Int J Adv Manuf Technol (2009) 42:606–620
DOI 10.1007/s00170-008-1628-9

ORIGINAL ARTICLE

An architecture for ubiquitous product life cycle support system and its extension to machine tools with product data model

Byeong-Eon Lee · Suk-Hwan Suh

Received: 16 November 2007 / Accepted: 18 June 2008 / Published online: 17 July 2008

Abstract Product life cycle information is utmost important for the stakeholders involved from product design, manufacturing, through product retirement. However, information on products after they become available to the customer becomes vague or unrecognized. Furthermore, various real-time data from the shop floor or usage data from customers are not gathered in real time. To overcome this problem, we present an ubiquitous product life cycle support (UPLS) system based on the UbiDM: a new paradigm for design and manufacturing via ubiquitous technology proposed by our research center. Specifically, in this paper, we derive a generic architecture for the UPLS system based on the UbiDM and requirement analysis from the various functional- and data-level perspectives. As an extension, we apply the generic architecture for the machine tools by developing a life cycle data model to be used for the UPLS system for machine tools. The machine tools were taken as an example among the products considering the importance of every life cycle of the product (i.e., Begin-of-Life, Middle-of-Life, and End-of-Life) and the necessity of international standards under establishment by the ISO TC184/SC1 and SC4. The impact and validity of the developed architecture and data model were made by some realistic usage scenarios with the UPLS system incorporating the architecture and data model developed in this paper.

Keywords UbiDM · Product life cycle engineering · Service oriented architecture · Machine tool data model · u-Design

1 Introduction

Product life cycle information becomes very crucial in the product-centric environment, including product design, manufacturing, and product retirement. In general, the product life cycle is divided into three stages: Begin-of-Life (BOL), which encompasses product design to production stage; Middle-of-Life (MOL), which encompasses usage and maintenance stage; and End-of-Life (EOL), which encompasses the disposal/recycle stage. In this regard, the contemporary Product Life Cycle Management (PLM) system can be viewed as an extension of Product Data Management (PDM) system within the scope of BOL, without covering much of the MOL and EOL. As a result, the users have to mostly rely on the BOL information in design and manufacturing. To solve the above problem, there should be (1) a means to acquire information of the product at MOL and EOL and an (2) information highway encompassing BOL, MOL, and EOL.

In this paper, we attempt to approach the above problem via ubiquitous-computing technology (henceforth, ubiquitous technology). Looking over the ubiquitous technology, a great advancement has been made since it was introduced in early 1980s. Up to the present time, ubiquitous technology has been mostly applied to aid people, and only a few attempts were made to product design and manufacturing. Until recently, research related to the

B.-E. Lee
e-Manufacturing Lab,
Division of Mechanical Engineering for Emerging Technology,
POSTECH,
Pohang, South Korea

S.-H. Suh (✉)
Center for Ubiquitous Manufacturing, POSTECH,
790-784 Pohang, South Korea
e-mail: shs@postech.ac.kr

manufacturing domain has been mostly concerned with the use of radiofrequency identification (RFID) technology to get information of in-use products [1–4]. Some European projects, such as PROMISE [5] and AMI4SME [6], respectively, propose a more advanced scenario for acquiring product information and networked collaboration by using ubiquitous technology, such as sensors embedded on the product, and context awareness via speech recognition, etc.

From the perspective of information acquisition, RFID tag, mobile communication, and embedded systems made it possible to obtain the physical data of individual products during MOL and EOL from the real world in which the product is used. Furthermore, the ubiquitous sensor network technology made it possible to get shop floor data during BOL and usage data during MOL in (near) real time. Based on the observations made during those stages, we recently proposed a new paradigm called UbiDM standing for 'design and manufacturing via ubiquitous technology' [7].

In order to realize UbiDM paradigm, a variety of technologies, including ubiquitous technology (UT), information technology (IT), and manufacturing technology (MT), were required. Figure 1 shows the detailed technology and relationship among the technology proposed in [7]. From the functionality point of view, the UbiDM technology can be largely classified into three categories: (1) vertical integration technology for collection and transmission of field data in (near) real-time, (2) horizontal integration technology for information flow encompassing the whole product life cycle, and (3) system engineering technology for design and validation of ubiquitous system.

Specifically, the vertical integration technology is related to sensors and devices and network communication technology for collection, processing, and transmission of the field data for the product and environment. Horizontal integration technology is related to information infrastructure called ubiquitous product life cycle information highway (UPLI) for the seamless exchange of information between stakeholders associated with the product life cycle. System engineering technology is related to the development tools for *ubiquitous system* including design, simulation, analysis, and evaluation on the computer, like computer-aided design (CAD)/computer-aided engineering (CAE)/computer-aided manufacturing (CAM) system for design, analysis, and manufacturing of the *product*.

The theme of this paper is related to horizontal integration technology for UPLI and its supporting system called the ubiquitous product life cycle support (UPLS) system. The UPLS system is a kind of information system running on the UPLI, the backbone of UPLS system for the seamless information flow, and it provides adequate information to the stakeholders based on the information obtained from the product via vertical integration technology, as indicated by the dotted box in Fig. 1. In this paper, we develop a generic architecture of UPLS system based on the various perspectives including requirement analysis, followed by the life cycle data model for the machine tools. The machine tools were taken as an example among the products considering the importance of every life cycle of the product (i.e., BOL, MOL, and EOL) and the necessity of international standards under establishment by the ISO TC184/SC1 and SC4.

The remainder of this paper is organized as follows: in Section 2, we investigate requirements for the architecture design of UPLS system described in Section 3; in Section 4, a life cycle data model for machine tools is derived and validated via usage scenarios in Section 5. This paper concludes with a discussion in Section 6.

2 Requirement analysis

During the life cycle of the product, a great deal of information like digital models, such as CAD drawings, technical documents, and structured and unstructured data, is created, changed, transferred, stored, and converted by different stakeholders and application systems. In designing the architecture of a UPLS system, various aspects should be taken into consideration: categorized into two levels: functional level and data interface level.

2.1 Functional-level requirements

Functional-level requirements are related to functions that a UPLS system should possess for successful and efficient execution of a particular mission. The crucial functions are described in what follows.

[FR-1] Real-time data acquisition Ubiquitous technology makes it possible to collect the individual product data in real time throughout the entire product life cycle. The collected data should include individual product data and context data related to the product. On the shop floor, for instance, the real-time data from various resources and manufacturing process will be required for the transparent management of the shop floor.

[FR-2] Closed-loop information flow Closed-loop information flow means the transparent and bi-directional information flow between every stakeholder involved in BOL, MOL, and EOL. For example, service and maintenance (MOL) and recycling (EOL) stakeholders should be able to obtain the product data, such as CAD drawings and technical information from the designers or producers of the products (BOL). Likewise, BOL stakeholders should be able to obtain feedback information generated by the

