Quantitative Evaluations on the Query Modeling and System Integrating Cost of SQL/MDR

Dongwon Jeong, Young-Gab Kim, and Hoh Peter In

SQL/MDR is a metadata registry query language used to consistently exchange and share the data between distributed metadata registries. It is an extension of the international SQL and is familiar to most database builders and administrators. It provides many advantages such as simplicity of query language, ease of use, an independent description for distributed querying, low cost for adding new systems, simplicity of exchanging mechanism, and so on. The goal of this paper is to evaluate and show its merits quantitatively. To achieve this goal, we define simulation models to compare with an existing approach and then describe the evaluation results. In the quantitative evaluation results, the good points of SQL/MDR can be identified and known.

Keywords: SQL/MDR, consistent access, metadata registry, data element, quantitative evaluation, interoperability.

I. Introduction

A metadata registry (MDR) is a set of data elements and is one of the most important components of ISO/IEC 11179 [1], [2]. In various fields, many metadata registries have been built as have many MDR management systems [3]-[7]. These metadata registries depend on the specific application domains. As more metadata are exchanged and shared between distributed metadata registries in order to provide more quality services, interoperability becomes an increasingly important issue.

The interoperability between metadata registries in different fields depends on effective sharing of the metadata. Successful sharing of metadata requires a standardized and consistent access method to exchange the metadata between the distributed metadata registries.

ISO/IEC 11179, the international standard specifying the method to build metadata registries, does not provide a standard access method to solve the above issue. Therefore, previous metadata registry management systems have been developed using different access methods. However, inconsistent access methods cause several problems:

- Incompatibility of defined metadata
- Heterogeneity of access method between metadata registry management systems
- High processing complexity for integrated metadata retrieval

To solve these problems, [8] proposed the SQL/MDR, a SQL-based query language for consistent access to a variety of metadata registries. It provides a standardized access method to exchange and share metadata between distributed metadata registries. As a result, this approach has many advantages such as query modeling simplicity, independency of query description, lower query modeling cost, lower distributed query
processing complexity, consistency of access method, and so forth. The work shows its merits with qualitative comparisons and several sample queries, but more explicit evaluation results are needed to prove its good points.

In this paper, both a simulation and experiment are illustrated to show the effectiveness of the SQL/MDR approach. In section II, we introduce key concepts of the MDR and SQL/MDR, with which our research is deeply related. We present the evaluation models for comparisons in section III and evaluations results in section IV. Finally, we conclude the paper in section V.

II. MDR and SQL/MDR

Since our research relates deeply to the works of, we will first introduce each.

1. Metadata Registry

ISO/IEC JTC 1/SC 32 developed ISO/IEC 11179 to enhance the interoperability of databases. We recognize things through their properties, and data represents the properties of them. In ISO/IEC 11179, a data element (DE) is a unit for which definition, identification, representation, and permissible values are specified by means of a set of attributes. The documented data elements are managed in an MDR [2].

A data element consists of mandatory and optional attributes, which completely describe the data. Documentation of the data elements is accomplished through the standardized registration process, thus a data element has one of the several registration statuses: submitted, recorded, qualified, standard, preferred standard, and retired. Figure 1 shows the lifecycle of data elements.

A metadata registry is composed of components to manage a set of data elements. A component contains the conceptual domain, data element concept, value domain, object class, and representation class as well as the data element. Most components classify data elements into several logical groups.

2. SQL/MDR Overview

SQL/MDR was proposed as a method to consistently exchange and share metadata between metadata registries. It has a role as a communication protocol between various management systems that manage their metadata registries [8], [10]. This query language is based on SQL and is similar to many SQL-based query languages [16]. It can be applied for semantic consistency maintenance of applications such as RFID data management, component-based application, etc [17], [18].

Query 1 is a query statement to simply show SQL/MDR
efficiency and its operating process. We assume the following situation.

