Open and Scalable Accumulation and Reuse of Common Design Resources

Cheng Hsu
Decision Sciences and Engineering Systems
Rensselaer Polytechnic Institute
Troy, NY 12180-3590
hsuc@rpi.edu

Abstract **"Common design resources" refers to a wide range of practices in the field of enterprise engineering. They also include tools and knowledge for collaboration among scientists and engineers. We develop them for service science: the design of complex one-of-a-kind knowledge-intensive systems. A new method, called Modelbase, accumulates reusable models (e.g., information, mathematical, and simulation) into randomly searchable repositories, and flexibly configures them for reuse. An implementation design is provided for structuring an open and scalable database of model resources and facilitating collaboration.**

Keywords— **service science, design, virtual organization, metadata management, metadatabase**

I. The Population Approach to Service Design

Presentation slides represent a simple example of common design resources. Every significant company can be assumed to have accumulated a vast amount of slides (charts, statistics, texts, etc.) which may be useful for new presentations and publications. The company may create an enterprise-wide searchable repository of these slides to tap into them for future use and hence improve professionals productivity. The same is true for a company's software assets, homepage resources, and video and music resources. In fact, we recognize this basic logic of ***population orientation***: accumulating reusable design resources from anywhere in the population and for anyone in the population for any digitized resources [18], including analytic and information modeling.

Such a repository requires sufficient metadata technology to represent the knowledge contents, along with the logic structure and software implementation, of these complex objects in the population. The usual technique of tagging (keywords) by itself, with or without full text search capability (e.g., Apache Lucene), is insufficient since it does not provide any common representation to information, mathematical and simulation models [12]. Therefore, we develop a new models representation method, based on the previous Metadatabase method [1, 3, 8, 13], to support the population approach.

In a general sense, the population orientation has major implications in any scientific field. One might call it the virtual enterprise of collaboration for scientific research. When the repositories are based on the common cyber-infrastructure, then it is also a cyber-infrastructure-enabled collaboration tool [15, 16]. Beyond reuse for productivity per se, the accumulation also amounts to gradual integration of scientific facts and findings throughout the field, such as modeling the entire universe in astronomy, the entire planet earth in geology, and the entire human genealogy in biology..

However, the particular implementation design of such a model representation method may depend on the specific scientific discipline concerned. In this paper, we consider only the problem of reducing the learning curves of designing highly customized knowledge intensive service systems [4, 7, 11, 17, 18]. These systems are inherently one of a kind, for value cocreation between the customer and the provider [2, 9, 21, 25]. They tend to cover the most value-added, significant service activities, and be recognized as the most difficult to model and automate [6, 10].

We embrace the premise of population modeling [18] to tackle such highly customized, knowledge-intensive service systems: the collection of all value cocreation mechanisms from anywhere, any enterprises of the space – the population – can show patterns for design and yield reusable design results. The population concept seeks relevant results rather than exact matches. That is, it does not attempt to generalize any design laws for all value cocreations, but to accumulate reusable results that help reduce the learning curves of design. It follows that we focus on the open and scalable ***accumulation and reuse*** of model resources for the population. We might add that collaboration is also a form of value cocreation – i.e., scientific endeavors maybe compared to service enterprises.

We present a new metadata method, the ***Modelbase***, to accomplish this goal. The population modeling approach for service is discussed next in Section II, using some industrial reference points. Section III presents the Modelbase design as a new general metadata representation method, and Section IV discusses its implementation and application for service systems design under the population modeling approach. Section V reviews the new method as a cyber-infrastructure-enabled collaboration tool for general application.

