NISTIR 7388
2007

Machine Shop Information Model Application, Next Step

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

Abstract: Simulation is defined as the imitation of the operation of a system or real-world process over time, and in many cases, manufacturing provides one of the most important applications of simulation (Zolfaghari and Roa, 2006). Standard interfaces could make information effective sharing, and hence promote the utilization of simulators. An information model (McLean et al., 2005), which represents machine shop data and facilitates data sharing among machine shop's manufacturing execution system, scheduling system, and simulation system, has been developed at the National Institute of Standards and Technology (NIST). Recently NIST researchers in collaboration with industrial partners have been working on a standards development effort titled Core Manufacturing Simulation Data (CMSD) Product Development Group (PDG) under the guidelines, policies and procedures of the Simulation Interoperability Standards Organization (SISO). A key activity of the CMSD PDG is to develop a CMSD information model using the machine shop information model as the strawman. This paper briefs the machine shop information model and the CMSD information model. This paper discusses information exchange, using NIST's information model, between different representations and presents an algorithm to exchange data between a database system and an eXtensible Markup Language (XML) [1] document. The algorithm has been built based on Document Object Model (DOM), XML Path Language (XPath), and Open Database Connectivity Database Engine (ODBC). The paper also describes interfaces for XML schema's validation, structured query, and data transfer.

Keywords: database, data interface, information model, manufacturing, simulation, XML

1 BACKGROUND

Simulation technology remains largely underutilized by industry today because of complex and costly custom development. Data communication among heterogeneous systems is always a concern, particularly in situations where the support of a universal data exchange standard is unavailable. There is a new data standard for the Web called the eXtensible Markup Language (XML) (Rezayat, 2000, Landau et al., 2002, Robert et al., 2002). XML is used to manipulate, access, and exchange data or metadata over different platforms and systems. XML is regarded as next generation data representation (Manola, 1999). There are still problems to transfer data between existing database system and XML representation format. The paper focuses on this issue.

1.1 Information model

Many integration projects today rely on shared semantic models based on standards represented using XML technologies (Morris, 2004). An information model is a representation of concepts, relationships, constraints, rules, and operations to specify data semantics for a chosen domain of discourse (Lee, 1999). 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. Information model is important for effective information sharing and integration. 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.

Figure 1 Sample schema

```
 <?xml version="1.0" encoding="UTF-8" ?>
- <xs:schema
      xmlns:xs="http://www.w3.org/2001/XMLSchema
      " elementFormDefault="qualified">
        + <!--    $Id: calendars.rnc,v 1.5
        2003/07/24 21:57:17 MSG Exp $
           RELAX NG compact syntax
        representation of "calendars" from shop
        data model  -->
   - <xs:complexType name="calendars">
      - <xs:complexContent>
         - <xs:extension
              base="all.front.matter">
           - <xs:sequence>
                <xs:element
                   maxOccurs="unb
                   ounded"
                   ref="calendar" />
             </xs:sequence>
           </xs:extension>
        </xs:complexContent>
     </xs:complexType>
     <xs:element name="calendar" type="calendar"
        />
   - <xs:complexType name="calendar">
      - <xs:annotation>
           <xs:documentation>information
              about the shift schedules that
              are in effect for a period of
              time, breaks, and
              holidays</xs:documentation>
        </xs:annotation>
      - <xs:complexContent>
         - <xs:extension
              base="all.front.matter">
           - <xs:sequence>
                <xs:element
                   ref="effective-
                   start-date" />
                <xs:element
                   ref="effective-
                   end-date" />
                <xs:element
                   maxOccurs="unb
                   ounded"
                   ref="shift-
                   schedule" />
        ...
 </xs:schema>
```

A machine shop information model (McLean et al., 2005) has been developed at the National Institute of Standards and Technology (NIST) as a part of efforts that support the development of standard data interfaces. The information model is intended to be used for representing and exchanging machine shop data, initially among manufacturing execution, scheduling, and simulation systems.