**Situation.**

1. Two metadata registries, MDR1 and MDR2, have different structures.
2. The first metadata registry, MDR1, is designed as follows: a table, `data_element_table`, includes all the data elements; the `data_element_name` field in `data_element_table` indicates the name of the data elements; and `Status`, another field in `data_element_table`, states the registration status of data elements.
3. The second metadata registry, MDR2, is created with the following conditions: Two tables, `table1` and `table2`, have all the data elements together. Some data elements are in `table1` and the others are in `table2`. `Table1` and `table2` use the `name` field as a key for joining the two tables. The `reg_status` field in `table2` states the registration status of data elements.

(3) Situation.

(1) Two metadata registries, MDR1 and MDR2, have different structures.

(2) The first metadata registry, MDR1, is designed as follows: a table, `data_element_table`, includes all the data elements; the `data_element_name` field in `data_element_table` indicates the name of the data elements; and `Status`, another field in `data_element_table`, states the registration status of data elements.

(3) The second metadata registry, MDR2, is created with the following conditions: Two tables, `table1` and `table2`, have all the data elements together. Some data elements are in `table1` and the others are in `table2`. `Table1` and `table2` use the `name` field as a key for joining the two tables. The `reg_status` field in `table2` states the registration status of data elements.

**Query 1.** Distributed access to two metadata registries.

Retrieve names of all of the data elements where registration status is 'RECORDED' from metadata registries, MDR1 and MDR2.

SQL/MDR: SELECT `DE_status(RECORDED)`
FROM `data_element`  
(Q1)

SQL1: SELECT `data_element_name`
FROM `data_element_table`
WHERE `status` = 'RECORDED'  
(Q2)

SQL2: SELECT `table1.name`
FROM `table1, table2`
WHERE `table1.name` = `table2.name`
AND `reg_status` = 'RECORDED'  
(Q3)

If two metadata registries are designed and developed based on the SQL/MDR approach, only one query statement (Q1) is required to get the final result. However, the previous approach requires two query statements, (Q2) and (Q3), because they have a different metadata registry structure and because there is no consistent access interface between them. If the number of metadata registries is set to N, then the previous approach requires N-query statements. Therefore, query modeling cost, distributed query process cost, preprocessing cost, and complexity of the exchanging mechanism increase exponentially.

In [8], query patterns were analyzed to design the SQL/MDR. Query operators based on the analysis results were defined and integrated into SQL3. The analysis targets include the key components of the MDR, data element lifecycle, data element attributes, and query patterns of services provided by the existing systems. Tables 1 and 2 show the defined operators and a brief Backus-Naur Form of SQL/MDR.

Although the SQL/MDR approach is simple and consistent, there is still one problem. In other words, many database platforms currently do not support the SQL/MDR. There are two resolutions to solve this issue. If SQL/MDR is adopted as an international standard, the problem could be solved easily. However, this process needs much time. The second is to solve it using an interface definition language (IDL)-based approach. We also implemented an SQL/MDR query processor using this approach [10].

In addition, there is a question about how the SQL/MDR approach solves the semantics of data fields used in various metadata registries. SQL/MDR includes standard-like names of data fields and is similar to IDL. Therefore, a name related to schemas and data fields in the SQL/MDR is mapped to one or more data fields of each metadata registry. In mapping, we can consider two cases. First, if metadata registries follow schematic structures and names of the standard, ISO/IEC 11179, they can be achieved by one-to-one mapping processing easily.

---

**Table 1. The defined query operators for metadata registries.**

<table>
<thead>
<tr>
<th>Operators</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>DE_name(KW, OPT), DE_definition(KW, OPT), DE_reg_organization(KW, OPT), . . .</code></td>
<td>Operators to retrieve data elements based on their mandatory attributes with the given keyword and search option.</td>
</tr>
<tr>
<td><code>DE_status(RA, DA, KW, OPT), DE_status_submitted(DA, KW, OPT), DE_status_registered(DA, KW, OPT), . . .</code></td>
<td>Operators to retrieve data elements based on their registration statuses with the mandatory attribute name, keyword, and search option.</td>
</tr>
<tr>
<td><code>DE_object_class(KW, OPT), DE_conceptual_domain(KW, OPT), DE_concept(KW, OPT), . . .</code></td>
<td>Operators to retrieve data elements based on group elements with a given the keyword and search option.</td>
</tr>
<tr>
<td><code>object_class(KW, OPT), conceptual_domain(KW, OPT), element_concept(KW, OPT), . . .</code></td>
<td>Operators to retrieve group elements; Results of these operators are group elements, not data elements.</td>
</tr>
</tbody>
</table>

