE INTERMEDIATE K to TA3000	8.00
CHARGE INTERMEDIATE L to TA3000	24.00
HEAT TA3000 to 88 DEGC	31.40
CHARGE INTERMEDIATE K to TA3000	8.00
COOL TA3000 to 10 DEC	15.60
AGE TA3000 for 60 minutes at 10 DEGC	60.00
CENTRIFUGE	
TRANSFER INTERMEDIATE M to CE4000	40.00
TRANSFER HYDRAZIME to HP5000	13.00
TRANSFER SPENT FLUID to TA4010	27.00

STANDARD_STREAMS					
process_step_cod	revision_id_internal	detail_rec_id	detail_rec_id_parent	parent_table_id	detail_sequence
Z-543C	1	30	29	6	1
Z-543C	1	43	42	6	1
Z-543C	1	62	61	6	1
Z-543C	1	69	68	6	1
Z-543C	1	76	75	6	1
Z-543C	1	93	92	6	1
Z-543C	1	104	103	6	1
Z-543C	1	117	116	6	1
Z-543C	1	148	147	6	1
Z-543C	1	154	153	6	1
Z-543C	1	160	159	6	1

STANDARD OP EQUIPMENT							
process_step_code	rev_id_int	det_rec_id	det_rec_id_par	par_tab_id	det_seq	equip_id_internal	equip_context
Z-543C	1	33	30	7	1	20400132	TO
Z-543C	1	41	30	7	1	20400145	FROM
Z-543C	1	46	43	7	1	20400132	TO
Z-543C	1	47	43	7	1	20400146	FROM
Z-543C	1	48	30	7	1	20400141	VIA
Z-543C	1	49	43	7	1	20400142	VIA
Z-543C	1	51	50	6	1	20400132	IN
Z-543C	1	57	56	6	1	20400132	IN
Z-543C	1	65	62	7	1	20400134	TO
Z-543C	1	66	62	7	1	20400147	FROM
Z-543C	1	67	62	7	1	20000008	VIA
Z-543C	1	72	69	7	1	20400134	TO
Z-543C	1	73	69	7	1	20400148	FROM
Z-543C	1	74	69	7	1	20000010	VIA
Z-543C	1	79	76	7	1	20400134	TO
Z-543C	1	80	76	7	1	20400149	FROM
Z-543C	1	81	76	7	1	20400132	VIA
Z-543C	1	83	82	6	1	20400134	IN
Z-543C	1	88	87	6	1	20400134	IN
Z-543C	1	89	88	10	2	20400132	
Z-543C	1	96	93	7	1	20400136	TO
Z-543C	1	97	93	7	1	20400132	FROM
Z-543C	1	98	93	7	1	20000014	VIA
Z-543C	1	99	93	7	2	20400143	VIA
Z-543C	1	107	104	7	1	20400136	TO
Z-543C	1	108	104	7	1	20400134	FROM
Z-543C	1	109	104	7	1	20000016	VIA
Z-543C	1	110	104	7	2	20400144	VIA
Z-543C	1	113	112	6	1	20400136	IN
Z-543C	1	120	117	7	1	20400136	TO
Z-543C	1	121	117	7	1	20400132	FROM
Z-543C	1	122	117	7	1	20000014	VIA
Z-543C	1	123	117	7	2	20400143	VIA
Z-543C	1	126	125	6	1	20400136	IN
Z-543C	1	127	126	10	2	20400132	
Z-543C	1	132	131	6	1	20400136	IN
Z-543C	1	133	132	10	2	20400132	
Z-543C	1	151	148	7	1	20400139	TO
Z-543C	1	152	148	7	1	20400136	FROM
Z-543C	1	157	154	7	1	20400140	TO
Z-543C	1	158	154	7	1	20400139	FROM
Z-543C	1	163	160	7	1	20400138	TO
Z-543C	1	164	160	7	1	20400139	FROM