II. The Service Systems Design Problem

Computer-Aided Design (CAD) is a gold standard of design science, and hence sets a reference point for any new design methods for value cocreation systems. No comparable science of services exists today to allow for a duplicate of

CAD for one-of-a-kind cocreation. In this sense, service is an open endeavor provable only by empirical evidence. Since service is utility-oriented, it can be designed with many alternative configurations. The design may vary even for the same value proposition, to the same customer by the same provider, when delivered at different time. As such, past designs may not stay accurate, let alone optimal, to future service systems; and hence cannot be reused in their whole without reviewing for possible reconfiguration. These fundamental characteristics have been detrimental to adopting CAD ideas for service [17, 18]. A poignant example is knowledge workers: they are not amenable to standardization and do not give rise to definitive "bill-of-materials" [11]. Their standardized assignment is a goal that always evades the field.

Therefore, a more meaningful interpretation of a CAD for service systems will be a window onto the cyberspace of relevant experiences and knowledge, in the forms of, e.g., business component models, simulation models, and optimization models, from any proven sources. Designers can seek precedents and references and obtain building blocks through such windows. An open and scalable accumulation of reusable model resources in a Modelbase serves this purpose. It promises to complement the traditional CAD approach and contribute to the design science for service. Furthermore, as products are increasingly customized, the new design method may also apply for, e.g., mass customization.

From an empirical perspective, knowledge-intensive services tend to concentrate in the consulting industry that provides on-demand solutions to globally integrated enterprises [45]. This focus leads to our recognition of significant industrial results for population modeling. In this context, a service system design is defined as a set of, e.g., high level component business models, simulation and analytic models, and IT implementation models, following the industrial practices in the field. It turns out that this definition also qualifies the contents of the accumulation sought, and hence permits a definitive representation method to be developed for these types of models. That is, the open and scalable representation and management of such models, along with the data and knowledge resources pertaining to them, as metadata [12, 18].

The metadata representation required is to recognize the common nature of all models: data, knowledge, simulation and analytic algorithms. The metradata management required is to achieve the ability of query and reuse of model resources in a database manner. The designer will save results into the Modelbase as well as retrieve them on demand, to flexibly configure them into any new design, in whole or in part.

Compared to collections of generic algorithms such as mathematic libraries (e.g., Mathematica and MatLab), simulation systems (e.g., Arena, Promodel, and Simul8), and statistics packages (e.g., SAS), the Modelbase concept requires model resources to be represented, managed, and processed at an inter-related hierarchy of abstractions: generic algorithms, application models, and linked super-models, which are integrated with data and knowledge models that may substantiate their applications. In addition, the Modelbase has to help integrate the simulation and analytic models with industrial business component models, which include application domain meta-models, particular component models, and possible IT design models. Finally, it has to deliver results at all levels and configure them as integrated models for new service system design; and the configuration may include any combination of any levels of representation across the entire repository – or, joining, zooming in and out to explore alternative configurations for alternative designs.

Our research joins previous data and knowledge methods with simulation and analytic mdoels by exploring their common basis: rules; since logic and mathematical expressions are both menable to rule-based representation, just as rules have already been linked to common data and knowledge representation [1, 3]. On this basis, the Modelbase method taps into the Metadatabase (see, e.g., [8, 13, 20]) for an open and scalable design, and the previous ontology systems (see, e.g., [14, 19, 26]) for common representation.

Population modeling (recognition of the characteristics of the applications in terms of, e.g., comprehensive business components) is difficult for academia but natural to industry, because the latter "owns" the application domains. The problem for industry is that the previous practices have not yet amounted to a complete design science for knowledge-intensive services because these results tend to be company-specific and lack comprehensive scientific proofs (e.g., capable of prediction of performance). We recognize, in particular, IBM's Component Business Modeling (CBM) approach and Key Performance Indicators (KPI) (see, e.g., [6, 22, 23, 24]) as a basis for compiling the population models into the Modelbase.

For example, the CBM results may serve as a (first) reference point to help recognize and categorize new model resources expressly contributed by researchers and practitioners to the open source, as well as previous results taken from the literature. We can also use the CBM to help test the Modelbase, concerning, e.g., its ability to grow with the field and to reduce the learning curves.

We now turn to develop the particular design of the models representation method: the Modelbase.

