Simulation Modelling Practice and Theory 18 (2010) 253–276

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

# A review of Web-based simulation and supporting tools

James Byrne [a,*], Cathal Heavey [a], P.J. Byrne [b]

[a] Enterprise Research Centre, University of Limerick, Limerick, Ireland
[b] Dublin City University Business School, Dublin City University, Dublin, Ireland

ARTICLE INFO

*Article history:*
Received 12 November 2008
Received in revised form 25 September 2009
Accepted 29 September 2009
Available online 6 October 2009

*Keywords:*
Web-based simulation
Discrete-event simulation
Simulation modelling
Web-based application

ABSTRACT

The area of Web-based simulation (the integration of the Web with the field of simulation) (WBS) has grown since the mid-1990s. The Web itself has evolved rapidly, and current Web-related research areas include Web 2.0, service-oriented architectures and the Semantic Web. This paper gives a review of the area of WBS, exploring the advantages and disadvantages of WBS over classical simulation systems, a classification of different sub- and related-areas of WBS, an exploration of technologies that enable WBS, and the evolution of the Web in terms of its relationship to WBS.



## 1. Introduction

The Internet and its multimedia front-end, the World-Wide-Web (WWW), has experienced tremendous growth since its development by Tim Berners-Lee of CERN (Centre for European Nuclear Research) in the early 1990s, achieving world wide acceptance [1–6]. Many disciplines are re-evaluating their strategies and techniques in view of the services offered by the Internet [4]. Simulation is no less affected by this technology than any other technique [7], as it represents a fertile area in which to perform computer simulation research [8]. Kuljis and Paul [9] go so far as to state that the pressure imposed by the proliferation of Web uses has forced the simulation community to migrate to the Web in order to stay "alive". Whether the Web has enabled, caused or forced the simulation community to move to the Web is open for debate (see [10]), but one thing is clear, it has resulted in the emergence of the area of Web-based simulation (WBS).

Web-based simulation is the integration of the Web with the field of simulation and has been growing during the past few years [11]. It is currently not an existing field, but rather an idea that represents an interest by simulation practitioners to exploit Web technology [12]. It can be defined as the use of resources and technologies offered by the World-Wide-Web (WWW) for interaction with client and server modelling and simulation tools, therefore a common characteristic of all WBS applications is that they use a browser as a support for graphical interfaces connecting the user with simulation [12]. The downloading of a simulation package from a server to the client's computer and its execution as an application wholly independent of the browser and the Web is not included in the Web-based simulation category: a browser always has to play an active role in the modelling or simulation process, either as a mere graphical interface or, additionally, as a container for the simulation numerical engine [12].

Although the notion of Web-based simulation is probably as old as the Web itself [13], early Web-based simulation efforts began in 1995, first by providing Web-front ends to simulations running as Common Gateway Interface (CGI)

* Corresponding author. Tel.: +353 857254678.
*E-mail address:* james.byrne@ul.ie (J. Byrne).

1569-190X/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.simpat.2009.09.013

scripts/programs, and in addition, work began on Java-based simulation packages, systems and environments that would run anywhere on the Web [14,15]. One of the first research papers to describe the issue of WBS as a reality that was already present in the ideas of many computer simulation researchers was a paper by Fishwick [16] at one of the work sessions at the 1996 Winter Simulation Conference in San Francisco [9,12,15–17]. In this paper, Fishwick [16] presented some issues and concepts in Web-based simulation in order to serve as a backdrop for more formal discussion and potentially form a new simulation sub-area. From this event, interest was aroused in the scientific and industrial community leading to sessions devoted solely to numerous aspects of Web-based simulation at international congresses organised by different international scientific associations (IFAC, IEEE, SCS) [12]. At this time, research in the area of Web-based simulation grew rapidly [18]. The first conference dedicated to Web-based modelling and simulation was held in 1998, and covered issues concerning Web agents, distributed simulation, simulation toolkits/products and applications [9,15,17]. This conference was held again in 1999, however in the year 2000 there was a large drop in papers being presented [9]. By this time, Kuljis and Paul [9] note that there were only a few WBS-related papers published in journals. However Miller et al. [15] and Miller et al. [19] report that WBS has exhibited explosive growth in the simulation research community. Indeed, Fishwick [20] notes that the area of WBS had good coverage at conferences and in special journal issues.

Still quite active on the research-side, the area of Web-based simulation is still in its infancy but is maturing, with efforts to expand Web-based simulation to include new capabilities beyond those found in conventional simulation technology or provide interoperation with other information processing technology being particularly promising [12,15]. However, despite the great promise held by Web-based simulation [3], the number of real applications and efficient tools for Web-based simulation is still very small [21]. For example, Wiedemann [21] states that after a review of actual Web-based simulation technologies, a lot of Web-based simulation projects were done as a test scenario, and the requirements of real customers were not taken into account.

The future holds great promise in terms of developing applications in the area of Web-based simulation, with the advent of such Web-related areas as service-oriented architectures, Web 2.0 technologies and the Semantic Web. The following sections in this paper aim to give an overview of Web-based simulation and tools in existence to support the area.

Fig. 1 gives an overview of the areas of WBS explored in this paper. This section (Section 1) presents an introduction to WBS, and attempts to: provide a definition of WBS; describe the evolution of the area of WBS; introduce the area of WBS. Section 2 gives the advantages and disadvantages of WBS over classical simulation systems. Section 3 presents a classification and description the different categories of WBS. A review of current Web technologies to enable WBS is given in Section 4, while Section 5 explores the relationship between WBS and the evolution of the Web. Finally, Section 6 presents conclusions based on findings from previous sections.

## 2. Web-Based simulation advantages and disadvantages

Web-based simulation has many benefits in comparison to classical systems, and many authors attempt to classify these, see [22–24]. A number of advantages of Web-based simulation over classical systems have been identified and can be classified as follows:

