On Creating Industry-Wide Reference Architectures

Liming Zhu, Mark Staples, Vladimir Tosic
Managing Complexity Research Group – Sydney (ATP), NICTA[*], Australia
School of Computer Science and Engineering, University of New South Wales, Australia
{Liming.Zhu, Mark.Staples, Vladimir.Tosic}@nicta.com.au

Abstract

Many industries have been developing e-business standards to improve business-to-business interoperability on a mass scale. Most such standards are composed of business data models with some message exchange patterns. Such data-only standards leave a very large interpretation space for the implementation stage at each individual organization. Thus, true industry-wide interoperability is still hard to achieve. In this industry report, we describe our experiences in creating and evaluating reference architectures for the Australian lending industry. To achieve the right level of prescriptiveness, our reference architectures are deliberately non-structural. Instead, they are based on a set of quality-centric architectural rules. We devised new methods for analyzing interoperability and evaluating such industry-level reference architectures. The first reference architecture has now been adopted and achieved positive effects. We also summarize several other lessons we learned, such as the need to align reference architectures with industry structures.

1. Introduction and motivation

Businesses continually seek to get work done faster, better, and cheaper. Industries increasingly realize that the optimization of efficiencies across organizations is the key to the success. This puts distributed enterprise computing in the context of industry-wide mass interoperation. Pair-wise and centrally coordinated integration effort will not scale to an exponentially connected industry ecosystem.

In recent years, industry-specific standardization bodies have been devising domain-specific e-business standards, such as ACORD (Agent-Company Organization for Research and Development) for insurance, MISMO (Mortgage Industry Standards Maintenance Organization) for lending in North America, AUTOSAR (Automotive Open System Architecture) for automobile, and HL7 (Health Level 7) for health informatics. Many of them focus on the standardization of business data and business message exchange patterns with the hope that once this is defined, mass interoperability will be automatically achieved within the industry. However, this has not been the case [10, 27]. Costs of pair-wise integration are still prohibitively high, even when both sides of the pair claim to be data standard "compliant". (Our own experiences from working with the Australian lending industry confirm this.) There are many social and technical reasons for this, ranging from lack of alignment with industry structures to heterogeneity of software architectures that inhibits technical-level (as opposed to business data level) interoperability.

For addressing these problems, a large number of, usually uncoordinated, small technical notes are developed in conjunction with the data standard. Their purpose is to explain mappings to technical implementations, the business processes involved, and different standard interpretations. Many such technical notes are often developed by technology vendors whose main concern is about mapping from data standards to their own technology products. Such biased and uncoordinated efforts introduce even more problems in achieving genuine mass interoperation.

From a distributed enterprise computing point of view, these problems are not new in terms of a particular integration among multiple parties. Some degree of control and centralized coordination can alleviate many of the problems. Making a system more evolvable, using service-oriented architecture (SOA) concepts, and adopting associated technical WS-* standards can also improve point-to-point interoperability.

[*] NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

However, these problems are significantly different in the context of a whole industry, which is effectively an ultra-large-scale ecosystem. Establishing control and centralized coordination, even in modest amounts, is very difficult on the scale of a whole industry. Thus, the enterprise computing solutions are often not applicable in such circumstances. Too much prescription in industry-wide standards is inappropriate because it does not accommodate heterogeneity and competitiveness among industry members. While standardization in addition to data standards is obviously needed, the challenge is to determine what and how much more to additionally standardize, so that a right balance between too much prescription and not enough prescription can be reached.

In this industry report, we describe our experiences in helping the Australian lending industry to tackle these challenges. We created non-structural reference architectures in the form of rules to influence (rather than control) sound technical software architecture derivation. We tailored existing architecture evaluation methods to evaluate such reference architectures. We extracted interoperability tactics and created an informal reasoning framework for more systematic analysis of interoperability. During this project, we learned several lessons transferable across vertical domains.

In the following section, we briefly review related work. In Section 3, we describe the context of our collaboration with the Australian lending industry. In Section 4, we describe our approach to industry-wide reference architectures and our evaluation tactics. In Section 5, we discuss the lessons we learned. The final section summarizes conclusions.