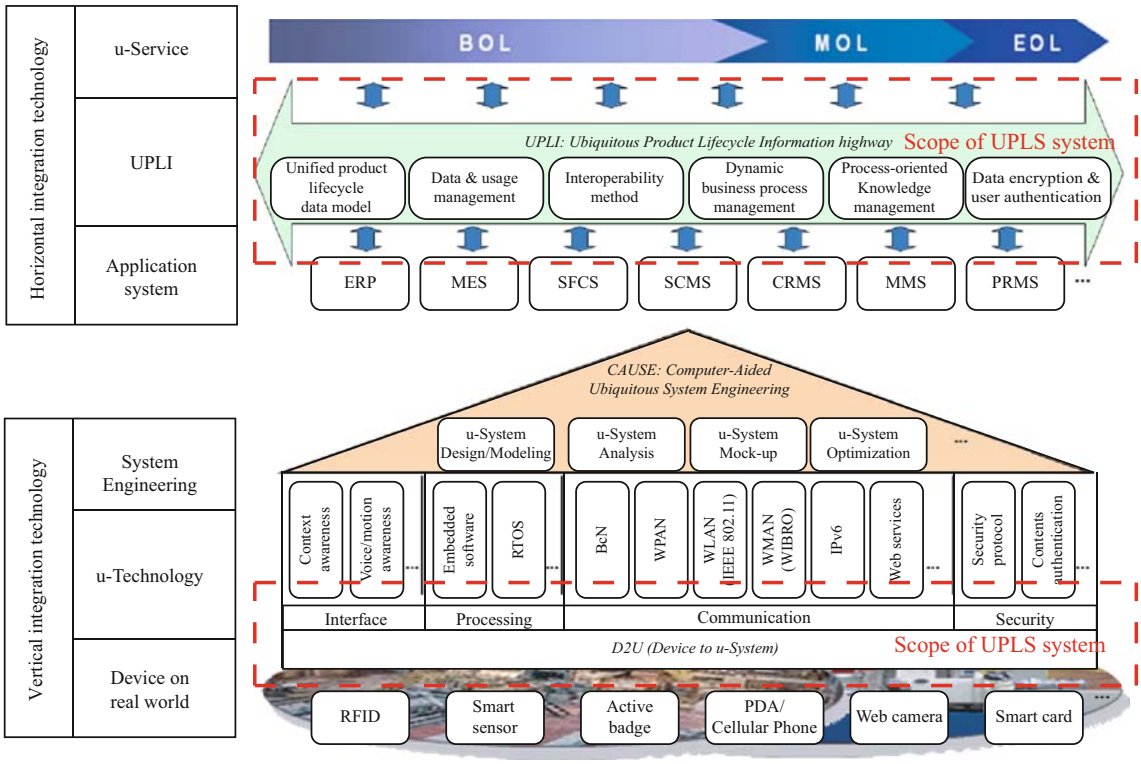

Fig. 1 Research spectrum of the UbiDM

stakeholders of MOL (corr. EOL) who used (remanufactured) the product.

[FR-3] Interoperability between devices and application systems In the ubiquitous environment, various devices and application systems used at the different life cycle stages communicate with each other via wired or wireless network. Since these systems may run on various platforms, it is very important to guarantee interoperability between the various devices, platforms, and application systems.

[FR-4] Integration with existing systems and services Many application systems such as CAD/CAM/CAE system, enterprise resource planning, supply chain management, customer relationship management, and manufacturing execution system are used to support the activities of the stakeholders during the product life cycle. Therefore, the integration and/or interface between existing systems and newly added services of UPLS system must be taken into consideration. For the seamless integration, it is required to remove the redundant functions while maximally reusing the existing functions.

[FR-5] Collaborative environment throughout the entire product life cycle Currently, collaboration is focused on the activities and stakeholders at the BOL stage. However, as ubiquitous technology enables transparent information exchange and feedback, collaborative development between all the participants at the BOL, MOL, and EOL must be supported.

2.2 Data-level requirements

The data interface level is associated with the data interface scheme, which supports the entire product life cycle. These data include not only the product data itself, such as the specification, assembly/disassembly, and BOM of the product, but product-related data such as business, maintenance, and expiration data generated during MOL and EOL. For the seamless information flow between various stakeholders and application system, the data-level interface is key. Detailed requirements of the data-level are itemized below.

[DR-1] Use of standardized data The standardization of the data for the UPLS system is required for the seamless information flow of the product for the entire product life cycle. To support the e-manufacturing paradigm, where Design Anywhere, Build Anywhere, Service Anywhere is accentuated, use of international standard data model is strongly recommended.

[DR-2] Data interoperability Since it is impossible to enforce the same standardized data model on all the stakeholders, existing data models used by the various stakeholders (henceforth *local* data) should be interoperated with the standardized data (henceforth *unified* data). The ontology-based technology for information integration is necessary to transmit and use data in various application systems.

[DR-3] Traceability of individual product information To realize the UbiDM paradigm, it is a requirement to trace and manage the information of individual products. Furthermore, stakeholders should be able to access the information related to individual products and its main components generated at any other stage of the product life cycle.

[DR-4] Data encryption and user authentication Since an enormous volume of information flow on UPLI will be requested and used by many stakeholders, they should be accessible in a limited fashion. For such a purpose, UPLS system should be able to manage access authentication for various information and to control the certification of users requesting these information by their application systems.

3 Architecture design for UPLS system

3.1 SOA approach

As various stakeholders and application systems use different software on the heterogeneous platform used in the various devices, it is very important to support the interoperability for leveraging various activities throughout the entire product life cycle (FR-3). For the interoperability between application systems, middleware technologies (e.g., Distributed Computing Environment [8, 9], Common Object Request Broker Architecture [10], Remote Method Invocation [11]) have been used. These middleware technologies ensure interoperability only if the same middleware is used. On the other hand, in the environment in which a variety of stakeholders are involved, such as UPLS, the middleware methodology is not promising at all. Considering the above facts, we take a service-oriented architecture approach to UPLS.

Service-oriented architecture (SOA) is considered a new computing paradigm providing the design framework to integrate distributed applications, so that their functionality can be accessed as services on a network [12]. A service is an independent piece of functionality that can be discovered on the network and that is interfaced only with the service description describing what it can do and with which ways it can be interacted. An interoperable service description consists of a protocol and a data format using open standards, so that each of the potential clients of the service understands the format. Interoperability is achieved by supporting standardized protocol and data formats of the service's current and potential clients.

Figure 2 shows the composition of the SOA. Service consumers find the required service and request if the service is suitable for their current needs and invoke a service to the service provider who registers the required service in the service directory.

3.2 SOA approach to UPLS system

From the requirement perspectives of UPLS system, SOA is a very powerful approach. Specifically, (1) in the UPLS system, application systems at different stages of the product life cycle should use the functions on different systems at the other stage (FR-3). By SOA, a service can be invoked over distributed networks regardless of the physical location, the implementation methodology such as c++, c#, java, and so on, or the implementing platform, including Windows, Linux, other embedded OS, and so on.

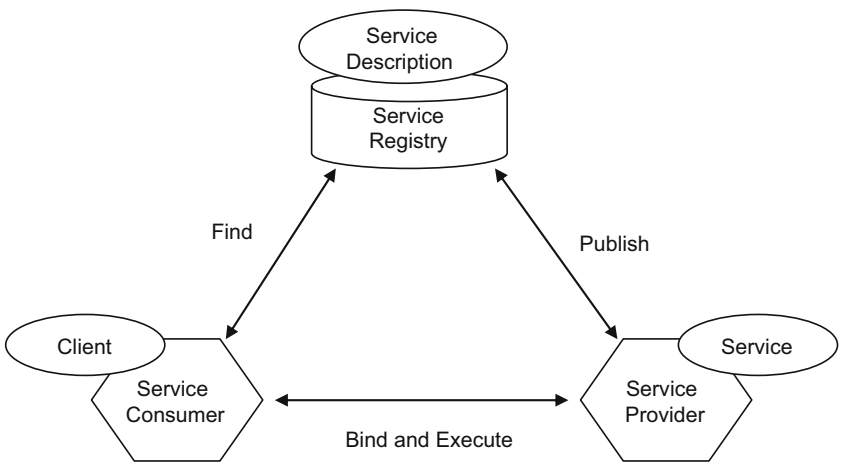

Fig. 2 Components of the service oriented architecture

(2) To integrate various systems with minimal efforts (FR-4), it is required to remove many redundant and non-reusable functions through the entire product life cycle. Services can be reused by numerous consumers by interfacing the service description that defines inputs, outputs, and associated semantics of the service in a standardized format.

To deploy SOA architecture, we need to define the following: (1) service list expected from the stakeholders of UPLS system and (2) providers and consumers of the services within UPLS system.