An XML schema (Laurent, 1998, Freire and Benedikt, 2004) describes the structure of an XML document. The purpose of an XML Schema is to define the legal building blocks of an XML document. It defines elements, attributes, elements' child elements, the order and number of child elements, data types, etc. The schema for the machine shop information model has been developed in the XML Schema language [15]. A sample schema is presented in Figure 1.

1.2 Database model

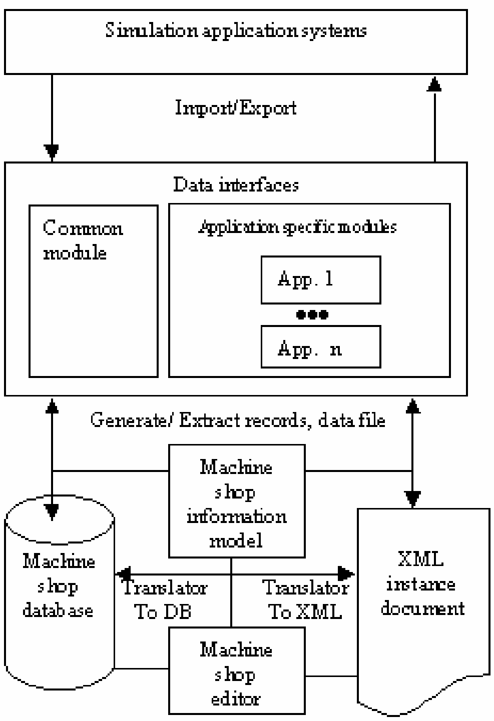
Database provides a structured means for storing and querying data. Most existing databases are relational databases. A database management system (DBMS), such as Microsoft Access [2] or Oracle [4], provides software tools for users to organize data in a flexible manner. The machine shop database, described in this paper, has been developed using Access (Luo, 2003). The database is designed to represent the machine shop information model. Access can import and export data using data access interface, such as Data Access Objects (DAO) [5], Open Database Connectivity (ODBC) [6], and Dynamic Data Exchange (DDE) [7].

2 DATA EXCHANGE FRAMEWORK

NIST has developed a software architecture, standard data interfaces, and a prototype generic machine shop simulator that can be readily reconfigured for use by a large number of small machine shops (McLean et al., 2002). The architecture for the generic machine shop simulator is divided into the following component elements: neutral shop data file, XML data processor, system supervisor and reporting, machine shop emulator, discrete event simulator, and user interface system. The machine shop information model is a key factor in effectively and efficiently integrating the generic machine shop simulator.

Standard interface is helpful for data exchange/sharing. A framework has been proposed to integrate simulation application systems, as shown in Figure 2. Prototype tools have been developed to demonstrate data exchange between the machine shop database and XML document. The development is built based on the widely accepted standards, such as Document Object Model (DOM) [8] for XML document operation, and XML Path Language (XPath) [9] for nodes query. The machine shop database or XML document is used to store machine shop data. Translators support data exchange between the database and XML document. The data interfaces of simulation systems access XML document and query machine shop database. The interface includes two major parts, common module and application specific modules. The common module is built based on the machine shop information model. Application system gets input/output data through its export/import modules. For example, an Arena simulation system (Rockwell Automation, Inc.'s simulation software) [14] can access its simulation data from an Access database that contains special structures defined Arena.

Figure 2 The application architecture for simulation



3 DATA TRANSLATOR TO MACHINE SHOP DATABASE

This section presents an algorithm to transfer data from an XML document to a relational database.

3.1 XML structure

XML provides a format for describing structured data, it is used to describe the machine shop information model. The information model is organized as tree shape, the top element of this structure is named *shop-data* that has branches or XML elements such as *calendars*, *resources*, *setup-definitions*, and *work*. These branches/elements might have their child elements. Figure 3 presents *setup-definitions, a machine shop XML element*, as a sample.

3.2 Schema validation

An XML schema provides details about the content model: which elements it contains and in what order, what its content can be, and which content these attributes can contain. It describes the vocabulary for use by others creating XML documents and defines the elements that can appear within an XML document and the attributes that can be associated with an element. It is used to verify that the incoming XML documents are in the expected format. It is used to validate the content of an XML document, determine whether the XML document is a valid instance of the vocabulary (grammar or rules) expressed by the XML schema.

