A Database Application for Manufacturing Simulation System Integration

Yan Luo
Y. Tina Lee
Manufacturing Systems Integration Division
National Institute of Standards and Technology
Gaithersburg, MD 20899-8260

Abstract: This report describes a database model currently under development at the Manufacturing System Integration Division (MSID) of the National Institute of Standards and Technology (NIST). The database model contains a set of tables that are mapped onto the machine shop information model developed by MSID. The information model provides neutral data interfaces for integrating machine shop software applications with simulation. The interface data include organizations, calendars, work, resources, schedules, parts, process plans, and layout within a machine shop environment. The database model is implemented using Microsoft Access; it is used to support the integration of manufacturing applications and simulations for machine shops. The database's structures, relationships, and usages are presented in this report.

Keywords: database, data structures, information model, machine shop operations, simulation

1. INTRODUCTION

Standard interfaces could help reduce the costs associated with simulation model construction and data exchange between simulation and other software applications -- and thus make simulation technology more affordable and accessible to a wide range of potential industrial users. Currently, small machine shops do not typically use simulation technology because of various difficulties and obstacles associated with model development and data translation. Small shops typically do not have staff with the appropriate technical qualifications required to develop custom simulations of their operations or custom translators to import their data from other software applications.

NIST is working with a number of industrial partners and researchers to develop neutral formats for machine shop data to facilitate simulation and modeling activities. A machine shop data model, as a neutral interface format, has been under development to support both NIST's System Integration of Manufacturing Application (SIMA) program and the Software Engineering Institute's (SEI) Technology Insertion Demonstration and Evaluation (TIDE) Program. SIMA supports NIST projects in applying information technologies and standards-based approaches to manufacturing software integration problems [1]. The TIDE Program is sponsored by the Department of Defense and SEI and it is currently engaged in a number of other projects with various small manufacturers in the Pittsburgh, Pennsylvania area. The technical work is being carried out as a collaboration between NIST, SEI, Carnegie Mellon University, Duquesne University, the iTAC Corporation, and the Kurt J. Lesker Company (KJLC).

KJLC is an international manufacturer and distributor of vacuum products and systems to the research and industrial vacuum markets. KJLC manufactures complete, automatically controlled vacuum systems with a special emphasis on custom-designed, thin-film deposition systems for research in alloys, semiconductors, superconductors, optical and opto-electronics. A small machine shop is contained within the KJLC manufacturing facility. KJLC's machine shop operation has been used to help define the requirements for simulation modeling and data interface specification activities described in this paper. Their facility will also be used as a pilot site for testing and evaluation of the simulation models, neutral data interfaces, and other software developed under this TIDE project. For more information on KJLC, see <www.lesker.com>.

The machine shop information model was developed with two goals in mind: a) support for the integration of software applications at a pilot facility – the KJLC's machine shop, and b) promotion as a standard data interface for manufacturing simulators and possibly for other software applications. The information model is continuing to evolve based on experience and feedback from KJLC's implementations and others involved in this effort. The objective of the information modeling effort is to develop a standardized, computer-interpretable representation that allows for exchange of information in a machine shop environment. The information model, when completed, must satisfy the following needs: to support data requirements for the entire manufacturing life cycle, to enable data exchange between simulation and other manufacturing software for machine shops, to provide for the construction of machine shop simulators, and to support testing and evaluation of machine shops' manufacturing software. Data structures contained within the information model include organizations, calendars, resources, parts, process plans, schedules, and work orders for machine shops.

An information model is a representation of concepts, relationships, constraints, rules, and operations to specify data semantics for a chosen domain of discourse. The advantage of using an information model is that it can provide shareable, stable, and organized structure of information requirements for the domain context. An information model serves as a medium for transferring data among computer systems that have some degree of compliance with this information model. For proprietary data, implementation-specific arrangements can be made when transferring those data [2].

In general, the contents of an information model include a scope, a set of information requirements, and a specification. Information requirements serve as the foundation of the specification of the information model. A thorough requirements analysis is a necessity. The initial goal for the machine shop information model is to support data transfer needed for KJLC's machine shop operations. This information model, ultimately, will be presented as a candidate for a standard data interface to be used by other machine shops. Thus, the completeness and correctness of the information requirements and a consensus on the data requirements from the industry are also important requisites.