2. Related work

The difference of building a "software city" (an ecosystem of systems) from building a "software building" (an individual system) has been argued for a long time. The original speculative pieces were on models of software development environments [21] and law-governed systems [15, 16]. The recently re-ignited interest in Ultra-Large-Scale (ULS) systems [18, 9, 24] provided illustrations of these differences in health, defense and space contexts. Although the problem definitions have been refined overtime, the solution has not been crystallized. Many research solutions for systems-of-systems, distributed systems and self-managing systems [14] are very applicable to solving these problems. On the other hand, it has been argued that software cities should not be "designed" in the traditional way, but rather "planned" and "regulated" similarly to urban planning. Business-IT alignment is also an active research area, but only a few works [25, 26] study its extension with a focus on inter-enterprise and complex environments. However, they do not provide general conclusions. Some work has been done on creating industry-specific XML (Extensible Markup Language) schemas [10, 23] and mapping them to SOA-based implementations [1]. However, the inadequacy of standards based on data is widely recognized [17]. One possibility for its improvement is through more flexible process-intensive systems [6, 25].

3. Project context description

Lending Industry XML Initiative (LIXI, http://www.lixi.org.au) is an Australian e-business standardization body serving the consumer loan industry. It covers different aspects of the lending chain, such as:

- Loan product information dissemination: Lenders (e.g., banks) communicate new/updated loan product information to brokers, mortgage houses, borrowers.
- Loan origination and approval: Lenders accept loan applications via various channels and execute a complicated process to decide whether the loan applications are approved. During this process, the lender interacts with many external parties, e.g., property valuation firms, insurance companies, credit bureaus.
- Property valuations. Property valuation firms perform independent property value estimations for lenders or borrowers.
- Mortgage insurance. Insurance companies provide insurance to lenders for selected loans.
- Settlement: The settlement involves a number of parties, including property buyers and sellers, financial institutions, legal and government land title offices. A good settlement process ensures transaction integrity.

Similarly to other industry standard organizations, LIXI first developed a business data standard in the form of a controlled vocabulary with corresponding XML schema and message exchange patterns. However, the adoption of this standard was not as fast as expected. The "build-once, interoperate with everybody" slogan was not materialized in reality. Many companies were asking for more standard implementation guidance. Essentially, the underlying causes were the challenges we described in Section 1.

Thus, LIXI asked for help from NICTA (old name: National ICT Australia). Initially, it was identified from a pure business point view that the natural missing part in LIXI work was development of the business process models accompanying the business data. Business processes put business data into context and formally capture control flows and parties involved. We evalu-

ated several specification languages and used the Business Process Modeling Notation (BPMN) for documenting a number of LIXI processes [27]. LIXI expects that these business process models will eventually be mapped to software implementations through Web service technologies, such as SOAP and Web Services Business Process Execution Language (BPEL) [19].

However, we quickly realized that business data and process models alone have limited power to achieve the desired mass interoperability. Lack of agreement about architecture and technical details has resulted in gaps between business standards and particular implementations. We consequently proposed to devise reference architectures and associated development guidelines to supplement LIXI e-business standards.

For a whole industry, the role of a reference architecture is significantly different from the traditional technical reference architectures, which only exemplify a possible arrangement of structural components and connectors. An industry-level reference architecture can not prescribe too much structure as it may prevent new business relationships/models among different parties and adoption of new technologies. An ultra-large-scale ecosystem also has a number of unique characteristics compared to traditional large systems:

- **Decentralization:** Data, development, evolution and operational control are all decentralized.
- **Inherently conflicting requirements:** Most parties want complexity to reside in others' parts of the overall system and want information to be shared, but do not want to share their own information. Technical solution companies provide/favor custom-built applications and intermediary gateways, while smaller players typically want commoditized applications and no intermediaries.
- **Continuous evolution with heterogeneous elements:** The whole ecosystem cannot be stopped and re-engineered. Day-to-day lending activities have to go on, and horizontal interactions with the larger financial and government systems also exert constant influence.
- **No clear people/system boundary:** The scale of involved companies varies widely. Some companies have sophisticated systems that can automate most tasks, while others still rely on fax and manual processing. Messages and activities in e-business standards can map to systems or people depending on specific parties and characteristics of individual transactions.