An XML instance file document will be validated against the machine shop schema before it is transferred to a database or used as a data source for a simulation application.

Figure 3 Sample XML structure


```
<?xml version="1.0" encoding="UTF-8"?>
<setup-definitions type="setup-definitions1" identifier="10000"
number="setupdefinitions1">
  <setup-definition type="setupdefinition1" identifier="1" number="1">
        <name>"Empty table"</name>
        <setup-components>
          <fixture-definition-key fixture-definition-number="F001"/>
          <fixtureset-definition-key fixtureset-definition-number="F1"/>
          <tool-definition-key tool-definition-number="T001"/>
          <toolset-definition-key toolset-definition-number="T1"/>
        </setup-components>
        <setup-resource-keys>
          <machine-key machine-number="310"/>
          ...
        </setup-resource-keys>
        <child-setup-keys>
          <setup-definition-key setup-definition-number="2"/>
          ...
        </child-setup-keys>
  </setup-definition>
  ...
  <setup-changeovers>
        <setup-changeover>
          <current-setup>
                <setup-definition-key setup-definition-number="1"/>
          </current-setup>
          <new-setup>
                <setup-definition-key setup-definition-number="2"/>
                <estimated-durations>
                  <setup-duration>
                    <nominal-duration>30</nominal-duration>
                  </setup-duration>
                </estimated-durations>
          </new-setup>
          <new-setup>
                <setup-definition-key setup-definition-number="3"/>
                <estimated-durations>
                  <setup-duration>
                    <nominal-duration>15</nominal-duration>
                  </setup-duration>
                </estimated-durations>
          </new-setup>
        </setup-changeover>
  </setup-changeovers>
  ...
</setup-definitions>
```


3.3 XML mapping

A database model is designed to map onto the machine shop model. The database contains a set of relational tables presented in a tree structure. The tables comprise the fundamental blocks of a relational 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.

Figure 4 shows a sample mapping from an XML element to a database table. The sample element is *group-technology-code*, which describes a code system that defines a particular part. The system identifies the part's shape, material, color, surface finish, function, weight, process, and cost using a predefined set of codes.

Figure 4 Structure mapping

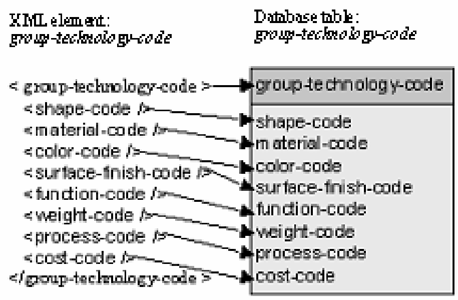


Figure 5 Data mapping

XML document: *holiday*

```
<holidays>
  <holiday identifier="101" >
    <name>New Year</name>
    <description/>
    <date>2003-01-01</date>
  </holiday>
  <holiday identifier="214" >
    <name>Valentines Day</name>
    <description/>
    <date>2003-02-14</date>
  </holiday>
  <holiday identifier="1225" >
    <name> Christmas </name>
    <description/>
    <date>2003-12-25</date>
  </holiday>
```

Table: *holiday*

identifier	name	description	date
101	New Year		2003-01-01
214	Valentines Day		2003-02-14
1225	Christmas		2003-12-25

Data contained in a machine shop XML document can be extracted and populated into a machine shop database. Figure 5 presents a data mapping sample. The *holiday* class provides a means to indicate that no production is scheduled on a specific date for the complete day. The *holiday* class has *identifier*, *name, description, and date* attributes. The *identifier* attribute is used to uniquely identify the object. The *name* and *description* attributes provide a means to specify related information about the holiday being defined. The *date* attribute allows the specification of the date on which the holiday is to take place. Only the *identifier* and *date* attributes required for valid instances of *Holiday*. In Figure 5, three holidays, New Year, Valentines Day, and Christmas, are defined, their corresponding identifiers are 101, 214, and 1225.