Based on the functional requirement, we converted the service list into five groups: (1) data acquisition services required to get the real-time data from the real world (FR-1), (2) data management services to provide all the participants with the product life cycle to store, retrieve, and transfer required data (FR-2), (3) collaborative process services to support the business process between companies by encapsulating each process or a sequence of the process as service (FR-5), (4) user-interaction services, which make it possible for the human operators to communicate with other stakeholders using various computing devices (FR-3), and (5) application interface services for the use of the various application systems by decomposing them into sub-systems or sub-functions (FR-4).

Based on the five services and technologies in the UbiDM paradigm, we classified the service providers and consumers into four units, each of which can be used as a building block in implementing UPLS system as follows: (1) D2U (Device-to-UPLI) unit—ubiquitous devices (such as RFID, sensor network, web camera, and so on) to be used as a means of providing a data acquisition service, (2) UPLI database unit—various databases in the UPLI to provide the data management service, (3) business process manager unit—the process-related technologies in the UPLI for providing collaborative process service, (4) user and application interface unit—the interface service between many services and user device/application system to perform the task requested by the user. Moreover, all the required functions will be encapsulated as services and interfaced by service descriptions stored in the service repository. The service repository contains not service itself, but the service description in a standard format. By referencing the service description, the user can get the service provided by the relevant service provider wherever it is.

Based on the discussion so far, we propose a generic architecture of UPLS system as shown in Fig. 3. The proposed architecture is composed of five units: (1) UPLS core unit, (2) UPLI database unit, (3) D2U unit, (4) business process manager unit, and (5) user and application interface unit. Compared with the contemporary PLM solution, the UPLS core and D2U units are new. Furthermore, the UPLI database has a large deviation from the contemporary PLM solution due to the extended scope of the UPLS. Detailed explanations for these units will be given in Sections 3.3, 3.4, and 3.5.

It is worth noting there are four *interface agents*, one for each unit, in the UPLS core unit (Fig. 3). This is due to the fact that SOA defines a kind of structure to find, request, and provide services but does *not* specify the methodology on how to use the services. Therefore, it is necessary to define a systematic mechanism for finding, requesting, and providing services (that may exist anywhere in the network) in an autonomous fashion. For such a purpose, we adopt a software agent concept defined as an autonomous, goal-oriented computer program, which formulates actions toward its goals, reacts to external stimuli, and communicates with other agents [13].

Based on this concept, we define the agent as an *interface* agent. The basic architecture of the interface agent is shown in Fig. 4. The interface agent is composed of four modules and one database: (1) communication module, (2) decision making module, (3) service interface module, (4) ontology integration module, and (5) service history DB.

The communication module monitors the messages to the interface agent and delivers them to the decision-making module. According to the messages, the decision-making module makes a command to the service interface module to meet its goal. The service interface module finds, requests, and provides services to the three sub-modules: (1) service search module, (2) service request module, and (3) service provide module. Each module uses standardized protocols, e.g., Simple Object Access Protocol as a messaging protocol, and Web Service Description Language (WSDL) as a service description language defined in eXtensible Markup Language. Ontology integration module describes the relationship between the unified product life cycle data model and local data model. Service history DB is a kind of database to save the history of the services that have been performed in this interface agent.

3.3 UPLS core unit

The UPLS core unit is a set of interface functions to support all the units that can carry out the services requested by the stakeholders. It is composed of four interface agents: (1) ubiquitous device interface agent, (2) user and application interface agent, (3) business process and knowledge interface agent, and (4) UPLI interface agent, and service repository composed of five repositories as shown in Fig. 3. The details are given below.

Ubiquitous device interface agent This interface agent connects D2U unit and other units requesting the in-use product data. This agent provides the requested data to the user device or application systems.

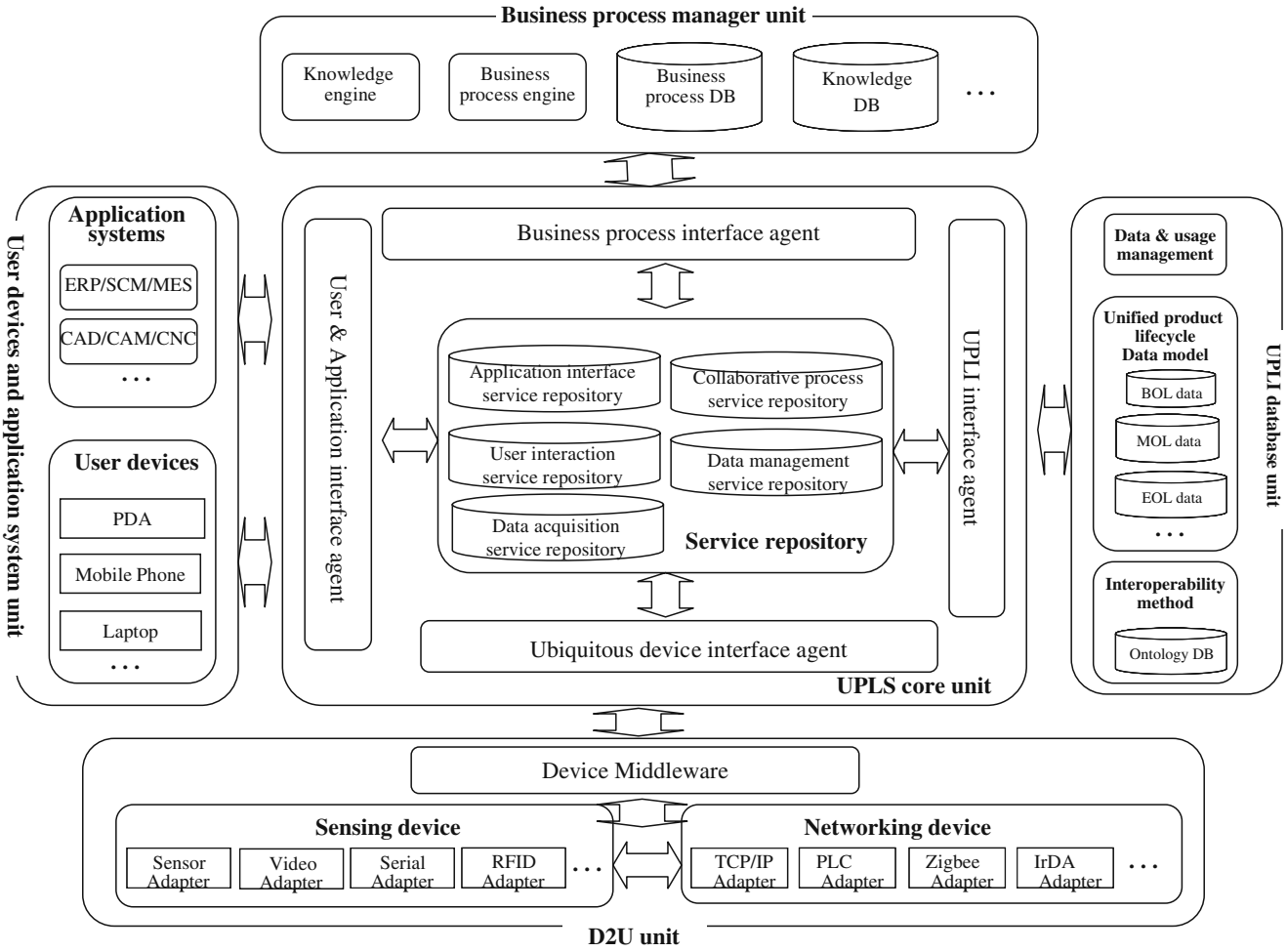

Fig. 3 Generic architecture of UPLS system

Business process interface agent This interface agent provides business process and knowledge to the other units of UPLS system. Business functions, ranging from simple request–reply functions to the full business process interactions, are encapsulated as services and registered in the service repository.

UPLI DB interface agent UPLI DB interface agent makes it possible to exchange the *local* data (used by application system of stakeholders) with the *unified* data (data model covering the whole life cycle used by UPLI). This agent provides the services defining mapping relationship between local and unified data model.