The LIXI ecosystem resembles the characteristics of an ultra-large-scale system [18]. In order to address both the business perspective and ultra-large-scale system challenges, the reference architecture needs to balance consistency and variety, address competing needs from different parties and consider trade-offs between prescriptive guidance and an ability to evolve.

4. Our approach

Our approach consists of a rule-based reference architecture for the whole (Australian) lending industry, specific mappings to technologies and an architecture evaluation method for rule-based architectures.

4.1. Rule-based reference architecture

We devised a set of quality-centric architectural rules. An architectural rule is defined as principles that need to be followed by structures within a system [4]. An architectural rule may be satisfied by several potential structural architectures. There are two major inputs into our architectural rules. One is a set of design tactics for achieving certain quality attributes [2]. The other is the specific context of the Australian lending industry, including problems, requirements, and industry structures. Essentially, we instantiated quality-specific design tactics in the context of LIXI.

Our rule-based approach is in line with other IT-related industries. For example, Google OpenSocial's specification and APIs (Application Programming Interface) [8] are a form of technology-level guidance for the social networking industry. This standard has both a data focus and an API focus. An industry-wide reference architecture is implied in this specification, but not explicitly outlined. In a way, we created an equivalent of OpenSocial for the Australian lending industry.

The following are sample sets of rules from a list of 40 rules in the LIXI context [29], with commercially sensitive information removed:

- **Influence, but do not control others.** Decentralization is one of the main characteristics of LIXI. Also, LIXI is a voluntary non-profit organization with no standard-enforcement power. In such settings, influence (instead of control) is the main mechanism to achieve interoperability and improve overall system quality. Our rule set encourages influence through micro-format proposals and optional design alternatives.
- **Use minimal service interface.** The modern business world is service-oriented. The technology world has recently been catching up by introducing the "service" concept, e.g., as SOAP-based Web services or RESTful (REpresentational State Transfer) services. A LIXI-compliant system should use message-centric (rather than operation-centric) interfaces. That is, service interfaces should not expose abstractions in the form of remote procedures. Essentially, we advocate the use of a single operation on a service (ProcessLIXIMessage), but allow more complicated inter-

faces to exist. Messaging behaviors are specified by LIXI content structure and LIXI message exchange protocols. This rule encourages maximum flexibility in the face of constant evolution. Ever-changing shared contexts are carried within LIXI messages. Message processing logic can either be hidden behind the service or exposed as protocol related metadata. This approach is related to other rules, e.g.: i) Use the LIXI canonical message model on public interface as much as possible; ii) Assume that LIXI services are autonomous. iii) Make LIXI services to share LIXI schemas and contracts, but not implementation classes.

- **Share metadata and context.** Metadata is usually described in service contracts. It can be related to policies (e.g. security requirements or encryption capabilities), quality of service characteristics (e.g. required response time), and semantic descriptions. Contexts are more instance-specific. We encourage metadata and contexts to be shared in all possible ways. Through the sharing of metadata and context, interoperability can be achieved at both design-time and run-time with little top-down prescriptive planning.
- **Specify semantic alignment.** LIXI standards have provided an ontology vocabulary and associated XML schemas for all defined messages. Semantic alignment links technical implementation elements with semantics. Interoperability between technical elements is improved by consulting the LIXI-related meaning at both design-time and run-time. There is still enough flexibility for implementing technical elements. Semantic alignment mechanisms minimize the effort needed to integrate components built independently.
- **Avoid explicit intermediaries.** We do not introduce the role of an intermediary explicitly in the reference architecture. However, we allow such intermediaries to organically appear in the overall ecosystem. This is very different from the existing e-business meta-standards, such as ebXML, which have an explicit concept of central registry and repositories through which companies post business processes, capability profiles and collaboration protocol agreements. Technically, this is appealing and simplifies some business scenarios. However, we found very difficult to introduce such a structure within LIXI because of complex business issues such as who the intermediaires should be, legal issues such as confidentiality concerns, and practical issues such as the difficulty of semi-automated agreement negotiation. Thus, in our reference architecture, interacting directly with another business party or through an intermediary is treated as the same general mechanism. Local intermediaries within certain areas can be introduced. Dynamic binding and proxy solutions can help achieve various relationships in practice.