III. THE MODELBASE METHOD

The new Modelbase is developed from integrating two previous results: the model-base concept [12] and the Metadatabase [13]. The former provides a starting design, while the latter the technical foundations of the design, for integrating three basic types of information resources: data, knowledge, and mathematical expressions and logic. The Metadatabase uniquely develops an ontology based on the general theory of modeling, i.e., the above-mentioned concepts of objects, entities, relationships, and rules, rather than based on applications [14]. Thus, it promises better practicality, openness and scalability to new applications.

Using the Metadatabase, as described in [18], the Modelbase structure in Figure 1 becomes a common schema for an open community of heterogeneous models.

From the perspective of its operation, the Modelbase is a database of metadata representing the models contained. Its structure comprises a basic set of generic information modeling concepts: the Entity-Relationship-Attribute sub-model; the Object-Orientation (Classification) sub-model; the Rule-Based sub-model (Rule-Condition-Action-Fact); and the Software, Hardware, Application, and User sub-model. They constitute the ontology of the Modelbase in the sense of a set of common ***types*** of metadata readily representing a wide range of information modeling methods. These types are further mapped into normalized meta-relations which define the structure of the Modelbase. Particular models are (reverse-) represented in these concepts, and become instances of these types that populate the meta-relations.

The Modelbase ontology is reasonably comprehensive and supported by the reverse-representation methodology accompanying the Metadatabase [14]. It encompasses UML, EXPRESS, and many other standard modeling tools in the field, To the extent that these concepts are applicable to capture the constructs of all enterprise data, knowledge, and analytic models, the ontology makes the Modelbase representation ***logically*** open and scalable. That is, with translation, they can accommodate any data, knowledge, and analytic models consistent with these neutral concepts. Previous results [14] show that the translation is amenable to CAD-style implementation.

When implemented as a relational database, the Modelbase manages its metadata contents through ordinary database operations: deletion, updating, retrieval, and insertion. Models are added, removed, and modified this way without causing disruption to the community. The Modelbase will function at three levels of community models accumulation and reuse: information repository (metadata) management, global query, and models (re-)configuration.

In Figure 1, each icon represents either a meta-relation (table) of metadata or a particular type of integrity control rules. The normalized meta-relations are categorized into four inter-related groups: Users-Applications, Database Models, Contextual Knowledge, and Software and Hardware Resources. The Database Models group is comprised of metadata types that define application data models for enterprises. They include Subject (comparable to Object and View), Entity-Relationship, and Data Item. The meta-relationship Define supports recursive Subjects, showing the super-sub hierarchies between Subjects in a bill-of-materials manner. This group of constructs serves as a basis to representing simulation and analytic models, as well as for representing databases involved in the service systems.

The Contextual Knowledge group is structured in terms of the metadata types Rule, Condition, Action, Operator, and Fact. These types constitute a Rulebase Model as a particular representation method for Rules [3]. The Condition and Action metadata types represent the declaration of conditions and actions for the predicates of rules, both of which are further substantiated with meta-relations Operator and Fact. Fact is corresponded to Data Items of the Database Models, and thereby integrates Database Models with Contextual Knowledge. These two groups are further linked, on the one direction, with the aggregated definition of Users-Applications which use the data and knowledge models represented; and on the other, with Software and Hardware Resources which implement these models. The Contextual Knowledge group is poised to capture the logical and mathematical expressions of analytic models.

The User-Application group is defined to represent multiple enterprises, application families, and user interface types (including natural language input) of the Database Models and their Contextual Knowledge. The Software and Hardware Resources group, on the other hand, represents the particular computing platforms, networking middleware, and other IT solutions involved in the implementation design of the databases and knowledge systems. Both groups play important role in the Modelbase, since the former readily supports CBM meta-models and link reusable results to new designs; while the latter uniquely accommodates software implementation of simulation and analytic models as well as supports the representation of IT solutions for CBM results.