KW: A given keyword,  OPT: Search options for matching,  RA: Registration attribute (including submitted, recorded, qualified, standard, etc)  
DA: Mandatory attribute names of data elements

---

ETRI Journal, Volume 27, Number 4, August 2005  
Dongwon Jeong et al.  369
Table 2. A brief syntax of SQL/MDR.

```sql
<extended query specification> ::= SELECT <extended attribute list> FROM <extended relation list>
WHERE <extended attribute qualification>;

<extended attribute list> ::= <attribute list> | <MDR attribute list> | <MDR operator>;
<extended relation list> ::= [COMMA] <general relation list> | <MDR relation list> [<extend attribute list>];
<extended attribute qualification> ::= <general qualification> | <MDR qualification>;
<MDR qualfication> ::= [<boolean term>] <MDR operator> [<extend attribute qualification>];
<MDR operator> ::= <DE mandatory attribute name> | <DE registration status> | DE_STATUS L_PAREN <MDR param list> R_PAREN;
<DE mandatory attribute name> ::= NAME | DEFINITION | … | ORGANIZATION;
<DE registration status> ::= DE_STATUS_SUBMITTED | DE_STATUS_RECORDED | DE_STATUS_QUALIFIED | DE_STATUS_STANDARD | DE_STATUS_PREFERRED | DE_STATUS_RETIRED;
<MDR relation list> ::= DATA_ELEMENT | DATA_ELEMENT_CONCEPT | … | OBJECT_CLASS;
```

If a metadata registry does not follow schematic or naming rules of the standard, schema(s) and field name(s) of the registry must be mapped to each schema name and field name of the SQL/MDR, respectively. Thus, one interfacing program for processing the mapping operations must be at the level between the metadata registry and the SQL/MDR. As a result, it is assumed that there are ten metadata registries and all of them do not follow the rules; ten interface programs are required.

An adapter-based approach can be considered. In this approach, if it has the same situation above (let the number of metadata registries be N), then it requires N(N-1)/2 adapters. This approach needs greater cost than the SQL/MDR approach. This issue will be discussed in detail in the following sections.

III. Evaluation Models

Comparative items and evaluation models are defined to quantitatively prove the advantages of the SQL/MDR. This section describes preliminary constraints, notations, and symbols including the items and models.

1. Comparative Items

Figure 3 summarizes the two comparative items and describes the associations with the other items. This paper focuses on two items, query modeling cost and processor development cost (system integrating cost), because the predominance of the remainder has been qualitatively shown with query examples in [8]. In addition, the remainder can be recognized in a logical and conceptual aspect. However, the two items directly affect the performance of the physical development.

2. Notations and Symbols

Notations and symbols are used to describe evaluations, and Table 3 shows their description. This paper also defines several assumptions. The assumptions will be explained with the evaluation model description.

3. Evaluation Models for Comparison of Query Modeling Cost

In the previous approach, we had to write as many queries as the number of metadata registries that have been written because metadata registries use different structures, and there is no standardized method to exchange information between them. However, in the case of using the SQL/MDR, only one query statement is necessary to accomplish this.

Figure 4 illustrates the models to evaluate query modeling cost. For the simulation according to the models, several assumptions are needed:

- The semantic analysis time of components or attributes in MDRs is generated by the random number generator.
- `DomK` is uniform.
- Each MDR has the same number of components.
- Each component is mapped to only one table. This means
Table 3. Summary of notations and symbols.

