Organizational Advancements through Enterprise Information Systems:
Emerging Applications and Developments

Angappa Gunasekaran
University of Massachusetts—Dartmouth, USA

Timothy Shea
University of Massachusetts—Dartmouth, USA

BUSINESS SCIENCE REFERENCE

Hershey · New York

Director of Editorial Content: Kristin Klinger
Senior Managing Editor: Jamie Snavely
Assistant Managing Editor: Michael Brehm
Publishing Assistant: Sean Woznicki
Typesetter: Michael Brehm, Michael Killian
Cover Design: Lisa Tosheff
Printed at: Yurchak Printing Inc.

Published in the United States of America by
Business Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue
Hershey PA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: http://www.igi-global.com/reference

Library of Congress Cataloging-in-Publication Data

Organizational advancements through enterprise information systems : emerging applications and development / Angappa Gunasekaran and Timothy Shea, editors.
p. cm.
Includes bibliographical references and index.
Summary: "This book provides a comprehensive assessment of the latest developments in the EIS revolution. including Enterprise Resource Planning (ERP) adoption, the integration of enterprise systems, personalized ERP, and the Semantic Web, and ideas and solutions for the future of the global enterprise"--Provided by publisher.
ISBN 978-1-60566-968-7 (hardcover : alk. paper) -- ISBN 978-1-60566-969-4 (ebook : alk. paper) 1. Information technology--Management. 2. Management information systems. 3. Business planning. 4. Information resources management. I. Gunasekaran, A. II. Shea, Timothy, 1954- III. Title.

HD30.2.O73773 2010
658.4'038011--dc22

2009042278

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

Chapter 21
Semantic Web Services for Simulation Component Reuse and Interoperability: An Ontology Approach

Simon J. E. Taylor
Brunel University, UK

David Bell
Brunel University, UK

Navonil Mustafee
Brunel University, UK

Sergio de Cesare
Brunel University, UK

Mark Lycett
Brunel University, UK

Paul A. Fishwick
University of Florida, USA

ABSTRACT

Commercial-off-the-shelf (COTS) Simulation Packages (CSPs) are widely used in industry primarily due to economic factors associated with developing proprietary software platforms. Regardless of their widespread use, CSPs have yet to operate across organizational boundaries. The limited reuse and interoperability of CSPs are affected by the same semantic issues that restrict the inter-organizational use of software components and web services. The current representations of Web components are predominantly syntactic in nature lacking the fundamental semantic underpinning required to support discovery on the emerging Semantic Web. The authors present new research that partially alleviates the problem of limited semantic reuse and interoperability of simulation components in CSPs. Semantic models, in the form

DOI: 10.4018/978-1-60566-968-7.ch021

of ontologies, utilized by the authors' Web service discovery and deployment architecture, provide one approach to support simulation model reuse. Semantic interoperation is achieved through a simulation component ontology that is used to identify required components at varying levels of granularity (i.e. including both abstract and specialized components). Selected simulation components are loaded into a CSP, modified according to the requirements of the new model and executed. The research presented here is based on the development of an ontology, connector software, and a Web service discovery architecture. The ontology is extracted from example simulation scenarios involving airport, restaurant and kitchen service suppliers. The ontology engineering framework and discovery architecture provide a novel approach to inter-organizational simulation, by adopting a less intrusive interface between participants Although specific to CSPs this work has wider implications for the simulation community. The reason being that the community as a whole stands to benefit through from an increased awareness of the state-of-the-art in Software Engineering (for example, ontology-supported component discovery and reuse, and service-oriented computing), and it is expected that this will eventually lead to the development of a unique Software Engineering-inspired methodology to build simulations in future.

INTRODUCTION

Commercial-Off-The-Shelf (COTS) Simulation Packages (CSPs) offer an interactive and visual modeling development environment for creating computer models of existing and proposed systems as well as for experimenting with the models themselves. Simulation practitioners in industry extensively use CSPs such as Simul8 (Concannon, et al., 2003), Witness, AnyLogic, AutoMod and Arena to model their simulations. These packages allow reuse of standard simulation components like workstations, queues, conveyors, resources, etc. and thereby provide the building blocks which facilitate the creation of larger models. As these models grow larger and more complex the prospect of simulation model reuse and interoperability is appealing as it has the potential to reduce the time and cost incurred in developing future models. An extension of model reusability is the concept of separate development and user groups, whereby models are developed and validated by one group and then used to specify simulations by another group (Bortscheller & Saulnier, 1992). This is collaborative model building. Collaborative model building is increasingly gaining prominence as models become large and complex and there is an increasing need among modelers, who may be specialising in different domains, to join together to conduct a simulation study. A few software vendors have started integrating solutions that facilitate such parallel and co-operative model development, for example, the Teamwork and Concurrent Version System (CVS) Integration provided by AnyLogic (XJ Technologies). In particular, the opportunity to interoperate models (running together in separate CSPs on separate computers linked via a network) is attractive as this approach avoids the costs of "cut and paste" integration. In this paper we look at the *discovery and import* of CSP-created models across organizational boundaries within the context of industrial supply chains, thus enabling development and user groups to exist in different organizations. This approach does not allow model information hiding between enterprises and contrasts with the *distributed simulation* approach to model reuse that enables an organization to hide model specific information and data from the other participants.

