An Architecture of Equipment Connectivity in an e-Business Environment

Sue X. Ye and Robin G. Qiu, *Member, IEEE*

***Abstract* -- This paper presents an integration architecture for equipment on the shop floor, which could provide suitable, intelligent, and cost-effective shop floor equipment connectivity for manufacturers' e-business solutions. First, a generic equipment interface structure and engine is defined as a platform capable for customization, leading to the creation of needed and designated equipment connectivity. By loading configurable equipment connectivity components and an equipment characteristic descriptor, generic equipment connectivity is then enabled for supporting proper communication protocols. In addition, a universal data format for the created equipment connectivity is considered, aiming at facilitating effective communication to heterogeneous host applications from the enterprise integration perspective. As a result, the proposed architecture provides equipment with a downloadable, configurable, extensible, and universal equipment connectivity.**

***Index terms* -- Equipment Connectivity, Architecture, Configurability, and Manufacturing Information System**

I. INTRODUCTION

Business environments are challenged by the rapid changes in business needs and customer demands, for instance, the increasing rate of new product introductions and technology innovations, a variety of regulations for factories in different geographical locations, and more complicated supply chains due to the globalization of manufacturing. As a result, the conduct of manufacturing businesses has become more and more dependent upon well defined, developed, and deployed information systems. Due to the explosive growth of data acquisition and analysis requirements, an increasing number of manufacturing processes, and numerous types of nonintegrated information flows, the delivery of accurate and pertinent information to the proper users at the right time becomes extremely difficult.

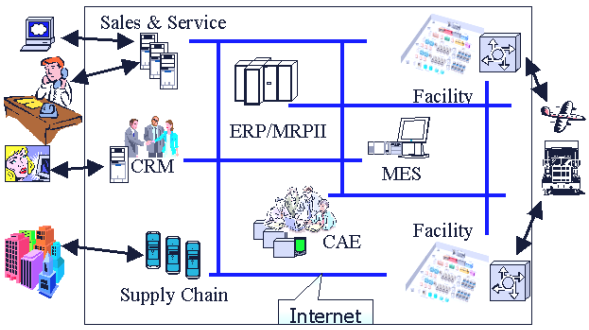

Figure 1. A Well-integrated and Operational e-Business Environment

A well-integrated and operational e-business environment by deploying enterprise-wide information systems is abstractedly illustrated in Fig. 1, where ERP stands for Enterprise Resources Planning, MRPII for Manufacturing Resource Planning, CRM for Customer Relationship Management, CAE for Computer-Aided Engineering including computer-aided design and manufacturing (CAD/CAM), and MES for Manufacturing Execution Systems. Note that it is essential for the completeness of both "e" and "business", i.e., orders are taken over the Internet and the products are made and delivered as promised [5]. Accordingly an enterprise-wide information system is divided into a front end (i.e., business- or office-level system) and a back end (i.e., plant-level system) [18]. The front end normally includes all the office planning, scheduling, sales, supply chain, and services applications, while the back end includes applications directly tied with value-adding manufacturing operations, such as logistics, engineering, manufacturing execution, and shop floor controls. Apparently due to the emergence of e-businesses and their supportive technologies, manufacturing enterprises could increase revenues by increasing sales or services through efficient multiple channels on the front end, while decreasing expenses using optimal and cost-effective manufacturing on the back end [5, 8].

Unfortunately, no such a well-integrated enterprise-wide information system exists yet in the world even though many world-class manufacturing enterprises have deployed many kinds of business unit domains (e.g., human resource, finances, engineering, supply chain management, etc.) information systems. Because of the non-existence of a standardized

---

[1] Sue Ye is with GL AgilityTech, Inc. Collegeville, PA 19426 USA (e-mail: xye@GLAgilityTech.com)

[2] Robin Qiu is with Penn State University, Malvern, PA 19355 USA (e-mail: robinqiu@psu.edu)

* The project was partially supported by Pennsylvania State University Great Valley 2002 SRS award and 2002-2003 Research Development Grant.

enterprise integration framework the investment in addressing different business issues from different business unit domains creates many isolated and sometimes overlapped information systems in most manufacturing enterprises [5, 18], thus forming many non-collaborating and unshared information silos. In particular, the great diversity of equipment types and lack of communication interface standards on the shop floor make the design and development of shop floor control systems capable for enterprise integration extremely complex and costly [15, 16].