EQUIPMENT_BASIC_INFORMATION			
equipment_id_internal	plant_name	equipment_tag_number	equipment_tag_type_descr
20000008	ZANZIBAR	PU2001	PUMP
20000010	ZANZIBAR	PU2002	PUMP
20000012	ZANZIBAR	PU2003	PUMP
20000014	ZANZIBAR	PU3001	PUMP
20000016	ZANZIBAR	PU3002	PUMP
20400132	ZANZIBAR	TA1000	TANK
20400133	ZANZIBAR	JKT1000	JACKET
20400134	ZANZIBAR	TA2000	TANK
20400135	ZANZIBAR	JKT2000	JACKET
20400136	ZANZIBAR	TA3000	TANK
20400137	ZANZIBAR	JKT3000	JACKET
20400138	ZANZIBAR	TA4010	TANK
20400139	ZANZIBAR	CE4000	CENTRIFUGE
20400140	ZANZIBAR	HP5000	HOPPER
20400141	ZANZIBAR	PU1001	PUMP
20400142	ZANZIBAR	PU1002	PUMP
20400143	ZANZIBAR	FQ1002	FLOW TOTALIZER
20400144	ZANZIBAR	FQ2002	FLOW TOTALIZER
20400145	ZANZIBAR	STORE-A	STORAGE TANK
20400146	ZANZIBAR	STORE-B	STORAGE TANK
20400147	ZANZIBAR	STORE-X	STORAGE TANK
20400148	ZANZIBAR	STORE-Y	STORAGE TANK
20400149	ZANZIBAR	STORE-Z	STORAGE TANK
20400150	ZANZIBAR	XV1000B	VALVE
20400151	ZANZIBAR	XV1000A	VALVE
20400152	ZANZIBAR	XV2000X	VALVE
20400153	ZANZIBAR	XV2000Y	VALVE
20400154	ZANZIBAR	XV2000Z	VALVE
20400155	ZANZIBAR	XV3000K	VALVE
20400156	ZANZIBAR	XV3000L	VALVE

STANDARD_OP_MATERIALS										
process_ step_ code	rev_ id_ int	det_ rec_ id	det_ rec_ id_ par	par_ tab_ id	det_ seq	material_ id_ int	amt_ std	amt_ units	accu_ as_ matl	matl_ i/o_ flag
Z-543C	1	31	30	7	1	122000755	1000	LBS	I	I
Z-543C	1	44	43	7	1	122000756	600	LBS	I	I
Z-543C	1	53	51	10	1	122000761		LBS	W	
Z-543C	1	59	56	6	1	122000761	1600	LBS	I	O
Z-543C	1	63	62	7	1	122000758	900	LBS	I	I
Z-543C	1	70	69	7	1	122000759	800	LBS	I	I
Z-543C	1	77	76	7	1	122000760	700	LBS	I	I
Z-543C	1	90	87	6	1	122000762	2400	LBS	I	O
Z-543C	1	94	93	7	1	122000761	800	LBS	I	I
Z-543C	1	105	104	7	1	122000762	2400	LBS	I	I
Z-543C	1	118	117	7	1	122000761	800	LBS	I	I
Z-543C	1	128	126	10	1	122000763		LBS	W	
Z-543C	1	134	131	6	1	122000763	4000	LBS	I	O
Z-543C	1	149	148	7	1	122000763	4000	LBS	I	I
Z-543C	1	155	154	7	1	122000765	1300	LBS	I	O
Z-543C	1	161	160	7	1	122000764	2700	LBS	S	O

MATERIALS_BASIC		
material_id_internal	material_title	stat_no
122000755	REACTANT-A	
122000756	REACTANT-B	
122000758	REACTANT-X	
122000759	REACTANT-Y	
122000760	REACTANT-Z	
122000761	INTERMEDIATE K	IK_0101
122000762	INTERMEDIATE L	IL_0102
122000763	INTERMEDIATE M	IM_0103
122000764	SPENT FLUID	
122000765	HYDRAZIME	HYD_5555

Appendix C: Restructured MIR Data

This appendix contains the manually restructured MIR data and was created to understand the MIR data structure better. Using the pointers mentioned in Chapter 4, the MIR was first queried in a top down manner from STANDARD_PROCESS (SP) down to STANDARD_PROCESS_STEP_REVISION (SPSR). The results of the SPS and SPSR tables are included in the top five rows of the table.