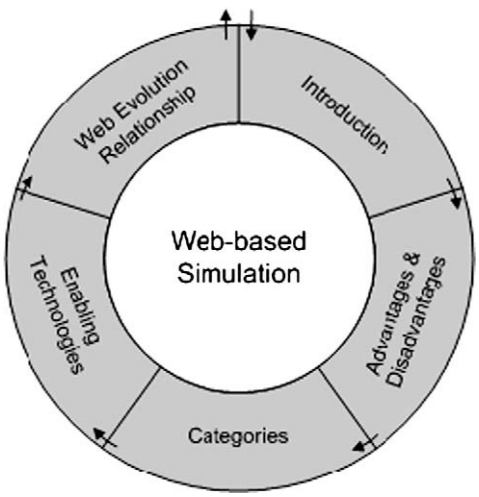


**Fig. 1.** Web-based simulation areas explored.

- *Ease of use*. The act of simulation model construction is accepted in the simulation community to be a fundamentally hard problem [25]. According to Kuljis and Paul [9] and Wang and Liao [26], simulation is and always was a highly specialist application area, with a high degree of difficulty. Simulation beginners often spend a great amount of time to accumulate the knowledge as well as the experience to overcome the technical complexity of simulation [27], and even for the experienced user, building, running, and analysing a simulation model can be a very time-consuming and error-prone process [28]. One of the main characteristics of the Web is its ease of navigation and use [9]. Obtaining data from the Web has become second nature to most users [29], and the Internet provides a familiar interface for both interacting with and controlling a simulation. Web-based simulation environments insulate the user from the intricacies involved in utilising third party models and the overheads of distributed simulation [23]. However, as Wiedemann [21] notes, it seems to be very dangerous to make an extrapolation from the ease of use of common Web-based systems to Web-based simulation systems, as the typical work and necessary features are very different.
- *Collaboration*. Communication and interaction are one of the essential factors to achieve a successful simulation project [26]. It is possible to develop environments with coherent Web-based support for collaborative model development [9], where people can communicate with each other from different places to develop the same simulation model over the Web. Such collaboration can include monitoring and debugging with the help from experts [9,26]. This approach may result in a reduction in simulation model development cost and time, over the use of simulation modelling packages such as Arena and eM-Plant which are only used at a local machine and lack group interaction capability [26].
- *Licence and deployment models*. The tool can be deployed locally, in an intranet setting or over the Internet, regardless of how it has been developed. The possibility exists to tailor the GUI allowing the use of the application on any device with a relevant Web browser installed. New licence models for simulation software have become possible, which is particularly useful for individuals and small companies where simulation capabilities are only rarely needed [30]. Traditionally, the starting investment for a typical simulation environment or external simulation consultant are on a high level [21]. The risk of losing money rises with increasing investment costs, as the revenue of a simulation study is unknown at the beginning [21]. Web-based simulation allows new business models for the use of simulation services, where these services can be rented for an interval of time (such as in an application server provider (ASP) or software as a service (SaaS) environment [31]) which can result in savings in terms of the possible previous prohibitive factors of time and cost [21].
- *Model reuse*. Since the appearance of Web-based simulation at the Winter Simulation Conference (see http://www.wintersim.org/) in 1996, the theme of model reuse and its methodological and technological support has attracted much discussion and debate in the modelling and simulation community [32]. The viewpoint of simulation modellers who use COTS simulation packages suggests that model reuse may in fact cost more than developing new models as trust must be established through testing [32]. The Web can support the reuse of existing simulation models very well by its distributed nature and the content management function like search machines and common data access protocols [21].
- *Cross-platform capability*. The Web allows for the ability to run an application on any Web browser on any operating system without compiling [18,33,34]. This capability relieves the application developer from having to worry about a client's configuration.
- *Controlled access*. Access can be controlled to a Web-based simulation application through the use of passwords, and limited time-span access can be allocated [18]. Users who need access to specific or limited areas of an application can be given access merely by being added to a password list, instead of having their client machines updated [33].
- *Wide availability*. A Web-based simulation application can be used from anywhere in the world with an Internet connection and outside of normal business hours without having to transport hardware or software [18,35].
- *Versioning, customisation and maintenance*. In using a Web-based system, maintenance is minimized [18,34]. All modifications can be made through the server [18], enabling frequent modifications, customisations and updates to be made and instantly distributed to the application, reducing error potential and eliminating virtually all on-site maintenance.
- *Integration and interoperability*. A Web-based tool can integrate and interoperate with both existing and future Web-based applications, as well as Web-enabled desktop applications [18,33,34].

A number of disadvantages of Web-based simulation over classical systems have also been identified, and these can be classified as follows:

- *Loss in speed*. The user can experience loss in speed when interacting with the tool due to downloading time and network traffic [33].
- *Graphical user interface limitation*. The interface provided by the Web as opposed to desktop simulation tools is limited [33], although this is starting to change with the evolution of multimedia authoring tools for the Web. Wiedemann [21] notes that the effort in replicating a complex interface provided by traditional desktop-based simulation tools on the Web is very high and might not be possible.
- *Security vulnerability*. Web-based applications are more vulnerable to malicious attacks than client-based applications [33]. Also if an ASP or SaaS licensing model is utilised by a company, sensitive data might be stored in a database external to the company [21].

- *Web-based simulation application stability*. The stability of the Web itself is not affected by the addition of new sites, removal of existing sites or any changes to sites, whereas the stability of a Web-based simulation application can be affected or disabled by, for example, the disappearance of any site that hosts parts of the modelling environment [9]. Any Web-based simulation application should be designed for robustness in coping with this issue.
- *Licensing restrictions*. If the simulation engine is developed using traditional proprietary simulation software, there may be license restriction issues, as traditional software licenses of simulation packages only allow a single place usage whereas a Web-based system must have the same license model as a network license [21]. This could be overcome through the use of open source simulation software for the simulation engine.
- *Difficulty in simplifying simulation*. Wiedemann [21] notes that the Web cannot simplify the simulation modelling process in the near future, concluding that simulation models should be developed by using traditional simulation tools, and the Web used to support the reuse of existing simulation models.

## 3. Categories of Web-based simulation

Several authors classify Web-based simulation (i.e. see [7,24,36–39]). Through desktop research into literature in the Web-based simulation area, the following seven main categories of Web-based simulation were identified:

- Local simulation and visualisation
- Remote simulation and visualisation
- Hybrid simulation and visualisation
- Web-based simulation documentation
- Web-based simulation model repository
- Component-based simulation in relation to Web-based simulation
- Distributed simulation in relation to Web-based simulation

The first three categories relate to descriptions of how Web-based simulation applications can be developed architecturally. Web-based simulation *documentation* relates to efforts to provide on-line documentation to existing simulations. Web-based simulation *model repository* concerns a description of Web-based repositories used for storing simulation models. Component-based simulation and distributed simulation concern areas related to Web-based simulation. The following sections provide a description of these seven categories in greater detail.

### 3.1. Local simulation and visualisation

Local simulation and visualisation (S&V) is where both the simulation engine and visualisation components are downloaded seamlessly by the client to the user's local computer, so that the graphical interface and the simulation engine coexist in the same environment (i.e. within the browser) shifting the responsibility for execution completely from the server to the client, making the server a central distribution point to the simulation but performing no real work [12,38,39]. The most common approach of the use of this technique is in the form of Java applets (see [7,37,38,40,41] for examples), although there is some study on the viability of using multimedia tools, such as Adobe Flash, for the simulation of real processes [12]. There are also researchers who, although working in local mode, separate the engine and the interface [12]. Fig. 2 gives the basic local simulation and visualisation configuration, after the initial loading phase. During this loading phase, the user opens a browser and navigates to a Web page typically containing an applet, and the browser automatically calls for the applet which is seamlessly downloaded to the client's computer. In this case, the applet contains both the simulation engine and visualisation/animation engine, which execute locally on the client-side. There may be a more advanced configuration in which, for example, a database might be present on the server-side.

Local simulation and visualisation is also known under a number of other terms. Lorenz et al. [38] use the term "local simulation and animation (S&A)", while Myers [39] uses the term "local simulation visualisation". Bencomo [12] uses the

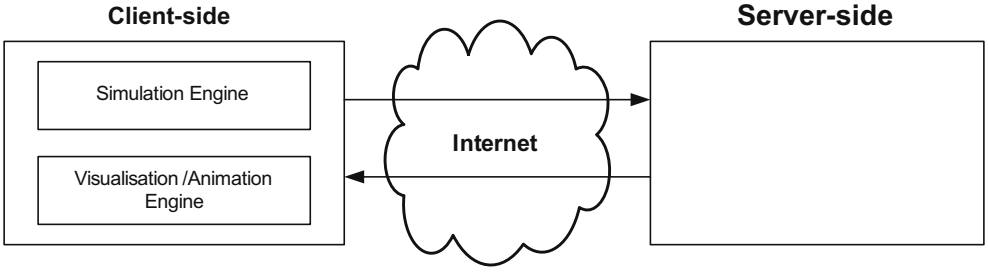


**Fig. 2.** Basic local simulation and visualisation configuration post initial loading phase.

term "local simulation" but also describes a variation of local simulation and visualisation where "monolithic approximation" is explained as being where "the graphical interface and the simulation engine are a monolithic application that is executed within the WWW browser and resides on the client's computer", and also where he explains "semi-distributed approximation" as being where "the graphical interface and the engine are independent applications and they both reside in the same computer, i.e., in the client computer. The interface is located in the browser, whilst the engine is usually a Matlab/Simulink, Labview or ACSL simulation environment". Whitman et al. [24] use the term "client-executed simulation" to describe local simulation and visualisation.