When office-planning systems have no effective connectivity to shop floor equipment, i.e., no real-time and pertinent data (e.g., inventory availability, equipment utilization, supplier's status, customer orders, commitments, and production schedules) flowing upwards from the shop floor into office-planning systems, how could the executives of a manufacturing enterprise make informed business decisions [5, 14, 21]?

Apparently, one of the most challenging issues in deploying a well-integrated enterprise information system is how shop floor equipment would be able to be linked to office-planning systems systematically and cost-effectively in guarantee of pertinent information being consolidated into the context of needs in different business unit domains. From the architectural perspective, this paper proposes a scientific and systematic approach to the design and development of a universal equipment interface by taking enterprise integration and cutting-edge technology requirements into consideration. The ultimate goal is to make the proposed equipment interface configurable, extensible, and universal, leading into the standardization of such an integration-ready equipment interface, so the shop floor connectivity to office-planning information systems could be created cost-effectively and efficiently as required in the promising e-business environment.

II. CURRENT EQUIPMENT'S ARCHITECTURAL PROBLEMS

Shop Floor Control Systems (SFCS) running in manufacturing facilities are usually composed of hybrid hardware/software systems, such as programmable logic controllers, distributed numerical controls, supervisory control and data acquisition systems, robotics supervision systems, material control systems, and other computerized process control systems. Traditionally SFCS is only required for promptly activating and coordinating all the manufacturing processes and related activities on the shop floor such that all the assigned tasks from office-planning systems would be fully and efficiently performed.

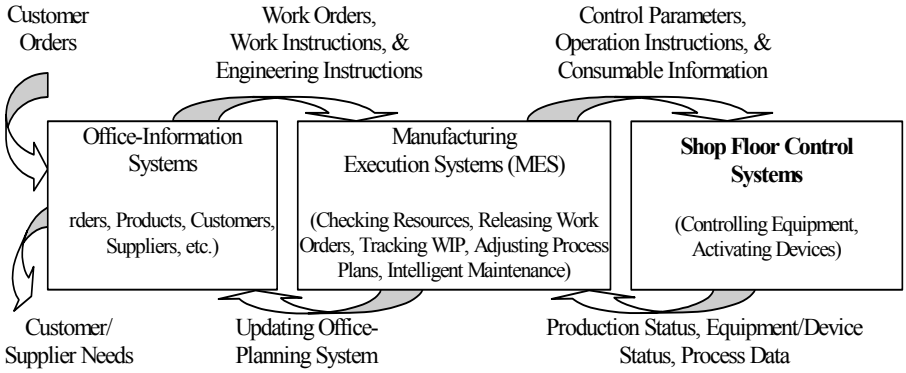

Figure 2. Data Flow in an Integrated Enterprise-wide Information System [11]

Manufacturing Execution Systems (MES) have been researched and developed for narrowing the gap between the shop floor and office-planning systems [8, 14, 21]. In great details Fig. 2 shows how office information systems, MES, and SFCS could be integrated from enterprise integration and data flow perspectives. In other words, it is MES that would bridge the gap between office planning and shop-floor manufacturing. On the one hand, MES takes in pertinent data from office-planning systems and ensures that their information is acted on intelligently on the shop floor; on the other hand, MES consolidates shop-floor data and put them into business-contexts, delivering accurate production states to office planning in a timely manner [10]. Therefore, only if there would exist effective connectivity from the shop floor way up to all the levels of office planning, SFCS could in real time execute all the tasks assigned by MES and in the meantime provide all the shop floor information (e.g., machine statuses, materials consumption, process data, alerts, etc.) for MES.

Obviously system integration to make shop floor data available in a timely way is fundamental to the realization of an integrated enterprise-wide information system, where effective connectivity to the shop floor is not an option but the cornerstone for the well-integrated enterprise-wide information system [11, 21]. The remaining part of this section

discusses problems related to three sequential and interrelated areas: *Integration Architecture, Generic Interface Structure and Engine, and Universal Data Format*.

