

A Translational Web Services Bridge for Meaningful Interoperability Among Information Systems

Kym Pohl
 CDM Technologies Inc.
 2975 McMillan Ave.
 San Luis Obispo, CA. 93401
 805-541-3750 x233
 kpohl@cdmtech.com

and

Lakshmi Vempati
 CDM Technologies Inc.
 2975 McMillan Ave.
 San Luis Obispo, CA. 93401
 805-541-3750 x248
 lvempati@cdmtech.com

Abstract—An emerging issue in the world of context-centric software-based decision-support is the need for potentially disparate systems to interoperate in meaningful and useful ways. Such interoperability must go beyond the elementary communication of data and endeavor to support a more powerful context-oriented inter-system relationship. A key issue in such functionality is the support, moreover the promotion, of meaningful interoperability while still retaining individual system representations, or perspectives. In other words, the meaningful integration of potentially disparate systems in a manner that allows each collaborating system to retain its potentially unique means of representing, or perceiving, the domain over which it operates. In the past, several approaches to this problem have been postulated, such as development of a specific translator for each source/target system pair combination, development of a universal ontology to encompass both systems, and so on. Specific, one-off translators are usually tightly coupled with both systems and have limited support for dealing with representational changes. The alternate approach of developing a universal representation is not only highly impractical but also requires an ongoing effort of monumental proportions to achieve even a remotely acceptable solution. Considering the potential complexity inherent in mapping between possibly disparate perspectives it is the opinion of the authors that a suitable solution will require the employment of reasoning-enabling technologies capable of supporting the high level analysis involved in performing such context-based translation. Above and beyond the need for complex translation among differing perspectives, the authors see an additional critical ingredient in supporting meaningful interoperability among systems as being the application of a web services-oriented model of inter-system collaboration. In this paradigm, both formalized and more ad hoc system capabilities are essentially defined and exposed as accessible web services. Interoperability in this sense involves systems employing

each other's services in an effort to perform their desired tasks. Reliant on support for complex translation to map between perspectives, this notion of remote service invocation offers a simple yet effective metaphor for addressing the increasing need for useful interaction among potentially disparate systems. The focus of this paper^{1,2} is to provide both a vision and supporting design for a translation-based web services interoperability bridge capable of supporting web services-oriented interoperability among systems operating over potentially disparate representations. Capitalizing on offerings from both the artificial intelligence and semantic web-based worlds the presented design incorporates technologies such as inference engines, rule-based systems, XML, XSLT, web services and service-oriented architectures to provide the needed infrastructure to support meaningful interoperability among context-based systems in an information age.

TABLE OF CONTENTS

1. THE PROBLEM	1
2. INGREDIENTS FOR A SOLUTION	3
3. A SOLUTION: TRANSLATIONAL WEB SERVICES INTEROPERABILITY BRIDGE	4
4. CONCLUSION	7
REFERENCES	7
AUTHOR BIOGRAPHIES	8

1. THE PROBLEM

As the demand for sharing information increases, an additional burden is placed on the tools and systems that support the decision-making process. Context-oriented

¹ IEEE Copyright 0-7803-8870-4/05/\$20.00©2005 IEEE

² IEEEAC paper #1467, Version 3, Updated December 10, 2004

systems, as opposed to data-centric systems, rely heavily on the contextual depth of the descriptions over which they operate. Contextual depth, or semantics, forms the fundamental enabler for such systems to offer users helpful assistance in the decision making process. Driven by the need for systems to understand more about the problems they are helping to solve is the need for such systems to interoperate. Whether interacting with other context-enabled systems or accepting data feeds from legacy data-centric systems, the need to understand the semantics of what is being communicated places a significantly higher burden on the representational depth of the overall exchange. Providing support for such context-centric interoperability is the topic of much research within academia and industry alike [15].