To motivate our approach, consider the area of Supply Chain Management (SCM). This consists of a series of tasks such as manufacturing,

transport and distribution that are undertaken by organizations with the aim of delivering products to their customers. Simulation of the supply chain can identify manufacturing bottlenecks, resources required for on time delivery, adequate stock levels for distribution etc. and help to improve the performance of the underlying supply chain. Each organization that forms a part of the supply chain normally develops models that simulate their own part of the supply chain using CSPs (Fujimoto, 2000). Assuming that all necessary individual simulation components are available then the question is how to link or interoperate them together. Distributed simulation offers one such solution. Distributed simulation can be defined as the distribution of the execution of a single run of a simulation program across multiple processors (Taylor et al., 2001). It allows each organization to run its model within its own site (thereby encapsulating model details within the organization itself) and participating with other sites through information exchange using distributed simulation middleware. Gan et al. (2000), Boer et al. (2002), Mertins et al. (2000), Gan et al. (2005), Taylor et al. (2005), Mustafee & Taylor (2006), Mustafee et al. (2009) are examples of successful distributed simulation using CSPs. There is a growing body of research dedicated to creating distributed simulation with CSPs and the High Level Architecture (HLA), the IEEE 1516 standard for distributed simulation. In an attempt to unify this research, the COTS Simulation Package Interoperability Product Development Group (CSPI-PDG), a Simulation Interoperability Standards Organization (SISO) standardization group, began operation in October 2004 (http://www.sisostds.org/).

The distributed simulation approach to achieving reusability in the context of CSPs faces the following challenges: (1) A lack of widespread demand for distributed simulation in industry has meant that the CSP vendors have not currently incorporated distributed simulation support into their products. Consequently, the organizations that want to use this approach do not have ready-made solutions; (2) Research projects that create CSP-based distributed simulations do not have access to the source code and are thus limited by the functionality offered by the vendor; and (3) Execution of a distributed simulation tends to be much slower than traditional standalone simulation. For example, the straightforward use of the conservative HLA time advance mechanisms results in a simulation that runs extremely slowly, at times a few factors slower that its corresponding sequential runs (Gan et al., 2005). However, for larger and more complex models, distributed simulation could be a feasible alternative (Mustafee et al., 2009). In order to progress, these issues have to be resolved before the industry can fully benefit from the application of CSP-based distributed simulation. Our approach is a step towards this as it facilitates the discovery of models.

Our discovery and import approach to model reuse in the context of CSPs offer an alternative to the distributed simulation approach. By *discovery* we mean that individual simulation models, which are created by organizations to model their activity in the supply chain, are discovered from among an inter-organizational repository of models spread across the web. The selected models are then loaded into a CSP, modified according to the requirements of the new model and executed. We believe that our approach to enabling CSP-based supply chain simulation has fewer technical limitations, especially when compared to using distributed simulation technique to connect different CSP-based components of the supply chain simulation. Mustafee et al. (2006, 2009) have previously implemented such an approach to model the UK National Blood Service (NBS) blood supply chain through use of the HLA. The authors have concluded that the level of technical expertise required to implement a CSP-based distributed simulation is significant, and for wider adoption of this approach it may be required that distributed simulation middleware be integrated with the CSP packages. This, in turn,

would generally require intervention of the COTS package vendors, as source code changes may be necessary. However, the alternative approach to reusing CSP-based components that we present in this paper will alleviate the steep-learning curve that is associated with learning distributed simulation technique. Furthermore, the requirement for the CSP vendor to intervene in the short run may also be by-passed. We therefore refer to this CSP-model reuse approach as the "lighter" approach (the distributed simulation approach being considered "heavier").

Our vision is a web of Simulation Component (SC) models that are accessible to the practitioner. The current representations of web components are predominantly syntactic in nature lacking the fundamental semantic underpinning required to support discovery on the emerging Semantic Web (Bell et al., 2005). Semantic models, in the form of web ontologies, utilized by web service discovery and deployment architectures provide one approach to support simulation model reuse. Improved component reuse supported by ontological models has already been proposed in simulation (Fishwick & Miller, 2004). When considering COTS simulation packages, intrusive activities are not possible when dealing with packaged software as only import or export capabilities are achievable. The tools of the Semantic Web provide a means to construct external descriptions of the CSP models. This external description, or ontology, can then be used to support the reuse of simulation components. Consider a scenario where a large multinational organization uses CSPs to model many of its business activities. Two human processes are undertaken when a simulation is required – the creation of the model and its execution. In order to fully utilize the capabilities within the organization we propose that *model parts* be reused more effectively, better utilizing the expertise within distinct models. In order to support component reuse, methods for describing the models that enable semantic discovery are proposed. The system supports the discovery of specific model components and their loading into the COTS simulation package. Semantic interoperation is achieved through the use of simulation component ontology to identify required components at varying levels of granularity (including both abstract and specialized components). Once selected, simulation components are loaded into a CSP, modified according to the requirements of the new model and executed. The ontology is derived from existing CSP simulation components and is contrasted to the current simulation ontology. We propose that the evolutionary construction of domain-grounded simulation component ontology better supports the semantic discovery of simulation components. In addition, when combined with hard simulation semantics (i.e., state), concepts from both vocabularies provide improved matching terms.

The chapter is organized as follows. Section 2 presents a summary of pertinent literature. Section 3 describes the Discrete Event Simulation Component (DESC) ontology and the process undertaken to engineer it. Section 4 covers the software tools that use the DESC ontology – the semantic search and component integration software. A conclusion summarizes the work presented.

RELATED LITERATURE

Three areas of research are relevant to the work presented here: *COTS simulation package interoperability*, *semantic web services* and *grid resource discovery*. Together they provide an insight into the decoupling of component simulation models from their execution environment and are used for discovery and synthesis. To outline reuse and interoperability problems in this area we first discuss *COTS simulation package interoperability*. We then introduce the precepts to our approach: *semantic search and ontology*.

COTS Simulation Package (CSP) Interoperability

The simple act of linking together, or interoperating, two or more CSPs and their models, can be extremely complex. This is due to time synchronization requirements and the complexity of distributed simulation algorithms and/or software used to create the link (such as the runtime infrastructures based on the IEEE 1516 High Level Architecture standard (IEEE 2000)) (Fujimoto, 2000). This complexity can often hide the precise nature of what is being shared between these interoperating CSPs. To attempt to simplify this, the Simulation Interoperability Standards Organization's (SISO) COTS Simulation Package Interoperability Product Development Group (CSPI PDG) are developing approaches to the standardization and simplification of CSP interoperability. The first major development by the CSPI PDG is a set of Interoperability Reference Models (IRMs) to help make this simplification possible. First introduced in detail in Taylor, et al. (2006), these IRMs are effectively design patterns for CSP interoperability. The IRMs are a set of guidelines for CSP interoperability and were first introduced in Taylor et al. (2008). The IRMs are discussed next.