The 6th, 7th and 8th rows of figure 11 describe how to read this table and are repeated on each subsequent page. The columns under the heading "TABLE (Recur level --->)" contain the abbreviated names of the MIR tables and indicate, by indentation, the level of recursion. The "a", "b", and "c" columns describe the hierarchy of the system establishing a link between parent and child entities while the "d" column indicates the sequence of the process steps. The detail_rec_id field (column "a") is a unique internal identifier used throughout the MIR data structure. The "b" column contains (as its name depicts) the detail_rec_id of its parent. The "c" column (parent_table_id) indicates the internal numbering system associated with each table. Finally, the "other" column represents the field name within the table listed to the left, and the "value" column contains the current value for the respective field.

STANDARD_PROCESS_STEP		STANDARD_PROCESS_STEP_REVISION	
process_code	Z-543	revision_id_internal	1
proc_step_code	Z-543C	detail_rec_id	2
proc_step_name	HYDRAZIME	ph_id	Z-543C00001000002
detail_rec_id	1		
a - detail_rec_id		c - parent_table_id	
b - detail_rec_id_parent		d - detail_sequence	

TABLE (Recur Level —>)					a	b	c	d	other	value
SO					3	2	4	1	op_code	OPERATION
									op_name	CRUDE HYDRAZIME
	SO				4	3	6	3	op_code	UNIT OPERATION
									op_name	PREPARE INTERMEDIATE
		SO			29	4	6	1	op_code	CHARGE
									op-name	CHARGE R-A TO TA1000
									duration	10
			STRM		30	29	6	1		
				SOM	31	30	7	1	matl_id	755
									amt_std	1000
									amt_units	LBS
									acc_as_matl	I
									matl_I_O_flag	I
				M_BAS	——	——	——	——	chemical_name	REACTANT-A
				SOP	32	30	7	1	val_std	1000
									val_units	LBS
				SOE	33	30	7	1	eq_id	132
									eq_context	TO
				E_BAS	——	——	——	——	eq_tag_num	TA1000
				SOP	35	30	7	1	prop_typ	C
				SOE	41	30	7	1	eq_id	145
									eq_context	FROM
				E_BAS	——	——	——	——	eq_tag_num	STORE-A
		SO			42	4	6	2	op_code	CHARGE
									op_name	CHARGE R-B to TA1000
									duration	6
			STRM		43	42	6	1		
				SOM	44	43	7	1	matl_id	756
									amt_std	600
									amt_units	LBS
									acc_as_matl	I
									matl_I_O_flag	I
				M_BAS	——	——	——	——	chemical_name	REACTANT-B
				SOP	45	43	7	1	val_std	600
									val_unts	LBS

a - detail_rec_id							c - parent_table_id			
b - detail_rec_id_parent							d - detail_sequence			
TABLE (Recur Level —>)					a	b	c	d	other	value
				SOE	46	43	7	1	eq_id	132
									eq_context	TO
				E_BAS	—	—	—	—	eq_tag_num	TA1000
				SOE	47	43	7	1	eq_id	146
									ea_context	FROM
				E_BAS	—	—	—	—	eq_tag_num	STORE-B
		SO			50	4	6	3	op_code	HEAT
									op_name	HEAT TA1000 TO 69C
									duration	30
			SOE		51	50	6	1	eq_id	132
									eq_context	IN
			E_BAS		—	—	—	—	eq_tag_num	TA1000
				SOP	54	51	10	1	prop_type	B
									val_std	25
									val_units	DEGC
				SOP	55	51	10	2	prop_type	E
									val_std	69
									val_unts	DEGC
		SO			56	4	6	1	op_code	AGE
									op_name	AGE TA1000 60' @69C
									duration	60
			SOE		57	56	6	1	eq_id	132
									eq_context	IN
			E_BAS		—	—	—	—	eq_tag_num	TA1000
				SOE	58	57	10	2	eq_id	
									eq_context	
			SOM		59	56	6	1	matl_id	761
									amt_std	1600
									amt_units	LBS
									acc_as_matl	I
									matl_I_O_flag	O
			M_BA		—	—	—	—	chemical_name	INTERMEDIATE K
				SOP	60	57	10	1	prop_type	C
									val_std	69
									val_units	DEGC

a - detail_rec_id	c - parent_table_id
b - detail_rec_id_parent	d - detail_sequence