As discussed in the last section, currently most office information systems are not designed to work at the shop floor level. Howells observed that these systems began as financial applications, moved into human resource, logistics, marketing, supply chain applications, and often as an afterthought, attempted to link to shop floor manufacturing systems [5]. Lacking a comprehensive design-for-integration consideration in the design stage, office information systems can hardly be connected to the shop floor. When there is a need, the point-to-point application programming interface (API) approach, consequently, has been adopted by most manufacturing enterprises, giving rise to extremely costly, non-flexible, non-scalable, and hardly sustainable system architecture [8, 9, 21]. Therefore, *the first problem* becomes what kind of *Integration Architecture* should be established for an integrated enterprise-wide information system, which allows for the system to be flexible, scalable, and easy to maintain.

Howells also pointed out in [5], the reality of manufacturing businesses was based on the fact that

- Regardless of the products manufactured, a financial package can be implemented in any industry.
- Regardless of the products manufactured, most logistics requirements are similar.
- It's in the manufacturing plant that the unique differentiators and business data exist.

In other words, different manufacturing equipment (i.e., processes) is used in making different products, such as autos, foods, chemicals, etc.). It is the variety of types of manufacturing equipment that complicates the shop floor connectivity. Because of the lack of an equipment-independent communication interface standard for computerization of control and management operations, enterprise/system integration on the shop floor becomes a proprietary, long-hauling, and costly process. Thus *the second problem* becomes what kind of *Generic Equipment Interface Structure and Engine* should be defined, which will be best suitable for the *Integration Architecture* and allow for customization in order to comply with different characteristics of equipment models.

On the shop floor data collected from equipment, devices, and other instrumentations are normally at the necessary level of granularity for fine-tuning manufacturing processes. For instance, they are widely used for statistical process control and run-to-run process adjustments [19]. But they will be overwhelming if directly acquired by upper office level information systems. As a matter of fact, they will most likely be of no value to a user without being filtered and put into the context of the relevant business unit domain [5, 16]. Besides, numerous types of built-in equipment interfaces provided by different suppliers typically use different communication protocols and accordingly different data formats. Correspondingly, *the third problem* becomes what kind of a *Universal Data Format* should be defined/adapted, which will not only well fit into the defined *Integration Architecture* and *Generic Equipment Interface* but also simplify information conveyance and data processing in an integrated enterprise-wide information system [16].

III. A PROPOSED ARCHITECTURE FOR FUTURE EQUIPMENT CONNECTIVITY

A well-known equipment controller specification has been defined by IEEE/NEMI [6]. Although the actual functionalities and software/hardware modules of a piece of equipment may vary substantially, the general constituents and their relationships are illustrated in Fig. 3. Equipment connectivity is the enterprise integration view of equipment interfaces. "It is preferable that off-the-shelf third party components are available to be integrated in the controller for this [integration] purpose so that the use of higher cost proprietary technology can be avoided." [1]

Implementation of the IEEE/NEMI Controller Specification involves complicated processes of software design and development, which is in the stage requiring a revolutionary change due to the emergence of e-business technologies. In particular, they are mainly driven by short-product life cycles and high quality of customer services. To meet the new challenges, software applications should be made the same as products in other industries like autos; they could be assembled from pre-build, fully-tested components systematically rather than writing codes line by line [3].

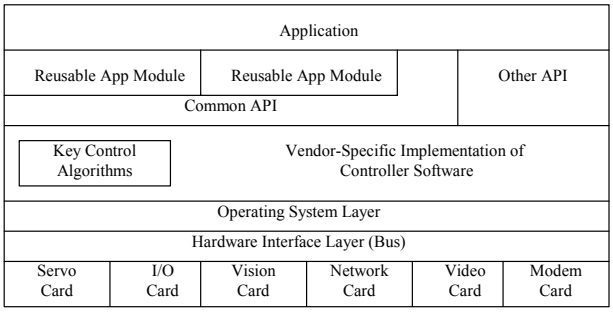

Figure 3. IEEE/NEMI Controller Specification. Adapted from [6]

A. Component-based Integration Architecture

Two lessons are learnt from the evolution of the component-based software development models. One lesson is the lack of consideration of application integrations; these components built in one computing environment (e.g., Common Object Model - COM) are hardly capable of being assembled in a different computing environment (e.g., Enterprise JavaBean - EJB). Another lesson is the lack of mechanisms for promoting reusability as the computing technologies advance; for instance, COM is eliminated from Microsoft's latest release of .NET framework due to the booming of Internet Technologies [12]. To overcome these issues, five fundamentals shall be fully considered in the proposed architecture for the equipment interface (Fig. 4). They are