The Equivalent meta-relationship cross-references data items from one application data model to their equivalents in others. Together with the conversion routines represented in Software and Hardware Resources, they achieve data semantics reconciliation and translation for the community. This part of the Modelbase structure also represents a core set of metadata that a distributed Modelbase design can install in local sites to facilitate peer-to-peer data reconciliation and collaboration [28]. They promise to facilitate connection of analytic models, too, across heterogeneous paradigms. This topic represents a significant future research.

Models will be “saved” into the Modelbase by first reverse-representing their constructs into the neutral Modelbase constructs, and then entering this neutral representation (e.g., the metadata) into the corresponding meta-relations in the Modelbase. Fundamentally, simulation and analytic models are represented at three levels: the metadata that describe a model, the core logic, or algorithm (e.g., the pseudo code), and the software implementation. In general, the model description metadata are directly construed in terms of the Modelbase concepts, with modifications to suit the extra model definitions. The logic of a model is stored as text, or a character stream, in the Modelbase. The software implementation of a model is referenced (by pointers) in the Modelbase but actually stored separately as, e.g., BLOB (binary large objects).

Thus, a model will have a number of entries in a number of meta-tables in the Modelbase representing it. New models are added to the community in the same way, and this process extends to the modification of existing models represented in the Modelbase, too. Thus, the Modelbase does not need to shut down during the insertion and update. In this sense, the Common Schema, i.e., the Modelbase structure, is ***operationally*** open and scalable, as well. This is a unique property of its technical foundation, the Metadatabase model.