The advantage of using the approach of local simulation and visualisation when developing a Web-based simulation application is that the network latency between the user and the simulator is reduced to nothing. A disadvantage of this approach is that the power and flexibility of the tool are minimized, as it relies on the power of the client to execute the simulations efficiently, and some users may lack the hardware needed for this [24]. Also, according to Lorenz et al. [38], existing simulation and visualisation tools are not suitable for this approach.

### 3.2. Remote simulation and visualisation

In remote simulation and visualisation, both the simulation engine and any animation generation engine are located and execute remotely, on the server-side, see [12,37,42]. Access to these is through a browser on the client-side, which is typically thin. This approach can be considered a job request on a batch processing system [39]. Parameters are submitted to the simulation engine through the Web server, and results are returned to the user once the simulation has finished running [39]. As Bencomo [12] states, communication between the graphical interface in the browser and the simulation engine is carried out using, for example, Common Gateway Interfaces (CGIs), sockets, Java remote method invocation (RMI), JavaBeans, Common Object Request Broker Architecture (CORBA), remote procedure call (RPC) or via front-end applications for simulator software. These types of communication and middleware, amongst others, are discussed in Section 4.1. Fig. 3 gives the basic remote simulation and visualisation configuration, after the initial loading phase. During this loading phase, the user opens a browser and navigates to a Web page and a basic interface is displayed in the browser. Both the simulation engine and any visualisation/animation engine reside on the server-side, and are interfaced with through the browser. There may be a more advanced configuration in which, for example, a database might be present on the server-side.

Remote simulation and visualisation is also known under a number of other terms. Lorenz et al. [38] use the term "remote simulation and animation based on loading applets", Holzinger et al. [42] use the term "server-side-based Web simulation solution", whilst Myers [39] uses the term "remote simulation visualisation". Bencomo [12] uses the term "remote simulation" but also describes a variation of remote simulation and visualisation where "distributed approximation" is explained as being where "the graphical interface and engine are independent applications and they are physically separate. The interface is located in the client browser but the simulation resides in the remote server, and may consist of an application developed for this purpose, or of modelling and simulation tools. In both instances, communication with the interface can be front-ends with CGIs or ad hoc servers for their communication with the interface, sockets, etc. Within this category the possibility of many users working independently with the same server should be considered" [12]. Whitman et al. [24] use the term "server-hosted simulation" to describe remote simulation and visualisation.

An advantage of this approach is that larger simulations can run on powerful, high-end computers and the analysts can get access to the results from any low-end computer with a browser [37]. Another advantage is that in addition to providing a familiar interface [24], it works well for adapting existing simulation products such as Arena or eM-Plant to a Web-based environment [24,38,39]. Myers [39] notes an additional benefit as being that a centralized application exists using this approach, making maintenance easier for developers. A disadvantage of this approach is that it is not good for observing dynamic processes at work [38,39]. Another drawback is that it does not allow the user to interrupt a running simulation [38,39], in real-time, due to the lag experienced between the server and the client. The client can only view pre-specified outputs at a specified time, which is useful for providing "snap-shot" views of the model [24,38], although video can be generated on the server-side of the animated model which can be downloaded and viewed after the model has finished running.

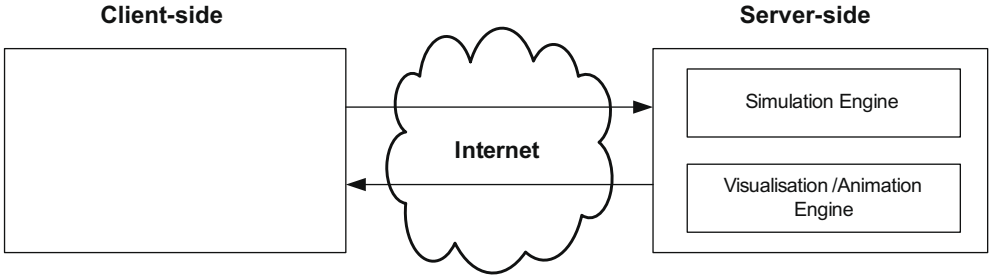


**Fig. 3.** Basic remote simulation and visualisation configuration post initial loading phase.

Narayanan [37] notes another shortcoming as being that the developer requires a knowledge of middleware technologies. Myers [39] states that long running simulations may take an extended time to generate output, especially under heavy load, resulting in user having to wait long periods of time or having the client timeout by waiting too long for a server response. The server can easily be overloaded if all of the responsibilities for the simulation are placed on the server for several concurrent user simulations [39], or if there is a heavy need for, for example, interactivity, visualisation data or numerical solvers [42]. If complex visualisations/animations are being generated on the server, these are notoriously processor intensive to generate and when combined with simulation execution cause a heavy server load even with a small number of users accessing the simulation server [39]. This affect can be negated somewhat by spreading the server load over a number of processors, as discussed in Section 3.7.

Examples of where remote simulation and visualisation is used to practically implement a Web-based simulation application include the development of a Web-based interface for storing and executing SIMAN simulation models over the Internet (see [34]), the development of eSCA which extends the capability of IBM's Supply Chain Analyzer (SCA) (see [43]), and its use to coordinate order processing in decentralized production units (see [44]).

### 3.3. Hybrid simulation and visualisation

According to Lorenz et al. [38] and Myers [39], hybrid simulation and visualisation was introduced in 1997. By combining the approaches of remote simulation and local visualisation, a hybrid combination can be created that yields the benefits of both [19,24,39]. In this approach, the simulation runs remotely on a simulation server, and when the user connects to the server through a WWW browser, a visualisation/animation engine is downloaded to the client-side [37–39], which is typically thick, see Fig. 4. A dedicated data connection may be established back to the server, in which the results of the simulation are transferred to the client-side allowing the visualisation engine to display results to the user in a dynamic nature [39]. This data can change continuously, delayed only by the executing simulation model and the latency present on the network connection [39]. This approach is typically carried out by utilising a Java server and Java applets (see Section 4.3, [38,39]), but may be carried out using other methods. Another implementation of the hybrid simulation and visualisation approach is where the visualisation/animation engine is used on the client-side to "build" the simulation model through an animated interface, and display the results, without tying the animation to the simulation as the model is executing.

Hybrid simulation and visualisation is also known under a number of other terms. Lorenz et al. [38] use the term "animation and manipulation using a Java data server", while Myers [39] uses the term "remote simulation/local visualisation". Whitman et al. [24] use the term "hybrid client/server simulation".

The hybrid approach combines the best features of local and remote simulation and visualisation, [19,24,39]. With the majority of the simulation execution being performed on the server-side, the simulation can take advantage of more powerful hardware, and maintenance is eased with a centralized application [39]. The workload on the server is reduced and more user interaction than traditional remote simulation and visualisation approaches by sending the animation/visualisation portion to the client-side [39]. One disadvantage of this approach is that the communication between the client and server is delayed by network latency, which might be several seconds on a wide area network [39]. This can be circumvented by offering no interaction while the simulation is running. Another issue is one of standards – many of the systems using this approach utilise Java applets through a browser interface, in which case the user must have the specific JRE installed in order to use the application, not a certainty on the Internet today [39].

There were not many examples identified by the author of where hybrid simulation and visualisation is used to practically implement a Web-based simulation application, indeed Whitman et al. [24] were not aware of any examples, but one identified example is an effort by Wang and Liao [26] whereby the simulation model runs on the server-side and the clients, through applets, update an XML document on the server which is used to describe the form of the model. Another is where Graupner et al. [35] take a hybrid simulation and visualisation approach in their development of a configuration system for manufacturing systems via the Internet.

