The Journal of Korean
Institute of CALS/EC
Vol 7, No. 1, April 2002

# 데이터사전을 이용한 ERP애플리케이션 개발

장민수*, 손주찬*, 백종명*

# ERP Application Development Using Business Data Dictionary

Minsu Jang, Joo-Chan Sohn, Jong-Myoung Baik

## Abstract

Data dictionary is a collection of meta-data, which describes data produced and consumed while performing business processes. Data dictionary is an essential element for business process standardization and automation, and has a fundamental role in ERP application management and customization. Also, data dictionary facilitates B2B processes by enabling painless integration of business processes between various enterprises. We implemented data dictionary support in SEA+, a component-based scalable ERP system developed in ETRI, and found out that it's a plausible feature of business information system. We discovered that data dictionary promotes semantic, not syntactic, data management, which can make it possible to leverage viability of the tool in the coming age of more meta-data oriented computing world. We envision that business data dictionary is a firm foundation of adapting business knowledge, applications and processes into the semantic web based enterprise infra-structure.

***Key Words : ERP, Data Dictionary, Data Exchange, Workflow, EAI, Metadata, Semantic Web***

* EC/CALS Dept., ETRI

## 1. Data Dictionary

### 1.1. Introducing Data Dictionary

It's very important to strategically manage business data information. 'Business data information' is defined as '*metadata about various data produced and consumed while processing business processes*'. Each metadata item is called 'data element'.

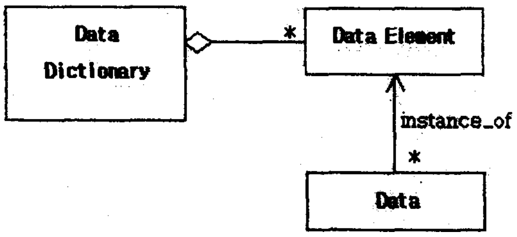
Data element can be speculated in a number of abstraction levels. In a very low level, a type of data in a program, such as int, char, string, can be a data element. Data type provides information about the size of data and the way of how to interpret the data to extract meaning from it.

On the other hand, business data bear a lot more information and lie on a higher abstraction level than simple data types. Business data is defined as an information entity used in business processes. For example, 'AmountOf Money' is a basic business data element. 'AmountOfMoney' contains semantic information about a specific category of data as well as information on physical properties of the data. 'AmountOf Money' can be represented simply by an integer data, but it is not complete only with it. To make it a useful business data element, monetary information should be accompanied. On the usage of this data element, 'AmountOfMoney' can be used to represent a specific class of data in the same semantic boundary, like 'Total Amount Of Purchase', 'Salary', 'Interest For This Month' etc. That is, data element can be reused in different business processes to represent similar data, which indicates that data elements are 'context-free'. By reuse of data elements, different data can be classified into the same category with an implication of data interchangeability. For example, if you represented both 'Salary' and 'Debt' with a data element 'AmountOfMoney', then you can do some meaningful computation with these two kinds of data to produce new information. More surprisingly, computing machines can understand the relationship between 'Salary' and 'Debt' by analyzing data elements used to represent them. Data element, a kind of metadata, implants basic knowledge about data into computer, in turn improves the level of ability of the computers to automatically manipulate data.

The following list shows some important properties of data elements.

- Data elements can represent data in different levels of abstraction.
- A data element can be a composite entity of data elements.
- Data elements are context-free. A data element can be reused to represent many kinds of business data of the same type.

Data Dictionary can be defined as a collection of data elements. Figure 1-1 shows the relationship between data dictionary and data element. Actual data produced and consumed in business processes are defined as instances of data elements. Interchangeability between data depends on the equality of data elements referenced by the data.