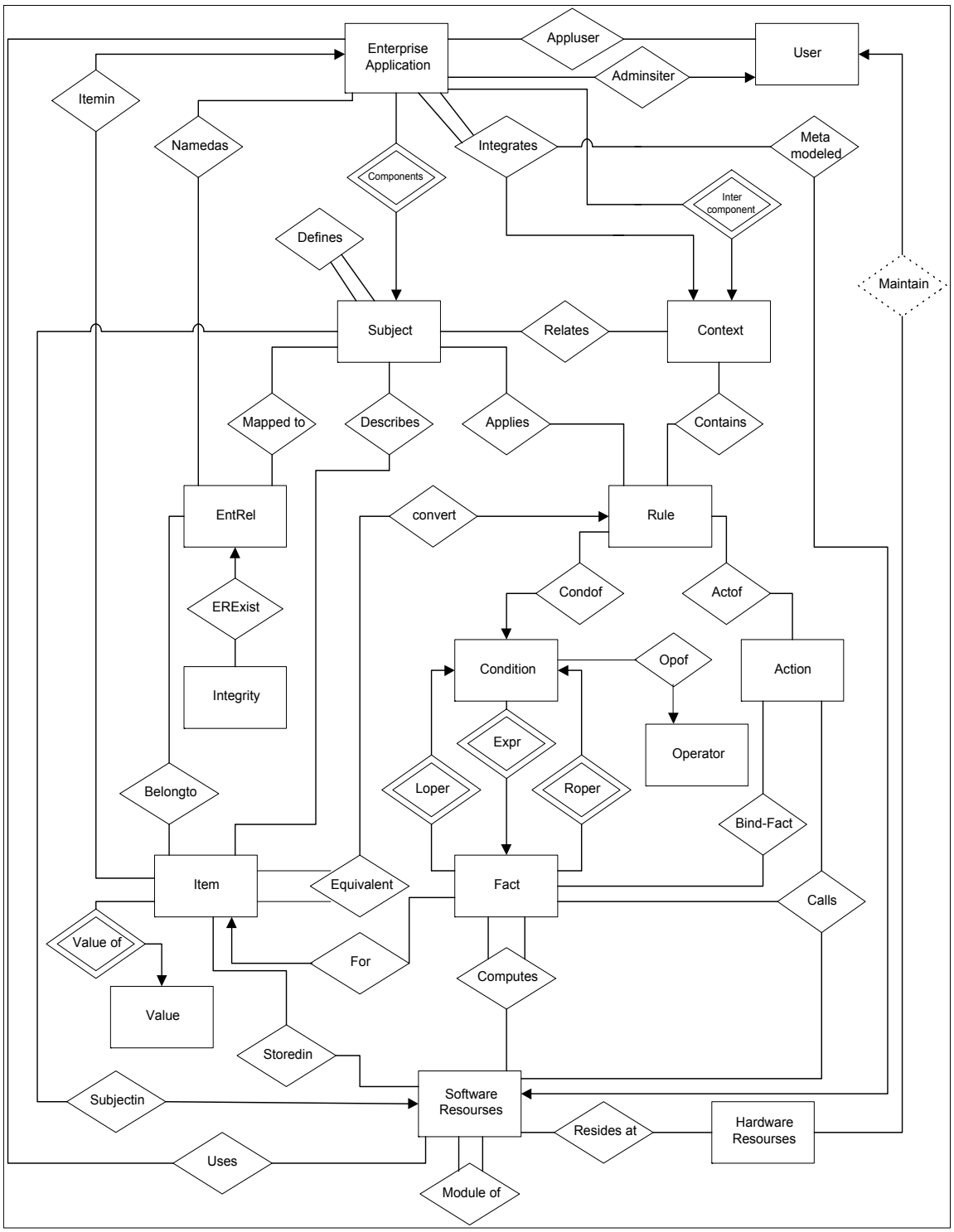

Figure 1: The Common Schema of the Modelbase.

The meta-attributes that represent models are listed below. These attributes "inherit" the constructs of the original Metadatabase model and map them to the basic concepts and constructs of simulation and analytic models. The notions of the logic and the software implementation of a model are flexible: they may either be further decomposed into multiple rules and elementary programs, respectively; or they in their entirety may be represented as a single, aggregate "rule" and a single program. In other words, multiple levels of decomposition are allowed by the recursive nature of the schema to either "zoom out" to the level of whole models or zoom in to individual rules and routines.

List of Categories of Model Representation Metadata

System (eco-system/CBM map/overall management-

decision-operation systems)
Application (application domain/meta-models) /i.e., meta-models for applications/
Subject (application/model/component/IT solution) /i.e., models for an application/
Context (associations between models)
Item (parameter and decision variable)
Rule (model structural rules, solution logic/procedure and operation/assertion) /i.e., logic/
Condition (constraint, function, and mathematical/logical statement)
Action (feasibility, optimality, and mathematical/logical statement)
File (input, output, and other interfaces)
Program (executable algorithm/code) /i.e., software implementation/

List of Meta-entities: relations of model representation metadata

System (Systname, descript, addedby, dateadded, uname)

Application (Aname, SubAname, descript, addedby, dateadded, Uname)
Subject (Sname, SubSname, descript, addedby, dateadded, Uname)
Context (Cname, descript, addedby, dateadded, uname)
Item (ItemCode, iname, represent, length, domain, descript, defvalue, addedby, dateadded)
Rule (Rname, rtype, descript, addedby, dateadded)
Condition (Condid, iname, operator, value)
Action (Actid, acctype, exemode, actcont)
EntRel (OEname, PKey, AKey, descript, addedby, dateadded) /***i.e., normalized models***/
PR (PRname, PKey, AKey, descript, addedby, dateadded)
MR (MRname, PKey, AKey, descript, addedby, dateadded)
FR (FRname, Dant, Ded, descript, addedby, dateadded)
File (Fname, ftype, accessmode, descript, addedby, dateadded)
Document (Dname, doctype, numpages, author, addedby, dateadded, descript)
Hardware (Hname, htype, location, comnet, addedby, dateadded, descript)
User (Uname, Aname-usagename, position, phone, office, address)
Program (Progname, descript, language, addedby, dateadded)

List of Meta-relationships: mostly binary relations associating meta-entities

Relates (Cname, Sname)

CondOf (Condid, Rname)
Describes (Iname, Sname)
Actof (Actid, Rname)
Contains (Cname, Rname)
Applies (Sname, Rname) etc.