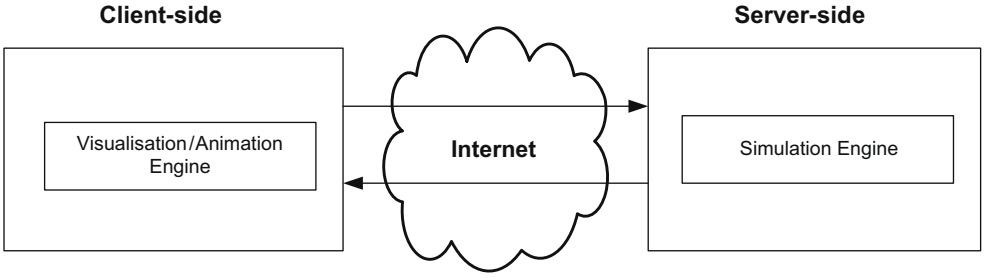


**Fig. 4.** Basic hybrid simulation and visualisation configuration post initial loading phase.

### 3.4. Documentation

The area of documentation in Web-based simulation relates to efforts that use the Web to provide on-line documentation to existing simulations [37]. The nature of the Web enables the production, storage and retrieval of "documents" containing any or all of the following elements: text, images, audio and video [7,17]. Examples of where documentation is utilised in Web-based simulation include description(s) of modelling components, results of output analysis for a set of inputs, and animations displaying simulation modelling results [37].

### 3.5. Model repository

This approach concerns the use of a server-based centralized repository that can be used to store simulation models [36]. This allows for the ability to rapidly disseminate models, results and publications [7,17]. Just as a Web document may be hyperlinked to several other Web documents, models may be linked together to form a model federation [19]. It has been proposed to use a repository of such models as interchangeable units in the development of larger simulation models [8]. This category is related to the categories of component-based simulation and distributed simulation; see Sections 3.6 and 3.7, respectively.

### 3.6. Component-based simulation in relation to Web-based simulation

Component-based simulation is a related area to Web-based simulation, and as such is described here. Although Web-based simulation is useful and viable without software component technology, utilization of components increases the potential of Web-based simulation [19], and Web-based simulation can easily subscribe to such an approach [9]. In Web-based simulation environments, the models for simulations have usually been developed using component-based modelling techniques [23,45].

Component-based software development (CBSD) (also known as component-based software engineering (CBSE) [46,47]) has become an important alternative for building software applications, and in particular for distributed systems [48]. Bertoa et al. [48] state that the component-based software development approach tries to improve the flexibility, reusability and maintainability of applications, helping develop complex and distributed applications deployed on a wide range of platforms, by plugging pre-fabricated software components, rather than building these applications from scratch. The goal is to reduce the development costs and efforts, while improving the quality of the final product due to the (re)use of software components already tested and validated [48]. It builds on the previous achievements of object-oriented software construction [47].

The idea of software components has been a major theme of software engineering for some time [45]. Syzyperski [49] defines a software component having to be a *unit of deployment* and, to enable dynamic scenarios, it has to be a *unit of versioning and replacement*. Components are independent of one another and communicate only by defined means [45]. Component-based software development systems may support some or all of the following capabilities [19]:

- *Object-oriented programming*. Under OOP, software is developed as objects consisting of attributes (data members) and methods (member functions). The advantages of encapsulation, inheritance and polymorphism have been well documented.
- *Persistence*. Mechanisms are provided to save and restore the state of executing objects with little or no programming. Traditionally, these operations required a substantial amount of custom coding.
- *Introspection*. By following certain coding conventions (design patterns) and by providing supplementary classes, different software components can be made to interact without any custom coding. For example one class may inquire about properties, methods or events of other classes. Properties are appearance or behavioural attributes that are exposed (e.g. by get/set methods) to other classes or visual tools.
- *Distribution*. Although not a requirement, it is useful to be able to have components work together even if they are not working on the same machine. This is facilitated by providing high-level mechanisms for distributed object-to-object communication typically through remote method calls or remote handling of events.
- *Platform independence*. While "Write-Once, Run Anywhere" is not a requirement, it certainly simplifies the developer's job. If this is fully supported, any object can be downloaded to any machine and executed without recoding or recompiling.

Component-based software can be used to develop highly modular simulation environments supporting high reusability of software components [19]. It permits visual programming that does not require a user to write code, therefore applying it to discrete simulation can significantly reduce model development time due to the simplicity of development [50]. In a component-based model, a system is represented as a set of interconnected components [23]. A component is a well defined entity which is viewed as a "black box", i.e., only its interface is of interest and not its implementation [23]. A component could in turn be specified using a set of sub-components [23]. During simulation, each atomic component is associated with a specific, well defined software module that implements its behaviour and functionality [23]. The software module could be those implemented by the modeller, available locally, or those obtained via the Web from other third party model developers [23].

To be regarded as a software component to be used in a discrete simulation, a program should meet the following criteria [45]:

- Its functionality should be entirely defined: that is, its stated function should be fully implemented and there should be no possibility of unintended side effects. This implies encapsulation in which all variables of the components are defined as local or private.
- All communication with any other program fragments or components should be through a fully defined interface. This implies that computation will be organised via message passing as in conventional object-orientation.
- The interface should be wholly unambiguous: that is, it should be entirely clear what inputs are required and what outputs are produced by the component and this must include error handling should incorrect inputs be provided in use. The interface definition will include the messages to which the component can respond and that it may itself produce.
- Its functionality and interface should be defined in a 'directory service' to which other components have access and from which components may be selected and used.

Component-based simulation has many advantages, such as reusability of components, support of visual development and interoperability by following a framework [50]. Such component-based development may give significant gains in productivity and quality as well as known obvious benefits to Web-based software development [9].

### 3.7. Distributed simulation in relation to Web-based simulation

By its nature, all of Web-based simulation can be described as distributed simulation [4,51], and it is predicted that in the future Web-based simulation may become synonymous with distributed simulation [51]. Indeed, Page et al. [51] classifies distributed simulation as a category of Web-based simulation. Web technologies and distributed simulation technologies have grown up largely independently, and though distributed simulation preceded the Web (it has been a research area since at least the late 1970s, see [52]), the Web is now influencing distributed simulation [13], as the Web offers services that can be exploited by the area of distributed simulation [4]. Soliman et al. [53] reported that even previous to 1995, it was an area that attracted a considerable amount of interest. At that stage, the reason for this interest arose from the fact that large simulations consume enormous amounts of time on sequential computers, and the irregular and data-dependent nature of simulation programs makes them hard to parallelize. Although simulation studies can be completed in a fraction of the time in which it required in the past, complex simulations take greater amounts of time and processing power to complete and the need for fast and reliable simulation and the area is as relevant now as it has been in the past [54].

Due to the ever-growing computational capacity of the Web [19], the main driver for the use of the Web with distributed simulation is the desire to manage simulation(s) over a network of computers through the Web, whose processors can be utilised to complete the simulation(s) in a timelier fashion [54,55]. Other related drivers are the aspiration to organise independent simulation models into a larger, single simulation, the ability to reuse simulation models and simulation components, and the ability to hide modelling details of a simulation [4,56]. Indeed, according to Pidd et al. [45], various forms of distributed simulation are possible over the Web, including simple multiple replications of the same model, client–server architectures for one or more simultaneously running models and the distributed operation of one or more linked models. Narayanan [37] also backs this up by stating that one category of Web-based simulation represents efforts on distributed, interactive simulations on the Web, where potentially multiple users can interconnect with the same underlying simulation model through Web browsers from different locations.

In a distributed simulation system, the model designer should not have to worry about the details of the technology used by the compiler to produce distributed simulations [4,10]. An important and heavily researched standard to support distributed simulations is the High Level Architecture (HLA). In the past two decades, the US Department of Defence (DoD) made significant investment in distributed simulation, fostering an evolution of standards to support the interoperability of simulations, culminating in the development of the HLA in the mid-1990s [7,17]. The HLA is a generalization and extension of the Distributed Interactive Simulation (DIS) protocols and the Aggregate Level Simulation Protocol (ALSP), [7,57]. The HLA grew out of cost and quality concerns that illustrated a need within the DoD to reuse existing modelling and simulation applications [17]. Defined under IEEE standard 1516 [58], the HLA is essentially a set of rules that simulation developers use that allow for interoperability and reuse of simulations, in which simulations can communicate with other simulations regardless of the platforms they are run on [59].