3.5 DOM specification

DOM is a platform- and language-neutral interface that will allow programs and scripts to dynamically access and update the content, structure, and style of documents. When an XML instance document is loaded into memory, the document is first converted into a DOM structure, the structure will then be analyzed for retrieving the data. Similarly, when generate an XML representation document, the DOM structure will be generated first. A DOM structure includes a root node, which is required, comments, instructions and version information.

DOM is used in the development of interface tools that transfer data between the machine shop database and XML document.

3.6 XPath specification

XPath is the result of an effort to provide a common syntax and semantics for functionality shared between the eXtensible Stylesheet Language Family (XSL) [16] Transformations and XML Pointer Language (XPointer) [17]. The primary purpose of XPath is to address parts of an XML document. XPath uses a compact, non-XML syntax to facilitate use of XPath within Uniform Resource Identifiers (URIs) and XML attribute values. XPath operates on the abstract, logical structure of an XML document, rather than its surface syntax. XPath gets its name from its use of a path notation as in URLs for navigating through the hierarchical structure of an XML document. XPath is also designed so that it has a natural subset that can be used for matching. XPath models an XML document as a tree of nodes.

XPath is mainly used to enquire a DOM document through the nodes tree. For example, the machine shop data can be accessed by using the XPath grammar.

4 DATA TRANSLATOR FROM MACHINE SHOP DATABASE

This section discusses the data exporting from a machine shop database and generating XML document.

4.1 XML document

An XML structure is built based on the schema of machine shop information model. XML data may come from table records of a machine shop database. Figure 6 presents a skeleton of a sample XML document.

Each XML document has both a logical and a physical structure [1]. Physically, the document is composed of units called entities. An entity may refer to other entities to cause their inclusion in the document. A document begins in a "root" document entity or prolog. Logically, the document is composed of declarations, elements, comments, character references, and processing instructions, all of which are indicated in the document by explicit markup. The function of the markup in an XML document is to describe its storage and logical structure and to associate attribute name-value pairs with its logical structures. An XML document is valid if it has an associated document type declaration and if the document complies with the constraints expressed in it.

Figure 6 XML document skeleton

```
<?xml version="1.0" encoding="UTF-8"?>
<element1 xmlns:eg="" attribute="">
        <element2> data </element2>
        <element3> data </element3>
        <element4 attribute="">
        ...
        </element4>
        ...
  </element1>
```