IRMs or "interoperability design patterns" are effectively a set of simulation patterns or templates, which enable modelers, vendors and solution developers to specify the interoperability problems that must be solved. The Interoperability Reference Models (IRMs) are intended to be used as follows:

- To clearly *identify* the model/CSP interoperability *capabilities* of an *existing* distributed simulation, e.g. the distributed supply chain simulation is compliant with IRMs Type A and B.
- To clearly *specify* the model/CSP interoperability *requirements* of a *proposed* distributed simulation, e.g. the distributed hospital simulation must be compliant with IRMs Type A and C.

An IRM is defined as the simplest representation of a problem within an identified interoperability problem type. Each IRM can be subdivided into different subcategories of problem. As IRMs are usually relevant to the boundary between two or more interoperating models, models specified in IRMs will be as simple as possible to "capture" the interoperability problem and to avoid possible confusion. These simulation models are intended to be representative of real model/CSPs but use a set of "common" model elements that can be mapped onto specific CSP elements. Where appropriate, IRMs will specify time synchronization requirements and will present alternatives. IRMs are intended to be cumulative (i.e. some problems may well consist of several IRMs). Most importantly, IRMs are intended to be understandable by *simulation developers, CSP vendors and technology solution providers.*

There are presently four different types of IRMs. These are described in Table 1. The reader is referred to Taylor et al. (2008) for an extensive discussion on the IRMs.

An example of **Type A Entity Transfer IRM** is presented next in relation to a distributed blood supply chain simulation created using CSP Simul8.

UK National Blood Service (UK NBS) is a public funded body in the UK that is responsible for distributing blood and associated products. The analysis of this health care supply chain is of particular interest as blood donors are in short supply, the shelf-life of blood products is relatively short and blood product ordering policies are potentially complex. The UK NBS is a part of the National Health Service (NHS) Blood and Transplant (NHSBT) organization. The NBS is responsible for collecting blood through voluntary donations, testing the blood for ABO and Rhesus grouping and infectious diseases such as HIV, processing the blood into around 120 different

Table 1. Interoperability reference models

IRM Type	IRM Name	IRM Description
Type A	Entity Transfer	Deals with the requirement of transferring entities between simulation models, such as an entity *Part* leaves one model and arrives at the next.
Type B	Shared Resource	Deals with sharing of resources across simulation models. For example, a resource R might be common between two models and represents a pool of workers. In this scenario, when a machine in a model attempts to process an entity waiting in its queue it must also have a worker. If a worker is available in R then processing can take place. If not then work must be suspended until one is available.
Type C	Shared Event	Deals with the sharing of events across simulation models. For example, when a variable within a model reaches a given threshold value (a quantity of production, an average machine utilization, etc.) it should be able to signal this fact to all models that have an interest in this fact (to throttle down throughput, route materials via a different path, etc.).
Type D	Shared Data Structure	Deals with the sharing of variables and data structures across simulation models. Such data structures are semantically different to resources, for example a bill of materials or a common inventory.

products (of which the main three are Red Blood Cells, plasma and platelets), storing the stockpile and transferring excess stock between different NBS centers, and finally issuing the different blood products to the hospitals as per their needs. The NBS infrastructure consists of 15 Process, Testing and Issuing (PTI) centers which together serve 316 hospitals across England and North Wales.

Blood products are stored in the PTI Centers until they are requested by the hospitals served by that Center. A hospital places an order for blood products when its inventory falls below a predetermined order point, or when rare products not held in stock are requested for particular patients. Hospitals normally receive their orders daily and the blood remains in the hospital bank until it is cross-matched (tested for compatibility) for a named patient. It is then placed in "assigned inventory" for that patient for a fixed time after the operation. If it is not used, it is returned to "unassigned inventory" and can be cross-matched again for another patient. On average a unit will be cross-matched four times before it is used or outdated. In practice, however, only half of the cross-matched blood is actually transfused. The original simulation ran on one PC and is described in (Katsaliaki and Brailsford 2006).

The problem faced by this simulation is speed. Mustafee et al. (2009) developed a distributed simulation that demonstrated that considerably length runtimes on a single computer could be reduced by distributing the simulation over several PCs. Without the use of the IRMs it would be difficult to write down the interoperability requirements in a common "language." With the IRMs this task becomes quite straightforward. There are no shared resources, events or data structures; the distributed simulation only requires the exchange of entities. There are two types of entity: orders and blood units. There are no bounded buffers in this model and there is no need to preserve queuing discipline when multiple entities arrive simultaneously. There is a travel time between the PTI Centre and hospitals. We can therefore quite clearly and simply state that the NBS distributed simulation implementation is compliant with **IRM Type A Entity Transfer.**

The distributed simulation (interoperability) implementation of the NBS blood supply chain shows how other supply chain simulations might be implemented (and some of the associated issues). However, the models still need to be found/discovered. This is particularly difficult if models exist across organizational boundaries. To began to explain our approach we now discuss the background to semantic search and ontologies.