Among the multitude of issues surrounding the subject of meaningful interoperability, it is the fundamental strength of context-centric, decision-support systems that poses the most challenging problem to this endeavor. This critical enabler, and simultaneous nemesis, is *representational depth*. As the name implies, the context-oriented approach to building decision-support systems endeavors to go beyond the classical *nuts and bolts* approach to representation (i.e., isolated chunks of typically numeric or string-based data with little or no inter-relationships and essentially void of any embedded semantics) and incorporate the potentially numerous relationships, implications, and rules that are needed for the more complex analysis inherent in agent-based, decision-support systems. A critical aspect of such representational depth is *perspective*. The biases associated with how something is *viewed* are very significant to the decision-making process within a particular domain. As such, perspective is a critical ingredient to effective context-oriented representation. Supporting the perspective of viewing a truck as a sequenced collection of assembly stages may be more appropriate, and effective, to an assembly-line management decision-support system than to view the automobile as a means of transporting cargo, the latter being more appropriate for a shipment planning system. The flip side to such representational depth is the increased disparity that inevitably develops between the representations upon which particular systems operate. In other words, the most empowering ingredient in context-oriented computing also presents one of the most difficult issues to deal with when such systems are asked to interoperate [18] [20].

The result of this paradigm is that for any meaningful interaction to occur between context-oriented systems there must be a translational component to the solution. To preserve the native biases (perspective) inherent in each interacting system, exchanged context must be transformed in a manner that incorporates the applicable perspectives of the receiver. It is this focus that drove the design and development of the web services-based, translational interoperability bridge presented in this discussion. The following sections describe the criteria for an effective solution to this paradigm, various technologies that show

significant applicability to this endeavor, and finally a discussion of a solution in the form of a translational, web services-based interoperability bridge successfully incorporating these ingredients.

Criteria

To successfully address the issues presented above, candidate solutions are required to meet several criteria. These properties range from adoption of available standards to exuding characteristics compatible with flexibility and reuse.

One of the primary goals of any solution intended for repeated application to varying interoperability scenarios is the ability to be flexible. A key aspect to such adaptability is the clear separation of framework from application specifics. In other words, support for the various abstractions associated with translation-based interoperability should be designed and implemented as a reusable framework. This framework should also identify the necessary interfaces outlining its connection to the application-specific side of the equation. The latter requiring the necessary functionality to effectively *adapt* client systems to the particular interoperability framework and interaction model presented by the solution.

Another important quality of a candidate solution is the promotion of available industry standards. This is particularly significant when a high degree of reusability by numerous parties is intended. Accordingly, the application of such a tool should center on industry-familiar technologies, standards, and tools. Not only does adherence to available standards aid in adoption of a particular solution but it constitutes an endeavor of significant importance in a field where complexity and *one-off* solutions abound.

As was identified in the previous section, a critical ingredient of interoperability among context-oriented systems is the preservation of individual perspective. This requires the ability to understand the subtleties inherent in such a concept (e.g., implied domain-specific constraints that have equally obscure and individual counterparts when considered from other domain-related perspectives). The logic required to support translation between such perspectives presupposes a level of reasoning akin to expert systems. In many respects, for the more complex context-oriented interaction a level of decision-support on par with solutions supporting multi-variable, complex problems may be appropriate. At times this may even suggest the inclusion of human decision makers to represent higher-level concepts not able to be adequately represented with current technology.

An additional, yet often times overlooked, criterion for a successful solution to the interoperability problem is not only the support of complex context translation but also the ability to support a more straightforward transformation without incurring the overhead associated with support for

the former. This is often the case where solutions targeting complex problems offer over-engineered and subsequently inefficient solutions for less taxing scenarios. The goal is to support a range of complexities and to limit any incurred overhead to situations where it cannot be avoided.

2. INGREDIENTS FOR A SOLUTION

The solution offered in this discussion takes the form of a translation-based, web service-oriented interoperability bridge based on a reusable framework. The interoperability bridge enhances the traditional web services architecture by also addressing the issue of differing representations between service users and service providers. As indicated earlier, supporting meaningful interaction among potentially disparate perspectives and subsequent representations is particularly important when dealing with context-oriented systems.