The anchoring constructs in the Modelbase representation method include Subject, (representing models for an application), EntRel (representing normalized, sharable models), Rule (representing logic), and Program (representing software implementation). They are also the anchoring constructs for the original Metadatabase.

IV. IMPLEMENTION OF THE MODELBASE

The Modelbase is designed for implementation as an open source repository for the population, either administered for an enterprise within the purview of some proprietary authority (e.g., IBM), or for an open community. The implementation will focus on two primary activities: reverse-representation of models (e.g., the CBM and other possible industrial results, simulation and analytic models, and other population model resources according to Figure 1) and the creation and maintenance of the Modelbase.

The first activity includes decomposition of (composite) models into some common reusable modules shared by these models. The results of reverse-representation will show the models in three levels of abstraction: model, core logic/algorithms, and software, as stated above. Thus, models may be retrieved and reused as they were in the literature, or as how the applications will use them (the user views), either independently or as components for use in other models. A protocol for decomposition will lead to recognition and normalization of model resources at the level of elements – i.e., unit modules and basic "modeling objects". These elements can be flexibly assembled and reused to form new models as well as to recover the original ones. This recursive structure is captured by the similarly recursive concept of Subject of Figure 1.

The Modelbase implementation can either use a central global server or distribute copies of the Modelbase at major sites of the community. Both approaches promise to work because the Modelbase does not involve high volume real time data transactions unlike ordinary production databases. It handles only metadata whose frequency of update and retrieval are expected to be relatively moderate. The Modelbase itself will be implemented as a relational database using open source technology such as PostgreSQL and PERL.

V. GENERALIZATION OF THE MODELBASE

A new Modelbase method is presented in the paper. The concept is general, while the particular design is motivated by the need of a new science for service. The Modelbase design can be readily constructed; however, to make it work, a comprehensive protocol and methodology are needed to help populate relevant models into it.

In addition, research opportunities exist for exploring methods of normalization (or, consolidation) of model resources up to the level of individual rules and routines, in a way similar to data normalization for databases. A normalized Modelbase promises to facilitate maximum flexibility in model sharing, reuse, and aggregation. The Modelbase attributes defined in Section 3 support this concept.

Finally, the Modelbase design uniquely supports the development of an open source repository for any scientific collaboration enterprises. For example, it promises to contribute to the notion of cyber-infrastructure-enabled discovery and collaboration in the science community at large. The particular design presented in the paper is applicable to any scientific fields to the extent that the same types of metadata (classes of data, knowledge, and simulation and analytic models) apply to these fields. This is a significant potential.