4.2. Mapping to implementation technologies

There are essentially two types of problems in LIXI. One is transactional business activities, such as loan application processing or property valuations. The other type is non-transactional, e.g. loan product information dissemination or back channel status updates.

For transactional scenarios, we further mapped [29] the reference architecture to SOAP-based Web services in order to leverage the sophisticated infrastructure support on handling security, transactions and state. BPEL-based workflow engine is used.

For non-transactional document exchange (such as large scale secure data dissemination), we further mapped the reference architecture to RESTful Web services [7] and feed technologies (e.g. Atom [11]). Products are modeled as REST resources and changes in resources are communicated through Atom feeds.

4.3. Mapping to implementation technologies

For evaluating this type of rule-based reference architecture, the existing architecture evaluation methods (e.g. the Architecture Tradeoff Analysis Method - ATAM [13]) for structural architecture can not be used directly. We analyzed the 10 common techniques [12] in architecture analysis and evaluation for their suitability, and adapted them for rule-centric architectures. Our method Evaluation Process for Rule-based Architecture (EPRA) is shown in Figure 1. Its details were described in [28]. There are four phases: I) business goals elicitation, II) rule analysis, III) architectural tactics analysis and IV) trade-off analysis. Each phase uses different tailored architecture evaluation techniques.

We now give an example of how we applied EPRA Phase III for interoperability analysis. A number of papers and technical reports (e.g.,[5, 22])have documented current approaches for achieving interoperability. However, currently there are no large collections of interoperability tactics. By analyzing the above-mentioned reference architectural rules (during Phase II), we extracted many architectural tactics for interoperability. Some example architectural tactics, organized into more general categories, are:

- **Tactics for increasing common understanding**

1. Use technical standards to increase interoperability on technical level.
2. Use a canonical model on the public interface.
3. Annotate technical elements with common semantics using a bottom-up approach.

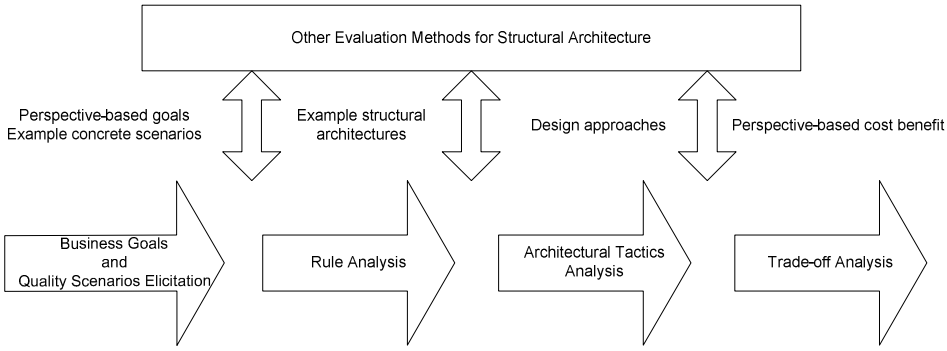

Figure 1. Evaluation process for rule-based architectures

4. Annotate information with expiry time.
5. Treat local optimization as constraints, not goals.
6. Use format indicator in messages.

- **Tactics for advertising and finding capabilities**

1. Use a service discovery mechanism for advertising and finding service capabilities.
2. Publicize capability profiles accessible to others.
3. Give mechanisms for propagating needs and offers.
4. Provide mechanisms for managing intentions and expectations.

In addition to the two general categories of tactics described above, we have also extracted tactics in the following general categories: tactics for predicting end-to-end outcomes, tactics for supporting participant flexibility, tactics for influencing/controlling others and tactics for reducing costs of interoperability.

A critical analysis activity in this phase is to use a reasoning model for quality attributes to determine if a proposed rule or tactic improves or hinders important quality attributes. Formal reasoning models for architectural interoperability (the main quality attribute we investigate) do not yet exist, and thus we had to invent an informal model with informal issues (rather than well-defined parameters) for our evaluation. Our informal reasoning model is intended to reduce the development that is required to achieve interoperability. It considers issues in two dimensions:

- **Interface element dimension:** operations, data and events. This is adapted from the abstract architectural model representation for service divergence evaluation, developed in [3]. That is, a divergence between two service interfaces can be further divided into operation divergence, data divergence, event divergence.
- **Syntax-semantics dimension:** syntax and semantics. For interface elements of each kind (operation, data or event), interoperability concerns can relate primarily either to syntactic or semantic issues.