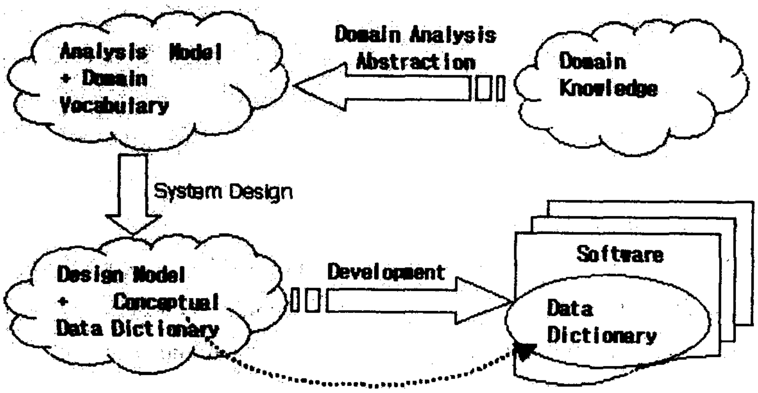
<Figure 1-1> Structure of Data Dictionary

Real-world examples of data dictionary can be found in many places. ABAP dictionary of SAP, Core Components of ebXML and Master Dictionary of RosettaNET are all good examples of data dictionary. These dictionaries aim at different usage areas from each other but basic concepts built into them are all similar. The basic concept of these dictionaries is 'data dictionary is a collection of context-free data elements encountered in business processes'. Of special interest of them are dictionaries of B2B standards, ebXML and RosettaNET. These dictionaries are designed to maximize the opportunity of data element reuse to ensure exchangeability of data between separate enterprises, which results in process interoperability.

## 1.2. Importance of Data Dictionary

### 1.2.1. A View from Software Development and Maintenance Perspective

The most important stages of software analysis and design process are domain analysis and class design. Through domain analysis, we get analytical documents about the problem domain we're going to solve with a software system. The caliber of domain analysis is building domain vocabulary. If the targeting domain is business area, this vocabulary should be a set of data elements used in business processes. That is, in this case, the outcome of domain analysis is a set of business data, which is a basic form of data dictionary. This basic data dictionary will be extended and refined through design process, and will be utilized in software development phase. This scenario is illustrated in Figure 1-2.


Analysis Model + Domain Vocabulary
Domain Analysis Abstraction
Domain Knowledge
System Design
Design Model + Conceptual Data Dictionary
Development
Software
Data Dictionary

<Figure 1-2> The Lifecycle Of Data Dictionary

The process shown in Figure 1-2 is almost the same as usual software development cycle. In the figure, there are two kinds of data dictionary, conceptual data dictionary and data dictionary. Conceptual data dictionary takes shape in the design phase, and contains mostly conceptual data elements that are to be refined and extended to incorporate development specific data element structures. The final data dictionary is complete in that it imposes conceptual structure of the final software system as well as the information on the low-level data organization of the system. Therefore, data dictionary gives you with logical model and physical model of the software system. These models provide important, sometimes invaluable and essential, clues about the system under manipulation when performing some maintenance or customization tasks. That is, when requirements to add, remove or modify some software system features arise, we can fulfill the requirements by observing the system in various levels, from conceptual to physical level, retaining system integrity. All these things are well accomplished by data dictionary.

As shown in Figure 1-1, data elements contained in data dictionary are metadata about data. Data refers to a data element. Data element represents data instances that have reference link to it. A simple consequence of this is that data with the same data element reference are interchangeable. Actually, data instances with the same data element reference are the same in type, and difference lies only in the value. So if data elements are used whenever a new data is declared, we can assure that data are always in control, because we can manipulate and monitor every kind of data through data dictionary. And if some logical or physical change occurs to any data elements, that change is immediately propagated to every data that refers to the modified data element. This is a powerful metadata scheme that brings up the flexibility and stability of the system to the higher ground.

The following bullets sum up the necessity of data dictionary in the light of software development:

- Data dictionary provides logical and physical views about the business process and software system. These models are essential for software customization and maintenance.
- By maximizing data element reuse, data interchangeability and integrity is maximized. This is also a firm ground from which data repackaging like data mining and analysis is based.

### 1.2.2. A View from B2B perspective

Recently, B2B is getting bright spotlight. B2B projects are abundant and companies show great interest in B2B. But we think that they firstly need to retrospect about ERP system they have, and data dictionary.