Semantic Web Services, Semantic search and Ontology

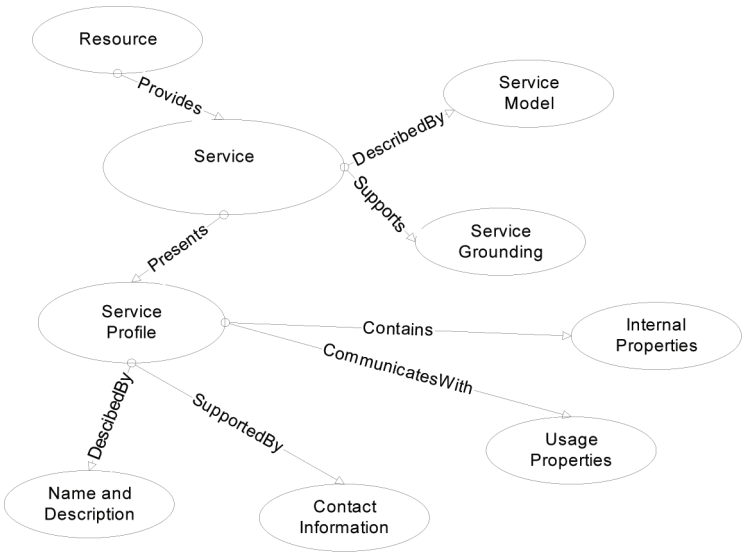
Semantic search has been applied to both semantic web services and grid resource discovery with a common reliance on knowledge modeled through ontologies. Ontology itself is a specification of a representational vocabulary for a shared domain of discourse – with definitions of classes, relations, functions, and other objects (Gruber, 1993). It is an explicit specification of a conceptualization. The term is borrowed from philosophy, where an ontology is a systematic account of existence (Gruber, 1993). In borrowing the term ontology and placing it into an engineering discipline, two distinct usage types emerge in the creation of these specifications: The theoretic (deductive) approach and the pragmatic (inductive approach) (Geerts & McCarthy, 1999). It is the pragmatic approach that is adopted in this paper – focusing on the engineering of knowledge from CSP models.

The Semantic Web provides the knowledge structure and reasoning about a web of models. Such knowledge is applied within the context of a grid of CSPs that are able to execute discovered models. The Semantic Web (Berners-Lee et al, 2001) aims to uncover knowledge about domains so as to better support discovery, integration and understanding of resident objects. Semantic web services (SWS) refine this vision (McIlraith et al., 2001) making web services "computer-interpretable, use apparent, and agent-ready". With this web of services comes a need to describe explicitly and in a form able to be read by computers.

Current intersections between web services and the Semantic Web have delivered a diverse body of research. The agent community (McIlraith et al., 2001; Gibbins et al., 2003; Martin et al., 1999) has recognized the benefit of ontology if computer-to-computer web architectures are to be achieved. Combining service and domain ontology is seen as a key to achieving service synthesis (Chen et al., 2003). Work on service ontology is currently centered on the OWL-S and WSMO groups. Recognizing the progress, by the DAML Consortium and others, attention has moved from ontology languages to specific application areas like services. A discussion of semantic web services would not be complete without coverage of the OWL-S upper ontology model (WSMO being similar in nature). The OWL-S high level model describes the relationship between the differing service decompositions (see Figure 1) (Chen at al., 2003; Ankolekar et al., 2001). A resource provides a service that is represented by the ServiceProfile, described by the ServiceModel and supported by the ServiceGrounding. Generally, the profile describes the service in a high level way (enough to discover the service), the model describes the detail of how it works and can be used to: (1) perform more in-depth analysis of whether the service meets a need, (2) to compose service descriptions from multiple services to perform a specific task, (3) during enactment, to co-ordinate activities from participants and (4) to monitor execution (Ankolekar et al., 2001). The service grounding details practical access and has converged with WSDL.

OWL-S (and WSMO) (Lara et al., 2004) provide generalized models for describing services. Others have identified the need for specialized common concepts within a web service context (Lara et al., 2004; Cardoso & Sheth, 2003; Paolucci et al., 2002; Curbera et al., 2002; Tosic et al, 2002), with one example being quality of service. These concepts represent glue homogenizing a wealth of asymmetrically described web resources. New issues become pertinent in a Semantic Web of a "great number of small ontological components consisting largely of pointers to each other" (Hendler, 2001). This semantic web service environment, with recognition of the need to combine service and domain ontologies, warrants research that identifies practical approaches for businesses to combine the service ontology with existing or new domain ontologies. The foremost question in semantic service orientation is how best this should be undertaken in the context of simulation.

Figure 1. OWL-S upper ontology



Transporting this vision to a simulation environment with a web of simulation components has several challenges. Combining distributed SC models into a new model requires that they are discovered. Consequently, explicit, computer readable knowledge is required for such search tasks. Knowledge in the form of ontologies has already been applied to simulation (Fishwick & Miller, 2004) with work by the University of Florida on simulation translation and University of Georgia on a taxonomy of simulation objects called DeMO. DeMO provides a precise description of simulation models with hard semantics. In order to realize a vision for SCs, similar to that of SWS, requires that the domain being simulated is represented explicitly (an OWL ontology (Smith et al., 2004)). The DeMO ontology (Fishwick & Miller, 2004) is an upper ontology that details events, activities and processes. Hard semantics work perfectly if all stakeholders adopt the single model. If this is not the case, and with only the CSP SCs, a transformation directly to such a model will likely miss tacit domain concepts that may help any subsequent SC search activity.

The eXtensible Modeling and Simulation Framework (XMSF) is defined as a set of composable standards, profiles and recommended practices for web-based modeling and simulation. XMSF prescribes the use of ontologies for the definition, approval and interoperability of complimentary taxonomies that may be applied across multiple simulation domains (Bhatt et al., 2004). In military modeling and simulation, the study of ontology is recognized as important in developing techniques that would allow semantic interoperability between simulation systems and to this effect the ontology of C2IEDM (Command and Control Information Exchange Data Model) has been created to further studies on enabling interchange of data between two or more systems (Tolk & Turnitsa, 2004). Work is also underway for creating an ontology for physics which would represent physics-based model semantics in modeling and simulation. Its intention is to capture the concepts of physical theories in a formal language so as to support various forms of automated processing that are currently not supported (Collins, 2004). An ontology for the representation of data pertaining to a Synthetic Environment called

sedOnto (Synthetic Environment Data Representation Ontology) has been proposed Bhatt et al., 2004). Finally, ongoing work is looking into establishing an ontology for BML, an unambiguous language to command and control forces and equipment (Tolk & Blais, 2005). We now present our ontology-based approach.