The specification of the information model defines elements, attributes, constraints, and relationships between elements in the domain context. The specification should be laid out using some formal information modeling languages. An information modeling language provides a formal syntax that allows users to unambiguously capture data semantics and constraints. Three types of methods that implement information models and currently used by the manufacturing community are:

- Data transfer via a working form, which is a structured, in-memory representation of data. The method uses a mechanism that accesses and changes data sequentially without actually moving the data around. All shared data are stored in memory.
- Data transfer via an exchange file, which is a file with a predefined structure or format. This method requires a neutral file format for storing the data. The application systems read from and write into files.
- Data transfer using a database management system. This method uses a database management system where information is mapped onto and retrieved from databases.

These implementation methods can be accomplished through translators that are developed using programming languages and database management systems. The selection of an implementation method is heavily dependent on the target environment where the application system resides. While the relational database is generally desirable for data transfer, the traditional file-oriented systems are being used still by many manufacturing applications.

A machine shop database implementation, which generates relational database tables from the information model using Microsoft Access [3], has been developed at the

Manufacturing System Integration Division (MSID) of NIST. The database model will be used to support the integration of manufacturing application and simulation in the shop environment. This report provides a detailed description of the database model. Section 2 presents the concept of standard interfaces that support machine shop simulations. Section 3 overviews the information model of machine shop data. Section 4 describes the objectives of the database development. Section 5 introduces the database model. Section 6 presents sample queries. Section 7 provides conclusions and a discussion on future work. The database model itself is presented in Appendices A-C, and a few sample data tables are presented in Appendix D.

2. STANDARD INTERFACES

This section describes our approach to developing standard data interfaces that support the machine shop manufacturing simulation. We have proposed an architecture for a generic data-driven machine shop simulator [4], and have been constructing a prototype simulator based on the architecture using commercial off-the-shelf software. The architecture for the generic machine shop simulator is divided into the following component elements: a neutral shop data file, an Extensible Markup Language (XML) [5] data processor, a system supervisor and reporting module, a machine shop emulator, a discrete event simulator, and a user interface system. The machine shop information model is a key factor in effectively and efficiently integrating the generic machine shop simulator.

The information model is now being formulated into a schema using the XML schema language [6]. The information model/XML schema serves as a neutral data format for representing and exchanging machine shop data. With the neutral data format, XML parsers, DBMS translators, and/or XML translators, machine shop data can be represented in working forms (structured, in-memory representations), in database tables, or in XML instance documents. Figure 1 depicts the role of the standard interfaces. The XML parsers, “to/from Database Management System (DBMS) translators,” and “to/from XML translators” are custom-built software programs. XML parsers convert XML schemas’ data elements to structural in-memory presentations, such as C++ data structures. “To/from DBMS translators” and “to/from XML translators” allow data to be converted among a user’s data formats, database structures, and XML document formats.

To facilitate an implementation of the machine shop information model, two translators are being developed at NIST. One converts an XML instance document to an Access database; the other converts a database back to XML. XML data structures, which are parsed from the XML Schemas, are used as intermediate representation. A graphical user interface (GUI) system will also be generated to execute various functions, such as import, export, and translator execution.