User and application interface agent This agent connects the human operator or application system to the UPLS system. To provide services (data or collaborative process, etc.) requested by the user, this agent searches via the service repository, and requests to the service provider.

Service repository Service repository contains the service descriptions registered by the interface agents. Each interface agent can search the desired service by referencing service descriptions written in a standardized protocol such as WSDL.

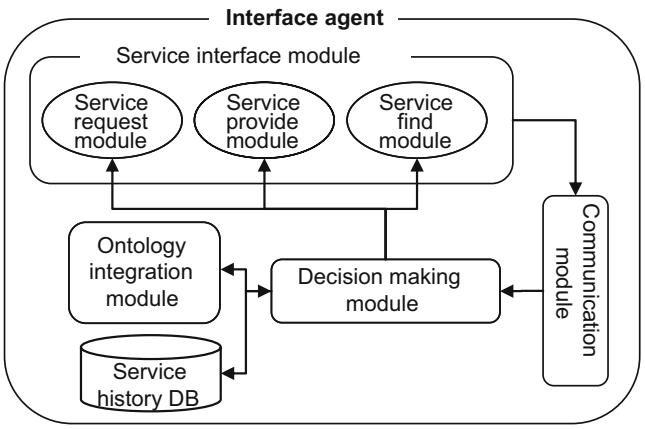

Fig. 4 Modular architecture of service interface agent

3.4 D2U unit

This unit is designed to get the in-use product data, throughout the entire product life cycle, and to transfer them to the other units. D2U unit is categorized into three types: sensing device, networking device, and middleware.

Sensing device Sensing devices such as embedded devices, RFID, active badge, multi-function sensors, and smart cards collect the shop floor data and in-use product data at MOL/EOL stages.

Networking device Networking device transfers the obtained data from sensing device to the other units. The networking device is grouped as wired network technology such as TCP/IP, power line cable, and wireless network technology such as wireless personal area network, wireless local area network, IEEE802.11, wireless metropolitan area network, and so on.

Device middleware Device middleware collects and filters raw data from sensing device and transfers data to the other units.

3.5 UPLI database unit

The UPLI database unit is based on the unified product life cycle data model (requirement DR-1) to provide ontology service (for interoperability between the local data and unified data, DR-2) in a limited fashion (authentication and security management, DR-3). The details are given below.

Unified product life cycle data model This is the key for achieving seamless information flow in UPLI. For ontology integration, the data model must be harmonized with the existing international standards such as ISO 10303 STEP [14], ISO 15531 MANDATE [15], ISO 14649 STEP-NC [16], and so on. One for the machine tools product will be derived in Section 4.

Data and usage management As a product passes through the entire life cycle, related information, such as design, production, and usage and history information, must be managed systematically. Furthermore, authentication and security management for accessibility is necessary.

Interoperability method To convert the local data into the unified data used in UPLI and vice versa, an interoperability protocol must be established. For such a purpose, ontology DB storing mapping relationship between the unified data model and local data model is necessary.

4 Unified data model for machine tools

For seamless information flow and interoperability between the stakeholders, a unified product life cycle data model is required. In this section, we develop a unified product life cycle data model for machine tools. From an architectural point of view, the architecture developed in Section 3 is generic and can be applied regardless of product type. In this sense, a data model for machine tools can be thought of as an example product for the specification of the UPLS system. Note also that machine tools is a capital intensive product used in the major industry having distinct life cycles involved in a variety of stakeholders. In other words, the impact of the UPLS system can be accentuated by the UPLS for machine tools. Furthermore, the machine tool data model is also of great concern and currently under establishment by ISO/ASME community including ISO TC184/SC1 and SC4, ISO TC39/SC2, ASME B5/TC56, etc.

4.1 Design consideration

Before designing the data model, we need to consider some aspects that the unified data model should have. The following principles are taken into consideration in designing the data model:

1. Comprehensiveness: The data model should be comprehensive enough to cover the important concept, terminology, and attributes used in the machine shop, MTB catalogue, research papers, software systems (e.g., CAD/CAM/CAE), etc.
2. Uniqueness: Each object defined in the data model should be uniquely defined within the whole schema and other related schema defined in the existing standards (e.g., ISO).
3. Completeness: Data model should reflect the input and output information used in the life cycle activity including BOL, MOL, and EOL.
4. Extensibility: The structure of the data model should be designed such that a new type of data can be readily accommodated without modifying the existing structure.
5. Inter-operability: The data model should be interoperable with existing standards for which harmonization is required.

In addition to the general rules for data modelling, we investigated previous works from the following perspectives: (1) information framework modeling that can be used to design overall structure of data model, (2) information contents modeling that can be used to design detail description of each data, and (3) reference data model for machine tools including the international standard related to the product life cycle of machine tools

(Table 1). Note that, at the moment, the international standard data model is defining a fraction of data model for BOL mostly developed by the ISO TC39 and ASME B5/TC56 committees.

The main procedure that we utilize for developing the unified data model is functional modeling related to the product life cycle via IDEF0, followed by information contents derivation.

4.2 Functional modeling for the product life cycle of machine tools

To capture the functional data requirements for the product life cycle, we analyzed the application activity by IDEF0 (Integration DEFinition) model. Note that the IDEF0 modeling method is a powerful tool typically used by the ISO community to determine data requirement of each activity in terms of input, control, output, and mechanism.

Figure 5 shows the top-level IDEF0 diagram for the product life cycle of machine tools. We divided the main functions into three stages: BOL, MOL, and EOL. By analysis of each function, we specified the required information (input) and generated information (output) during each stage. Figure 6 shows the activities related to the BOL stage of the product. A variety of information is generated by various activities including machine tool design, manufacturing process design, machine tool production, and inspection. Largely, the generated data includes specification of the machine tools and main components, assembly/disassembly information, BOM data, configuration of machine tool including kinematics information, and manufacturing information such as operation/routing data and resource data. More data can be derived by decomposing each function into sub-functions (third level, forth level of IDEF0).

Similarly, for MOL and EOL, we analyzed detailed activities and extracted the required and generated information. For MOL, product specification data generated at BOL (such as configuration, assembly/ disassembly, and BOM) and product-related data including planning and supplying information are necessary. The generated data include status of the product, usage history, maintenance history, breakdown information, and so on. For EOL, product specification data and MOL data such as usage condition, maintenance history, part replacement, and the like are required. Furthermore, additional information for specifying options and constraints on the EOL activity, such as remanufacturing, deposing, etc., are necessary. The generated data during EOL includes the machine tool lifespan and its components and the recycle/reuse information of the machine tools and components. Again, more data can be derived by decomposing the functions into sub-functions.

Table 1 Classification of related research and international standards on machine tools

Usage	Product life cycle			Subjects	Author/institute	Types
Information framework modeling	O			A product information modeling framework for product life cycle management	R. Sudarsan et al. [24]	Academic research
Information framework modeling	O	O	O	Typology of standards and their convergence for PLM	Eswaran Subrahmanian et al. [25]	Academic research
Information contents modeling		O		Life cycle information associated with decisions made in the usage phase of products	Auto-ID Center	Academic research
Information contents modeling			O	Information requirements for End-of-Life Decision Making	Auto-ID Center	Academic research
Information contents modeling	O			Manufacturing related data according to the source where the data comes from	Auto-ID Center	Academic research
Referenced data model	O			ASME B5.59–1: "Data Specification for Machine Tool Performance Tests	ASME B5/TC56 [26]	International standard
Referenced data model	O			ASME B5.59–2: "Data Specification for Proper ties of Machine Tools for Milling and Turning"	ASME B5/TC56 [27]	International standard
Referenced data model	O			STEP AP 203: Configuration controlled design	ISO TC184/SC4	International standard
Referenced data model		O		STEP AP 239: PLCS (Product Life Cycle Supports)	ISO TC184/SC4	International standard
Referenced data model	O			ISO 14649: Data model for Computerized Numerical Controllers	ISO TC184/SC1	International standard
Information contents modeling	O			Machine tool catalogues for various machine tool types from different machine tool builder		Machine tool catalogue
Information contents modeling	O			Simulation softwares for the machining in the machine tools		Simulation software