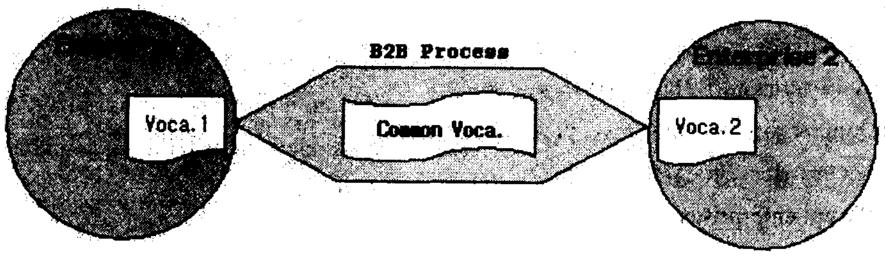
For a B2B process between two enterprises to be properly conducted, enterprises need to be connected among each other's relevant business processes. Business data exchange is an essential part of this kind of business process integration.

[3] Business data exchange is possible through standard business data dictionary. Standard business data dictionary is a collection of core business data with standard representation schema. Core business data items are collected by analyzing various business processes and by identifying common data elements. Those identified business data items make a good data dictionary.

There are a number of data exchange standards, and they are mostly specified in XML. The most notable of them all is ebXML standard. ebXML, sponsored by UN/CEFACT and OASIS, is a modular suite of specifications that enables enterprises of any size and in any geographical location to conduct business over the Internet. Using ebXML, companies now have a standard method to exchange business messages, conduct trading relationships, communicate data in common terms and define and register business processes. [4] EbXML specification suite includes business process schema, metadata registry and repository, messaging framework and core components.

Core component specification defines a data dictionary for ebXML. Enterprises achieve interoperability of business processes with other enterprises by translating enterprise specific business data vocabulary into core component vocabulary, which makes business data from different enterprises exchangeable.

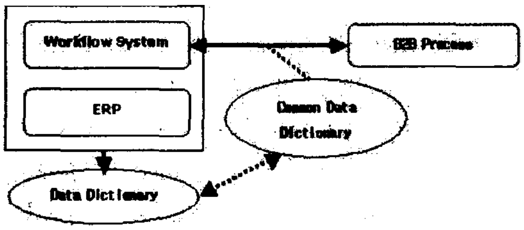
Figure 1-3 shows the structure of B2B business process integration using common data vocabulary. This looks quite ordinary and common. But actually, translation from Vocabulary 1 or 2 to the common vocabulary is not simple. It seems like that many developers and managers think that it is a casual task, but the reality is not like that. How can you draw a regular and simple methodology to pick up appropriate data elements to be used for a B2B process from diverse and dispersed particles of enterprise data? How can you electronically connect your business process with B2B process seamlessly without human intervention? The necessary precondition for realizing real profitable B2B structure should be 'business process standardization and automation' and 'well-structured data dictionary'.



<Figure 1-3> B2B through Common Data Vocabulary

We need to bring (some of) our attention to ERP and workflow systems, an IT base structure of an enterprise. Only after an enterprise makes ERP and workflow system a seamless part of its business operation, it finally gets ready for higher ground of B2B. Without this IT base, enterprise should cut down the vision of B2B electronic commerce. The profit and advantage to be gained from B2B would be reduced significantly.

Figure 1-4 illustrates the hierarchical structure of B2B structure.



<Figure 1-4> Hierarchical B2B Structure

As shown in the figure, data dictionary is placed at the very basic part of the structure. Data dictionary plays critical roles in the following two aspects:

1) Data exchange between enterprise applications and workflow system
   ➔ *interoperability among applications in an* enterprise
2) Data exchange between enterprise workflow system and B2B workflow system
   ➔ interoperability of business processes among enterprises (integration of services from different enterprises)

To achieve 1), enterprise applications should inherently be data exchangeable. That is, from the stage of application development, data declarations should incorporate data elements defined in central data dictionary whenever possible, which results in data openness and exchangeability. Without this kind of data standardization, it would be very difficult to build interoperation scheme between enterprise applications and workflow system. Every application would require specific integration scheme, and integration itself would become a big project, which is not affordable or feasible for most of the cases.

As for 2), the situation is similar to 1). Data exchange occurs at the connection point of enterprise business process and B2B process, and at this point we need a standardized way of gracefully match and translate enterprise specific data into B2B's common vocabulary. Without well-structured data dictionary, it would be very hard to design general-purpose vocabulary translation scheme. In this case also, data dictionary is essential in that it eases the pain of selecting appropriate data elements and matching them with common data elements. Without data dictionary, translation scheme would be coded by case-by-case engineering, which is, again, infeasible and unaffordable.