At the heart of the HLA is the notion of a federation, which is a collection of federates that interoperate using the protocols described by the architecture [17]. A federate is a HLA compliant simulation. Interactions between federates are controlled by a Run-Time Infrastructure (RTI) [59]. If multiple simulations are connected via the RTI, they are referred to as a federation. A collection of related data sent between simulations is referred to as an object. These objects have attributes (data fields). Events sent between simulations are referred to as interactions, and these interactions have parameters (data fields). An object model template (OMT) specifies what information is communicated between simulations and how it is documented.

According to the Defence Modelling and Simulation Office (DMSO) Web site (see http://www.dmso.mil/public/transition/hla/), HLA is now being revised to include a number of improvements, including the following:

- Extended XML support for Federate Object Model (FOM/SOM), such as schemas and extensibility.

- Fault tolerance support services.
- Web Services (WSDL) support/API.
- Encoding helpers.

Simulation in terms of grid computing (or grids) has emerged as an area of research within the domain of distributed simulation, and indeed it has been stated that Grid computing has dominated distributed computing research for more than a decade [60]. It has been facilitated by an increased need for collaborative research, together with continuing advances in communication technology and computer hardware [61]. Grid computing can be described as a significant advancement in the field of distributed computing, in which distributed systems have been developed that can provide users access to geographically dispersed computing resources that are administered in multiple computer domains [61,62]. The word "Grid" is used as an analogy to the electric power grid, which provides pervasive access to electricity and many believe that by allowing all components of our information technology infrastructure – computational capabilities, databases, sensors, and people – to be shared flexibly as true collaborative tools, the Grid will have a similar transforming effect, allowing new classes of application to emerge [63]. Architecturally, Grids have been closely aligned with Web Services and SOA [60].

The software component that makes grid computing possible is commonly referred to as grid middleware [61]. Grid middleware usually provides mechanisms for security, job submission, job monitoring, resource management and file transfers, amongst others, as well as providing a distributed computing infrastructure to cater to the computing needs of the grid user available [61]. Examples of grid middleware include Globus, European data Grid and Condor, see [60,61].

In relation to simulation, grid computing may provide an opportunity to further improve the use of simulation in industry, [61]. It has a great potential to help solve problems faster by using a much larger resource pool, running iterations that can be distributed to networked computing resources to achieve significant speedup, [62,64]. Simulation is characterized by the need to run multiple sets of computationally intensive experiments, and large-scale scientific simulations have traditionally been the primary benefactor of grid computing [61]. The application of this technology to simulation in industry has, however, been negligible [61]. For examples of grid computing applied to simulation, see [61,62,64].

Virtualization is an area closely related to distributed/parallel computing and grid computing, and concerns linking a computer cluster together so that they form a single computer, the components of which are commonly connected through a LAN. Perumalla et al. [65] give an example of executing parallel/distributed discrete event simulation (PDES) programs using virtualization, in which a "virtual" supercomputer is utilised as opposed to a supercomputer where CPU time is limited and expensive.

Areas related or heavily linked with distributed simulation include component-based simulation (see Section 3.6) and middleware (see Section 4.1).

## 4. Enabling technologies for Web-based simulation

Section 3 gave the different architectural configuration options in developing a Web-based simulation application as local, remote and hybrid S&V, depending on whether the simulation engine and application visualisation takes place on the server-side or client-side in the application. Fig. 5 gives an overview of the technical modular options available in developing and deploying a Web-based simulation application, in the context of its possible architecture.

There are some examples in literature where authors have given architectural options for Web-based simulation applications. For example, in presenting his framework, Gyimesi [66] gives a similar diagram for his modular concept of a tool he developed called SimASP, displaying only one "Web" server rather than multiple "application" servers, and without middleware on the server-side or plug-ins on the client-side. Wiedemann [21] gives an architecture of an application service providing (ASP) system, with browser, Web server, a Winsock-component integrating with a simulation database, and another database.

Henriksen et al. [67] present a layered architecture for their *Web-Based Simulation Centre* (WBSC) in which the central layer consists of a Web server and database management system (DBMS), and working out the next layer includes all functions and data structures that fulfil integration with the simulation engine, the third layer consists of the simulation, data input, graphical output, components, groupware, administration and animation, and the fourth layer is the user interface. Liston et al. [68] and Byrne et al. [69] present an n-tier architecture with an RIA plug-in on the client-side, and the simulation engine sitting in the processing tier. Clegg et al. [70] gave a basic client–server architecture with database for their design of a simulator.

Referring to Fig. 5, the sub-sections in this section give a breakdown of each of the technical modular options as given. Middleware is important as it enables different modules in a Web-based simulation application to interoperate, which may be between modules on the same server or between different servers, enabling distributed simulation. The section on middleware is presented in Section 4.1. The simulation engine is an integral part of the application, and this might take the form of a commercial-off-the-shelf tool or a language or library, with the possibility of it being split over more than one server or being run on the client-side. Section 4.2 gives an overview of simulation tools, languages and libraries for this purpose.

There is a wide range of languages and tools that can be used in developing and deploying Web applications, on both the server and client-sides. Section 4.3 gives an overview of the options for this module in a Web-based simulation application.

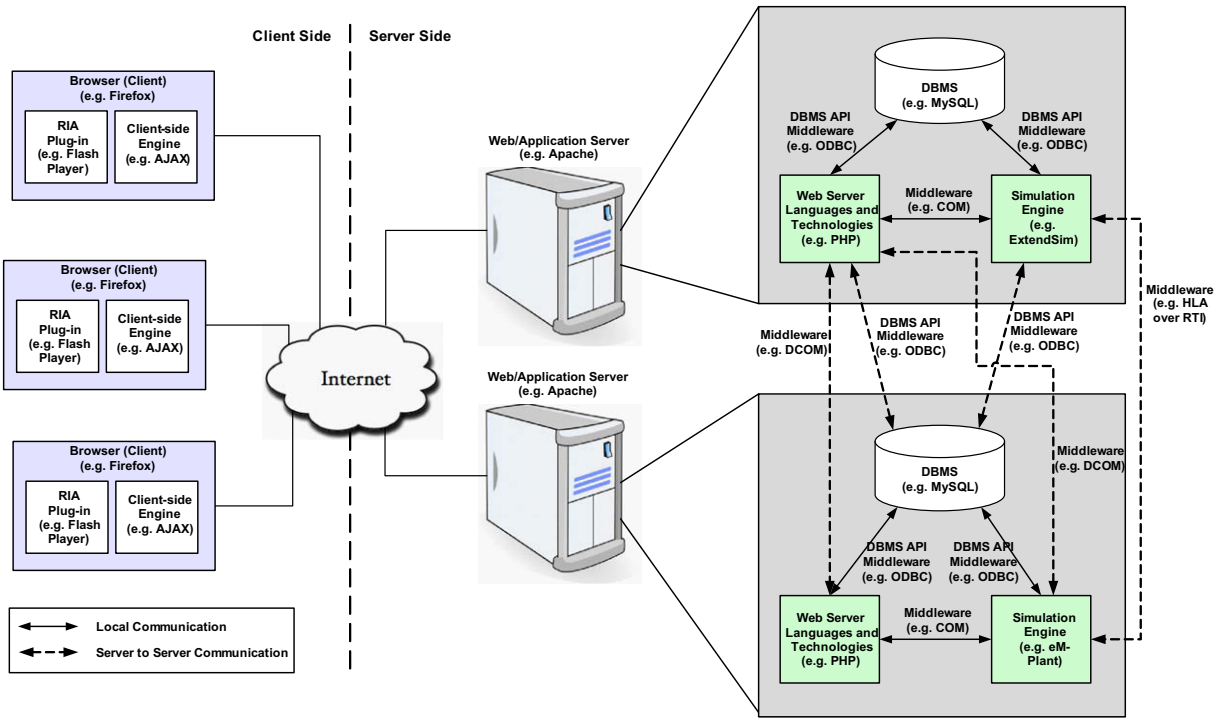


**Fig. 5.** Web-based simulation modular architectural options.

Finally, Section 4.4 gives a summary of a number of relational database management systems and possible APIs for interoperating with them.