- Integration: the proposed architecture shall make an equipment interface easy for integration.
- Equipment-independent: the built equipment interface baseline shall be equipment-independent in the sense that equipment interface communicating mechanisms are generic without being confined to equipment's idiosyncrasies.
- Configurability: each piece of equipment shall have its specific characteristics in terms of what information will be communicated with a host application and what kind of communication protocols will be used. Thus the interface shall be able to be self-configured by loading equipment-specific descriptor on the fly.
- Extensibility: the computing technologies advance, so do equipment, control, manufacturing, and others. The architecture shall allow the built interface to be capable for extension in light of functionalities.
- Technology adaptability: the proposed architecture and interface mechanisms shall be capable of being adaptive to the advances of networks and computing technologies.

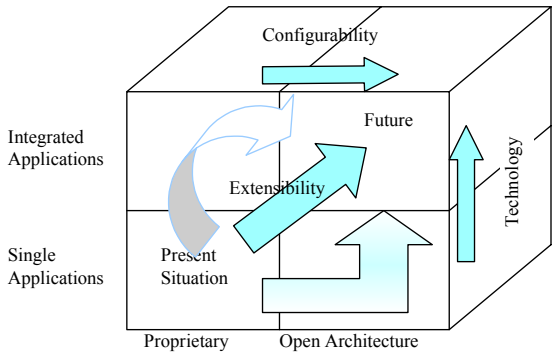

Figure 4. Evolution of Integration Architecture

An innovative equipment interface architecture is accordingly proposed in Fig. 5 (b). As seen in Fig. 5 (a) and (b), the proposed equipment interface will be equipment-independent rather than equipment-dependent. Different from providing some industrial standard-compliant and proprietary APIs for host applications, the proposed interface will provide a generic computing environment, where functionalities/services can be configured, customized, and reconfigured by downloading different components.

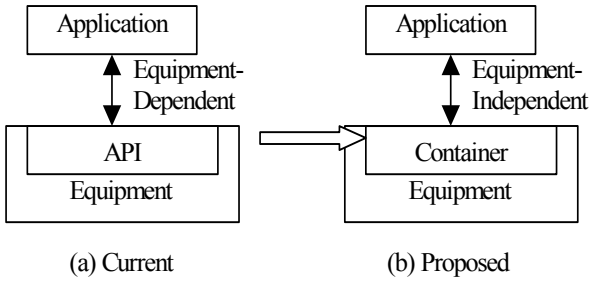

Figure 5. Defining Integration Architecture for Equipment Interface

B. Standardization of Equipment Interfaces

With reference to the capability of well-established equipment interfaces like open and modular architecture controller (OMAC) [13], a standard is in need for the proposed equipment interface to be easily integrated into manufacturing information systems. Fig. 6 shows a new abstracted equipment interface structure, where a new module embraced for the ease of enterprise integration and the state-of-the-art network connectivity technologies is proposed. The integration module could co-exist with traditional API as illustrated in Fig. 7.

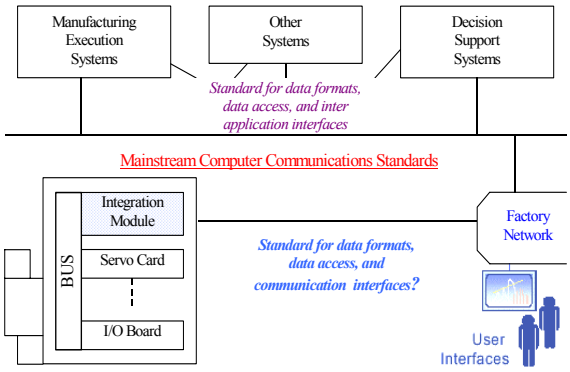

Figure 6. Integration View of Equipment Interface [19]