To put it in simple terms, data dictionary is an essential element in building sound and profitable B2B structure.

## 2. Data Dictionary Support in SEA+

As stated so far, data dictionary plays essential roles in enterprise information systems, both in the areas of ERP application and business integration. But most enterprise software development tools do not share this consensus. Major development tools, like Visual Basic™ and Power Builder™, don't have a support for data dictionary. These tools do not impose data element concept into software development methodology.

Realizing the importance of data dictionary and the deficiency of common tools in supporting data dictionary, we implemented native data dictionary support in SEA+, an enterprise information platform developed in ETRI. SEA+ has a software development environment, and this environment supports defining, building and utilizing data dictionary.

### 2.1. Introduction to SEA+

SEA+ is a component-based ERP package system targeted at small to medium sized companies. SEA+ application structure is based on business components that perform standardized business processes. SEA+ system contains both an execution tool set and a development tool set. With these tool sets, we can develop ERP applications and run them. [2]

In SEA+, every kind of detailed information on ERP application modeling, development, customization and maintenance is stored in a central data repository. Application source codes, database schema information, application user interface design information, transaction information etc are all stored and managed centrally. By central management of these data, integrated and secure management of the system is possible. This gives us with invaluable monitoring ability and decision clues in regard to application design, development and customization which guide us to more secure manipulation of the system without breaking system integrity

### 2.2. SEA+ Development Environment and Data Dictionary

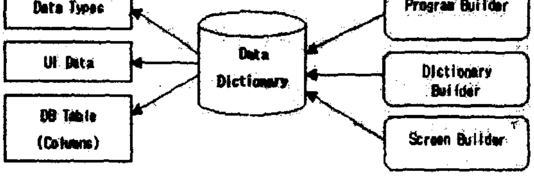
In SEA+, applications are developed by four specialized tools, which are Dictionary Builder, Program Builder, Menu Builder and Screen Builder. The main functions of these tools are as follows:

- Dictionary Builder - a tool to define and manage data dictionary and database tables.
- Program Builder - a tool to write server-side methods. The main roles of server-side methods are business logic processing and server-side data handling.
- Menu Builder - a tool to build menu hierarchy.
- Screen Builder - a tool to interactively design graphical user interfaces

As seen above, this tool set sums up to be a 4GL development environment like Visual

Basic™. But SEA+ development environment is basically different from other 4GL environment in that it has a data dictionary support implanted in it. SEA+ data dictionary contains various kinds of metadata like data types, UI data, database table information etc. The dictionary is referenced all the time by the four development tools, which means that applications should utilize information derived from the dictionary. If you wanted to create a data structure to hold a set of data returned from a server-side method, it should first be defined in the data dictionary as a data element. Then you can refer to that data element to specify the structure of the returned data. A reference link from the return data to the element is created. If you designed a dialog box and wanted to put a label inside it, you should first define a text string data element in the data dictionary that will hold label string values. These are simple examples of data dictionary use in SEA+ development environment.

The figure 2-1 shows the organization of the SEA+ tools with central data dictionary.



<Figure 2-1> SEA+ Development Environment

## 2.3. Examples of SEA+ Data Dictionary Support

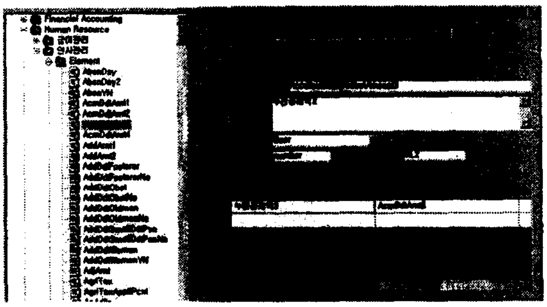
SEA+ data dictionary stores many types of data elements. Data elements can be simple elements or composite elements. A simple data element is defined on itself, but a composite data element is defined as a set of simple data elements. Composite data elements can be used to describe complex information entities like DB table schemas or structures.