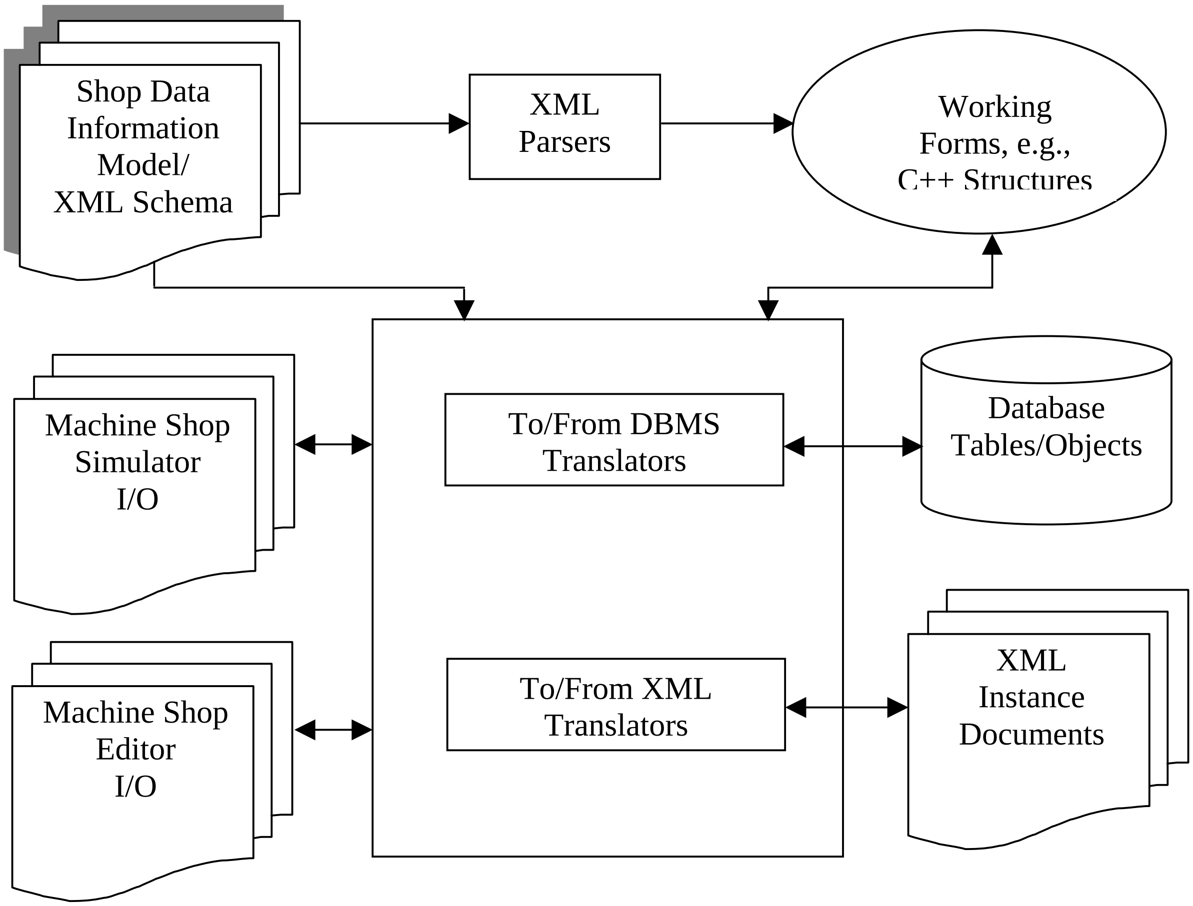

Figure 1: Standard data interfaces

3. OVERVIEW OF THE INFORMATION MODEL

NIST's machine shop data model [7] is presented in both a graphical form in the Unified Modeling Language (UML) [8], and a textual form in XML. The model contains twenty major manufacturing data elements. The primary objective is to develop a structure for exchanging shop data between various manufacturing software applications, including simulation. The idea is to use the same data structures for managing actual production operations and for simulating the machine shop. The rationale is that if one structure can serve both purposes, the need for translation and abstraction of the real data would be minimized when simulations are constructed. The mapping of real world data into simulation abstractions is not, for the most part, addressed in the current data model. Figure 2 illustrates the machine shop data model's top-level data structure that includes most of the major manufacturing data elements identified in the model. The name along with a brief description of each major element is described as follows.

• O*rganizations* is used to maintain the organizational structure, contacts and address information for the manufacturing organization and its customers and suppliers.

• *Calendars* identifies the shift schedules that are in effect for a period of time, breaks and holidays.

• *Resources* describes all the resources that may be assigned to tasks in the shop. The resource types available in the machine shop environment include: stations and machines, cranes, employees, and tool and fixture catalog items.

• *Skill-definitions* lists the skills that an employee may possess and the levels of proficiency associated with those skills.

• *Setup-definitions* typically specifies tool or fixture setups on a machine. Tool setups are typically the tools that are required in the tool magazine. Fixture setups are work holding devices mounted on the machine. Setups may also apply to cranes or stations.

• *Operation-definitions* defines the operations that may be performed at a particular station or group of stations in the shop.