The application of a *service request* metaphor to inter-system collaboration allows each interoperating system to view other systems, and expose itself, as a collection of available services. The resulting interoperability model promotes a decoupled environment requiring no notion of system identity other than the standard service descriptors registered with the bridge's web services registry. Further, due to the embedded translational quality of this solution, bridge clients (i.e., service users and service providers) need not be concerned that the other might *speak a different language*. In support of such an interoperability model, a number of established, and emerging technologies may be employed.

Technologies

The web service-based, translational interoperability bridge incorporates a number of prominent technologies to accomplish its goal. First among these is web services architectures [2] [3] [10] [11] [12] [13]. By supporting standardized service lookup registries and interaction protocols, web services architectures present an extensible decoupled, capability-oriented model for system interaction. In this model services are employed on an as-needed basis allowing the classical notion of operational boundaries to effectively expand and contract, as additional capabilities are needed. Furthermore, adhering to standard interaction protocols, systems are empowered with a vehicle for discovering and engaging new capabilities. Although the issue of semantic discovery is still an area of significant research, web services architectures lay an effective foundation for such discovery-oriented dynamics.

The eXtensible Markup Language (XML) [9] [14] together with its Extensible Stylesheet Language Transform (XSLT) [1] language counterpart are two additional technologies employed by the interoperability bridge. XML provides a flexible means of defining structure through the use of syntactical tags. XML schemas can be developed to describe the structural aspects of entities, notions, and concepts.

Receivers can process XML documents based on these schemas in an interpretational manner. The result is a means whereby software components can process incoming content based on a previously unknown representation. However, it should be noted that such discovery is limited to structural characteristics and does not include the discovery of semantics, *or meaning*, vital in context-oriented decision-support systems. Even considering a schema describing the domain of concepts, rules, and implications, there is still a requirement for a pre-defined understanding by the receiver of the basic concepts, rules, and implications of the domain. At some point, the semantics need to be adequately represented beyond simply their structure. That said, however, structural discovery does play a significant role in the eventual goal of true contextual discovery but is only a piece of the puzzle.

The ability to describe discrete, interpretable structure can be exploited to support structural transformation between XML schemas. XSLT is one such language that can be used to describe exactly how content based on one schema can be mapped into another. XSL transforms, or rules, can be defined statically or dynamically and can be effectively applied in the case of straightforward, property-to-property translation. Translation at this level is useful however, for the more complex transformation inherent in context-oriented representations a more powerful paradigm is required. The additional reasoning required for this level of transformation can be successfully addressed through the use of inference engine-based technology. Similar to XSLT, inference engine-based transformation represents transformation logic as sets of managed rules. However, in the case of inference engines, these rule sets can be significantly more complex with support for extensive condition-based pattern matching and the subsequent management of progressively satisfied pattern matches. Some examples of rule-based inference engines are the CLIPS expert system shell developed by NASA [17] and the JESS inference engine developed by Sandia Laboratories [6]. In either case, complex transformation logic can be implemented as expert systems applying various levels of reasoning to determine the appropriate transformation. An illustration of the benefits of such capabilities would be the case, for example, where the transformation of a heavily constrained plan may need its *truth* maintained as it moves from one representational *world* to another. Under these capabilities assured truth maintenance may require a level of decision-support capable of reorganizing the plan in a manner that complies with the additional constraints described in the target world while still representing the initial intent, or goals, outlined in the source *world*. Availability of this level of transformation capabilities allows such activities to be functionally and architecturally encapsulated within the conceptual inter-system bridge. This ensures that any artifacts passing into a connected system's representational world are fully compliant with native constraints. The resulting interoperability model avoids the representational contamination associated with having to distinguish foreign

content from native content within a particular world, the former requiring a level of additional local processing to become compliant with local constraints before it can be reacted to.