### 4.1. Middleware

Referring to Fig. 5, middleware is generally a key component in the architecture of a Web-based simulation application. Distributed computing has come into the mainstream due to the increasing popularity of PCs, the rapid growth of the Web and the advances of high-speed network access, [71,72]. As information technologies continue to evolve, computing environments are becoming more distributed and heterogeneous, [73–75]. Interoperability of heterogeneous applications is defined as the ability for multiple software applications written in different programming languages running on different platforms with different operating systems to communicate and interact with one another over different computer networks [73]. Software developers are faced with the task of writing more code for dealing with the issues of heterogeneity and interoperability, so middleware technologies are emerging which offer a standard infrastructure for dealing with these issues.

Middleware technologies help to improve the flexibility, extensibility, maintainability and reusability of distributed applications [76,77]. Component-based software engineering is revolutionizing the field of distributed object systems, and these technologies utilise a component-based approach, allowing the users to create distributed components, [76], see Section 3.6. Components are implementation/integration units with precisely-defined interfaces that can be installed in application server run-time environments [77]. They can reside on different servers and make their behaviour available as a service [78]. They can be reused and combined with other components, and are a part of the object-oriented paradigm. Example of a component are a single button in a GUI, or an interface to a database manager [71]. A component runs within a context called a container. Examples of containers include pages on a Web site, Web browsers, and word processors [71].

There are several types of middleware, and Hurwitz [79] offers a classification system for these:

- *Remote procedure call*. Client makes calls to procedures running on remote systems. Can be asynchronous or synchronous.
- *Message oriented middleware*. Messages sent to the client are collected and stored until they are acted upon, while the client continues with other processing.
- *Object request broker*. This type of middleware makes it possible for applications to send objects and request services in an object-oriented system.
- *SQL-oriented data access*. Middleware between applications and database servers.

In addition to these, when referring to simulation technology, middleware is generally used in the context of the high level architecture (HLA) that applies to many distributed simulations, see Section 4.1. In order to identify what technologies are classified as middleware, several papers were studied in the literature, all of which offer examples of middleware technologies, see Table 1. From this table, it can be seen that the following technologies are classified as middleware technologies: Object Management Group's (OMG), Common Object Request Broker Architecture (CORBA), Microsoft's Distributed Component Object Model (COM/DCOM), Sun's Enterprise Java Beans (EJB), Sun's Remote Method Invocation (RMI), Microsoft's .NET, Sun's Java 2 Enterprise Edition (J2EE), Remote Procedure Call (RPC), Service-Oriented Architecture Protocol (SOAP), Sockets, Java Message Service (JMS), Microsoft .NET Remoting and WWW Consortium's (W3C) Web Services.

In describing how these technologies have evolved (see Fig. 6), COM+ and EJB are both transactional component middleware (TCM), [94]. COM+ evolved from Microsoft Transaction Server (MTS) and includes both COM and DCOM. Microsoft COM+ provides a foundation to build component-based enterprise applications by using Microsoft Component Object Model (COM) objects. Microsoft .NET Enterprise Services is a feature that is included in the Microsoft .NET Framework in which .NET objects can use COM+ features plus some additional features for .NET components, to create Enterprise Service Components. .NET Remoting is an API for interprocess communication. Windows Communication Foundation (WCF) is a programming framework released in December 2006 to build applications that inter-communicate (see Fig. 7).

Object Management Group (OMG) was created in 1989 to develop, adopt, and promote standards for the development and deployment of applications in distributed heterogeneous environments. In response to this they developed the Common Object Request Broker Architecture (CORBA) [78]. CORBA is currently one of the most popular commercial middleware standards (note that it is not a product) for distributed object computing, [95,96].

Enterprise Java Beans is a managed, server-side component architecture for modular construction of enterprise applications. The EJB specification is one of the several Java APIs in the Java Platform, Enterprise Edition (Java EE). EJB includes Java RMI and the CORBA interface IIOP, which makes it possible to call an EJB server from a CORBA client.

A Web Service is defined by the W3C (see http://www.w3.org) as "a software system designed to support interoperable Machine to Machine interaction over a network". SOAP (originally Simple Object Access Protocol) is a protocol for exchanging XML based messages over computer networks. WSDL (Web Services Description Language) is an XML based language providing a model for describing Web Services. UDDI (Universal Description, Discovery and Integration) is a Web Service XML standard designed to be interrogated by SOAP messages and provide access to WSDL documents. Both .NET and Java EE provide platforms for the development of Web Services.

There are a number of examples of middleware used with (and for) Web-based simulation in the literature reviewed. For example, Cho and Kim [97] apply some of the component-based middleware frameworks to discrete event systems simulation to develop a component-based simulation environment. They base their environment on COM, and compare it to using both JavaBeans and CORBA frameworks. In enabling large-scale deployment of Web-based network simulation frameworks, Cholkar and Koopman [98] use a set of standard CORBA-IDL-based programming interfaces. Meyer Zu Eissen and Stein [30] give the different realization alternatives for Web-based simulation services using Web Services.

The next section describes simulation tools, languages and libraries, and gives a list of the common interfaces used to interact with such simulation tools in a Web-based environment.

**Table 1**
Examples of classifications of middleware technologies.

| Source | Technologies classified as middleware |
|---|---|
| [78] | CORBA, RMI, Web Services |
| [80] | CORBA, .NET, J2EE |
| [81] | J2EE, Web Services, CORBA |
| [82] | CORBA, J2EE |
| [83] | CORBA, RMI, SOAP |
| [84] | Web Services, .NET, J2EE and CORBA |
| [85] | CORBA, COM, RMI |
| [86] | CORBA, COM, RMI, EJB |
| [77] | CORBA Component Model (CCM), J2EE, .NET |
| [74] | CORBA, COM, RMI |
| [73] | CORBA, COM/DCOM, EJB |
| [87] | CORBA, Java RMI, DCOM |
| [88] | RPC, DCOM, CORBA, EJB, SOAP |
| [89] | COM/DCOM, SOAP |
| [72] | DCOM, CORBA |
| [90] | CORBA, DCOM, EJB |
| [91] | CORBA, DCOM, RMI |
| [92] | Java RMI, CORBA ORB, DCOM |
| [93] | Sockets, RPC, RMI, JMS, .NET Remoting, Web Services |