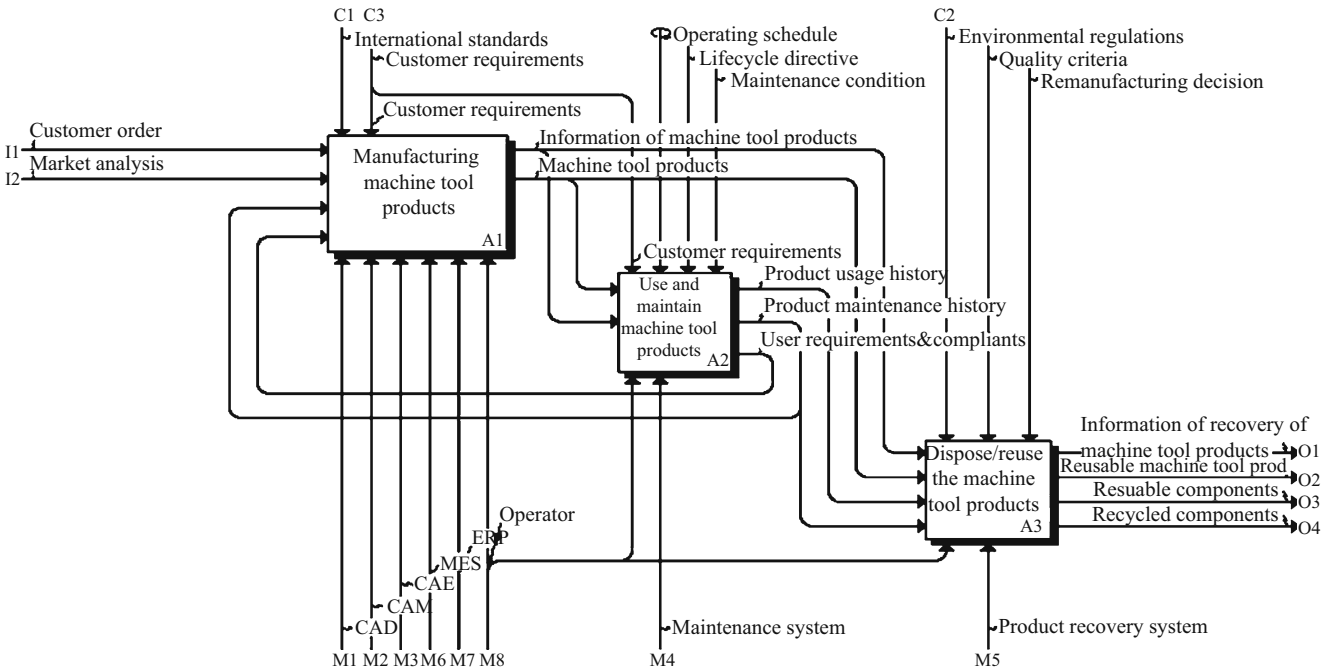

Fig. 5 Functional diagram for the product life cycle using IDEF0 modeling

4.3 Information contents modeling

Based on the functional modeling, we extracted the required and generated data. Based on those, we developed information content from the unified data model for machine tools. In defining terminology (ENTITY) and attributes, we utilized the available information including Table 1, catalogue of machine tools, commercial software on CAD/CAM/CAE, PLM, etc.

The result is shown in Fig. 7. More precisely, Fig. 7 is the overall structure of the product life cycle data model represented in an EXPRESS-G (formalized as ISO 10303–11 [17]). It should be remarked that: (1) the overall structure shown in Fig. 7 reflected only the major activities represented by IDEF0 in Section 4.2, and (2) more detailed data associated with the sub-activities can be readily accommodated by extending the schema.

For the BOL model, we defined the (1) detailed specification of the machine tools and its components, assembly/disassembly information, and kinematics information; we classified the (2) types of machine tools and main components and designed detailed attributes of the machine tools and its component; (3) we defined the assembly information relationship between the machine tools and its main components; (4) we developed a data model for kinematics information schema to represent the kinematics of the machine tools, which is harmonized with ISO 10303–105 [18]; and (5) we extracted other BOL data was designed by referencing the existing international standard such as ISO 10303-203 (CAD data) [19], ISO 10303-239 (product life cycle support) [20], ISO 10303-240 (macro-process planning) [21], ISO 14649 (micro-process planning) [16], and ISO 15531 (resource information model) [15].

In terms of EXPRESS-G schema shown in Fig. 7, the data model for machine tools, named *Machine_tool_ product* entity (italics mean the name used in the schema) (1) has four subtypes (*NC_machine_tool*, *complex_machine_tool*, etc), and (2) is a subtype of *Product* defined in ISO 10303-239. *Machine_tool_product* is composed of many components (*Machine_tool_axes*, *Machiint_tool_spindle*, *Auxiliary*, etc) with assembly relationship defined in the *Assembly_ component_relationship* (referenced from ISO 10303-239). Note that the oval shape means objects from other international standard model (in this case, ISO 10303-239). Furthermore, *Machine_tool_product* has an attribute of *Kinematic_structure* with attributes of *Kinematic_ joint* (referenced from ISO 10303-105), etc. In this way, the kinematics data model, machine type, assembly information, etc., used in BOL stage are unambiguously defined, harmonized with the existing standards.

Likewise, we designed MOL data for the usage of machine tools under the framework of ISO 10303-239: the international standard for the product life cycle support. We first classified the MOL information into four types: *Work_definition*, *Activity*, *Observation* and *Breakdown*, and specified detailed data for each class based on the characteristics of the machine tools as follows.