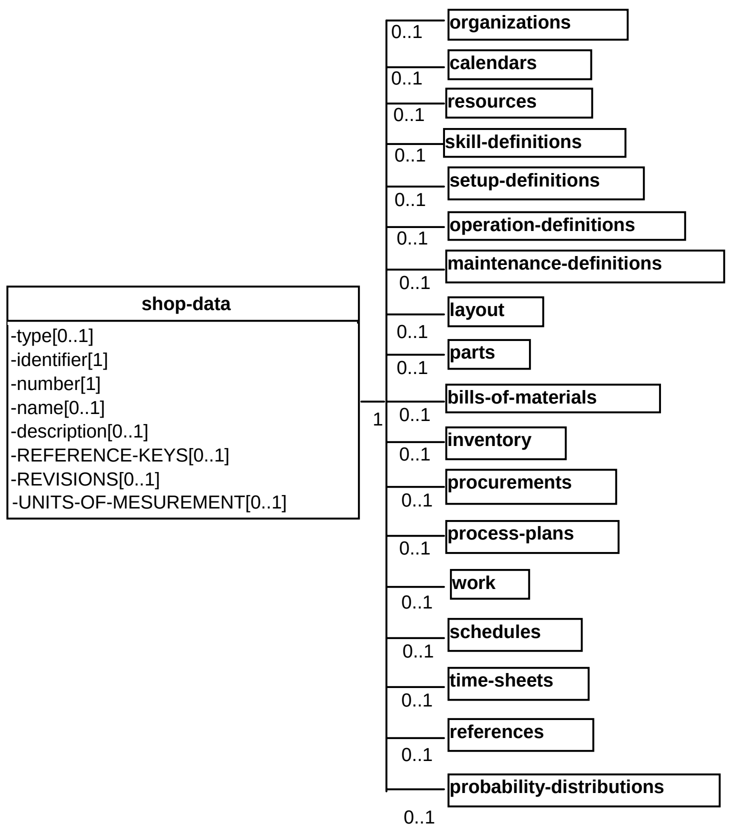

Figure 2: The top structure of the machine shop data model

• *Maintenance-definitions* defines preventive or corrective maintenance to be done on machines or other maintained resources.

• *Layout* defines the location of reference points within the shop, area boundaries, paths, resource, and part objects.

• *Parts* provides elements for part specifications, group technology codes, customers, suppliers, as well as links to bills of materials, process plans, drawings, part models and other references.

• *Bills-of-materials* cross-references the parts and quantities required in a hierarchical bill-of-materials structure.

• *Inventory* identifies the instances and locations for parts, materials, tools, and fixtures inventory.

• *Procurements* identifies the external purchases that have been created to satisfy the part inventory and manufacturing requirements.

• *Process-plans* specifies a set of process plans that are associated with production and support activities for a particular part or parts. A process plan has routing sheets and operation sheets that correspond to the job and task level in the work hierarchy.

• *Work* is used to specify a collection of a hierarchy of production orders, jobs, and tasks. It is also used to specify a collection of internal support orders for maintenance activities, inventory picking and tool preparation.

• *Schedules* lists planned assignment or mapping of work to resources and resources to work.

• *Revisions* specifies information about a set of revisions of the subjects. Information included in the element are each revision's identification, description, date, creators, etc.

• *Time-sheets* provides a list of individual time sheet elements. A time-sheet is used to log the hours that an employee has worked, the time an employee has taken off from work, and accrual of leave hours.

• *Probability-distributions* specifies distributions that are used to vary processing times, breakdown and repair times, and availability of resources, etc.

• *References* describes the information about reference materials that support or further define the data elements contained within the shop data structure.

• *Units-of-measurement* describes various measurement units used in the file, for example, the measurement may be for distance, speed, mass, time duration, or currency.

4. OBJECTIVES OF THE DATABASE IMPLEMENTATION

The objectives of the database implementation effort include:

- to demonstrate the feasibility of the information model,
- to develop a pilot database system and then to migrate to a large database management system, and
- to support the integration of manufacturing applications and simulations used in machine shops.

A database is a collection of related information. It provides a structured means for storing and querying data. Most existing databases are relational databases. A database management system (DBMS), such as Microsoft Access, and Oracle [9], provides software tools for users to organize data in a flexible manner. The database, described in this report, has been developed using Access, a personal computer (PC) based database product. Access comes with the Microsoft Office suite of products and contains many of the features of a relational database management system. Access is more economic to implement and maintain than mainframe DBMSs. Access is extensible. It has a capability to import and export data with various text file formats and database systems using the data access interface, such as Data Access Objects (DAO) [10], Open Database Connectivity (ODBC) [11], and Dynamic Data Exchange (DDE) [12].

5. DATABASE DEVELOPMENT

A database model is designed to map onto the machine shop information model. The database contains a set of relational tables presented in a tree shape structure. Tables comprise the fundamental blocks of a database. A table is a grouping of selected data organized into fields (columns) and records (rows) on a datasheet. A field identifies a data type for a set of value in a table while a record stores a set of values defined by fields. This section describes the guidelines that are used to develop the database. It also provides samples for a mapping from an XML element to a relational table, a design view of a table, and the relationships between tables.