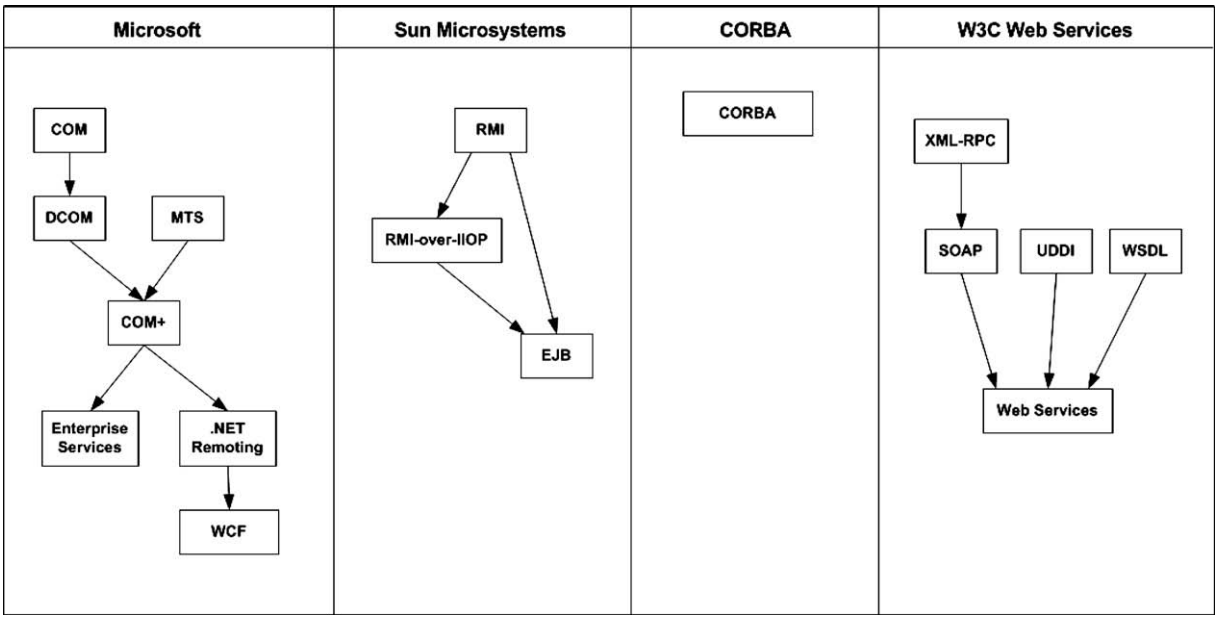


**Fig. 6.** Evolution of middleware technologies (standards) for application integration.

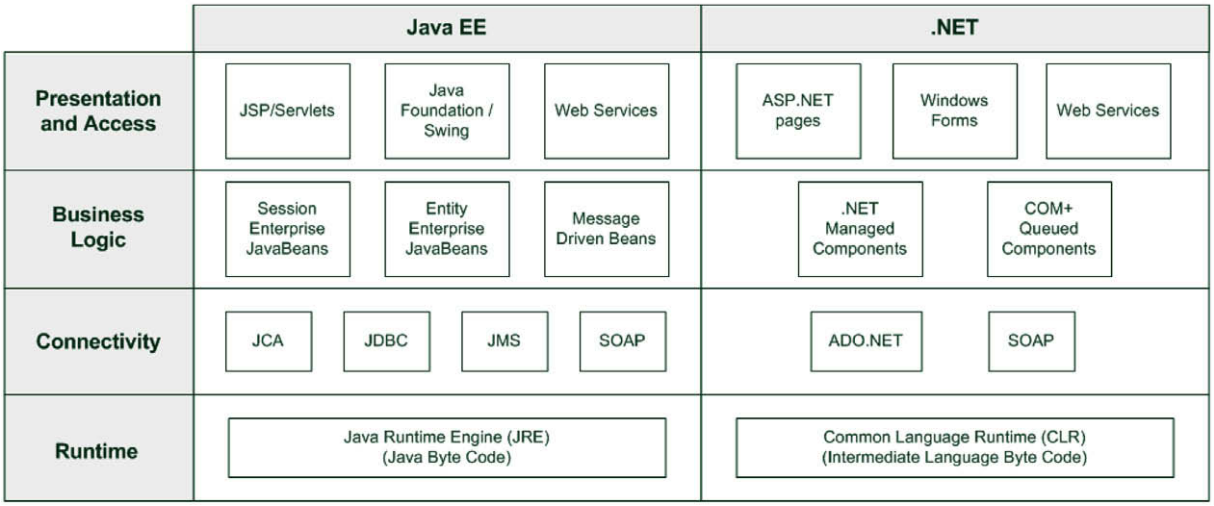


**Fig. 7.** Java EE and .NET Architectures, adapted from [112].

### *4.2. Simulation tools, languages and libraries for WBS*

Referring to Fig. 5, there are a number of different types of simulation tools, languages and libraries that can be exploited in a Web-based environment. Table 2 gives a sample list of common commercial-off-the-shelf (COTS) visual interactive modelling systems (VIMS) tools for discrete event simulation, (DES) while Table 3 gives a sample list of common simulation libraries and languages.

COTS VIMS tools for DES are shrink-wrapped software systems in which models are built by point and click using predefined objects for which the user must supply properties [99]. According to Pidd [99] they requiring little in the way of programming, although they commonly give the user a simulation language to program with, either specific to the tool (e.g. 'Plant Simulation' with *SimTalk*, 'Simul8' with *Visual Logic*) or a common language such as Java (e.g. 'Anylogic').

A typical method for utilising COTS VIMS tools for DES is to deploy them in a modular fashion into a remote S&V environment, see Section 3.2, as this method works well for adapting such tools to a Web-based environment [24,38,39]. In this case the tool is installed and deployed on an application server, and a separate visualisation/animation interface is developed

**Table 2**
Sample list of COTS VIMS tools for DES.

| Name and vendor | Interoperability/external connectivity | Vendor website |
|---|---|---|
| "eM-Plant/Plant Simulation" by Siemens/UGS/Tecnomatix | COM, DCOM, HTML, ODBC, ActiveX, Sockets | http://www.plm.automation.siemens.com/en_us/products/tecnomatix/plant_design/plant_simulation.shtml |
| "Anylogic" by XJ Technologies | As with Java | http://www.xjtek.com/ |
| "ARENA" by Rockwell Automation | ActiveX Object Model, ADO/ODBC, VBA, Import links for dxf and Visio | http://www.arenasimulation.com/ |
| "AutoMod" by Applied Materials Inc./Brooks Software | ActiveX, C code source files, links to dll | http://www.brookssoftware.com/pages/67_simulation_and_modeling.cfm |
| "Enterprise Dynamics"/"Taylor ED" by Incontrol | ActiveX, OLE for Process Control (OC), ODBC | http://enterprisedynamics.com |
| "Extendsim"/"Extend" by Imagine That! Inc. | COM/ActiveX, ODBC/SQL | http://www.extendsim.com |
| "Flexsim" by Flexsim Software Products Inc. | DDE, ODBC, Windows Sockets, As with C++ | http://www.flexsim.com |
| "Goldsim" by Goldsim Technology Group | ODBC | http://www.goldsim.com |
| "Micro Saint Sharp" by Micro Analysis and Design | COM | http://www.maad.com/ |
| "ProModel" by ProModel Corp. | ActiveX | http://www.promodel.com |
| "Simul8" by Simul9 Corp. | ActiveX/COM | http://www.simul8.com |
| "Witness" by Lanner Group Ltd. | ActiveX/COM | http://www.lanner.com |
| "Simio" by Simio LLC | As with .NET (public object model being developed to integrate with) | http://www.simio.biz/ |

**Table 3**
Sample list of simulation languages and libraries.

| Name and vendor | Language based upon (License) | Vendor website |
|---|---|---|
| "JavaSim" (now "J-Sim") by Department of Computer Science, University of Newcastle Upon Tyne, UK | Java simulation library (Academic, Open Source) | http://javasim.ncl.ac.uk/ |
| "JSIM" by Department of Computer Science, University of Georgia, USA | Java simulation library (Academic) | http://www.cs.uga.edu/~jam/jsim/ |
| Sage/HighMast by Highpoint Software Systems, LLC | .NET (COTS) | http://www.highpointsoftware.com/ |
| SIMSCRIPT by CACI International Inc. | Itself, can integrate with C/C++/Java/Fortran (COTS) | http://www.simscript.com/ |
| SLX/GPSS/H by Wolverine Software | Itself, modelled after C (COTS) | http://www.wolverinesoftware.com/ |
| "SSJ" by Département d'Informatique et de Recherche Opérationnelle (DIRO), Université de Montréal, Canada | Java simulation library (Academic) | http://www.iro.umontreal.ca/~simardr/ssj/indexe.html |
| "DSOL" by Simulation Group in Systems Engineering, Delft University of Technology, The Netherlands | Java simulation library (Open Source) | http://sk-3.tbm.tudelft.nl/simulation/index.php |
| "SimPy" by Simpy Developer Team | Python simulation library (Open Source) | http://simpy.sourceforge.net/ |
| Delsi 2.0 | .NET simulation library (COTS) | http://www.holushko.com/ |

to integrate with it. An advantage of this approach is that a pre-developed generic simulation model (or library of models) can be developed whose structure and/or parameters can be controlled through the use of a database. Due to the interactive nature of these tools, this generic simulation model can be developed more quickly than is typical with a simulation language or library. Disadvantages of this method include that it is not good for observing or interrupting running simulations [38,39], due to lag experienced between the server and the client, and interfaces are typically developed to allow for simulation input parameters to be sent to the simulation engine and to allow for simulation results to be viewed.