<table>
<thead>
<tr>
<th>Notations &amp; symbols</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nMDR</td>
<td># of MDRs (the number of metadata registries)</td>
</tr>
<tr>
<td>nComPer</td>
<td># of MDR components per each MDR (Assumption: each MDR has the same number of Coms; Comp:Table = 1:1)</td>
</tr>
<tr>
<td>nAttribPer</td>
<td># of attributes per each Com (Assumption: each Com has the same number of Attribs)</td>
</tr>
<tr>
<td>nComQ</td>
<td># of semantic MDR components to be described for creating OrigQ</td>
</tr>
<tr>
<td>nAttribQ</td>
<td># of semantic attributes to be described</td>
</tr>
<tr>
<td>ComPer</td>
<td>Actual components in local MDRs; Mapped to semantic components</td>
</tr>
<tr>
<td>AttribPer</td>
<td>Actual attributes in local MDRs; Mapped to semantic attributes</td>
</tr>
<tr>
<td>ComQ</td>
<td>Semantic components in OrigQ</td>
</tr>
<tr>
<td>AttribQ</td>
<td>Semantic attributes in OrigQ</td>
</tr>
<tr>
<td>TransQ</td>
<td>Translated query for local MDR systems</td>
</tr>
<tr>
<td>OrigQ</td>
<td>Original query of user</td>
</tr>
<tr>
<td>AnalT</td>
<td>Semantic analysis time (This time value be randomly generated with the value domain [0,1])</td>
</tr>
<tr>
<td>Ran()</td>
<td>Random number generator to create semantic analysis time</td>
</tr>
<tr>
<td>DomK</td>
<td>Domain knowledge of users (Assumption: be uniform. Every user has the same level of knowledge)</td>
</tr>
<tr>
<td>QDT</td>
<td>Query design time, i.e. description time with one or more metadata registries</td>
</tr>
<tr>
<td>WrtCom()</td>
<td>Interpreting and selecting time for proper component’s names</td>
</tr>
<tr>
<td>WrtAttrib()</td>
<td>Interpreting and selecting time for proper component’s attribute names</td>
</tr>
</tbody>
</table>

In the previous approach, there is no unified method to access metadata registries. Thus, as many query statements as metadata registries are required to generate an integrated result. In Fig. 4, the previous approach needs three query statements, TransQ_A, TransQ_B, and TransQ_C, for the metadata registries, MDR-A, MDR-B, and MDR-C. The calculation formula for estimating the query modeling cost of the previous approach is as follows:

\[
\text{Cost(QDT}_{\text{PREV}}) = nMDR \cdot \left( \sum_{i=1}^{nMDR} \text{WrtCom}(\text{ComQ}_i) + \sum_{j=1}^{nComPer} \text{WrtAttrib}(\text{AttribQ}_j) \right)
\]

where \( nMDR \) is the number of metadata registries, WrtCom(ComQ_i) is the time for analyzing and describing a ComQ and its ComPer set, WrtAttrib(AttribQ_j) is the time for analyzing and describing an AttribQ and its AttribPer set, and Ran( ) is the random number generator.

By contrast, for the SQL/MDR-based approach, only one query statement is written to gather data from the metadata registries. Therefore, the modeling cost of this approach is independent of the number of metadata registries, and its formula is as follows:

\[
\text{Cost(QDT}_{\text{SQL/MDR}}) = \sum_{i=1}^{nComQ} \text{WrtCom}(\text{ComQ}_i) + \sum_{j=1}^{nAttribQ} \text{WrtAttrib}(\text{AttribQ}_j)
\]

all attributes of each component are included in one table.

- Each component has the same number of attributes.
- Each respective MDR is constructed with a different structure.
4. Evaluation Models for Comparison of System Development Cost

The second comparative item pertains to the development cost for integrating n-MDR management systems. This item is subdivided into two sub-items: initial integrating (organization) cost and the cost corresponding to the addition of a new metadata registry.

A. Comparison Model for Initial Organizational Cost

Figure 5 illustrates the model for the cost comparison of the initial organization. In Fig. 5, the previous approach does not provide a standardized information exchanging method; thus, all systems should create adapters to exchange metadata with each other. Provided that the number of systems is N, each system has (N-1) adapters. As a result, the number of all adapters in the integrated system is N(N-1). The calculation model of the previous approach for initial organization cost is as follows:

\[
\text{Cost(InitORG\textsubscript{PREV})} = N(N-1) - C\textsubscript{PREV} = C\textsubscript{PREV} - (N^2 - N),
\]

where \(C\textsubscript{PREV}\) is the cost for establishing interfaces (adapters) for exchanging metadata between metadata registries in the systems.