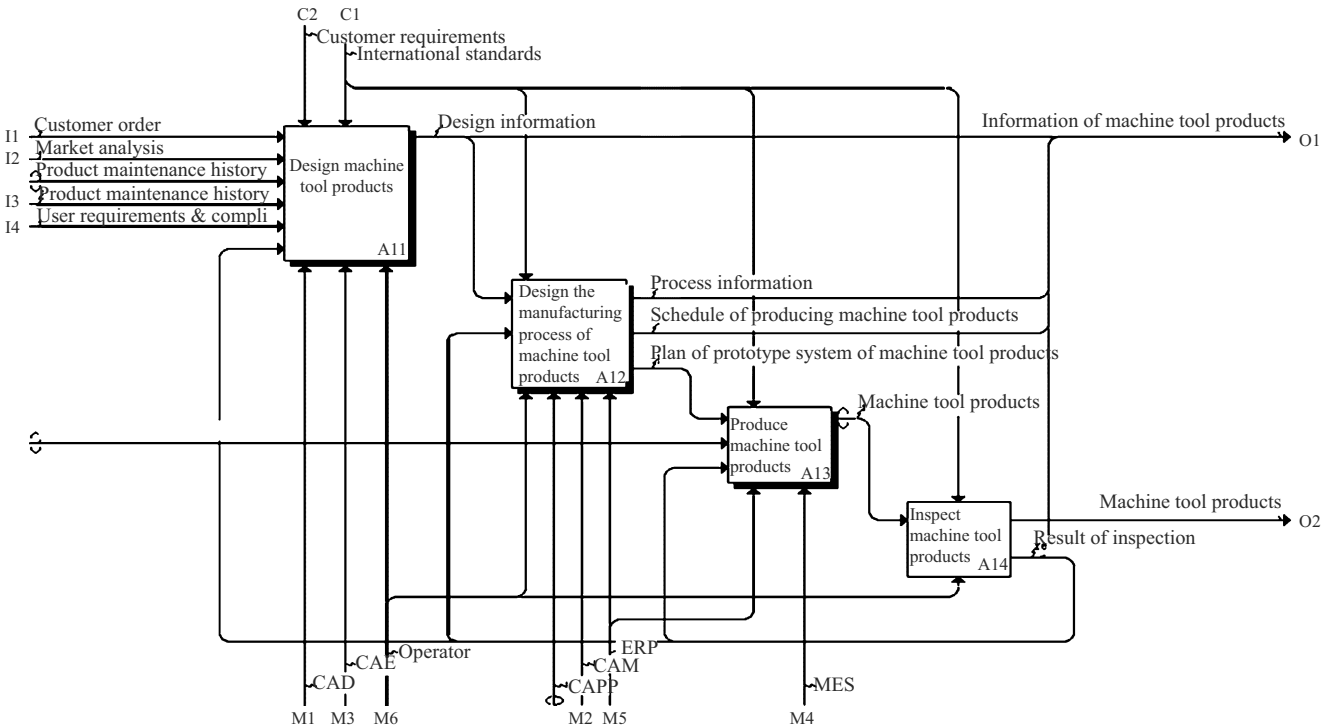

Fig. 6 Functional diagram for the Begin-Of-Life using IDEF0 modeling

Work_definition, defining the scheduled work during MOL, has an attribute of *Activity*, which is classified into three subtypes, e.g., *Maintenance_activity*, *Diagnosis_activity*, and *Machining_activity*. In addition, the status of machine tools is observed using *Observation* by the various sensors (*Sensor_observation*) and context awareness technology (*Context_observation*), and so on. And, we defined the details of the *Breakdown* information into five: (1) *Functional_breakdown*, (2) *Physical_breakdown*, (3) *System_breakdown*, (4) *Zonal_breakdown*, and (5) *Hybrid_breakdown*. Furthermore, we defined more details of the each breakdown based on the causes of the breakdown of machine tools. For instance, *Functional_breakdown* is classified into the following: (1) *Mechanical_component_breakdown*, (2) *Electronic_component_breakdown*, (3) *Motor_breakdown*, and (4) *Aux_breakdown*.

In regard to the EOL data model, we defined *EOL_activity* as being in one of the four following categories: *Remanufacturing*, *Repair*, *Recycling*, and *Dispose*. This is based on literature related to recycling and remanufacturing area [5]. The attributes of *EOL_activity* include (1) the life cycle data, which can be referenced from machine tools and its parts (i.e., *Machine_tool_product* and *Part*), generated by BOL and used by MOL, (2) *EOL_considerations*, which imposes constraints on the recovery options, and (3) *EOL_report*, including information about the result of the EOL processes, such as the life of machine tools and its component, product flaw information, product usability information, recycle/reuse information, etc. This is to support the physical activity that the EOL stakeholder (i.e., recycler or re-manufacturer) sends information to the BOL stakeholder (i.e., product designer and manufacturer). This can be thought of as a means to close the information loop among the stakeholders. In practice, the EOL report can be preciously used for various BOL activities to upgrade the product quality, performance, etc. Finally, *EOL_considerations* includes options for EOL activity, such as *Recovery_decision_criteria*, *Legislative_information*, *Market_information*, and *Corporate_policies*.

5 Operational scenarios

To show the validity of the presented architecture and data model, results based on implementation will be the most effective. However, since UPLS system is composed of many units running on an infrastructure encompassing broad range, implementation is not a practically feasible way at the moment. Thus, in this paper, we fabricated a usage scenario under the assumption that the UPLS system incorporating the presented architecture and data model is developed and available for the stakeholders. Since the UPLS system is applied to the various stakeholders, a number of scenarios could be developed for the machine tools; for example, machine tool designer, machine tool user, machine tool maintainer, machine tool re-manufacturer,

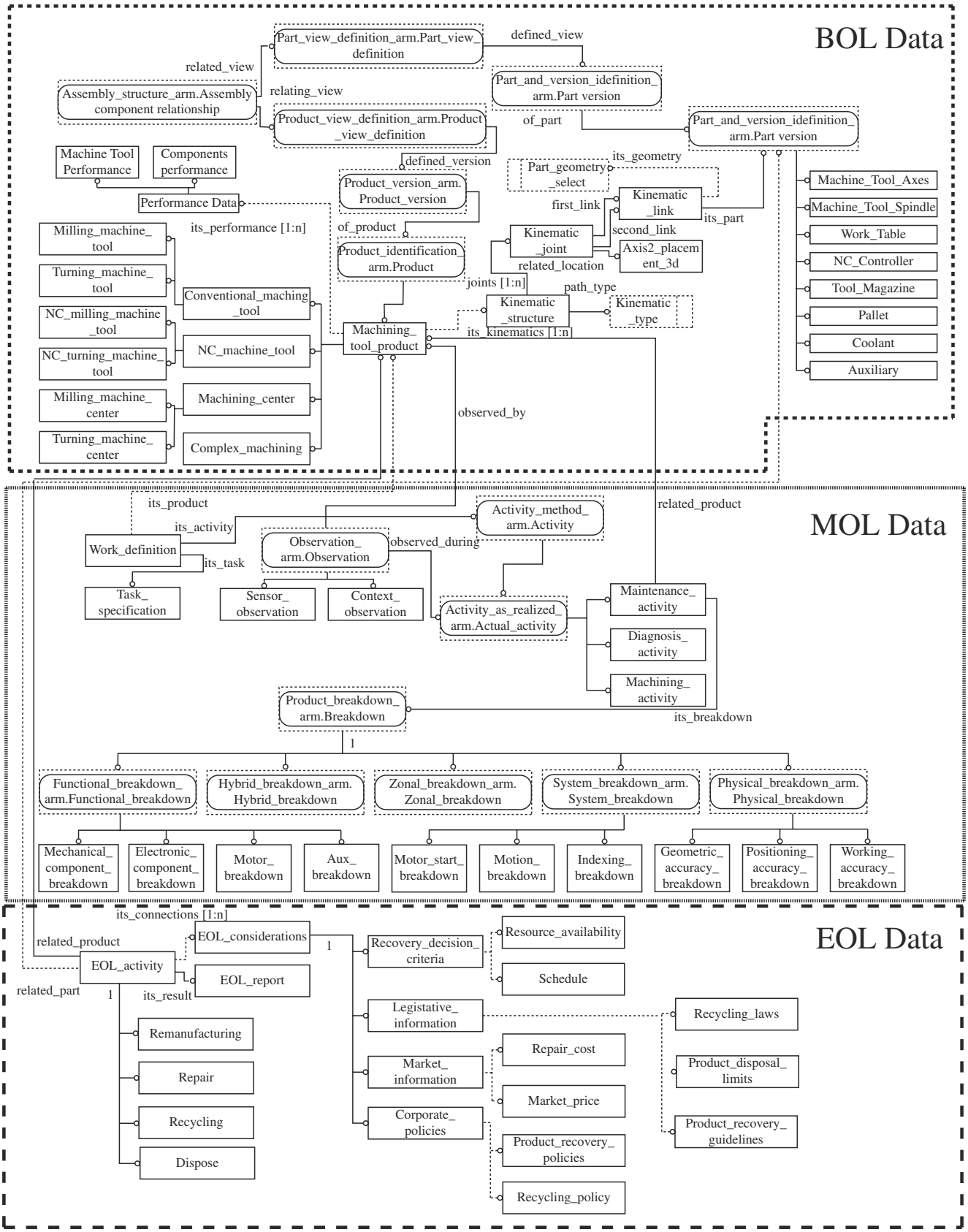

Fig. 7 Overall structure of unified product life cycle data model for machine tools

etc. Among them, we picked up a scenario for the designer within a paradigm of the so-called u-Design included in the UbiDM.