5.1 Guidelines

The following guidelines are used when developing the database structure:

- Elements in the information model are represented as "tables" in the database model.
- Attributes and child-elements of an element are represented as "fields" in the corresponding table.
- Parent-elements of an element are represented as "fields" in the corresponding table. These fields are required for system maintenance.
- A cardinality relationship is specified with the definition of each field if a child-element exists.
- INDEX is a required field for each table and is used by the system.
- A table name is presented in the lower case.
- A field that serves as a referenced table, or a referencing table, or a system used item (such as INDEX or parent-element) is presented in the upper case. All other fields are presented in the lower case.
- Names used for elements or attributes in the information model are kept the same in the database.
- Enumeration tables have names prefixed with "enumeration-" or "type-".

5.2 Relational tables and XML elements

Elements in the information model are used to describe fundamental features or common features of machine shop data. They are represented as tables in the database model. Figure 3 is a sample presentation view about the *stock-level-quantities* element. S*tock-level-quantities* specifies the quantities of tools, fixtures, materials, or parts for various purposes and is represented by the following data elements: *allocated-quantity*, *back-order-quantity*, *on-hand-quantity*, *on-order-quantity*, *required-quantity*, *safety-stock-quantity*, and w*ork-in-progress-quantity*. In Figure 3, *TOOL-INVENTORY-ITEM, TOOLSET-INVENTORY-ITEM, FIXTURE-INVENTORY-ITEM, FIXTURESET-*

INVENTORY-ITEM, PART-INVENTORY-ITEM and *MATERIALS-INVENTORY-ITEM* are parent-elements of *stock-level-quantities.*

stock-level-quantities
INDEX TOOL-INVENTORY-ITEM TOOLSET-INVENTORY-ITEM FIXTURE-INVENTORY-ITEM FIXTURESET-INVENTORY-ITEM PART-INVENTORY-ITEM MATERIALS-INVENTORY-ITEM on-hand-quantity allocated-quantity safety-stock-quantity required-quantity on-order-quantity back-order-quantity work-in-process-quantity

Figure 3: *Stock-level-quantities*

The XML definition of *stock-level-quantities* is presented as follows.

```
<stock-level-quantities>
        <on-hand-quantity />
        <allocated-quantity />
        <safety-stock-quantity />
        <required-quantity />
        <on-order-quantity />
        <back-order-quantity />
        <work-in-process-quantity />
</ stock-level-quantities >
```

5.3 Data structure in a design view

In Access, Design View is used to create fields in each table. A design view contains information presented in three columns: Field, Data Type, and Description. The “Field” column identifies attributes of the element and special elements that are used by the database system, or the referenced or referencing tables. The “Data Type” column defines the data type format for the field. The “Description” column is reserved for information about the attribute usage, domain, definition (that defines the internal relationship between tables), reference, cardinality relationship, etc. The cardinality relationship specifies how many specific instances of the child element could be related to the parent element. The cardinality relationship may be one to zero or one, one to zero or more, one to one or more, or exactly “n” occurrences, and is presented in the design view as [0..1], [0..*], [1..*], [n], respectively.

Shop-data is the top level of the data model. The design view of the *shop-data* table is presented in Table 1.

Table 1: Data structure of *shop-data*

Field	**Data Type**	**Description**
INDEX	Number	[1], system use
type	Enumeration	[0..1]
identifier	Number	[1]
number	Text	[1]
name	Text	[0..1]
description	Text	[0..1]
REFERENCE-KEYS	Text	[0..1], reference-keys.SOURCE
REVISIONS	Text	[0..1], revisions.SOURCE
ORGANIZATIONS	Number	[0..1], organizations.SHOP-DATA
CALENDARS	Number	[0..1], calendars.SHOP-DATA
RESOURCES	Number	[0..1], resources.SHOP-DATA
SKILL-DEFINITIONS	Number	[0..1], skill-definitions.SHOP-DATA
OPERATION-DEFINITIONS	Number	[0..1], setup-definitions.SHOP-DATA
SETUP-DEFINITIONS	Number	[0..1], setup-definitions.SHOP-DATA
MAINTENANCE-DEFINITIONS	Number	[0..1], maintenance-definitions.SHOP-DATA
LAYOUT	Number	[0..1], layout.SHOP-DATA
PARTS	Number	[0..1],parts.SHOP-DATA
BILLS-OF-MATERIALS	Number	[0..1], bills-of-materials.SHOP-DATA
INVENTORY	Number	[0..1], inventory.SHOP-DATA
PROCUREMENTS	Number	[0..1], procurements.SHOP-DATA
PROCESS-PLANS	Number	[0..1], process-plans.SHOP-DATA
WORK	Number	[0..1], work.SHOP-DATA
SCHEDULES	Number	[0..1], schedules.SHOP-DATA
TIME-SHEETS	Number	[0..1], time-sheets.SHOP-DATA
REFERENCES	Number	[0..1], references.SHOP-DATA
PROBABILITY-DISTRIBUTIONS	Number	[0..1], probability-distributions.SHOP-DATA
UNITS-OF-MEASUREMENT	Number	[0..1], units-of-measurement.SHOP-DATA