However, the SQL/MDR is used as a communication method to consistently exchange information between the MDR management systems. Hence, each system needs only one adapter for interfacing with the SQL/MDR, and its calculation model is as follows:

\[
\text{Cost(InitORG\textsubscript{SQL/MDR})} = C\textsubscript{SQL/MDR} \cdot N,
\]

We assume that \(C\textsubscript{PREV}\) is equal to \(C\textsubscript{SQL/MDR}\). Actually, \(C\textsubscript{PREV}\) is more than \(C\textsubscript{SQL/MDR}\) because the previous approach needs m-to-n interfacing, but the SQL/MDR approach requires 1-to-m (that is, SQL/MDR-to-systems) interfacing. Therefore, this assumption has no effect on our evaluation results to show the advantage of SQL/MDR. As a result, when we define \(C\textsubscript{PREV} = C\textsubscript{SQL/MDR} = c\), a constant value, then the two calculation models can be redefined as follows.

**Previous approach:**
\[
\text{Cost(InitORG\textsubscript{PREV})} = C\textsubscript{PREV} \cdot (N^2 - N) = c(N^2 - N)
\]

**SQL/MDR approach:**
\[
\text{Cost(InitORG\textsubscript{SQL/MDR})} = C\textsubscript{SQL/MDR} \cdot N = cN
\]

B. Comparison Model of Costs for Adding a New System

This comparative item evaluates costs for adding a new MDR system to an existing integrated MDR exchanging system. As for the previous approach in Fig. 6, a new system should create as many adapters as the number of the local MDR systems that the previous integrated system consists of. Existing local MDR systems also need to create an additional adapter to share their information with the new system.

\[
\text{Cost(InitORG\textsubscript{PREV})} = N^2 - N
\]

\[
\text{Cost(InitORG\textsubscript{SQL/MDR})} = N
\]

We assume that \(C\textsubscript{PREV}\) is equal to \(C\textsubscript{SQL/MDR}\). Actually, \(C\textsubscript{PREV}\) is more than \(C\textsubscript{SQL/MDR}\) because the previous approach needs m-to-n interfacing, but the SQL/MDR approach requires 1-to-m (that is, SQL/MDR-to-systems) interfacing. Therefore, this assumption has no effect on our evaluation results to show the advantage of SQL/MDR. As a result, when we define \(C\textsubscript{PREV} = C\textsubscript{SQL/MDR} = c\), a constant value, then the two calculation models can be redefined as follows.

**Previous approach:**
\[
\text{Cost(InitORG\textsubscript{PREV})} = C\textsubscript{PREV} \cdot (N^2 - N) = c(N^2 - N)
\]

**SQL/MDR approach:**
\[
\text{Cost(InitORG\textsubscript{SQL/MDR})} = C\textsubscript{SQL/MDR} \cdot N = cN
\]
Therefore, 2(N-1) adapters are required to add a new system, where N is the number of local metadata registry systems (initial metadata registry systems).

Provided that the number of new systems is M, the calculation model is as follows:

\[ \text{Cost(ADD)} = 2^{M+i+1} \]

On the other hand, in case of the SQL/MDR approach, a new system to be added requires one adapter, but the other systems do not. In other words, it is possible for the new systems to be absorbed into the integrated system independently. Hence, the calculation model of this approach is as follows:

\[ \text{Cost(ADD}_{\text{SQL/MDR}}) = C_{\text{SQL/MDR} \cdot M} \]

IV. Evaluation Results

This section describes the evaluation results that are obtained through the simulation models referred to in section III.

1. Comparative Evaluation Results on Query Modeling Costs

Table 4 summarizes parameters and their value setting. The parameter nMDR represents the number of metadata registries. Query modeling cost depends on the number of metadata registries, so this parameter is variable. Schema structures of MDRs can be different according to design, but the number of components must be unified because the MDR is a part of the ISO/IEC 11179 standard. This paper assumes that nComPer, the number of components, is uniform. It is also assumed that nAttribPer, the number of attributes of each component, is uniform.