For each rule or tactic, the two dimensions form a matrix where each cell represents a type of interoperability issue that can be affected by the rule or tactic and consequently impact the overall interoperability of the system. For example, rules about annotating Web service messages with LIXI message schema and vocabulary will have a positive impact on semantic data interoperability issues. Another example is that not defining a service operation in the reference architecture increases the flexibility of individual systems, but decreases both syntactic and semantic operation interoperability. Using this simple matrix approach, we were able to analyze each rule and tactic systematically.

5. Lessons learned

In addition to the overall methodological contribution to enterprise computing at an industry level, we have learned many important lessons during this work:

1) Industry-specific interoperability standards should address not only syntax and semantics of exchanged data and business processes, but also reference software architectures. Different aspects of interoperability need to be governed through different mechanisms. When a "standard" data schema is used in the field, variations are inevitably introduced and subsequently used for ad-hoc, point-to-point and non-repeatable integrations. Due to the high costs for such integrations, standard adoption rate is low. This is not because organizations do not have capabilities and willingness to implement these standards once or twice. It is because the costs of variations and repetitive point-to-point integrations are prohibitively high. This usually ends up with large players dictating one particular schema variation in a hub-spoke fashion, similar to the experiences with EDI (Electronic Data Interchange). Industry-wide mass interoperation among small and medium players remains missing. Adding semantics helps. Normalized reference business processes at organization edges help. However, an important issue is also providing technology-level guidance at the right prescriptive level with the right form. Reference software architectures can provide such guidance.

2) Rule-based reference architectures achieve much better balance between prescriptiveness and flexibility than structure-based architectures. Industry standards should guide industry members towards interoperation. However, industry members are heterogeneous and mutually competitive. Too much prescription in industry standards usually deters industry members from adopting the standard (particularly when standard adoption is optional). A balance between prescriptiveness and flexibility is needed. It is specific to an industry (and even problem domain) and depends more on social and business issues than technical issues. A balance can be usually achieved in more than one point (e.g., within a range). A rule-based reference architecture containing governing quality-centric rules (rather than structural prescriptions) with the right technology binding exemplars is a good way to such balance. The traditional structure-based reference architectures are too prescriptive on the industry-wide scale. Within a rule-based reference architecture, suggestive structural architectures can be provided as exemplary reference implementations to demonstrate specific technology bindings. When a rule-based reference architecture is developed, it is crucial to consider the trade-off and balancing nature of the rules.

3) Both technical issues and business issues should be the main concerns of a reference architecture. In particular, reference architectures should be aligned with industry structures. There are many factors in industry structures, such as business models, economic incentives, business relationships and IT investment capabilities. Without a full understanding of these issues, a sound technical solution may stifle new business models, contradict adoption incentives, hinder business relationships and mismatch IT capabilities. For a to-be ecosystem owned by nobody, it is very difficult to "plan" who will pay for which part of the system and shared infrastructure and how fast the paying party can recover their investment. Adoption incentive mechanisms and loosely specified architectures are often better than forced compliance and dictated structures. For example, we observe that many technical gateways that bridge a cottage sub-industry with big players are often one-sided and try to optimize the business process for the big players at the expense of the small players. It is not surprising that the big players subsidize the development of the gateway and want to push one-sided gateway solution into the reference architecture. Our reference architecture avoids such intermediaries, but allows different types of gateways to appear organically. Different types of gateways also take into considerations the IT capabilities of each side and incentive mechanisms.

4) Systematic evaluation of rule-based architectures requires new methodologies. Existing architecture evaluation methods are designed for structure-based architectures. However, rule-based architectures have significantly different organization and mechanisms. We analyzed existing architecture analysis and evaluation techniques and concluded that they were not appropriate for rule-based architectures. There are also additional issues for evaluation of industry-wide reference architectures. For example, systematic evaluation against business goals and industry structures is needed for industry-wide rule-based reference architectures. Our Evaluation Process for Rule-based Architecture (EPRA) [32] adapted existing architecture analysis and evaluation techniques for rule-centric architectures.