Shop-data is defined by an identifier, a number, and optionally a type, a name, a description, references, revisions, units of measurement, organizations, calendars, resources, skill definitions, setup definitions, operation definitions, maintenance definitions, a layout, parts, bills of materials, an inventory, procurements, process plans, work, schedules, time sheets, and probability distributions. The field *INDEX* in Table 1 is a special element used by the system, its instance should be an integer, and there will be exactly one instance for a *shop-data* instance. *Type* is an attribute of *shop-data* and is an enumeration to describe types about *shop-data*. *Name* is another attribute, and there may be zero or one instance for *a shop-data* instance. *ORGANIZATIONS* is an attribute served as a pointer pointing to the table of *organizations*. A *shop-data* instance may have zero or one instance of *ORGANIZATIONS*. *ORGANIZATIONS* in the *shop-data* table points to the field of *shop-data* in the *organizations* table. *REFERENCE-KEYS* points to the field of *source* in the *reference-keys* table. All referenced tables that are identified in Table 1 can be found in Appendix A.

5.4 Relationships between tables

The machine shop database model can be represented in a tree shape expression. There are different levels of relationships among elements. The first level carries the relationship between *shop-data* element and the major elements such as *revisions*, *organizations*, *calendars*, *resources*, etc. The second level has the relationship among major elements and their referencing elements.

This subsection describes the relationships between the *machines* element and its related elements. *Machines* lists each machine that exists within the shop. The *machine* element provides descriptive information, hourly rate, technical specifications, status, station associations, and reliability data for a machine. *Machines* is a collection of individual *machine* elements. It *is* referenced by *resources* and it references the elements of *machine*, *machine-group*, *reference-keys*, and *revisions*. *Type*, *identifier*, *number*, *name*, and *description* are attributes of *machines*. Figure 4 demonstrates how *machines* relates to other elements.