TABLE (Recur Level —>)					a	b	c	d	other	value
		SO			61	4	6	5	op_code	CHARGE
									op_name	CHARGE R-X TO TA2000
									duration	9
			STRM		62	61	6	1		
				SOM	63	62	7	1	matl_id	758
									amt std	900
									amt units	LBS
									acc as matl	I
									matl_I_O_flag	I
				M_BAS	——	——	——	——	chemical_name	REACTANT-X
				SOP	64	62	7	1		
				SOE	65	62	7	1	eq_id	134
									eq context	TO
				E_BAS	——	——	——	——	eq_tag_num	TA2000
				SOE	66	62	7	1	eq_id	147
									eq context	FROM
				E_BAS	——	——	——	——	eq_tag_num	STORE-X
		SO			68	4	6	6	op_code	CHARGE
									op_name	CHARGE R-Y TO TA2000
									duration	8
			STRM		69	68	6	1		
				SOM	70	69	7	1	matl_id	759
									amt std	800
									amt units	LBS
									acc as matl	I
									matl_I_O_flag	I
				M_BAS	——	——	——	——	chemical_name	REACTANT-Y
				SOP	71	69	7	1	prop type	
									val std	800
									val units	LBS
				SOE	72	69	7	1	eq_id	134
									eq context	TO
				E_BAS	——	——	——	——	eq_tag_num	TA2000
				SOE	73	69	7	1	eq_id	148
									eq context	FROM

a - detail_rec_id	c - parent_table_id
b - detail_rec_id_parent	d - detail_sequence

TABLE (Recur Level --->)					a	b	c	d	other	value
				E_BAS	—	—	—	—	eq_tag_num	STORE-Y
		SO			75	4	6	8	op_code	CHARGE
									op_name	CHARGE R-Z TO TA2000
									duration	7
			STRM		76	75	6	1		
					77	76	7	1	matl_id	760
									amt std	700
									amt units	LBS
									acc as matl	I
									matl_I_O_flag	I
				M_BAS	—	—	—	—	chemical_name	REACTANT-Z
				SOP	78	76	7	1	prop type	
									val std	700
									val units	LBS
				SOE	79	76	7	1	eq_id	134
									eq context	TO
				E_BAS	—	—	—	—	eq_tag_num	TA2000
				SOE	80	76	7	1	eq_id	149
									eq context	FROM
				E_BAS	—	—	—	—	eq_tag_num	STORE-Z
		SO			82	4	6	1	op_code	HEAT
									op_name	HEAT TA2000 TO 50C
									duration	10
			SOE		83	82	6	1	eq_id	
									eq context	IN
			E_BAS		—	—	—	—	eq_tag_num	TA2000
				SOP	86	83	10	2	prop type	E
									val std	50
									val units	DEGC
		SO			87	4	6	9	op_code	AGE
									op_name	AGE TA2000 60' @50C
									duration	60
			SOE		88	87	6	1	eq_id	134
									eq context	IN
			E_BAS		—	—	—	—	eq_tag_num	TA2000

a - detail_rec_id							c - parent_table_id			
b - detail_rec_id_parent							d - detail_sequence			
TABLE (Recur Level —>)					a	b	c	d	other	value
			SOM		90	87	6	1	matl_id	762
									amt_std	2400
									amt_units	LBS
									acc_as_matl	I
									matl_I_O_flag	O
			M_BA		——	——	——	——	chemical_name	INTERMEDIATE L
				SOP	91	88	10	1	prop_type	C
									val_std	50
									val_units	DEGC
	SO				5	3	6	5	op_code	UNIT OPERATION
									op_name	MAIN REACTION
		SO			92	5	6	1	op_code	CHARGE
									op_name	CHARGE I-K TO TA3000
									duration	8
			STRM		93	92	6	1		
				SOM	94	93	7	1	matl_id	761
									amt_std	800
									amt_units	LBS
									acc_as_matl	I
									matl_I_O_flag	I
				M_BAS	——	——	——	——	chemical_name	INTERMEDIATE K
				SOP	95	93	7	1	prop_type	
									val_std	800
									val_units	LBS
				SOE	96	93	7	1	eq_id	136
									eq_context	TO
				E_BAS	——	——	——	——	eq_tag_num	TA3000
				SOE	97	93	7	1	eq_id	132
									eq_context	FROM
				E_BAS	——	——	——	——	eq_tag_num	TA1000
		SO			103	5	6	1	op_code	CHARGE
									op_name	CHARGE I-L TO TA3000
									duration	24
			STRM		104	103	6	1		