3. A SOLUTION: TRANSLATIONAL WEB SERVICES INTEROPERABILITY BRIDGE

The overall design of the interoperability bridge is divided into two primary components. The first of these components, the Translational Web Services Center (or *Service Center* for short) forms the heart of the bridge and is

responsible for standard web services administration including management of the central web services registry as well as interaction between service requestors and the services they utilize. In addition, to effectively *bridge* representational differences between service requestors and providers, the Service Center is also responsible for transparently employing an appropriate translation service to perform any required translation. As such, both collaborators are effectively shielded from any differences in native representations yet are able to interact in a meaningful manner. The following section provides an in depth discussion of the Service Center and how it supports this level of interoperability.

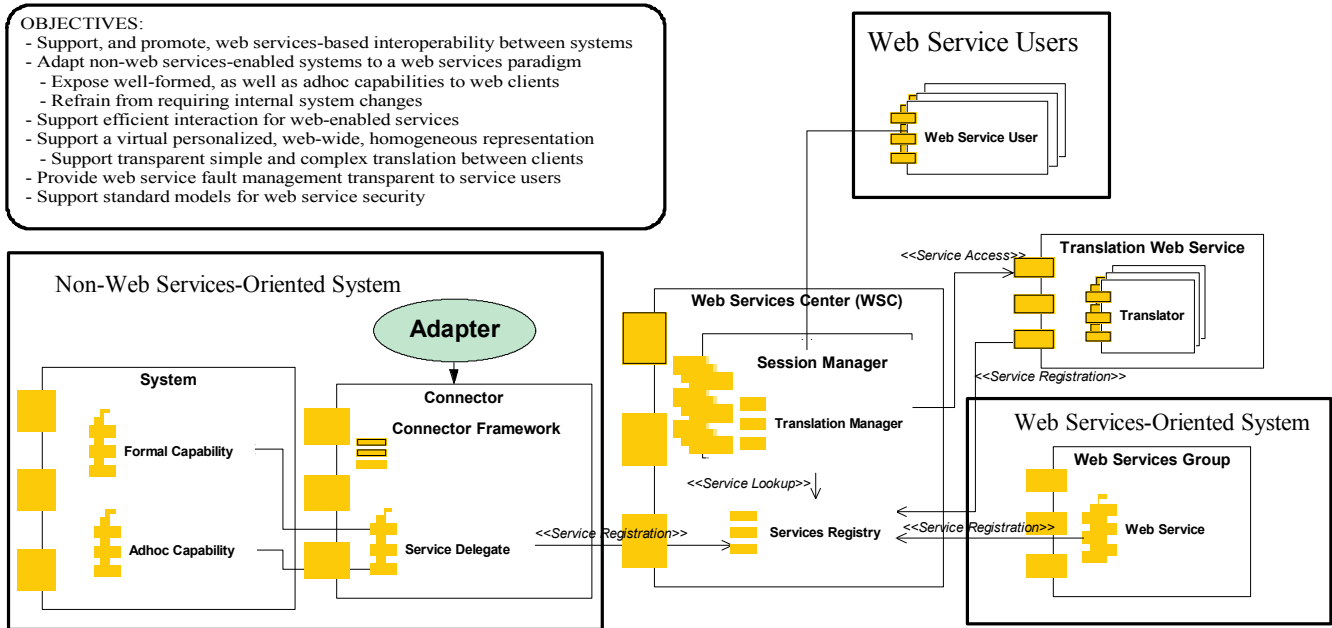


Figure 1 – Overall Session Architecture Supporting Meaningful Interoperability Between Collections of Both Formal and Ad Hoc Services

Translational Web Services Center (Service Center)

The Service Center is based on a standard web services framework [10], however, enhanced with a translation management facility. Figure 1 provides an illustration of how the Service Center integrates into a web services architecture consisting of both formal and more ad hoc-oriented services. As its primary role, the Service Center provides a web-enabled facility where service clients can discover and engage registered services using XML structured content over SOAP-based (Simple Object Access Protocol) [21] communication. As is illustrated in Figure 1, services can exist in fully formalized web-enabled form or in a more ad hoc manner that can be adapted to this web-services paradigm through employment of the Connector Framework (*the Connector Framework will be discussed in greater depth in a later section*). Regardless of the formality and sophistication of services, the Service Center presents its clients with a congruent web-services view regardless of native representation.