Next, nComQ represents the number of semantic components that must be examined and described to create the final result from metadata registries against a given original query. These components have several actual components, which are semantically the same. Therefore, if the number of MDRs is 5, semantic_component:actual_component = 1:5 because all MDRs have semantically the same component.

Also, nAttrib represents the number of semantic attributes to be examined for the final result of a given original query. One semantic attribute of a given original query is mapped to several actual attributes in metadata registries. Therefore, semantic_attribute : actual_attribute = 1:5, where the number of MDRs is 5.

Finally, AnalT is the time to semantically analyze actual attributes or actual components. Proper actual attributes or actual components should be selected against semantic attributes or semantic components in a given original query, and query statements that reflect schema characteristics of the each metadata registry should then be written. The analysis time is different depending on human factors because they are difficult to capture. In our simulation, AnalT is created by a random number generator that we have designed, and this issue remains as one of the further needed studies. Evaluation on query modeling cost has the three following cases:

• Case I. V(nMDR): The number of MDRs gradually increases.
• Case II. V(nComQ): The number of semantic components to be written increases in steps.
• Case III. V(nAttribQ): The number of semantic attributes to be described increases by degrees.

A. Result of Case I: V(nMDR)

Figure 7 shows the evaluation results of Case I. In this case, a variable factor is the number of MDRs. The value domain of nMDR is 1 to 5. Parameters nComPer, nAttribPer, nComQ, and nAttribQ have fixed values. As can be seen in the result, the SQL/MDR approach is proven to be more efficient against the previous approach. When the number of MDRs is 5, SQL/MDR : Previous = 21.05 : 112.69 (that is, about 1:5).

![Fig. 7. Simulation result of Case I: V(nMDR).](image)
B. Result of Case II: V(nComQ)

In Case II, a variable factor is the number of semantic components. Figure 8 illustrates the simulation results. As the nComQ value increases, the modeling cost difference between the two approaches continues to grow. When nComQ is 5, the difference between the approaches is about 40 and reaches to about two times that of the modeling cost that the SQL/MDR approach requires in the same situation.

C. Result of Case III: V(nAttribQ)

In the third query modeling cost comparison model, a variable factor is the number of semantic attributes. Figure 9 shows the simulation results. In these results, as the number of semantic attributes nAttribQ increases, the modeling time difference between the two approaches grows.

As we can see through the three simulation results, in the aspect of the query modeling, the previous approach requires much more time than the SQL/MDR approach. As a result, it can be explicitly seen that the SQL/MDR approach is more efficient than the previous approach.

2. Comparative Evaluation Results on System Development Costs

As aforementioned, another comparative item relates to the system development cost to integrate distributed metadata registries. This item includes two sub-items: initial organizational cost and a cost associated to the adding of a new system.

A. Evaluation Result for Initial Organizational Costs

Figure 10 illustrates the initial organizational costs of two approaches. In the SQL/MDR approach, it is required that all systems (local MDR systems) create just one adapter to interface with the SQL/MDR. Therefore, as many adapters as the number of local MDR systems is needed to organize an integrated system. The previous approach requires one-to-one mapping between all local systems.

In Fig. 10, when the number of local systems is ten, the number of necessary adapters in the previous approach is 90. On the other hand, the SQL/MDR approach requires ten adapters. Therefore, as the number of local systems increases, the difference between the adapters needed in both approaches increases by geometric progression.

B. Evaluation Result for Adding a New System

Figure 11 shows the evaluation result of the cost for adding a new system. As shown in the result, the SQL/MDR approach is efficient as compared with the previous approach. When the number of local systems is ten, the difference between the two approaches is 160. In other words, the previous approach

Fig. 8. Simulation result of Case II: V(nComQ).

Fig. 9. Simulation result of Case III: V(nAttribQ).

Fig. 10. A comparison result of system development cost for the initial organization.