a - detail_rec_id							c - parent_table_id			
b - detail_rec_id_parent							d - detail_sequence			
TABLE (Recur Level --->)					a	b	c	d	other	value
				SOM	105	104	7	1	matl_id	762
									amt_std	2400
									amt_units	LBS
									acc_as_matl	I
									matl_I_O_flag	I
				M_BAS	—	—	—	—	chemical_name	INTERMEDIATE L
				SOP	106	104	7	1	prop_type	
									val_std	2400
									val_units	LBS
				SOE	107	104	7	1	eq_id	136
									eq_context	TO
				E_BAS	—	—	—	—	eq_tag_num	TA3000
				SOE	108	104	7	1	eq_id	134
									eq_context	FROM
				E_BAS	—	—	—	—	eq_tag_num	TA2000
		SO			112	5	6	12	op_code	HEAT
									op_name	HEAT TA3000 TO 88C
									duration	31.4
			SOE		113	112	6	1	eq_id	136
									eq_context	IN
			E_BAS		—	—	—	—	eq_tag_num	TA3000
				SOP	115	113	10	2	prop_type	E
									val_std	88
									val_units	DEGC
		SO			116	5	6	13	op_code	CHARGE
									op_name	CHARGE I-K TO TA3000
									duration	8
			STRM		117	116	6	1		
				SOM	118	117	7	1	matl_id	761
									amt_std	800
									amt_units	LBS
									acc_as_matl	I
									matl_I_O_flag	I
				M_BAS	—	—	—	—	chemical_name	INTERMEDIATE K

a - detail_rec_id							c - parent_table_id			
b - detail_rec_id_parent							d - detail_sequence			
TABLE (Recur Level —>)					a	b	c	d	other	value
				SOP	119	117	7	1	prop type	
									val std	800
									val units	LBS
				SOE	120	117	7	1	eq_id	136
									eq context	TO
				E_BAS	——	——	——	——	eq_tag_num	TA3000
				SOE	121	117	7	1	eq_id	132
									eq context	FROM
				E_BAS	——	——	——	——	eq_tag_num	TA1000
		SO			125	5	6	1	op_code	COOL
									op_name	COOL TA3000 TO 10C
									duration	15.6
			SOE		126	125	6	1	eq_id	136
									eq context	IN
				E_BAS	——	——	——	——	eq_tag_num	TA3000
				SOP	129	126	10	1	prop type	
									val std	88
									val units	DEGC
				SOP	130	126	10	2	prop type	E
									val std	10
									val units	DEGC
		SO			131	5	6	6	op_code	AGE
									op_name	AGE TA3000 60' @10C
									duration	60
			SOE		132	131	6	1	eq_id	136
									eq context	IN
			E_BAS		——	——	——	——	eq_tag_num	TA3000
			SOM		134	131	6	1	matl_id	763
									amt std	4000
									amt units	LBS
									acc as matl	I
									matl_I_O_flag	O
			M_BA		——	——	——	——	chemical_name	INTERMEDIATE M

a - detail_rec_id | c - parent_table_id
b - detail_rec_id_parent | d - detail_sequence

TABLE (Recur Level —>)					a	b	c	d	other	value
				SOP	135	132	10	1	prop type	C
									val std	10
									val units	DEGC
	SO				144	3	6	6	op_code	UNIT OPERATION
									op_name	CENTRIFUGE
		SO			147	144	6	16	op_code	TRANSFER
									op_name	TRANS I-M TO CE4000
									duration	40
			STRM		148	147	6	1		
				SOM	149	148	7	1	matl_id	763
									amt std	4000
									amt units	LBS
									acc as matl	I
									matl_I_O_flag	I
				M_BAS	——	——	——	——	chemical_name	INTERMEDIATE M
				SOP	150	148	7	1	prop type	
									val std	4000
									val units	LBS
				SOE	151	148	7	1	eq_id	139
									eq context	TO
				E_BAS	——	——	——	——	eq_tag_num	CE4000
				SOE	152	148	7	1	eq_id	136
									eq context	FROM
				E_BAS	——	——	——	——	eq_tag_num	TA3000
		SO			153	144	6	17	op_code	TRANSFER
									op_name	TRANS HYDRA TO HP5000
									duration	13
			STRM		154	153	6	1		
				SOM	155	154	7	1	matl_id	765
									amt std	1300
									amt units	LBS
									acc as matl	I
									matl_I_O_flag	O
				M_BAS	——	——	——	——	chemical_name	HYDRAZIME