Although centered on a standard web services architecture, this solution extends such a model through its ability to seamlessly *bridge* representational differences between clients and the services they interact with. The Service Center manages this activity in the form of a Translation Manager. Not a translator itself but rather a manager of such activity, the Translation Manager is responsible for discovering and engaging suitable translation web services to perform the required representational mappings. Adhering to the common web services interaction model by itself acting as a client to the Service Center, determination and engagement of translation services by the Translation Manager is performed through Universal Description, Discovery, and Integration (UDDI)-based registry lookup and XML-based SOAP communication [10]. Where security is an issue, this process would include a level of authentication of trust since the chosen translation service would be decrypting the message content in order to perform the necessary translation. Since the translation

service is engaged in a standard web-services manner such authentication would follow well-established procedures.

Once web service clients locate a suitable service via the Service Center registry, interaction with that service occurs via the Service Center itself, rather than directly with the service. This is a clear departure from the typical model where once engaged, interaction between web service client and web service provider occurs in a direct point-to-point fashion. The motivation behind this deviation is based on the desire to shield both service client and provider from any responsibility for, or even notion of the representational translation occurring behind the scene. The disadvantage of this interaction model is the inclusion of an extra node (i.e., the Service Center) in service client and provider interaction. However, in practice there are numerous opportunities for various levels of optimization ranging from the spawning of dedicated Translation Managers per service session to actually supporting direct service client and provider communications when translation is not actually needed (i.e., client and provider *speak* the same language). In the case of direct interaction, the Service Center provides the client with the reference of the actual service as opposed to that of the Service Center itself. In any case, once a service has been located (i.e., a suitable service has been discovered and the Service Center has returned an appropriate reference) both parties collaborate with each other according to their own native languages. In essence, with respect to Service Center-based interaction each party is provided with a homogeneous representational view of the *world*.

In practice, the translation activity comes in multiple levels of complexity. In its simplest form, translation may be straightforward property-to-property mappings. In this case, a translation service employing XSLT-based transformation would suffice. However, in the case of context-centric systems (as service clients or providers), translation between representations may require a more sophisticated environment. In such situations an inference engine-based translator capable of managing communities of rule-based agents may be more appropriate. In either case, the Translation Manager can utilize the discovery-based registry to locate a suitable translation service.

As is alluded to above, the current design of the solution presented in this paper imparts no explicit responsibility on any part of the Service Center for configuring translation services with either the relevant representational schemas or the knowledge of how to map between them. Rather, the Service Center relies on its ability to locate and engage other web services capable of carrying out the needed translation. Outsourcing such responsibilities to those services that essentially *own* particular domains promotes the notions of maintainability, scalability, and design simplicity.

Connector Framework

The second component comprising the interoperability bridge, the Connector Framework, offers a reusable framework for adapting non-web services-savvy capabilities to the interoperability model promoted by the Service Center. Figure 2 illustrates the internal architecture of the Connector Framework along with the implemented interfaces encapsulating various client-specific details including exactly how to interact with local capabilities (i.e., actual services). Service Center clients requiring adaptation utilize a Connector configured with specific implementations of such interfaces to facilitate all direct interaction with the Service Center. Such interaction is essentially predicated on either the issuance or reception of web services-based communications (e.g., service registration, service lookup, service requests, and request results).