Fig. 11. Evaluation results on cost for adding new systems.
creates an additional 160 adapters to add ten local systems to the integrated system. This wastes uncountable time and effort.

These evaluations show the efficiency of the SQL/MDR, where its good points are easily seen. In this paper, several constraints (assumptions) are defined. However, they do not affect the evaluation results, shown in section III.

V. Conclusion

This paper illustrated the evaluation results to explicitly show the advantages of the SQL/MDR. It is referred in [8] that the SQL/MDR is superior to the existing approach in the following aspects: simplicity of query modeling, simplicity of the exchanging mechanism, ease of use, independent description for distributed querying, low cost of system development, and so on. Most of the advantages were qualitatively shown with query examples. However, quantitative evaluations are required to show its merits completely and definitely.

This paper aims at tackling the issue and defining two comparative items: query modeling cost and system development cost. Evaluation models are defined and calculation models for them are proposed. The SQL/MDR approach is more efficient than the previous approach in all of the evaluations. Consequently, this paper clearly proved that the SQL/MDR is an effective sharing method between metadata registries.

In this paper, we assumed that analysis time is uniform. However, the analysis time is different depending on human ability. Further work on the experiment needs to consider the human factor.

Acknowledgment

We appreciate Mr. Fran Jarnjak for his useful comments on this paper.

References

Dongwon Jeong received the BS degree in computer science from Kunsan National University, Korea, in 1997, the MS degree in computer science from Chungbuk National University, Korea, in 1999, and the PhD degree in computer science from Korea University, Korea, in 2004. He worked as a full-time instructor from 1999 to 2000 in the Advanced Institute of Information Technology, Korea. From 2001 to 2005, he worked as a Senior Research Engineer in the Lime Media Technology Laboratory. He was a Research Professor in the Research Institute of Information and Communication Technology, Korea University, from 2004 to 2005. From 2002 to present he has been a committee member of the Telecommunications Technology Association, Korea. Now, he is a Professor in the Department of Informatics and Statistics, Kunsan National University, Korea, and is on the Program Committee of SERA2005 (International Conference on Software Engineering Research, Management & Applications). His research interests include metadata registry-based integration and application, mobile agent security, XML-to-RDB conversion and hybrid security, ubiquitous computing & semantic maintenance, and ontology conversion.

Young-Gab Kim received the BS in biotechnology and genetic engineering and minored in computer science & engineering at Korea University, Seoul, Korea, in 2001. He received the MS in computer science & engineering from the same university in 2003. Since 2001, he has been a member of Institute of Computer & Information Communication from Korea University. Now, he is a PhD candidate in the Department of Computer Science & Engineering at Korea University, Seoul, Korea. His current research interests include metadata registry (MDR), mobile agents, security protocols, security for ubiquitous computing and home networks, and security formalization.

Hoh Peter In received the BS and MS degrees from Korea University in 1990 and 1992, and the PhD degree from the University of Southern California (USC) in 1998, all in computer science. From 1999 to 2003, he was the Associate Director of Software Process Improvement Lab (SPIL) and an Assistant Professor at Texas A&M University, College Station. He is now an Assistant Professor at Korea University and an Adjunct Professor at San Francisco State University. His current research interests include embedded and ubiquitous software engineering, situation-aware middleware, mobility in sensor networks, software security, and security risk management. He has served as the Program Chair of the International Workshop on Embedded and Ubiquitous Software Engineering (EUSE 2004), the Vice Chair of Software Engineering on the Internet at the 2002 SAINT (International Symposium on Applications and the Internet), the Publication Chair of the 2001 IEEE RTES (International Workshop on Real-Time Embedded Systems), and was on the Program Committee of the IEEE RE (Requirements Engineering) in 2003 and IEEE COMPSAC (International Computer Software and Applications Conference) from 2000 to 2005. His honors and awards include a Best Paper Award in IEEE International Conference on Requirements Engineering in 1996, Student Best Paper Awards in Korea Information Science Society in 1988 and 1989, and a Student Best Paper Award in Korea Academic Promotion Foundation (Korea Research Foundation) in 1988.