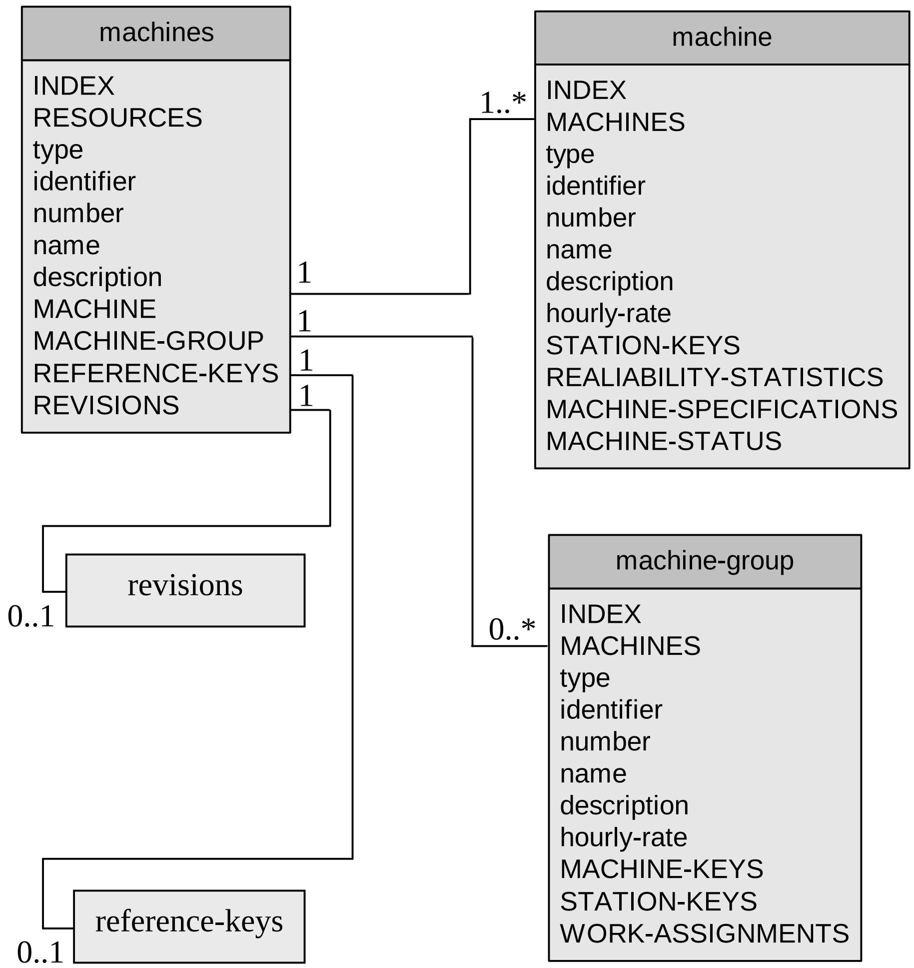

Figure 4: The relationships of *machines*

5.5 Summary of the development

The machine shop database model includes about 500 relational tables. The database model is presented in Appendices A and B. Appendix A presents the table structures for the shop data model. Each table defines a name, its data type and a description for a field. A field may be an attribute, a parent reference, or a child reference. A data type identifies the data format for the field. The data type can be a text string, a number (such as a double or an integer), or an enumeration. A description presents additional information. Appendix B lists enumeration elements that are used in the database model. Relationships among major manufacturing shop data elements are presented in Appendix C. A set of sample data tables is listed in Appendix D.

6. SAMPLE QUERIES

The Structured Query Language's "SELECT", "CREATE", "INSERT", "DELETE", "UPDATE," etc. are useful to manipulate data from a database [13]. This section demonstrates how to query a database for information.

6.1 Basic query

Basic query is used based on a single table. A sample basic query is introduced here. The e*numeration-units* element has the following fields: *angular-units*, *currency-units*, *distance-units*, *length-units*, *mass-units*, *speed-units*, *time-duration-units*, and *volume-units*. Sample data for *enumeration-units* are presented in Table 2. Table 3 shows the result from the sample query that uses the e*numeration-units* table to retrieve information about unit items applicable to a distance.

The basic query statement is:

SELECT [*enumeration-units*].[*distance-units*] FROM [*enumeration-units*];

Table 2: Enumeration items for *enumeration-units* (Query Input)

angular-units	**currency-units**	**distance-units**	**length-units**	**mass-units**	**speed-units**	**time-duration-units**	**volume-units**
DEGREE	DOLLAR	MILE	MILE	TON	MILE/HOUR	DAY	OUNCE
RADIAN	EURO	FOOT	FOOT	POUND	KILOMETER/HOUR	HOUR	CUBICFOOT
	YEN	INCH	INCH	OUNCE		MINUTE	LITER
		KILOMETER	KILOMETER	KILOGRAM		SECOND	GALLON
		METER	METER	GRAM			MILLILITER
		CENTIMETER	CENTIMETER				CUBICMETER
							CUBICINCH
							CUBICCENTIMETER

Table 3: Enumeration items for *distance-units* (Query Output)

distance-units
MILE
FOOT
INCH
KILOMETER
METER
CENTIMETER

6.2 Complex query

Complex queries are used in programs or database operations. The database can be manipulated using database interfaces such as Data Access DAO and ODBC. User can directly manipulate the tables and records by database interfaces, for example, using a *make-table* query, a *cross-table* query or a *parameter* query. The *make-table* query creates a new table from all or part of the data in one or more tables. The c*ross-table* query is a summary query that gives the user control over how the summary data appeared on the screen. The *cross-table* query generates a two-dimensional summary matrix created from tables. The results are displayed in a row/column spreadsheet-like format. The *parameter* query is a query that prompts the user for the criteria each time the query is run.

A sample complex query is described here. The source data comes from Appendix D. Table 4 shows the result from the following query.

The query conditions are:

(1) *shift-schedule*.name ="human-operated-machines-Fall2002"
(2) *shift*.number ="dms-one-of-two-shifts-five-days"
(3) Query for *shift-schedule*.name, *shift*.number, *shift*.start-time, *shift*.end-time and *shift*.duration.