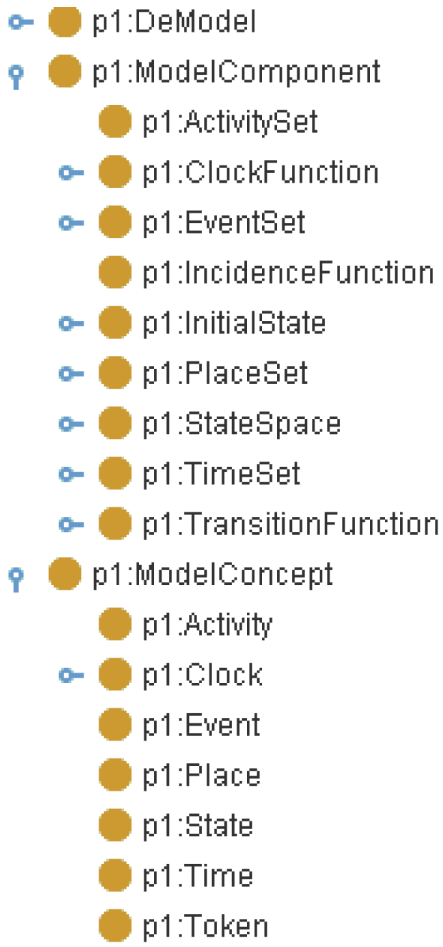
Figure 2. DESC-restaurant ontology structure

SIMULATION COMPONENT ONTOLOGY

Requirement for Semantic Search

The globalization of many organizations and industries often results in a fragmentation and heterogeneity of knowledge produced by its domain experts. In order to synthesize the most appropriate knowledge in a model, the best available model parts must first be found. Syntactic and taxonomic approaches limit the precision in which SCs can be related to the domain. Typical

Figure 3. DeMO ontology structure



issues are that a component may not fit neatly into a prescribed category or simple use of synonyms to describe the component.

The Discrete Event Simulation Component Ontology

The Discrete Event Simulation Component (DESC) ontology resulted from two distinct research activities: (1) the transformation of CSP models into OWL ontology files and (2) semantic search scenarios being carried out against the OWL files. Snapshots of DeMO and DESC ontologies are presented in figures 2 and 3. The differences are apparent with DeMO focusing on the component properties and DESC on the component in relation to the domain. Links between the two models are achieved through referencing the DeMO:ModelComponent from the DESC:SimulationConcept when it relates to an available component model. Additionally, the DeMO ontology is imported by the DESC ontology so that the latter can use classes and properties of the former (for example, when describing a business concept that is a specific *state* or *activity* in the simulation).

The ontology was created using the Protégé tool from Stamford University (with OWL plugins) (http://protege.stanford.edu/). A decision was made to ground the ontology in the domain

Table 2. Process for deriving semantic content from CSP models

Activities	Description	Impact
Component Extraction	Specific components are extracted to form distinct models. These are stored in the DESC library (a standard web server).	CSP models SC Models
Component Typing	A new class is added to the OWL ontology to represent the SC. Similar classes are grouped under a type.	OWL Classes
Component Dependency Models	Extended DeMO properties are used to define dependencies between services. E.g. StateDependency. Reference DeMO concepts when describing business properties (e.g. ThinkingTable has a DeMO state property). New classes and properties are created for previously implied activities etc. (e.g. Serving is a created from an analysis of table in ordering and eating).	OWL Properties New OWL Classes and properties implied from the model
Ontology Testing	The finalized ontology is loaded into the SEDI4G server and several search tasks are undertaken.	DESC OWL File

Figure 4. Simul8 model

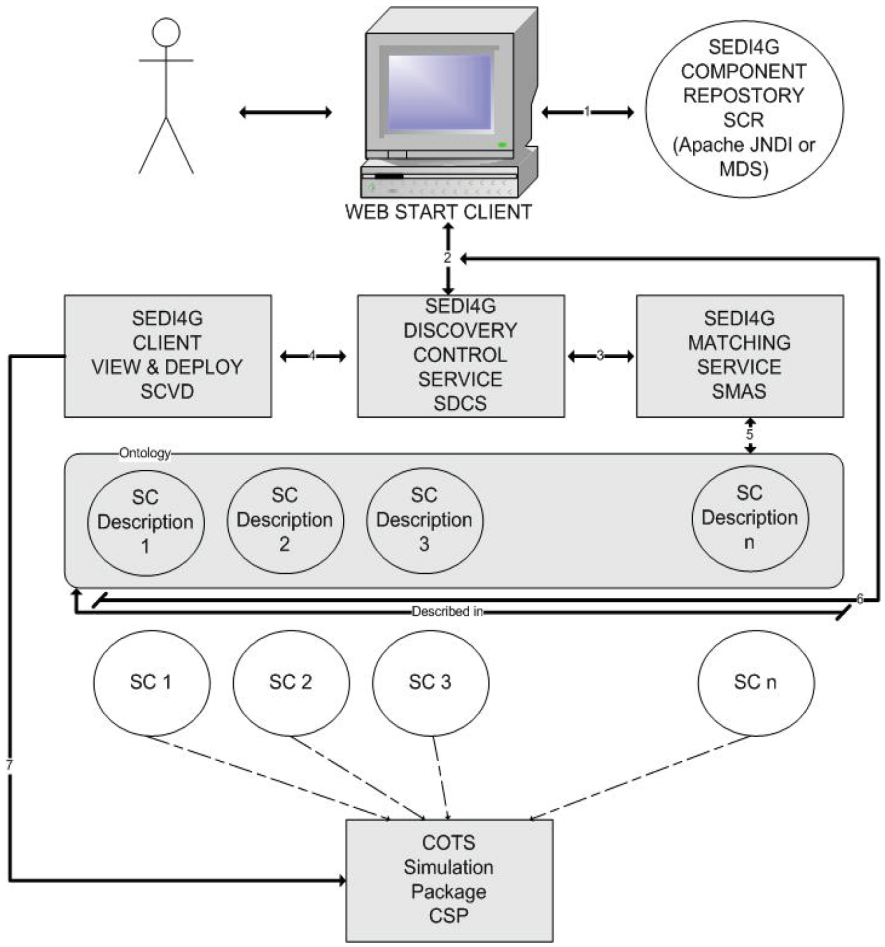
language of existing SCs as opposed to using a particular service ontology such as OWL-S or WSMO.