REFERENCES

1. G. Babin and C. Hsu, "Decomposition of Knowledge for Concurrent Processing," *IEEE Transactions on Knowledge and Data Engineering,* vol. 8, no. 5, 1996, pp 758-772.
2. M. Bitner and S. Brown "The Evolution and Discovery of Services Science in Business Schools", *Communications of the ACM*, July, 2006, pp. 73-78.
3. M. Bouziane and C. Hsu, "A Rulebase Management System Using Conceptual Modeling," J. *Artificial Intelligence Tools*, Vol. 6, No.1, 1997, pp. 37-61.
4. Cambridge Papers, *Succeeding Through Service Innovation,* University of Cambridge Institute for Manufacturing and IBM Corporation, April 2008.
5. W.K.V. Chan and C. Hsu, "Service Scaling on Hyper-Networks," *Service Science*, vol. 1, no.1, forthcoming.
6. L. Cherbakov, G. Galambos, R. Harishanka, S. Kalyana, and G. Rackham, "Impact of Service Orientation at the Business Level," *Special Issue on Service-Oriented Architecture of IBM Systems Journal*, vol. 44, no. 4, 2005
7. H. Chesbrough and J. Spohrer "A Research Manifesto for Services Science," *Communications of the ACM*, July, 2006.
8. W. Cheung and C. Hsu, "The Model-Assisted Global Query System for Multiple databases in Distributed Enterprises," *ACM Transactions on Information Systems,* vol. 14, no. 4, 1996, pp 421-470.
9. M. Dausch and C. Hsu, "Engineering Service Products: The Case of Mass-Customizing Service Agreements for Heavy Equipment Industry", *International Journal of Service Technology and Management*, Vol. 7, No. 1, 2006, pp. 32 - 51.
10. T. Davenport, "The Coming Commoditization of Processes," *Harvard Business Review*, June 2005.
11. B. Dietrich and T. Harrison "Service Science: Serving the Services", *OR/MS Today*, June, 2006.
12. C. Hsu and W.A. Wallace, "Model Representation in Information Resources Management," in Yuji Ijiri, ed., *Creative and Innovative Approaches to the Science of Management*, Quorum Books, Westport, Connecticut, 1993.
13. C. Hsu, M. Bouziane, L. Rattner and L. Yee, "Information Resources Management in Heterogeneous Distributed Environments: A Metadatabase Approach," *IEEE Transactions on Software Engineering,* vol. 17, no. 6, 1991, pp 604-625.
14. C. Hsu, Y. Tao, M. Bouziane, and G. Babin, "Paradigm Translations in Integrating Manufacturing Information Using a Meta-Model," *Journal of Information Systems Engineering,* vol. 1, September 1993, pp 325-352.
15. C. Hsu (ed.), *Service Enterprise Integration: an Enterprise Engineering Perspective,* Springer Science Publishers, 2007.
16. C. Hsu, D. Levermore, C. Carothers, and G. Babin, "Enterprises Collaboration: On Demand Information Exchange Using Enterprise Databases, Sensor Networks, and RFID Chips," *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, vol. 37, no. 4, 2007, pp. 519 - 532.
17. C. Hsu and J. Spohrer, "Improving Service Quality and Productivity: Exploring the Digital Connection Scaling Model," *International Journal of Service Technology and Management*, forthcoming.
18. C. Hsu, Service Science: Design for Scaling and Transformation, World Scientific and Imperial College Press, Singapore, 2009.
19. Y. Kalfoglou and M. Schorlemmer, "Ontology mapping: the state of the art," *The Knowledge Engineering Review*, vol.18, no.1, 2003, pp. 1-31.
20. D. Levermore, G. Babin, and C. Hsu, "A New Design for Open and Scalable Connection of Independent Databases in Digital Enterprises" *Journal of the Association for Information Systems*, forthcoming.
21. C. Lovelock and WIRTZ, *Services Marketing: People, Technology, Strategy*, 6th ed., Prentice Hall, Upper Saddle River NJ, 2007.
22. N. Nayak, M. Linehan, A. Nigam, D. Marston, F. Wu, D. Boullery, L. White, P. Nandi, and J. Sanz, "Core Business Architecture for a Service-Oriented Enterprise," IBM corp., 2007.
23. A. Nigam and N.S. Caswell, "Business Artifacts: An approach to operational specification," *IBM Systems Journal*, vol. 42, no. 3, 2003.
24. J. Sanz, V. Becker, J. Cappi, A. Chandra, J. Kramer, K. Lyman, N. Nayak, P. Pesce, I. Terrizzano, and J. Vergo, "Business Services and Business Componentization: New Gaps between Business and IT," *Proceedings of the IEEE International Conference on Service-Oriented Computing and Applications SOCA '07*, June, 2007.
25. J. L. Zhao, C. Hsu, H. J. Jain, J. Spohrer, M. Taniru, and H. J. Wang, "ICIS 2007 Panel Report: Bridging Service Computing and Service Management: How MIS Contributes to Service Orientation?" *Communications of the Association for Information Systems*, Vol. 22, March, 2008, pp. 413-428.