Figure 7 A sample of XML document prolog

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="/style.css" type="text/css" title="default
stylesheet"?>
<!--The model was generated on 2004-7-10-->
```

Figure 7 shows a sample prolog of an XML document. The *version* declaration specifies the version of XML being used. The *encoding* declaration identifies which encoding being used to represent the characters in the document. Content that is not intended for the XML parser, such as notes about document structure or editing, can be included in a comment. Processing instructions can be used to pass information to applications in a way that escapes most XML rules.

4.2 Generating XML document from database

This section discusses how to transfer data from a relational database to an XML document. Three query tools have been developed:

Creating document for whole database query

As discussed above, the machine shop database is built based on the machine shop data model. The database is a tree shape structure, the top level is element *shop-data*. In this tool, *shop-data* is considered as root element, the algorithm will enquiry all the records in every table of the database and hence generate an XML document that represents the entire database information.

Creating document for a root element

This tool provides a query to generate an XML document for a selected root element. This query searches the related records of tables. If the root element has more than one records, only one record will be selected as root element record.

Creating document for a SQL query

Structured Query Language (SQL) provides functions to support database query. This tool provides an algorithm, as shown in Figure 8, to generate XML file using SQL statements.

5 NEXT STEP - STANDARDIZATION

5.1 Standard Body

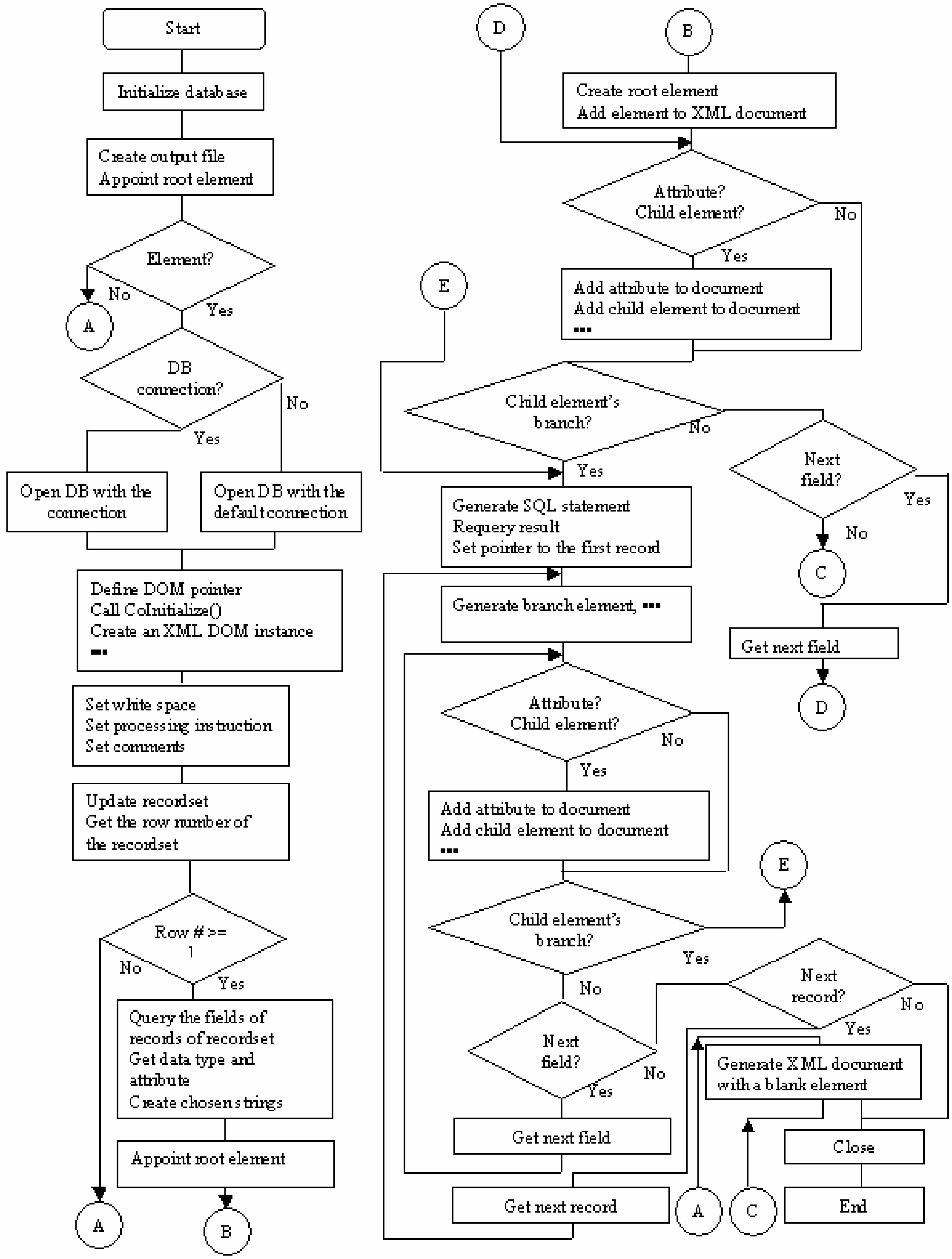
The research work described above demonstrates a feasibility of the machine shop information model. The ultimate objective in the development of the machine shop information model and its data sharing applications is to promote the establishment of a standard data interface for manufacturing simulators. The machine shop information model has been transferred to the Core Manufacturing Simulation Data (CMSD) Product Development Group (PDG) of the Simulation Interoperability Standards Organization (SISO) [10]. SISO, a standard body recognized by Institute of Electrical and Electronics Engineers (IEEE), focuses on facilitating simulation interoperability and component reuse.

5.2 Overview – the CMSD information model

The key activity of the CMSD PDG is to generate a CMSD information model (CMSDIM), using both the Unified Modeling Language (UML) [3] and XML. The machine shop information model is the strawman of the CMSD information model. The CMSDIM is to describe the essential entities in the manufacturing domain and the relationships between those entities that are necessary to create manufacturing simulations. The model has being developed with the following goals in mind: (1) to foster the development and use of simulations in manufacturing operations, (2) to facilitate data exchange between simulation and other manufacturing software applications, (3) to enable and facilitate better testing and evaluation of manufacturing software, and (4) to increase manufacturing application interoperability. The primary objective was to develop data structures for exchanging manufacturing data between various manufacturing software applications, including simulation. The major categories of manufacturing information that are defined in this information model include:

• *Organization* is used to maintain organizational structure, contacts and address information for the manufacturing organization and its customers and suppliers.
• *Calendar* identifies the shift schedules that are in effect for a period of time, breaks and holidays.