Ontology Engineering

A number of activities were carried out to transform three CSP models into an ontological form, i.e. as files written in the Web Ontology Language (OWL). The process included the decoupling of the SCs from the model by placing distinct component models into a web-based component library (URI accessible). The framework of the activities carried out in this work is detailed in Table 2. The framework evolved as each CSP model was deconstructed and transformed into ontology classes (including relations to dependent or related classes). Realization of the need for a DESC ontology resulted from this process – which included the adoption of DeMO for hard component semantics.

The ontology engineering process resulted in DESC-RESTAURANT (Figure 2), DESC-KITCHEN and DESC-AIRPORT models (OWL Files). Each provided more component returns as concept inferencing was able to traverse the concept tree and return additional suitable candidates. The process undertaken to engineer the domain simulation ontology provides the basis for subsequent modelers to reference and extend

Figure 5. Discovery architecture



the domain ontology; thus achieving richer search results and evolving into a large component ontology. The ontology engineering process systematically analyses the CSP model, of which figure 4 is a simple example.

DISCOVERY AND IMPORT OF SIMULATION COMPONENTS

Our *discovery and import* approach aimed at CSP model reuse enables us to (1) semantically search for the desired simulation models and (2) parse and import the identified models into a simulation package. For our demo application we have used CSP Simul8. Simul8 enables users to rapidly construct accurate, flexible and robust simulations using an easy-to-use visual modeling interface (Curbera et al., 2002). However, our discovery and import architecture has the potential to support any CSP that allows an external program to perform basic operations such as opening the CSP and loading a model through its Component Object Model (COM) interface (Gray, et al., 1998). COM is a Microsoft technology that allows different software components to communicate with each other by means of interfaces (Tosic et al, 2002). The discovery component of our architecture (described in section 4.1) can be used with very little change to support other CSPs. The parse and import component, however, would require implementation of a CSP specific parser (described in section 4.2) and cannot be reused.

Design of Component Discovery System

The component discovery system is an extension of the SEDI4G architecture (Bell & Ludwig, 2005). Extending the application to support simulation component (model) (SC) descriptions as well as grid services required only minor configuration changes to support the new OWL DESC ontology. The semantic discovery system shown is figure 5 comprises a set of web services (SCVD, SDCS and SMAS).

The discovery process begins by identifying the web services and ontology required to carry out semantic search. The choices are directed by the ontology size and service placement on the network (represented by the grey flexible services and data in Fig. 1). Thus, Step 1 involves the selection of which discovery control service (SDCS), knowledge base and matching service best fit the user requirement – specified as text strings. This information is sent to SDCS together with the search parameters (2). SDCS then calls the KB based matching service SMAS (based on OWLJessKB (http://edge.cs.drexel.edu/assemblies/software/owljesskb/)) (3) that in turn loads the KB and rules (5). The matching is carried out and returned to SDCS for use in one of the client components (4). The SDCS service can optionally provide the resource properties, the dynamic state of each service, alongside the service choices (6). Finally the returned components are displayed in a web start client (SCSV holding the component options on the server side) allowing selected components to be deployed into the CSP. The deployment is simple in nature, loading server side XML into the CSP. A more robust solution would provide transformation capabilities as has been done by Fishwick and Miller (2004).

The matching algorithm is semantic and uses an ontology and a reasoning engine. The assumption in this paper is that an ontology is a catalogue of the types of "things" derived from existing simulation models. Types in the ontology represent the predicates, word meanings, or concepts and relation types of the language when used to discuss topics in the domain (Bell and Ludwig, 2005) – in this paper these are SCs.

To summarize, the matching algorithm comprises two steps; the initialization of the knowledge base and the search. During the initialization phase the ontology is loaded transforming ontological classes into facts that have rules applied using the Rete algorithm (Forgy, 1982). During the search

inferences are made from the facts (using Java Expert System Shell (Jess) queries to identify similarities in properties and subclass relations – see (Bell and Ludwig, 2005)) identifying semantically matched SCs. For example, when searching for a component to simulate a restaurant table – several are returned that model different states.

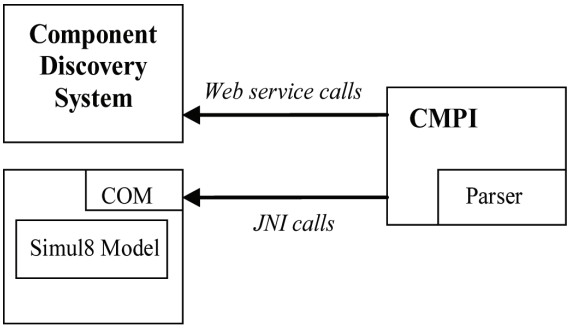
Design of CSP Model Parser and Importer

The discovery architecture detailed in the previous section is used by the CSP Model Parser and Importer (CMPI) software to conduct a semantic search for existing models. This search is conducted by calling a web service defined in the component discovery architecture, which takes a search string as parameter and returns an enumeration of unique resource names (URN) and corresponding unique resource locators (URL) for each model returned by the matching algorithm. CMPI then provides the user an option to (1) download the models into the local system for introspection or (2) import them directly into the new model being built through reuse of the discovered components.

If the user chooses option (1) the model can be downloaded into the local system by clicking on the URL, as with any file download from the Internet. The file downloaded is an XML representation of the Simul8 model that was discovered. If the user chooses option (2) the URN is passed as a parameter to yet another web service, which returns the XML representation of the model as a SOAP attachment. The nature of this web service is synchronous and this allows the CMPI to block further execution of the code until the XML file has been received.