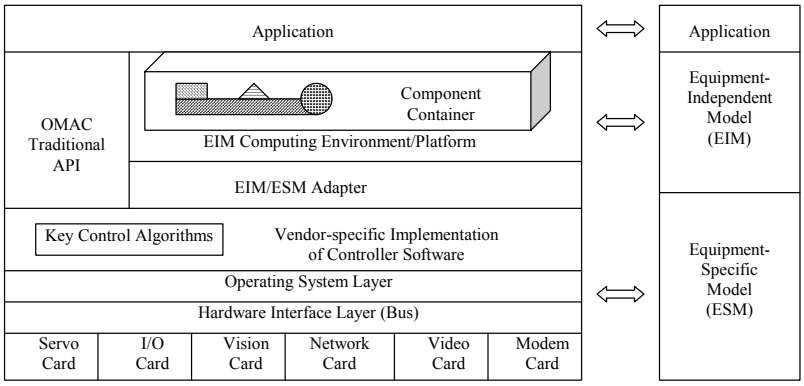

Figure 7. Structured and Standardized Universal Equipment Interface

Fig. 7 defines two levels of models for equipment controller; an equipment-specific model (ESM) contains the same bottom-half layers as defined in Fig. 3, while the top-half layers are replaced with an equipment-independent model (EIM) in order to make the newly structured controller suitable for the proposed component-based integration architecture. Without loss of generality, the EIM is divided into two parts. One part could be implemented as traditional OMAC API, while the other part could be implemented as proposed in this project.

C. Customizable Engine/Container

By incorporating JINI technologies (or "plug-and-play" network appliances) and OMAC API concepts into this proposed architecture [7, 13], a customizable and generic interface engine/container could be developed using the advanced software programming technologies including platform-independent computing, mobile computing, and distributed component theory (Fig. 8) [2, 12, 23]. It provides an environment for creating dynamically networked components, applications, and services, thus extending its capabilities from standalone communication to enterprise connectivity.

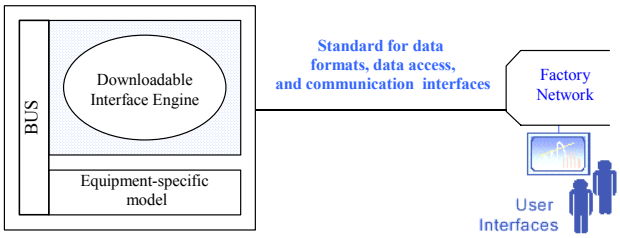

Figure 8. Customizable Interface Computing Engine for Integration

Because a variety of application types can act as manufacturing information systems, we standardize the vocabulary for the design and implementation of host application's responsibility. The termed EIM container is the configurable interface-computing environment. Fig. 9 shows the EIM container architecture, where an EIMObject defines the connectivity of a service component (e.g., EIMComp) and an EIMHome manages the service component's life cycle [23]. Components will be downloaded from a host application. In other words, it is the host's responsibility for customizing the network behaviors of a controlled machine.

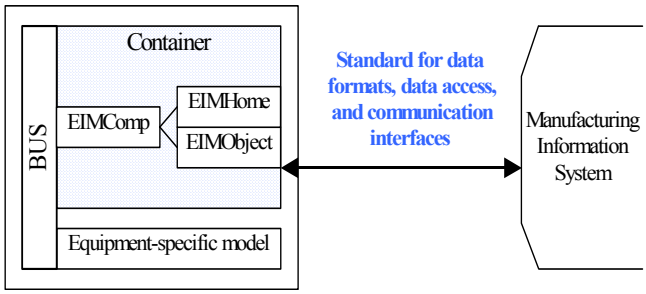

Figure 9. EIM Container Architecture

For example, when integration is required for a machine, some equipment-dependent components, communication components, and host-designated requirements components (e.g., EIM/ESM adapter, data acquisition, protocol enabler, standard compliance, events handler, finite-state machine, $\Psi = (\mathrm{T}, \amalg)$ coupler, data filter and fusion components) will be instantiated with designated parameters by its host application [15, 17]. As soon as these components get customized, they will be

downloaded into the equipment container. The customized equipment interface will then service as desired. Note that a computing component can be reconfigured and downloaded repeatedly as needed.

D. XML as the Universal Data Format

As is illustrated in Fig. 6, Standards for data formats and data access should fit into the mainstreams of computer communication standards across all the applications on both Intranet and Extranet in order to make synchronization, interoperation, and integration easier between different modules belonging to different information systems (e.g., equipment control and management systems, manufacturing information modules, and office-planning applications). Opposite to a proprietary format dictated by the sending and receiving application, for example, data formats in GEM [20], SMEMA [22], and G-Code [4], it is more cost-effective and universal to exchange information in a standard format whenever possible. Being nonproprietary, platform-independent, HyperText Transport Protocol (HTTP) – compatible, extensible, self-defining, and transformable, XML is widely accepted as the lingual franca of information systems [2, 16, 24]. XML has now been embraced by most of leading software vendors, such as Microsoft, IBM, Sun, and Oracle.