a - detail_rec_id							c - parent_table_id			
b - detail_rec_id_parent							d - detail_sequence			
TABLE (Recur Level —>)					a	b	c	d	other	value
				S_O_P	156	154	7	1	prop type	
									val std	1300
									val units	LBS
				SOE	157	154	7	1	eq_id	140
									eq context	TO
				E_BAS	—	—	—	—	eq_tag_num	HP5000
				SOE	158	154	7	1	eq_id	139
									eq context	FROM
				E_BAS	—	—	—	—	eq_tag_num	CE4000
		SO			159	144	6	18	op_code	TRANSFER
									op_name	TRANS SPENT TO TA4010
									duration	27
			STRM		160	159	6	1		
				SOM	161	160	7	1	matl_id	764
									amt std	2700
									amt units	LBS
									acc as matl	S
									matl_I_O_flag	O
				M_BAS	—	—	—	—	chemical name	SPENT FLUID
				SOP	162	160	71		prop type	
									val std	2700
									val units	LBS
				SOE	163	160	7	1	eq_id	138
									eq context	TO
				E_BAS	—	—	—	—	eq_tag_num	TA4010
				SOE	164	160	7	1	eq_id	139
									eq context	FROM
				E_BAS	—	—	—	—	eq_tag_num	CE4000

Appendix D: Populated MRPII Tables

This appendix contains the populated MRPII tables and follows the same format as shown in Appendix B: the table names are in uppercase at the top of each respective block; the field names as in lowercase in the shaded sections and the field contents are in uppercase listed below the field names. Where there is no data shown that field is null.

FM_ROUT_HDR

rout_id	rout_no	rout_ver	rout_desc	rout_qty	item_um	rout_class
761		1	INTERMEDIATE K	1600	LBS	ROUT1
762		1	INTERMEDIATE L	2400	LBS	ROUT1
763		1	INTERMEDIATE M	4000	LBS	ROUT1
765	Z-543C	1	HYDRAZIME	1300	LBS	ROUT1

FM_ROUT_DTL

rout_id	r_step_no	r_step_id	oprn_id	step_qty
761	5		29	1000
761	10		42	600
761	15		50	
761	20		56	1600
762	5		61	900
762	10		68	800
762	15		75	700
762	20		82	
762	25		87	2400
763	5	761	92	800
763	10	762	103	2400
763	15		112	
763	20	761	116	800
763	25		125	
763	30		131	4000
765	5	763	147	4000
765	10		153	1300
765	15		159	2700

FM_ROUT_MTL

formula_id	rout_id	routstep_no	formline_id
59	761	5	
59	761	10	
59	761	15	
59	761	20	
90	762	5	
90	762	10	
90	762	15	
90	762	20	
134	763	5	59
134	763	10	90
134	763	15	
134	763	20	59
134	763	25	
134	763	30	
155	765	5	134
155	765	10	
155	765	15	

FM_OPRN_DTL

oprn_id	activity	resrce	o_line_id	proc_qty	res_usage	usage_um
29	CHARGE	TA1000		1000	10.00	MIN
42	CHARGE	TA1000		600	6.00	MIN
50	HEAT	TA1000		69	30.00	MIN
56	AGE	TA1000		69	60.00	MIN
61	CHARGE	TA2000		900	9.00	MIN
68	CHARGE	TA2000		800	8.00	MIN
75	CHARGE	TA2000		700	7.00	MIN
82	HEAT	TA2000		50	10.00	MIN
87	AGE	TA2000		50	60.00	MIN
92	CHARGE	TA3000	56	800	8.00	MIN
103	CHARGE	TA3000	87	2400	24.00	MIN
112	HEAT	TA3000		88	31.40	MIN
116	CHARGE	TA3000	56	800	8.00	MIN
125	COOL	TA3000		10	15.60	MIN
131	AGE	TA3000		10	60.00	MIN
147	TRANSFER	CE4000	131	4000	40.00	MIN
153	TRANSFER	HP5000	147	1300	13.00	MIN
159	TRANSFER	TA4010	147	2700	27.00	MIN