The merging of the existing model (being built through reuse of discovered models and model components) with the new model requires a CSP specific parsing operation. Since both the models in question have an XML representation, we employ a text parsing mechanism which traverses

Figure 6. Architecture of dependencies of CMPI



through the XML hierarchy of these models and outputs a third XML file containing assimilated results from both. This new XML file is now loaded into the CSP and the user is presented with the overall model. It should be added that the text parsing mechanism is heavily dependent on the specific knowledge of Simul8. However, this is not a major problem because a model can be opened in Simul8, copied into the clipboard and pasted into another Simul8 model. This solution would alleviate the need for a model parser.

The CMPI software is written in Java and it uses the Simul8 COM interface to interact with Simul8 using Java Native Technology (Sun, 2003). CMPI invokes web service calls to communicate with the component discovery system. It also includes a CSP specific parser component which, as has been discussed in the previous paragraph, can be considered optional. The architecture and dependencies of CMPI is shown in Figure 6.

CONCLUSION

The paper presents a novel approach to CSP model reuse and interoperability. The approach adopts a simulation component ontology and semantic search architecture. The approach to modeling simulation components focuses on the specific application domains. In relating each component to a type collection and each other enables the search process to better identify likely semantic

matches. Several Simul8 models are transformed into OWL ontologies and then used by a web service based semantic search and component deployment architecture. The research has demonstrated: (1) a new, lighter approach to CSP model reuse and (2) the benefits of semantic search to this field of research. We now critically discuss the shortcomings of this research with regards to each of the aforementioned points in order to provide an overview of how we intend to address these limitations in our future work.

Although it can be argued that our 'lighter' semantics-based approach has a shorter learning curve when compared to the 'heavier' distributed simulation approach, it is still true that the simulation modeler in industry will have to be well acquainted with Software Engineering concepts such as semantics-based interoperability, software component reuse and ontologies. Furthermore, our approach is currently based on a particular simulation package (Simul8). Although it may be intuitive to imagine a scenario in which model components, developed in heterogeneous CSPs by different modelers, are discovered and then imported to a specific CSP to facilitate intra-CSP model reuse, in reality this is a distant objective. One reason for this is that the CSPs are "black boxes" and have been designed and implemented to exist in isolation.

Thus, one model component developed in a specific CSP can only be imported in other instance of the same CSP. In order to circumvent this limitation of our otherwise CSP-neutral ontology-based SEDI4G architecture, we plan to conduct further research using CSP AnyLogic and a three-phase CSP emulator. We intend to conduct a "proof-of-concept" study which would attempt to show that simulation model reuse across CSPs is achievable. The choice of CSP AnyLogic and the CSP emulator, which had been implemented for an earlier study by Mustafee and Taylor (2006), is dictated by the fact that both AnyLogic and the CSP emulator support the Java language. Thus we plan to investigate the scope for simulation component discovery (using the DESC ontology and the SEDI4G architecture) and reuse in the context of heterogeneous CSPs. We would consider CSP AnyLogic and CSP emulator as the exemplar CSP application for our study.

This research has also demonstrated the benefits of semantic search to the field of simulation. Semantic search and reusable software components are two concepts we have borrowed from Software Engineering. There is scope to learn more. The authors are particularly interested in building a framework, which would help create reusable simulation components and would ultimately enable modelers to build models using these reusable and interoperable components.

REFERENCES

Ankolekar, A., Burstein, M., Hobbs, J. R., Lassila, O., Martin, D., McDermott, D., et al. (2001). DAML-S: Semantic Markup for Web Services. *International Semantic Web Working Symposium (SWWS)*, 348-363.

Bell, D., de Cesare, S., & Lycett, M. (2005). Semantic transformation of Web services. *OnTheMove 2005 (SWWS 2005 Workshop),* 2005, 856-865.

Bell, D., & Ludwig, S. A. (2005). Grid Service Discovery in the Financial Markets Sector. *Journal of Computing and Information Technology*, *13*(4), 265–170. doi:10.2498/cit.2005.04.02

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, *284*, 34–43.

Bhatt, M., Rahayu, W., & Sterling, G. (2004). sedOnto: A Web enabled ontology for synthetic environment representation based on the SEDRIS specification. *Fall Simulation Interoperability Workshop,* Boer, C. A., Verbraeck, A., & Veeke, H. P. M. (2002). Distributed simulation of complex systems: Application in container handling. *European Simulation Interoperability Workshop*

Bortscheller, B. J., & Saulnier, E. T. (1992). Model reusability in a graphical simulation package. In *Proceedings of the 1992 Winter Simulation Conference* (pp. 764-772).

Cardoso, J., & Sheth, A. (2003). Semantic e-workflow composition. *Journal of Intelligent Information Systems*, *21*, 191–225. doi:10.1023/A:1025542915514

Chen, L., Shadbolt, N. R., Goble, C., Tao, F., Cox, S. J., Puleston, C., & Smart, P. (2003). Towards a knowledge-based approach to semantic service composition. *Second International Semantic Web Conference (ISWC2003)*.

Collins, J. B. (2004). Standardizing an ontology of physics for modeling and simulation. *Fall Simulation Interoperability Workshop.*

Concannon, K. H., Hunter, K. I., & Tremble, J. M. (2003). Dynamic scheduling II: SIMUL8-planner simulation-based planning and scheduling." In *Proceedings of the 2003 Winter Simulation Conference* (pp. 1488-1493).

Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., & Weerawarana, S. (2002). Unraveling the Web services Web - An introduction to SOAP, WSDL, and UDDI. *IEEE Internet Computing*, *6*, 86–93. doi:10.1109/4236.991449

Fishwick, P. A., & Miller, J. A. (2004). Ontologies for modeling and simulation: Issues and approaches. In *Proceedings of the 2004 Winter Simulation Conference* (pp. 259-264).