5) Research of ultra-large-scale systems is important for next-generation enterprise computing. We find some of the software engineering and enterprise computing techniques [9, 20, 26] invaluable in creating reference architectures. However, others are not designed with the industry-wide scale in mind. This is particularly the case with architecture evaluation methods, interoperability analysis, and quality issues. In the industry-wide scale, social and business issues are often much greater "pain points" than purely technical issues. Enterprise computing technologies need to adapt to very large, decentralized "systems of systems" for mass interoperability and economic concerns. Rather than re-inventing everything, many existing techniques can be adapted to solve the challenges. The approaches researched in Ultra-Large-Scale (ULS) systems [18] are particularly relevant to creating industry-level reference architectures. Some examples of these approaches are using design rules and design spaces, harnessing economics to promote good designs, and designing across socio-technical levels.

6. Conclusions and future work

In this industry report, we described our approach for creating reference architectures for the Australian lending industry. We went beyond the normal data standard and structural reference architecture and introduced rule-based architectures and architecture analysis methods. We considered both technical issues (interoperability, flexibility, evolvability) and business issues (incentive mechanisms and IT capabilities of industry participants). The resulting architecture is less structurally prescriptive to better support evolution, and points to a number of further research directions.

The first reference architecture (accompanied with derived technology exemplar bindings) that we developed was for property valuation aspects of lending

[29]. It has now been adopted in the Australian lending industry and achieved positive effects. We have recently published another reference architecture for lending product information dissemination [30]. Both reference architectures help individual organizations extend their enterprise computing practices towards industry-level mass interoperation. Additional reference architectures will be developed in the near future.

As systems become more complex, it is impossible to understand them through component-level interaction analysis. Macro-level measures are useful concepts for understanding the overall system. For further understanding of ultra-large-scale IT systems, we need to find macro-level metrics to indicate the overall quality of the system and we need methods for analyzing these new metrics. The analogy with urban planning and city building is just the first step.

7. References

[1] A. Allam and A. Tost, *Developing a Web Service Using an Industry-Specific Messaging Standard*, IBM, 05 July 2007; http://www.ibm.com/developerworks/webservices/library/ws-soa-messagingstandard/index.html.

[2] F. Bachmann, L. Bass, and M. Klein, *Deriving Architectural Tactics: A Step Toward Methodical Architectural Design*, CMU/SEI-2003-TR-004, SEI, CMU, 2004.

[3] S. Bhattacharya and D. Perry, "Architecture Assessment Model for System Evolution," *Proc. 6th Working IEEE/IFIP Conf. Software Architecture* (WICSA'07), IEEE, 2007, p. 8.

[4] J. Bosch, "Software Architecture: The Next Step," *Software Architecture: 1st Euro. Work.* (EWSA 2004), LNCS 3047, Springer, 2004, pp. 194-199.

[5] D. Carney, D. Fisher, E. Morris, and P. Place, *Some Current Approaches to Interoperability*, SEI CMU/SEI-2005-TN-033, SEI, CMU, 2005.

[6] R. Dijkman, "A Classification of Differences between Similar Business Processes," *Proc. EDOC 2007*, IEEE, 2007, pp. 37-50.

[7] R.T. Fielding and R.N. Taylor, "Principled design of the modern Web architecture," *ACM Trans. Internet Tech.* (TOIT), ACM, vol. 2, no. 2, May 2002, pp. 115-150.

[8] Google, "OpenSocial," accessed 27 June 2008; http://code.google.com/apis/opensocial/.

[9] A. Hess, B. Humm, M. Voss, and G.A.E.G. Engels, "Structuring Software Cities A Multidimensional Approach," *Proc. EDOC 2007*, IEEE, 2007, pp. 122-129.

[10] S. Hinkelman, D. Buddenbaum, and L.-J. Zhang, "Emerging Patterns in the Use of XML for Information Modeling in Vertical Industries," *IBM System Journal*, IBM, vol. 45, no. 2, 2006, pp. 373-388.

[11] IETF, *The ATOM Syndication Format*, RFC 4287, Dec. 2005; http://www.ietf.org/rfc/rfc4287.txt.