SEA+ data elements are used to describe various information entities. Here's a short list of data element types available in SEA+ data dictionary:

- database table schema
- database table column type
- label text
- method's return type
- method's parameter type

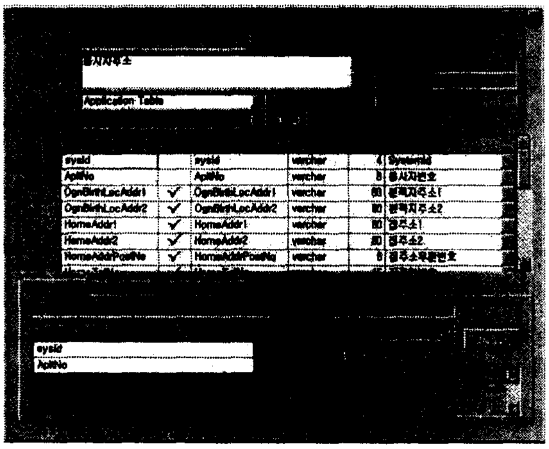
### 2.3.1. Simple Data Element

Figure 2-2 shows a screen shot from Dictionary Builder tool. The left pane lists a hierarchy of data elements. Data elements are organized into two levels – B-Domain and D-Domain. B-Domain represents business domain such as human resource management, financial accounting etc. D-Domain is a sub-category of business types in B-Domain. Under D-Domain, there comes a list of data elements. In the figure, a data element called 'AcmDdtAmt3' is shown. In the right pane, a property box that lists properties of the data element is shown. As shown, the properties include name, description, owner type, data type, and label texts. Data element abstracts a simple data into a logical entity that can be manipulated logically and be reused.


<Figure 2-2> Simple Data Element

### 2.3.2. Database Table Schema

SEA+ data dictionary integrates database schema information. As shown in Figure 2-3, every table column should have a reference to data element to specify the type of each column. By just referencing data element, a column's property is augmented with the data element's properties. If there's no appropriate data element to use in the dictionary, one should define a proper data element first.



종사자주소

Application Table

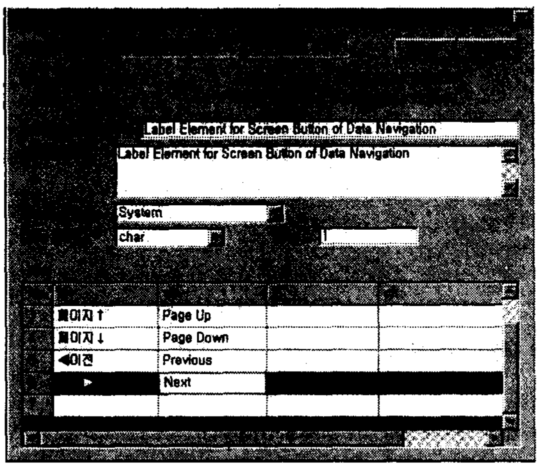
| | | | | | |
|---|---|---|---|---|---|
| sysid | | sysid | varchar | 4 | SystemId |
| AplNo | | AplNo | varchar | 8 | 종사자번호 |
| OgnBirthLocAddr1 | ✓ | OgnBirthLocAddr1 | varchar | 80 | 본적지주소1 |
| OgnBirthLocAddr2 | ✓ | OgnBirthLocAddr2 | varchar | 80 | 본적지주소2 |
| HomeAddr1 | ✓ | HomeAddr1 | varchar | 80 | 현주소1 |
| HomeAddr2 | ✓ | HomeAddr2 | varchar | 80 | 현주소2 |
| HomeAddrPostNo | ✓ | HomeAddrPostNo | varchar | 6 | 현주소우편번호 |

sysid
AplNo

<Figure 2-3> Database table declaration using data elements

### 2.3.3. UI component resource

SEA+ data dictionary stores UI component resources like label texts. Label text is important because label texts should be manipulated uniformly to customize user interface components. For example, if there's a need to port a UI into another locale, we have to rewrite every label texts exhaustively. By managing label texts in organized manner, we can work out the porting without any pain. Figure 2-4 shows a label text data element property box. You can see in the figure that there are Korean and English versions of label texts declared for each label item.