Forgy, C. L. (1982). Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problems. *Artificial Intelligence*, *19*, 17–37. doi:10.1016/0004-3702(82)90020-0

Fujimoto, R. M. (2000) *Parallel and Distributed Simulation Systems.* New York: John Wiley & Sons Inc.

Gan, B. P., Liu, L., Jain, S., Turner, S. J., Cai, W., & Hsu, W. (2000). Manufacturing supply chain management: Distributed supply chain simulation across enterprise boundaries. In *Proceedings of the 2000 Winter Simulation Conference* (pp. 1245-1251).

Gan, B. P., Yoke, M., Low, H., Wang, X., & Turner, S. J. (2005). Using manufacturing process flow for time synchronization in HLA-based simulation. *IEEE International Symposium on Distributed Simulation and Real-Time Applications* (pp. 148-157).

Geerts, G. & McCarthy, W.E. (1999). An accounting object infrastructure for knowledge-based enterprise models. *IEEE Intelligent Systems & their Applications*.

Gibbins, N., Harris, S., & Shadbolt, N. (2003). Agent-based semantic Web services. In *Proceedings of the 12th International Conference on World Wide Web* (pp. 710-717). Budapest, Hungary: ACM Press.

Gray, D. N., & Hotchkiss, J., LaForge, Shalit, S. A., & Weinberg, T. (1998). Modern languages and Microsoft's component object model. *Communications of the ACM*, *41*, 55–65. doi:10.1145/274946.274957

Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*, *5*, 199–220. doi:10.1006/knac.1993.1008

Hendler, J. (2001). Agents and the Semantic Web. *IEEE Intelligent Systems*, *16*, 30–37. doi:10.1109/5254.920597

IEEE. (2000). IEEE Standard 1516 (HLA Rules), 1516.1 (Interface Specification) and 1516.2 (Object Model Template).

Katsaliaki, K., & Brailsford, S. C. (2007). Using Simulation to Improve the Blood Supply Chain. *The Journal of the Operational Research Society*, *58*(2), 219–227.

Lara, R., Roman, D., Polleres, A., & Fensel, D. (2004). A conceptual comparison of WSMO and OWL-S. *European Conference on Web Services* (pp. 254-269).

Martin, D., Cheyer, A. J., & Moran, D. B. (1999). The Open Agent Architecture: A Framework for Building distributed Software Systems. *Applied Artificial Intelligence*, *13*, 91–128. doi:10.1080/088395199117504

McIlraith, S. A., Son, T. C., & Zeng, H. L. (2001). Semantic Web Services. IEEE *Intelligent Systems & their Applications,* 16, 46-53,

Mertins, K., Rabe, M., & Jaekel, F. (2000). Neutral template libraries for efficient distributed simulation within a manufacturing system engineering platform. In *Proceedings of the 32nd Conference on Winter Simulation,* 1549-1557.

Mustafee, N., & Taylor, S. J. E. (2006). Investigating distributed simulation with COTS simulation packages: Experiences with Simul8 and the HLA. *Operational Research Society Simulation Workshop* (pp. 33-42).

Mustafee, N., Taylor, S. J. E., Katsaliaki, K., & Brailsford, S. (2006). Distributed Simulation with COTS Simulation Packages: A Case Study in Health Care Supply Chain Simulation. In *Proceedings of the 2006 Winter Simulation Conference* (pp. 1136-1142), Monterey, CA, USA.

Mustafee, N., Taylor, S. J. E., Katsaliaki, K., & Brailsford, S. (2009). Facilitating the Analysis of a UK NBS Chain Using the HLA. *SIMULATION: Transactions of the Society of Modeling and Simulation International*, *85*(2), 113–128. doi:10.1177/0037549708100530

Paolucci, M., Kawamura, T., Payne, T. R., & Sycara, K. (2002). Semantic matching of web services capabilities. *International Semantic Web Conference* (pp. 333-347). Berlin: Springer-Verlag.

Smith, M. K., Welty, C., & McGuinness, D. L. (2004) OWL Web Ontology Language, W3C Recommendation 10 February 2004. Retrieved from http://www.w3.org/TR/owl-guide/

Sun Microsystems. (2003). Java Native Interface. Retrieved from http://java.sun.com/j2se/1.4.2/docs/guide/jni/.

Taylor, S. J. E., Bohli, L., Wang, X., Turner, S. J., & Ladbrook, J. (2005). Investigating Distributed Simulation at the Ford Motor Company. In *Proceedings of the Ninth IEEE International Symposium on Distributed Simulation and Real-Time Applications. IEEE Computer Society* (pp. 139-147).

Taylor, S. J. E., Sudra, R., Janahan, T., Tan, G., & Ladbrook, J. (2001). Towards COTS distributed simulation using GRIDS. In *Proceedings of the Winter Simulation Conference* (pp. 1372-1379).

Taylor, S. J. E., Turner, S. J., & Strassburger, S. (2008). Guidelines for commercial off-the-shelf simulation package interoperability. In *Proceedings of the Winter Simulation Conference* (pp. 193-204).

Taylor, S. J. E., Wang, X., Turner, S. J., & Low, M. Y. H. (2006). Integrating Heterogeneous Distributed COTS Discrete-Event Simulation Packages: An Emerging Standards-Based Approach. *IEEE Transactions on Systems . Man & Cybernetics: Part A*, *36*(1), 109–122. doi:10.1109/TSMCA.2005.859167

Tolk, A., & Blais, C. (2005). Taxonomies, ontologies, and battle management languages – recommendations for the coalition BML study group. *Spring Simulation Interoperability Workshop*.

Tolk, A., & Turnitsa, C. (2004). Ontology of the C2IEDM - further studies to enable semantic interoperability. *Fall Simulation Interoperability Workshop.*

Tosic, V., Esfandiari, B., Pagurek, B., & Patel, K. (2002). On requirements for ontologies in management of web services. *Web Services, E-Business, and the Semantic Web* (pp. 237-247). Berlin: Springer-Verlag.