Consider a machine tool designer specialized in the spindle of the machine tools running over 20,000rpm. However, since the release of the product, there have been some complaints from the users including MTB, machine operator, and often machine tool maintainers. The AS-IS process in Fig. 8 is the conventional product development process. The designer relies on his/her know-how or database and the direct or indirect customer voice intermittently delivered from the salesmen and/or AS department. Furthermore, the designer and field tester perform the test under the preset conditions, without considering the real environments where the spindle is used, such as the types of machine tools, cutting conditions, shop floor temperature, etc.

From the information perspective, this is due to the fact that (1) the usage data obtained via sensors installed in the machine tools cannot be delivered to the designer due to lack of transferring mechanism, (2) the usage data intermittently obtained by the customer (operator) and maintenance report is not delivered to the designer automatically and systematically due to lack of appropriate mechanism and infrastructure, and (3) often, information collected by a complex procedure (e.g., customer survey) is not accurate and complete.

To solve the above problem, the designer needs to have 'in-use' information of the spindle from the operator and maintainer directly in an automated fashion. The in-use information can be carefully integrated into the new design of the spindle, in particular for finite analysis, physical test, and field test as shown in TO-BE process of Fig. 8.

To accomplish the above TO-BE process, we now present an operational scenario with UPLS system based on the architecture and data model presented in this paper. In this scenario, there are three pre-defined services: (1) the maintenance history service, which provides the maintenance history; (2) the ontology mapping service, which converts the local data model to the unified data model and vice versa; and (3) the in-use information service, which provides the usage information of the machine tools. The designer asks the maintenance history service to the maintainer and the maintainer returns information from his Maintainer DB using the ontology mapping service, which is defined in UPLI DB. And the customer can get and save the status of the machine tools periodically using D2U technologies. The designer can get information of machine tool status from the customer using the in-use information service. The overall operation scenario is shown in Fig. 9 whose details are described in what follows. Note that the number in brackets in the scenario is indicated in Fig. 9.

[1] To analyze the major problems that occurred during usage, the designer uses his/her application system (e.g., CAD/CAE) to get product history including breakdown information of the spindle unit from various maintainers. Then, the application system

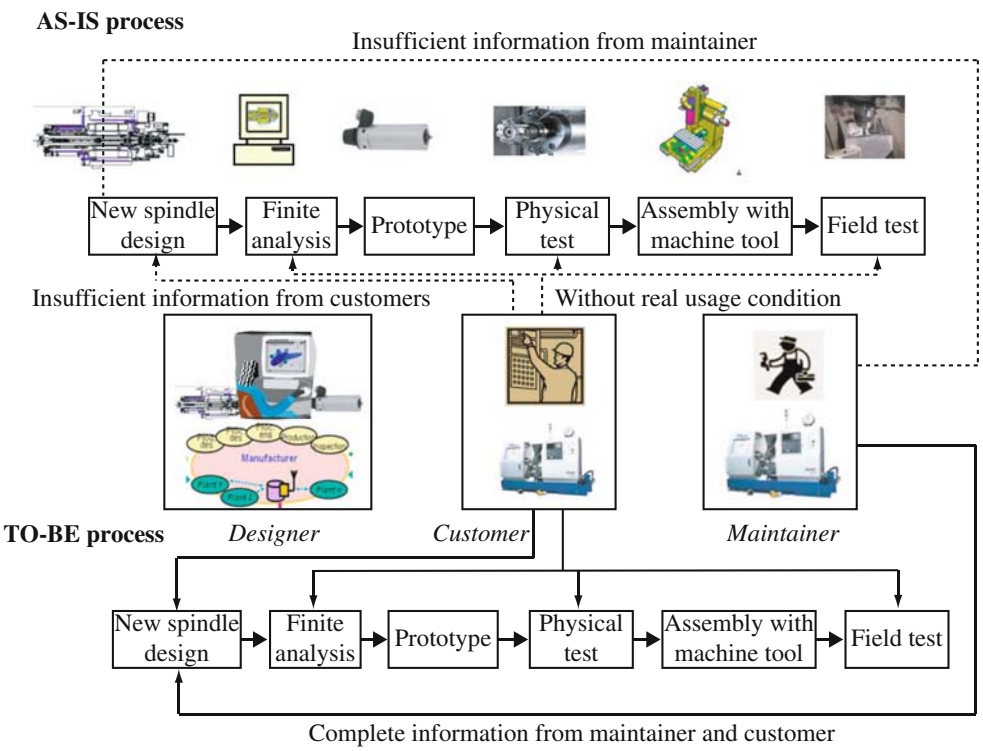

Fig. 8 As-is design process and to-be design process of high speed spindle unit

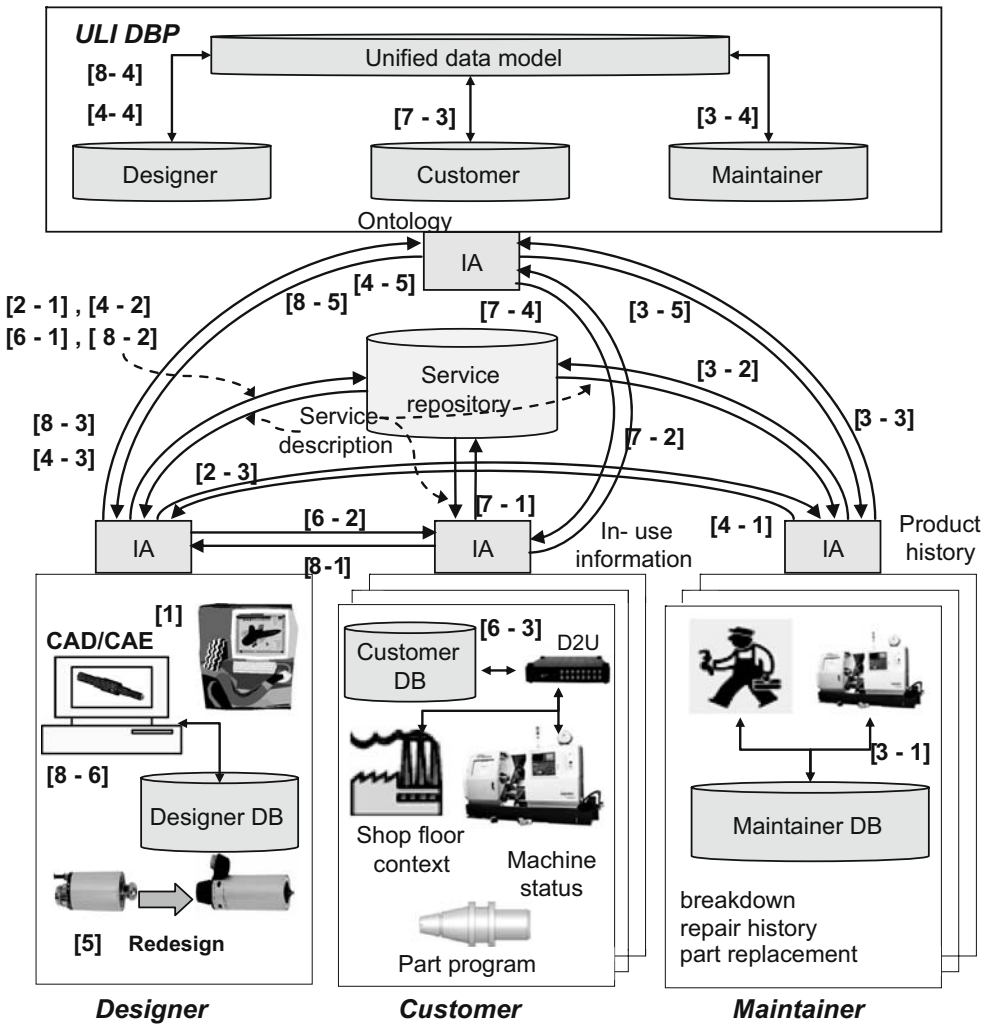

Fig. 9 Overall operation scenario with UPLS system for machine tools

connects the service repository to get the information via Designer interface agent (IA) as indicated [1] in Fig. 9.