[12] R. Kazman, L. Bass, and M. Klein, "The Essential Components of Software Architecture Design and Analysis," *J. Systems and Software*, Elsevier, vol. 79, no. 8, Aug. 2006, pp. 1207-1216.

[13] R. Kazman, M. Klein, and P. Clements, *ATAM: Method for Architecture Evaluation*, CMU/SEI-2000-TR-004, SEI, CMU, 2004.

[14] J. Kramer and J. Magee, "Self-Managed Systems: An Architectural Challenge," *Proc. Future of Software Eng.* (FOSE'07) at ICSE 2007, IEEE, 2007, pp. 259-268.

[15] N.H. Minsky, "Law-governed Software Processes," *Proc. 5th Int'l Software Process Work. Experience with Software Process Models*, IEEE-CS, 1989, pp. 98-100.

[16] N.H. Minsky and D. Rozenshtein, "A Software Development Environment for Law-Governed Systems," *ACM SIGPLAN Notices*, ACM, vol. 24, no. 2, 1989, pp. 65-75.

[17] B. Mutschler, J. Bumiller, and M. Reichert, "Why Process-Orientation is Scarce: An Empirical Study of Process-oriented Information Systems in the Automotive Industry," *Proc. EDOC 2006*, IEEE, 2006, pp. 433-440.

[18] L. Northrop, R. Kazman, M. Klein, D. Schmidt, K. Wallnau, and K. Sullivan, *Ultra-Large Scale Systems: The Software Challenge of the Future*, SEI, CMU, 2006.

[19] OASIS, *Web Services Business Process Execution Language Version 2.0*, OASIS Standard, 11 Apr. 2007; http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html

[20] OMG, *UML Profiles for Enterprise Distributed Object Computing (EDOC)*, updated 12 Dec. 2007; http://www.omg.org/technology/documents/formal/edoc.htm

[21] D.E. Perry and G.E. Kaiser, "Models of Software Development Environments," Proc. *10th Int'l Conf. Software Eng.* (ICSE'88), IEEE-CS, 1988, 60-88.

[22] D. Quartel and M. v. Sinderen, "On Interoperability and Conformance Assessment in Service Composition," *Proc. EDOC 2007*, IEEE, 2007, pp. 229-240.

[23] M. Roth, M.A. Hernandez, P. Coulthard, L. Yan, L. Popa, H.C.-T. Ho, and C.C. Salter, "XML Mapping Technology: Making Connections in an XML-Centric World," *IBM System Journal*, IBM, vol. 45, no. 2, 2006, pp. 389-410.

[24] R. Sterritt, C. A. Rouff, M.G. Hinchey, J.L. Rash, and W. Truszkowski, "Next Generation System and Software Architectures: Challenges from Future NASA Exploration Missions," *Science of Computer Programming*, Elsevier, vol. 61, no. 1, June 2006, pp. 48-57.

[25] A. Tao and J. Yang, "Context Aware Differentiated Services Development with Configurable Business Processes," *Proc. EDOC 2007*, IEEE, 2007, pp. 241-252.

[26] R.S. Tapia, M. Daneva, and P. v. Eck, "Validating Adequacy and Suitability of Business-IT Alignment Criteria in an Inter-Enterprise Maturity Model 202," *Proc. EDOC 2007*, IEEE, 2007, pp. 202-213.

[27] L. Zhu, L. Osterweil, M. Staples, U. Kannengiesser, and B.I. Simidchieva, "Desiderata for Languages to be Used in the Definition of Reference Business Processes," *Int'l J. Software and Informatics* (IJSI), Chinese Academy of Sciences, vol. 1, no. 1, 2007, pp. 37-66.

[28] L. Zhu, M. Staples, and R. Jeffery, "Scaling Up Software Architecture Evaluation Processes," *Making Globally Distributed Software Development a Success Story: Int'l Conf. Software Process* (ICSP'08), LNCS 5007, Springer, 2008, pp. 112-122.

[29] L. Zhu and B. Thomas, *LIXI Valuation Reference Architecture and Implementation Guide 1.0*, LIXI, 2007; http://www.lixi.org.au/ri.html.

[30] L. Zhu and B. Thomas, *LIXI Visible Loans: A Pub-Sub based Service*, LIXI, Mar. 2008; http://www.lixi.org.au/medrel/visibleloans.html.