The query statement is:

SELECT [shift-schedule].name, shift.number, shift.[start-time], shift.[end-time], shift.duration FROM ([shift-schedule] INNER JOIN shifts ON [shift-schedule].SHIFTS = shifts.[SHIFT-SCHEDULE]) INNER JOIN shift ON shifts.SHIFT = shift.SHIFTS
WHERE ((([shift-schedule].name)="human-operated-machines-Fall2002") AND ((shift.number)="dms-one-of-two-shifts-five-days"));

Table 4: Output of the sample complex query

shift-schedule.name	shift.number	shift.start-time	shift.end-time	shift.duration
human-operated-machines-Fall2002	dms-one-of-two-shifts-five-days	8:00:00 AM	3:59:00 PM	480

7. CONCLUSIONS AND FUTURE WORK

This report described the work being carried out at MSID in developing a database model for the machine shop data. The objectives of the database development are to demonstrate the feasibility of the information model, to develop a pilot database system and then to migrate to a large database management system, and to support the integration of manufacturing applications and simulations used in machine shops.

The information model will continue to evolve based on the experience and feedback from others involved in this effort. The model is now being transformed into a schema using an XML schema language. There are also plans to expand the model to include assembly line, supply chain, and other domain areas. Thus the database model will be revised accordingly.

The information model will be proposed as a candidate standard to be considered by a formal standards body. There are also experimental development activities underway to test the viability of the model with real world applications. A generic manufacturing simulator is being developed at NIST for the TIDE Program [3]. The model is also being used in the TIDE Program to integrate a manufacturing execution system with a real-time adaptive scheduler, and the manufacturing simulator. Future work also includes the development of the translators that exchange data among XML, the database, and certain proprietary shop data based on the machine shop information model.

ACKNOWLEDGEMENTS AND DISCLAIMER

This project is funded by NIST's SIMA Program and the SEI TIDE Program. SIMA supports NIST projects applying information technologies and standards-based approaches to manufacturing software integration problems. No approval or endorsement of any commercial product by the National Institute of Standards and Technology is intended or implied. The work described was funded by the United States Government and is not subject to copyright.

REFERENCES

[1] Carlisle, M., and J. Fowler. 2001. Systems Integration for Manufacturing Applications Biennial Report. Fiscal Years, NISTIR 6721. Gaithersburg, MD. National Institute of Standards and Technology.

[2] Lee, Y. T. 1999. Information Modeling: From Design To Implementation. Proceedings of the Second World Manufacturing Congress, ed. S. Nahavandi and M. Saadat, 315-321. Canada/Switzerland. International Computer Science Conventions.

[3] http://www.microsoft.com/office/access/default.asp

[4] McLean, C., A. Jones, Y. T. Lee, and F. Riddick. 2002. An Architecture for a Generic Data-Driven Machine Shop Simulator. Proceedings of the Winter Simulation Conference. eds. E. Yucesan, C. Chen, J. L. Snowdon, and J. M. Charnes, 1108-1116. Piscataway, NJ. Institute of Electrical and Electronics Engineers.

[5] http://www.w3.org/XML/ [accessed December 11, 2003].

[6] van der Vlist, E., 2002. XML Schema. Sebastopol, CA. O’Reilly & Associates, Inc.

[7] Lee, Y. T., C. McLean, and G. Shao, 2003. A Neutral Information Model For Simulating Machine Shop Operations. Proceedings of the Winter Simulation Conference.

eds. S. Chick, P.J. Sanchez, D. Ferrin, and D. J. Morrice, 1296-1304. Piscataway, NJ. Institute of Electrical and Electronics Engineers.

[8] http://www.omg.org/uml/ [accessed December 11, 2003].

[9] http://www.oracle.com/ip/deploy/database/oracle9i/ [accessed December 11, 2003].

[10] http://searchdatabase.techtarget.com/sDefinition/0,,sid13_gci213877,00.html [accessed December 11, 2003].

[11] http://searchvb.techtarget.com/sDefinition/0,,sid8_gci214133,00.html [accessed December 11, 2003].

[12] http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/WinUI/WindowsUserInterface/DataExchange/DynamicDataExchange/AboutDynamicDataExchange.asp [accessed December 11, 2003].

[13] http://www.microsoft.com/sql/ [accessed December 11, 2003].