FM_OPRN_MST					
oprn_id	oprn_no	oprn_desc	oprn_vers	proc_qty_	oprn_cl
29	CHARGE	CHARGE R-A TO TA1000	1	LBS	OP1
42	CHARGE	CHARGE R-B to TA1000	1	LBS	OP1
50	HEAT	HEAT TA1000 TO 69C	1	DEGC	OP1
56	AGE	AGE TA1000 60' @69C	1	DEGC	OP1
61	CHARGE	CHARGE R-X TO TA2000	1	LBS	OP1
68	CHARGE	CHARGE R-Y TO TA2000	1	LBS	OP1
75	CHARGE	CHARGE R-Z TO TA2000	1	LBS	OP1
82	HEAT	HEAT TA2000 TO 50C	1	DEGC	OP1
87	AGE	AGE TA2000 60' @50C	1	DEGC	OP1
92	CHARGE	CHARGE I-K TO TA3000	1	LBS	OP1
103	CHARGE	CHARGE I-L TO TA3000	1	LBS	OP1
112	HEAT	HEAT TA3000 TO 88C	1	DEGC	OP1
116	CHARGE	CHARGE I-K TO TA3000	1	LBS	OP1
125	COOL	COOL TA3000 TO 10C	1	DEGC	OP1
131	AGE	AGE TA3000 60' @10C	1	DEGC	OP1
147	TRANSFER	TRANS I-M TO CE4000	1	LBS	OP1
153	TRANSFER	TRANS HYDRA TO HP5000	1	LBS	OP1
159	TRANSFER	TRANS SPENT TO TA4010	1	LBS	OP1

FM_FORM_EFF		
item_id	formula_id	rout_id
761	59	761
762	90	762
763	134	763
765	155	765

CR_RESR_MST	
resrce	resource_desc
CE4000	CENTRIFUGE
HP5000	HOPPER
STORE-A	STORAGE TANK
STORE-B	STORAGE TANK
STORE-X	STORAGE TANK
STORE-Y	STORAGE TANK
STORE-Z	STORAGE TANK
TA1000	TANK
TA2000	TANK
TA3000	TANK
TA4010	TANK

CR_RESR_DTL		
resrce	orgn_code	gl_class
CE4000	MRK	GL1
HP5000	MRK	GL1
STORE-A	MRK	GL1
STORE-B	MRK	GL1
STORE-X	MRK	GL1
STORE-Y	MRK	GL1
STORE-Z	MRK	GL1
TA1000	MRK	GL1
TA2000	MRK	GL1
TA3000	MRK	GL1
TA4010	MRK	GL1

FM_ACT_MST	
activity	activity_desc
CHARGE	CHARGE R-A TO TA1000
CHARGE	CHARGE R-B to TA1000
HEAT	HEAT TA1000 TO 69C
AGE	AGE TA1000 60' @69C
CHARGE	CHARGE R-X TO TA2000
CHARGE	CHARGE R-Y TO TA2000
CHARGE	CHARGE R-Z TO TA2000
HEAT	HEAT TA2000 TO 50C
AGE	AGE TA2000 60' @50C
CHARGE	CHARGE I-K TO TA3000
CHARGE	CHARGE I-L TO TA3000
HEAT	HEAT TA3000 TO 88C
CHARGE	CHARGE I-K TO TA3000
COOL	COOL TA3000 TO 10C
AGE	AGE TA3000 60' @10C
TRANSFER	TRANS I-M TO CE4000
TRANSFER	TRANS HYDRA TO HP5000
TRANSFER	TRANS SPENT TO TA4010

IC_ITEM_MST		class				
item_id	item_desc	gl_	inv_	sales_	ship_	frt_
755	REACTANT-A	GL1	INV1	SALES1	SHIP1	FRT1
756	REACTANT-B	GL1	INV1	SALES1	SHIP1	FRT1
758	REACTANT-X	GL1	INV1	SALES1	SHIP1	FRT1
759	REACTANT-Y	GL1	INV1	SALES1	SHIP1	FRT1
760	REACTANT-Z	GL1	INV1	SALES1	SHIP1	FRT1
761	INTERMEDIATE K	GL1	INV1	SALES1	SHIP1	FRT1
762	INTERMEDIATE L	GL1	INV1	SALES1	SHIP1	FRT1
763	INTERMEDIATE M	GL1	INV1	SALES1	SHIP1	FRT1
764	SPENT FLUID	GL1	INV1	SALES1	SHIP1	FRT1
765	HYDRAZIME	GL1	INV1	SALES1	SHIP1	FRT1