A fundamental role of XML is the interchange of data between applications. As illustrated in Fig. 8, standards for data formats, data accesses, and communication interfaces are essential to a generic equipment interface for system integration. Using the widely accepted XML technology for data exchange between equipment and host applications, a set of standard methods can be formulated to facilitate host communications and data conversion from equipment-specific parameters to an XML or vice versa. A schema for the proposed equipment connectivity is needed as well. The schema shall be extensible for modifications/customizations [12, 24].

V. CONCLUSIONS

This paper briefly discusses the architectural needs of configurable equipment connectivity in a future manufacturing information system. The proposed standardized and scientific approach could substantially reduce the cost and implementation cycle of a generic, customizable, and configurable equipment interface, leading to the creation of effective shop floor equipment connectivity to the office planning systems. When pertinent information is timely collected from and delivered to the shop floor, a well-integrated manufacturing enterprise-wide rather than only office-level information system could be deployed as anticipated in the future e-business environment. Therefore, manufacturers will maximally benefit from the deployment of e-business solutions as promised.

REFERENCES

[1] Chrysler, Ford Motor Co., and General Motors. *"Requirements of Open, Modular, Architecture Controllers for Applications in the Automotive Industry"*, December 1994. White Paper - Version 1.1.

[2] Cummins, F., *Enterprise Integration: An Architecture for Enterprise Application and Systems Integration*, John Wiley & Sons, Inc., 2002.

[3] DOM, http://www.w3.org/DOM/.

[4] HASS, "Discovery Center," http://www.hasscnc.com.

[5] Howells, R., "ERP Needs Shop-Floor Data," *Manufacturing Engineering*, pp. 54-62, Oct. 2000.

[6] IEEE/NEMI, "Low-cost Open Architecture Controller Specification," *www.ieee.org*.

[7] JINI, "JINI Technologies," http://wwws.sun.com/software/jini/.

[8] Koch, C., "Why Your Integration Efforts End Up Looking Like This …," *CIO*, pp. 98-108, Vol. 15, No. 4, Nov. 15, 2001.

[9] Linthicum, D., *Enterprise Application Integration*, Addison-Wesley, 2000.

[10] MESA, "The Benefits of MES: A Report from the Field," *MESA International – White Paper Number 1*, 1997.

[11] MESA, "Controls Definition & MES to Controls Data Flow Possibilities," *MESA International – White Paper Number 3*, 2000.

[12] Microsoft, http://www.microsoft.com.

[13] OMAC, "OMAC Baseline Architecture," http://www.arcweb.com/omac.

[14] Owen, T. and Parker, K., "One App's Ceiling Is Another App's Floor," *Manufacturing Systems*, pp. 47-56, October 1999.

[15] Qiu, R. and Joshi, S., "A Structured Adaptive Supervisory Control Methodology for Modeling the Control of a Discrete Event Manufacturing System," *IEEE Transactions on Systems, Man, and Cybernetics*, Part A: Systems and Humans, Vol. 29, No. 6, pp. 573-586, Nov. 1999.

[16] Qiu, R. "A Data Fusion Framework for an Integrated Plant-wide Information System," *The 5th International Conference on Information Fusion*, Annapolis, MD, pp. 101-107, July 8–12, 2002.

[17] Qiu, R. and Grant, D., "Virtual Equipment Module – a Generic Equipment Control Interface," The *14th IFAC World Congress*, pp. 449-454, July 5-9, 1999.

[18] Rockwell, "Making Sense of e-Manufacturing: A Roadmap for Manufacturers," *Rockwell Automation White paper*.

[19] SEMI, "International Technology Roadmap for Semiconductors 2001 Edition," *www.sematech.org*.

[20] SEMI, "Generic Equipment Model," http://www.semi.org.

[21] Slater, D., "Talk to Your Plants," *http://www.cio.com/archive/031500/plants.html*.

[22] SMEMA, "SMEMA Interface Standard 1.2," http://www.smema.org.

[23] Sun Microsystems, http://java.sun.com.

[24] XML, http://www.w3.org/XML/.