• *Resource* describes all the resources that may be assigned to tasks in the shop. The resource types available in the shop environment include: stations and machines, cranes, employees, tool and fixture catalog items, and user-defined type of resource.

Figure 8 Export data from machine shop database



• *Skill definition* lists the skills that an employee may posses and the levels of proficiency associated with those skills.
• *Setup definition* 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 definition* defines the operations that may be performed at a particular station or group of stations in the shop.
• *Maintenance definition* defines preventive or corrective maintenance to be done on machines or other maintained resources.
• *Part* provides elements for part specifications, group technology codes, customers, suppliers, as well as links to bill of materials, process plans, drawings, part models and other references.
• *Bill-of-materials* cross-references the parts and quantities required in a hierarchical bill-of materials structure.
• *Inventory* identifies the instances and locations for part, materials, tool, and fixture inventory.
• *Process plan* 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.
• *Schedule* lists planned assignment or mapping of work to resources and resources to work.
• *Revision* specifies information about a set of revisions of the subjects. Information included in the element are each revision's description, date, creators, etc.
• *Probability distribution* specifies distributions that are used to vary processing times, breakdown and repair time, and availability of resources, etc.
• *Reference* describes the information about reference materials that support or further define that data elements contained within the CMSDIM data structure.

5.3 Status

The SISO CMSD PDG has released a draft version of the CMSDIM in UML [11] for review and comment in early 2006 [12]. A CMSDIM in the XML schema is under development. A database and editor that are based on the CMSDIM are under development by the NIST researchers. The CMSD database, in Microsoft Access, is designed to map onto the CMSDIM. The objective to develop the CMSD editor is to facilitate the use of the CMSDIM. The editor is developed using Microsoft .NET Framework 1.1 Service [13].

6 DISCUSSIONS AND CONCLUSION

Advanced manufacturing technologies are adopted in industry because of world wide competitive (Luo, 2002; Luo, 2003; Luo 2004). Simulation technology can reduce product cost, shorten product development time, and improve product quality. The machine shop information model developed at NIST provides neutral data interfaces for integrating machine shop software applications with simulation. The CMSD information model, an extension of the machine shop information model, is currently developed by the SISO. The CMSD information model defines a data specification for efficient exchange of manufacturing life cycle data in a simulation environment. The specification provides neutral data interfaces for integrating manufacturing software applications with simulation systems. The interface data includes organizations, calendars, work, resources, schedules, parts, process plans, and layout, etc. The paper discusses the data transfer between XML document and the machine shop database. The future work contains enhancing the information model and database model, studying the data sharing mechanism for distributed simulation system, and developing application platform for the manufacturing simulations.

DISCLAIMER

Commercial software products are identified in this paper. These products were used for demonstration purposes only. 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

Charles, R. H. and Rochelle N. P. (2003) "Simulation modeling using ProModel technology," Proceedings of the 2003 Winter Simulation Conference, New Orleans, LA.