IC_ITEM_MST (class cont)				
price_	stor_	purch_	tax_	custom_
PRICE1	STOR1	PURCH1	TAX1	CUST1
PRICE1	STOR1	PURCH1	TAX1	CUST1
PRICE1	STOR1	PURCH1	TAX1	CUST1
PRICE1	STOR1	PURCH1	TAX1	CUST1
PRICE1	STOR1	PURCH1	TAX1	CUST1
PRICE1	STOR1	PURCH1	TAX1	CUST1
PRICE1	STOR1	PURCH1	TAX1	CUST1
PRICE1	STOR1	PURCH1	TAX1	CUST1
PRICE1	STOR1	PURCH1	TAX1	CUST1
PRICE1	STOR1	PURCH1	TAX1	CUST1

FM_FORM_MST					
formula_no	form_vers	form_id	form_desc1	scale_typ	form_cl
IK_0101	1	59	INTERMEDIATE K	1	FM1
IL_0102	1	90	INTERMEDIATE L	1	FM1
IM_0103	1	134	INTERMEDIATE M	1	FM1
HYD_5555	1	155	HYDRAZIME	1	FM1

FM_MATL_DTL						
formula_id	f_line_id	line_no	item_id	qty	item_um	line_type
59		1	755	1000	LBS	1
59		2	756	600	LBS	1
59		3	761	1600	LBS	2
90		1	758	900	LBS	1
90		2	859	800	LBS	1
90		3	760	700	LBS	1
90		4	762	2400	LBS	2
134	59	1	761	800	LBS	1
134	90	2	762	2400	LBS	1
134	59	3	761	800	LBS	1
134		4	763	4000	LBS	2
155	134	1	763	4000	LBS	1
155		2	765	1300	LBS	2
155		3	764	2700	LBS	3

References

[Chat91] Chatterjee, A., Segev, A., "Data Manipulation in Heterogeneous Database", Sigmod Record, Vol. 20, No. 4, December 1991.

[Gara91] Garai, I., "Management of information Flow in Process Change Control", MS Thesis, Department of Mechanical Engineering, University of Maryland, College Park, 1991.

[Harh86] Harhalakis, G.,"Manufacturing Resource Planning Systems", Class notes for ENME 615 at the University of Maryland, College Park.

[Johr89] Johri, A., "Functional Design of an Integrated CIM System at the Facility Level", MS Thesis, Department of Mechanical Engineering, University of Maryland, College Park, 1989.

[Lin91] Lin, C.P., "Design, Verification an Implementation of Rule Based Information Systems for Integrated Manufacturing", PhD Thesis, Department of Mechanical Engineer, University of Maryland, College Park, 1991.

[Litw90] Litwin, W., Mark. L., Roussopoulos, N., "Interoperability of Multiple Autonomous Database", ACM Computing Surveys, Vol. 22, No. 3, September 1990.

[Mill86] Mills, H., Linger, R., Hevner, A., "Principles of Information Systems Analysis and Design", Academic Press, Inc. 1986.

[Rush93] Rush, D., "Comparison of Database Systems....", computer program for Systems Engineering Masters Thesis at the University of MD College Park, 1993.

[Rush94] Rush, D., "Upload of Data from MIR to MRPII...", computer program for Systems Engineering Masters Thesis at the University of MD, College Park, 1994.

[Shet90] Sheth, A., P., Larson, J.,A., "Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases", ACM Computing Surveys, Vol. 22, No. 3, September 1990.

[Thom90] Thomas, G., Thompson, G., Chung, C., Barkmeyer, E., Carter, F., Templeton, M., Fox. S., Hartman, B., "Heterogeneous Distributed Database Systems for Production Use", ACM Computing Surveys, Vol. 22, No. 3, September 1990.

[Voll88] Vollmann, T., E., Berry, W., L., Whybark, D., C., "Manufacturing Planning and Control Systems", Irwin, 1988.