Label Element for Screen Button of Data Navigation

Label Element for Screen Button of Data Navigation

System

char

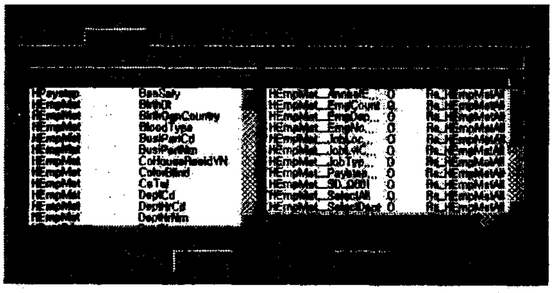
1

| | | | |
|---|---|---|---|
| 페이지↑ | Page Up | | |
| 페이지↓ | Page Down | | |
| ◀이전 | Previous | | |
| ▶ | Next | | |

<Figure 2-4> Label Text Data Element

2.3.4. Dependency Information

An important information part of data dictionary is dependency information. Dependency information indicates which system components refer to which data elements. This makes a very complex dependency graph structure, and this structure should be referred all the time when we need to modify data elements or system components. By referring to dependency information, SEA+ can determine if any modifications put to the system breaks integrity of the system. This is a necessary feature for system customization. Figure 2-5 shows a sample property box that lists dependency relationships between a table, 'HEmpMst', and various data elements.



| | |
|---|---|
| HPaystep | BasSaly |
| HEmpMst | BirthDt |
| HEmpMst | BirthOgnCountry |
| HEmpMst | BloodType |
| HEmpMst | BusiPartCd |
| HEmpMst | BusiPartNm |
| HEmpMst | CoHouseResidYN |
| HEmpMst | ColorBlind |
| HEmpMst | CoTel |
| HEmpMst | DeptCd |
| HEmpMst | DeptHrCd |
| HEmpMst | DeptHrNm |

| | | |
|---|---|---|
| HEmpMst_AnnualE... | 0 | Rs_HEmpMstAll |
| HEmpMst_EmpCount | 0 | Rs_HEmpMstAll |
| HEmpMst_EmpDep... | 0 | Rs_HEmpMstAll |
| HEmpMst_EmpNo... | 0 | Rs_HEmpMstAll |
| HEmpMst_JobLoc... | 0 | Rs_HEmpMstAll |
| HEmpMst_JobLvC... | 0 | Rs_HEmpMstAll |
| HEmpMst_JobTyp... | 0 | Rs_HEmpMstAll |
| HEmpMst_Payitem... | 0 | Rs_HEmpMstAll |
| HEmpMst_SD_0001 | 0 | Rs_HEmpMstAll |
| HEmpMst_SelectAll | 0 | Rs_HEmpMstAll |
| HEmpMst_SelectDept | 0 | Rs_HEmpMstAll |

<Figure 2-5> Dependency Information Example

## 3. Data Dictionary and Semantic Computing

### 3.1. Semantic Computing

Recently, along with the arrival of semantic web hype, semantic computing paradigm is getting hot spotlights. Semantic computing paradigm can be defined as "A computing methodology which depends on the semantics of data." The most differentiating element of semantic computing is the utilization of 'machine understandable data'. Machine understandable data is nothing but meta-data which describes information entities upon processing. Ontology is the representative means of specifying semantic structure of and relationships among information entities. By standardizing description syntax and resource designation mechanism, we can make software do automatically interpret the meaning and structure of information at hand and interoperate with other software systems.

To our understanding, data dictionary is not very different from ontology, terminology or semantic meta-data. These are all a well-organized set of standardized data elements that machines can interpret in some way. But practically, ontology and terminology encompass broader range of meta-data related technologies, like meta-data repository, standard terminology, web services etc.