Outgoing communications, whether a request for service or the results of a locally satisfied service request, are managed collectively by the Export Manager, Export Adapter, and Export Formatter. The main function of the Export Adapter is to employ the mechanism offered by the local capability to receive outgoing communications. In many cases this mechanism may take the form of an event service capable of notifying interested parties (in this case, the Export Adapter) of various events (e.g., issuance of a service request). In other cases, such a mechanism may simply take the form of explicit method calls made to an extended Export Adapter implementation. However, the Connector Framework makes no demands of the efficiency or sophistication of such a facility other than its existence.

also be noted that both the Export and Import Formatters do not perform the type of representational transformation undertaken by the translational component of the Service Center. These formatters simply convert non-XML formats to their XML equivalents, and vice versa.

Once the request has been converted into the native format, the particular service delegate invokes the native capability to perform the requested service and manage the returning of any results to the Service Center as outgoing communications. Details of exactly how local capabilities are invoked and interacted with are fully encapsulated inside the service delegate and may take a variety of forms including direct interface interaction or creation of a local event triggering the desired functionality. Regardless of the means of invocation, the functionality being requested may be at varying stages of formality. In other words, since the service delegates are essentially the web service-savvy representatives of a particular set of functionality, exactly what local functionality constitutes an externally exposed service is encapsulated, and can therefore be essentially determined, by the particular delegate. This is particularly useful when adapting legacy functionality to a web service-oriented interoperability paradigm. The actual functionality comprising a particular service need not be aware of the *grander scheme* of interoperating with other systems.

The scenario presented above is, of course, most suitable where there is no native concept of web services interoperability. However, in the case where native capabilities are designed to operate in a web services paradigm, the role of Service Delegates can be reduced to managing the reformatting of communications in the case of non-XML systems, or, in the case where XML is supported, omitted completely. In the latter scenario the native web services capability would manage its own exposure to the Service Center but would still benefit from the virtual homogeneous representational environment supported by the translational component of the Service Center.

4. CONCLUSION

The Translational Web Services Interoperability Bridge presents an effective means where by existing, perhaps loosely defined, system functionality can be adapted to operate in a web services paradigm. Through the use of Service Delegates, the details associated with directly interfacing with local system functionality are encapsulated and effectively isolated from reusable framework components. With flexibility as a fundamental theme, systems developed with such service-oriented concepts more native to their design are able to avoid any undue overhead associated with such adaptation and exploit the functionality offered by the Service Center in a more direct fashion.

The solution to interoperability presented in this discussion goes beyond traditional web services architectures by

supporting the representational disparity typically exhibited by context-oriented systems. Rather than constraining interoperating systems to common representations, the interoperability bridge provides a mechanism for managing the potentially complex representational translation between interoperating systems. As a result, interoperating systems can function in an extended service-oriented world while still maintaining their significantly biased perspectives critical to context-based decision-support.

REFERENCES

- [1] Cagle, K., M. Corning, J. Diamond, T. Duynstee, O. Gudmundsson, M. Mason, J. Pinnock, P. Spencer, J. Tang, A. Watt, J. Jirat, P. Tchistopolskii, and J. Tennison, "Professional XSL", Wrox Press Ltd., Birmingham, UK., 2001
- [2] Daconta M., L. Obrst and K. Smith, "The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management", Wiley, Indianapolis, IN., 2003
- [3] Ewalt D., "The Next Web", Information Week, October 2002, (www.informationweek.com/story/IWK20021010S0016)
- [4] Fikes R. and D. McGuinness, "An Axiomatic Semantics for RDF, RDF Schema and DAML+OIL", KSL Technical Report (KSL-01-01), October 2001, (www.ksl.stanford.edu/people/dlm/daml-semantic/abstract-axiomatic-semantic.html)
- [5] Fowler M and K Scott, "UML Distilled: Applying the Standard Object Modeling Language", Addison-Wesley, Reading, Massachusetts, 1997.
- [6] Friedman-Hill, E., "JESS In Action", Manning Publications Co., Greenwich, CT, 2003
- [7] Garshol L. and G. Moore (eds.), "The XML Topic Maps (XTM) Syntax", JTC1/SC34:ISO 13250, July 22, 2002, (www.y12.doe.gov/sgml/sc34/document/0328.htm)
- [8] Giarratano J. and Riley G., "Expert Systems: Principles and Programming", 2nd Edition, PWS Publishing Company, Boston, MA.
- [9] Gil Y. and V. Ratnakar, "Markup Languages: Comparison and Examples", Information Sciences Institute, University of Southern California, TRELIS project, 2002, (www.isi.edu/expect/web/semanticweb/comparison.html)
- [10] Graham S., S. Simeonov, T. Boubez, D. Davis, G. Daniels, Y. Nakamura, and R. Neyama, "Building Web Services with Java: Making Sense of XML, SOAP, WSDL, and UDDI", Sams Publishing, Indianapolis, IN, December 2001