Freire, J. and Benedikt, M. (2004) "Managing XML data: an abridged overview," Computing in Science & Engineering, Vol. 6, No. 4, pp. 12-19.

Landau, R. H.; Vediner, D.; Wattanakasiwich, P. and Kyle, K. R. (2002) "Future scientific digital documents with MathML, XML, and SVG," Computing in Science & Engineering, Vol. 4, No. 2, pp. 77–85.

Laurent, S. S. (1998) *XML: a primer*, Foster City, CA: IDG Books Worldwide.

Lee, Y. T. (1999) "Information modeling: from design To implementation." Proceedings of the Second World Manufacturing Congress, Durham, U.K.

Luo, Y. (2000) "Injection molding product application activities models," International Journal of Advance Manufacturing Technology, Vol. 16, No. 4, pp. 285-288.

Luo, Y. (2002) "Manufacturing features based solution for high speed cutting," International Journal of Production Engineering and Computers, Vol. 4, No. 5, pp. 25-30.

Luo, Y. (2003) "Chip formation analysis for high speed cutting," International Journal of Advanced Manufacturing Systems, Vol. 2, No. 2, pp. 247-254.

Luo, Y. (2004) "Parametric tool wear estimation solution of HSC appropriate machining," International Journal of Advanced Manufacturing Technology, Vol. 23, No. 7-8, pp. 546 – 552.

Luo, Y. and Lee, Y. T. (2003) "A database application for manufacturing simulation system integration," Proceedings of the IASTED International Conference: Applied Modeling and Simulation, Marina Del Rey, CA.

Luo, Y. and Lee, Y. T. (2005) "Application of machine shop data model in manufacturing simulation," Proceedings of the 2005 International Conference on Modeling, Simulation and Visualization Methods (MSV 05), Las Vegas, NV.

McLean, C.; Jones, A.; Lee, Y. T. and Riddick, F. (2002) "An architecture for a generic data-driven machine shop simulator," Proceedings of the Winter Simulation Conference, San Diego, CA.

Manola, F. (1999) "Technologies for a web object model," IEEE Internet Computing, Vol. 3 No.1, pp. 38-47.

McLean, C.; Lee, Y. T.; Shao, G. G. and Riddick, F. (2005) "Shop data model and interface specification," NISTIR 7198, Gaithersburg, MD.

Morris, K. C.; Kulvatunyou, B.; Frechette, S.P.; Lubell, J. and Goyal, P. (2004) "XML schema validation process for CORE.GOV," NISTIR 7187, Gaithersburg, MD.

Rezayat M. (2000) "Knowledge-based product development using XML and KCs," Computer-Aided Design, Vol. 32, No. 5-6, pp. 299-309.

Robert W. P. L.; Leong, H.V.; Dillon, T. S.; Chan, A. T. S.; Croft, W. B. and Allan, J. (2002) "A survey in indexing and searching XML documents," Journal of the American Society for Information Science and Technology, Vol. 53, No. 6, pp. 415-437.

van der Vlist, E. (2002) *XML Schema*, Sebastopol, CA, O'Reilly & Associates, Inc.

Zolfaghari, S. and Roa, E. V. L. (2006) "Cellular manufacturing versus a hybrid system: a comparative study," Journal of Manufacturing Technology Management, Vol. 17, No. 7, pp. 942-961.

NOTES

1 http://www.w3.org/TR/2006/REC-xml-20060816/
2 http://office.microsoft.com/en-us/access/default.aspx
3 http://www.omg.org/uml/
4 http://www.oracle.com/database/index.html
5 http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vccore/html/_core_dao.3a_.where_is.......asp
6 http://support.microsoft.com/kb/279721
7 http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/WinUI/WindowsUserInterface/DataExchange/DynamicDataExchange/AboutDynamicDataExchange.asp
8 http://www.w3.org/DOM/
9 http://www.w3.org/TR/xpath
10 http://www.sisostds.org/
11 http://www.microsoft.com/office/access/default.asp
12 http://discussions.sisostds.org/file.asp?file=CMSDPart102%2D28%2D06%2Epdf 2006
13 http://www.microsoft.com/downloads/details.aspx?displaylang=en&FamilyID=A8F5654F-088E-40B2-BBDB-A83353618B38
14 http://www.arenasimulation.com/
15 http://www.w3.org/XML/Schema
16 http://www.w3.org/TR/xsl/
17 http://www.w3.org/TR/xptr/