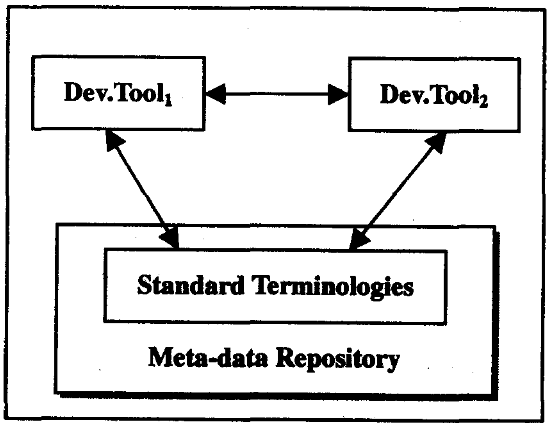
### 3.2. Standard Terminology

One very interesting effort in the area of semantic computing is the standardization of terminology. At ISO, they are trying to make international standards of meta-data repository and terminology specification.[7] The terminology standardization aims at specifying various computational data in a standard way and encourages reusing these terminologies across various computing field. Terminology is basically the same concept as the data element.

By globally standardizing terminology, computing systems will be able to talk to each other natively, without any translation of data types or semantics. Actually, terminology standardization has not been a viable approach for computing systems because of the dispersion and diversity of computing systems. But the time is ripe now to standardize terminologies and share the standard across the world with a big help from web ubiquity, XML, web services and meta-data repositories and all good things connectivity brought to the computer world.

### 3.3. The vision

Figure 3-1 shows a vision of the next generation ERP application development environment when the standard terminologies and meta-data repositories becomes fully present.



<Figure 3-1> Application Development Using Standard Terminologies

The figure stresses that data dictionary, which is meta-data repository in this figure, is no longer an enterprise specific entity, but a globally standardized entity. Every ERP applications would be forced to import data elements from this standardized meta-data repository, and if needed, any enterprise will be able to create a new set of data elements and register them for global reuse.

We think that this vision is not just an imagination, but a reality of the near future. The global structure of the web, ubiquitous technologies like web services and XML are just two snippets of the enabling technologies.

### 3.4. Data Dictionary Inspiration

The conclusion of this section is simple. Data dictionary is an invaluable and visionary element in the development of ERP applications. To well prepare for the future of meta-data oriented computing, we should realize the importance of data element, and revise the process of application development with data element modeling, specification and management. Also, some efforts should be taken to make, utilize and manage "standardized data elements" inside the ERP system.

## 4. Conclusion

In this paper, we speculated that data dictionary should be an essential part of software development environment. By utilizing data dictionary, we assure that the final software system could be manipulated in more integrated manner. And data dictionary, by exposing in a structured way the vocabulary of domain knowledge, encourages interoperability of the system. We can organize B2B processes and data exchange mechanisms more gracefully with data dictionary.

In the near future, meta-data oriented semantic computing systems will flourish. These systems will be built upon the basic idea of data element and data dictionary. By incorporating data dictionary as an explicit element of information systems and development processes, it will be possible to stand still through the transition toward semantic computing age.

## References

[1] SAP, "SAP R/3 Library Online Help (4.5B) : BC – ABAP Dictionary", 1999.
[2] S.J., Park, "A Development Technology of the Component-Based ERP package system", EC/CALS Technology Workshop, 1999, pp. 161-171.
[3] UN/CEFACT, OASIS, "ebXML Core Component Overview Version 1.05", 10 May 2001.
[4] UN/CEFACT, OASIS, "ebXML General Information", 2001.
[5] RosettaNet.org, "RosettaNet Overview", 1999
[6] WfMC, "WfMC Workflow Standard – Interoperability Abstract Specification", WFMC-TC-1012, 1996.
[7] ISO, "ISO/IEC 11179 Specification and Standardization of Data Elements "

**Minsu, Jang** is a researcher at the Electronics and Telecommunications Research Institute, South Korea. He currently works on web content transformation and adaptation system development. His research interests include software patterns, frameworks, metadata, software agents, programming languages, and PDAs. He can be reached at minsu@etri.re.kr.

**Joo-Chan, Sohn** is a senior researcher at the Electronics and Telecommunications Research Institute, South Korea. Mr. Sohn received M.S degree in MIS from HanKook University of Foreign Studies. He is currently a project leader for the web-to-mobile transcoding system and digital rights management system. He also worked as a software designer in an ERP project, developing software development tools for business applications. His research interests include DRM system and ubiquitous business contents publishing system. He can be reached at jcsohn@etri.re.kr.

**Jong-Myoung, Baik** is a principal researcher at the Electronics and Telecommunications Research Institute, South Korea. He can be reached at jmbaik@etri.re.kr.