[2] Upon receiving the request via IA of [1], *Service find module* of Designer IA searches service registered in service repository [2–1]. *Decision making module* of Designer IA references service description defined in service repository and activates *service request module* to obtain the product history from Maintainer IA [2–2].

[3] By the request from Designer IA, Maintainer IA searches the information from the Maintainer DB [3–1]. To convert the information into the standardized information defined in the unified data model, Maintainer IA finds its service [3–2] and asks UPLI IA [3–3], and obtains the standardized information from Maintainer Ontology DB [3–4] via UPLI IA [3–5].

[4] Maintainer IA transfers the standardized product history information to the original requester; i.e., Designer IA [4–1]. To convert the standard information to the local information used by the designer application system, Designer IA gets service description [4–2], and asks UPLI IA [4–3] and obtains the local information from Designer Ontology DB [4–4] via UPLI IA [4–5].

[5] Based on the spindle breakdown history information, the designer analyzes the breakdown cases and determines that the thermal deformation is the main cause. Thus, the main concept of the new design is to enhance the performance for the thermal deformation. To see the performance of the new design, the designer wants to analyze and simulate the spindle by the in-use information and shop floor environment.

[6] To obtain in-use information and environment, Designer IA obtains service description from Service Repository [6–1], asks Customer IA [6–2]. The Customer IA retrieves the in-use information from the Customer DB storing data acquired via D2U [6–3]. The information includes machining conditions, such as total operation time, cutting tools used, workpiece material, and sensed information including temperature, humidity of the shop floor, and vibration of the spindle, etc.

[7] Similar to Scenario [3–2] ~ [3–5], a series of procedures [7–1] ~ [7–4] are applied to convert local information from Customers to standard information.

[8] Similar to Scenario [4–1] ~ [4–5], procedure [8–1] ~ [8–5] is applied to convert the standard information to the designer local information. Using this information, the designer analyzes and simulates the newly designed spindle with the real operating conditions in the environmental information of the shop floor.

6 Conclusion

In this paper, we addressed the necessity for information supporting system encompassing the whole product life cycle via ubiquitous technology. Called a UPLS system, based on the paradigm of UbiDM, it supports the various stakeholders of the product life cycle by providing the real usage information for the product and environment in which the product is used. The two major procedures performed and subsequently described in this paper are (1) architecture for UPLS system and (2) data model for machine tools. We derived SOA-based architecture for UPLS system mainly for the sake of interoperability and fulfilling the requirements from various perspectives. For the specification of the generic UPLS system, we took the machine tools as an example by developing a life cycle data model for the machine tools.

To show the validity and usefulness of the developed works, we developed an operational scenario making use of UPLS system for the re-design of the high speed spindle unit. Through the operational scenario, we showed that UPLS system can be effectively used to perform the u-Design making use of the in-use information directly collected from the machine tools on the shop floor and failure/repair records from the maintainer (and possibly EOL report from the re-manufacturer/recycler even if it is not included in the scenario). The scenario derived is only one example to show the usage of UPLS system, and many more scenarios can be developed, e.g., u-Factory for manufacturing of the product [22] (categorized as BOL application) and u-PRMS (product recovery management system) for re-manufacturing and recycling [23] (categorized as EOL application), etc.

The developed works validated via the operational scenario can be used as the conceptual framework and data models for developing a UPLS system. However, since the UPLS system is a sizeable system considering its infrastructure, its development and completion will take time to realize. At the moment, a prototype system is under development.

References

1. Soga S, Hiroshige Y, Dobashi A, Okumura M, Kusuzaki T (1999) Products lifecycle management system using radio frequency identification system. 7th IEEE International Conference on Emerging Technologies and Factory Automation, vol. 2, October 18–21, UPC, Barcelona, Catalonia, Spain, pp 1459–1467
2. Parlikad A, McFarlane D (2003) The role of product identity in end-of-life decision making. Technical report, Auto-ID center, Institute of Manufacturing, Cambridge
3. Gross S, Parlikad A, McFarlane D, Fleisch E (2003) The role of the auto-ID enabled product information in a product's usage: a maintenance example. Technical report, Auto-ID center, Institute of Manufacturing, Cambridge
4. Chang Y, McFarlane D, Koh R, Floerkmeier C, Putta L (2002) Methodologies for integrating auto-ID data with existing manufacturing business information systems. Technical report, Auto-ID center, Institute of Manufacturing, Cambridge
5. Kiritsis D, Bufardi A, Xirouchakis P (2003) Research issues on product lifecycle management and information tracking using smart embedded systems. Adv Eng Inform 17:189–202 doi:10.1016/S1474-0346(04)00018-7
6. Gill S, Cormican K (2006) Support ambient intelligence solutions for small to medium size enterprises: typologies and taxonomies for developers. 12th International Conference on Concurrent Enterprising, Milan, Italy
7. Suh S (2007) UbiDM: A new paradigm for product design and manufacturing via ubiquitous computing technology. Technical report, Center for Ubiquitous Manufacturing, POSTECH, South Korea
8. Rosenberry W, Kenney D, Fisher G (1992) Understanding DCE. O'Reilly, Sebastapol, CA
9. Shirley J (1992) Guide to writing DCE applications. O'Reilly, Sebastapol, CA
10. Object Management Group (1996) The common object request broker: architecture and specification. Revision 2.0. http://www.omg.org
11. Grosso W (2001) Java RMI. O'Reilly, Sebastapol, CA
12. Matthew C, Laskey K, McCabe F (2006) Reference model for service oriented architecture 1.0. http://www.oasis-open.org
13. Nwana HS (1996) Software agents: an overview. Knowl Eng Rev 11(3):1–40
14. International Standards Organization, ISO 10303 (1994) Part 1: Overview and fundamental principles. ISO, Geneva, Switzerland
15. International Standards Organization, ISO 15531 (2004) Part 1: General Overview. ISO, Geneva, Switzerland
16. International Standards Organization, ISO 14649 (2003) Part 1: Overview and fundamental principles. ISO, Geneva, Switzerland
17. International Standards Organization, ISO 10303 (2003) Part 11: Description methods: The EXPRESS language reference manual. ISO, Geneva, Switzerland
18. International Standards Organization, ISO 10303 (1996) Part 105: Integrated application resource: Kinematics. ISO, Geneva, Switzerland
19. International Standards Organization, ISO 10303 (2005) Part 203: Application protocol: Configuration controlled 3D design of mechanical parts and assemblies. ISO, Geneva, Switzerland
20. International Standards Organization, ISO 10303 (2005) Part 239: Application protocol: Product lifecycle support. ISO, Geneva, Switzerland
21. International Standards Organization, ISO 10303 (2005) Part 240: Application protocol: process plans for machined products. ISO, Geneva, Switzerland
22. Suh S (2007) u-Factory for POSCO. Technical report, Center for Ubiquitous Manufacturing, POSTECH. http://u-mfg.postech.ac.kr
23. Um J, Yoon J, Suh S (2007) Product data modeling & architecture design for product recovery management system. 5th International

Symposium on Environmentally Conscious Design and Inverse Manufacturing, Tokyo, Japan
24. Sudarsan R, Fenves S, Sriram R, Wang F (2005) A product information modeling framework for product lifecycle management. Computer-Aided Des 37(13):1399–1411 doi:10.1016/j.cad.2005.02.010
25. Subrahmanian E, Rachuri S, Fenves S, Foufou S, Sriram R (2005) Product lifecycle management support: a challenge in supporting product design and manufacturing in a networked economy. Int J Prod Lifecycle Manage 1(1):4–25 doi:10.1504/IJPLM.2005.007342
26. ASME B5/TC56 Committee, ASME B5.59-1 (2005) Data specification for machine tool performance tests. ASME, New York
27. ASME B5/TC56 Committee, ASME B5.59–2 (2005) Data specification for properties of machine tools for milling and turning. ASME, New York