- [11] Heflin J., R. Volz and J. Dale (eds.), "Requirements for a Web Ontology Language", W3C Working Draft, July 8, 2002, (www.w3.org/TR/webont-req)
- [12] Hendler J., T. Berners-Lee and E. Miller, "Integrating Applications on the Semantic Web", Journal of the Institute of Electrical Engineers of Japan, 122(10), October 2002, (pp.676-680).
- [13] Horrocks I., "DAML+OIL: A Description Language for the Semantic Web", IEEE Intelligent Systems, Trends and Controversies., 2002
- [14] Hunter D., C. Cagle, D. Gibbons, N. Ozu, J. Pinnock, and P. Spencer, "Beginning XML", Wrox Press Ltd., Birmingham, UK., 2000
- [15] Karsai G., "Design Tool Integration: An Exercise in Semantic Interoperability", Proceedings of the IEEE Engineering of Computer Based Systems, Edinburgh, UK, March, 2000
- [16] Manola F. and E. Miller (eds.), "RDF Primer", W3C Working Draft, March 19, 2002, (www.w3.org/TR/2002/WD-rdf-primer-20020319/)
- [17] NASA, "CLIPS 6.0 Reference Manual", Software Technologies Branch, Lyndon B Space Center, Houston, Texas, 1992
- [18] Pohl J., "Information-Centric Decision-Support Systems: A Blueprint for Interoperability", Office of Naval Research (ONR) Workshop hosted by the CAD Research Center in Quantico, VA, June 5-7, 2001
- [19] Pohl J, A Chapman, K Pohl, J Primrose and A Wozniak, "Decision-Support Systems: Notions, Prototypes, and In-Use Applications", Technical Report, CADRU-11-97, CAD Research Center, Design Institute, College of Architecture and Environmental Design, Cal Poly, San Luis Obispo, CA, January, 1997
- [20] Pohl K., "Perspective Filters As A Means For Interoperability Among Information-Centric Decision-Support Systems", Office of Naval Research (ONR) Workshop hosted by the CAD Research Center in Quantico, VA, June 5-7, 2001
- [21] Simple Object Access Protocol (SOAP) Version 1.1; www.w3.org/TR/soap

AUTHOR BIOGRAPHIES

Kym J. Pohl is a senior software engineer with CDM Technologies Inc. in San Luis Obispo. His current focus is on agent-based, collaborative decision-support systems with particular interest in representation and collaboration architectures. Following an undergraduate degree in Computer Science he earned Master degrees in Computer Science and Architecture. Over the past 15 years he has provided technical leadership in the design and development of a number of multi-agent decision-support systems for the US Department of Defense, including the Integrated Marine Multi-Agent Command and Control System (IMMACCS) for tactical command and control and the SEAWAY system for the coordination of logistical sea-based sustainment operations.



Lakshmi Vempati is a Software Developer Specialist at CDM Technologies Inc. in San Luis Obispo, CA. Some of her interests include agent based decision support systems, modeling and simulation and all aspects of aviation and spaceflight. She has a Bachelors degree in Electrical Engineering and a Masters degree in Aerospace Engineering.