In order for COTS VIMS tools for DES to be deployed into a Web-based environment, they need a method by which they can integrate with the rest of the Web application, including the rest of the logic of the application and typically the database. Middleware interfaces are a common feature of these tools, and all tools listed in Table 2 provide at least one middleware interface for integrating with them, and/or to give them the ability to output to a database. For example, *Extendsim* provides ActiveX and COM components to allow for external application integration, and provides ODBC in order to output to a database. For more on middleware, see Section 4.1 and for more on integration with databases, see Section 4.4. An example of a Web-based simulation environment in which middleware is used with a COTS VIMS tool is given by Bodner et al. [100] where they integrate a Java applet with ARENA using CORBA. They propose that DCOM, Java sockets or Java remote method invocation (RMI) could also be used in this situation.

When COTS VIMS tools for DES are deployed to a Web environment, a number of different users can potentially use them at the same time, creating a number of instances of the one underlying generic model. This can cause the server to be overloaded with even a small number of users [39], as these tools are commonly processor intensive applications. This can be alleviated by spreading the server load over a number of processors, or possibly application servers.

Referring to Table 2, at least one tool can be used in a different way, namely *Anylogic*, as it gives features allowing it to be deployed using a local S&V architecture (see Section 3.1). The simulation can be developed on a client using the tool, and then automatically converted to a Java applet (see Section 4.2). This applet can then be deployed to the Web, to run on the client-side in a Web browser. This approach relies on the client to execute the simulations efficiently, and network latency between the simulation and the visualisation is non-existent due to both running on the client-side. Some users might lack the hardware necessary to run the simulation on the client-side, depending on the simulation. Bruzzone [101] state that while applets are usually not able to access local resources such as .xls files or any other database file, Anylogic applets can be digitally signed for allowing access to other resources. An applet developed by Bruzzone [101] is signed digitally to link with a Microsoft Excel spreadsheet.

Another variation on Web-based simulation by COTS companies is in a product called Witness Server by Lanner Group, see [102]. This is an additional module for Witness simulation customers allowing a Witness simulation model to be placed on the Web to be accessed by a customised browser interface by users.

Table 3 gives a list of simulation languages and libraries which can be exploited in a Web-based simulation environment. Languages and libraries are more flexible in their use than COTS VIMS tools for DES, but take more time to develop models in. Tools in this space can be deployed in local, remote or hybrid S&V architectures, see Section 3. JavaSim, JSIM, DSOL and SSJ are all simulation libraries written in Java to be used by the Java language. The basic entities in JSIM are components allowing integration with scripting languages. Delsi 2.0 is a COTS .NET simulation library as is Highmast, and SimPy is an open source simulation library developed for use with Python. SIMSCRIPT and SLX are both languages with integration features for other languages. There are examples in the literature reviewed of simulation languages and libraries being utilised for Web-based simulation. For example, Miller et al. [19] show that the use of component technology in JSIM (which was designed for Web-based simulation) allows simulation models to be treated as components that can be dynamically assembled to build model federations (see Section 4.1 for *federations*). According to Chin et al. [82] and Jacobs et al. [103], DSOL has successfully been deployed in a number of Web-based simulation projects. Merkuryeva et al. [104] state that SLX supports Web simulation. Lorenz et al. [38] give an example of the embedding of simulation software into the Web using GPSS/H as an example.

In conclusion, there is a large number of simulation tools, languages and libraries, some of which have been referred to in this section (all of which can be used to provide a modular simulation engine in a Web-based environment). The next section gives an overview of some of the most popular/relevant languages and technologies that can be used on the Web in developing and deploying a Web-based simulation solution.

### 4.3. Web-related languages and technologies

This section presents languages and technologies for creating Web-based tools that have both been used in the past and that can be potentially used to create Web-based simulation applications. This is a vast area of research; therefore this section attempts to give a high-level overview of the most common of these technologies.

Referring to Fig. 5, in developing Web-based applications the technologies used can be categorised into *server-side* and *client-side*, depending on whether they execute on the server or the client. Server-side scripting is a Web server technology in which a user's request is fulfilled by running a script directly on the Web server to generate dynamic hypertext mark-up language (HTML) pages. There are a number of different server-side languages and Web application frameworks, some of the most popular of which are as follows:

- *PHP*. PHP originally stood for "Personal Home Page", and is a server-side general purpose scripting language that can be embedded into HTML. It is very popular for Web development, and is commonly used in a Linux environment with the Apache server and MySql. An alternative to this configuration (known as LAMP) is to use Perl or Python instead of PHP. Henriksen et al. [67] and Meyer zu Eissen and Stein [30] give PHP as a possibility for inclusion in the development of a Web-based simulation application.
- *Perl*. Perl is a high level general purpose programming language, has been used since the early days of the Internet to write Common Gateway Interface (CGI) scripts and is a popular server-side scripting language for the Web. Lorenz et al. [38] give an example of implementing a Web-based simulation application using Perl to write CGI scripts. Meyer zu Eissen and Stein [30] give Perl as a possibility for inclusion in the development of a Web-based simulation application.
- *Python*. Python is a high level general purpose programming language that can be used for server-side scripting. McGrath et al. [105] use Python in the development of their distributed simulation for support of emergency response services.
- *Microsoft Active Server Pages (ASP)*. ASP is a server-side scripting language developed by Microsoft, and can be used, like PHP, Python and Perl, to dynamically generate Web pages. Most ASP pages are written in VBScript. Henriksen et al. [67] gives ASP as a possibility for inclusion in the development of a Web-based simulation application.
- *JavaServer Pages (JSP)*. JSP is a Java technology in which, upon receiving a request from a Web client, can generate HTML, XML or other types of documents dynamically. They can generate Java Servlets that is then compiled by a Java compiler. They are a part of Java Enterprise Edition (Java EE), as discussed further in this section. Henriksen et al. [67] give Java Servlets and Meyer zu Eissen and Stein [30] give JSPs as possibilities for inclusion in the development of a Web-based simulation application.

- *Microsoft ASP.NET*. ASP.NET is a Microsoft Web application framework and, as it's built on the Microsoft Common Language Runtime (CLR) ASP.NET code can be supported by other languages such as C# and VB.NET. ASP.NET is discussed further in this section below. As an example of its use, Chu et al. [106] use ASP.NET with C# in developing their universal Web-based simulation system for greenhouse crops.
- *Adobe Coldfusion*. Coldfusion is a Web application framework similar to ASP.NET, JSP and PHP providing server-side scripting capability, and is based on JavaEE. It is developed by Adobe. Bodner et al. [100] give Coldfusion as a possibility for the development of their Web-based simulation tool for material handling.
- *Smalltalk*. Smalltalk is an object-oriented, reflective programming language that can be used for server-side scripting. A popular implementation of a Smalltalk Web framework is the Seaside Web framework.

Client-side technologies have become more complex in recent years with the advent of Rich Internet Applications (RIAs). RIAs are Web applications that run on the client-side and have features and functionality akin to traditional desktop-based applications. Indeed Held and Blochinger [107] note that they strongly resemble desktop applications in quality and ease of use, but the difference is that they can be accessed anytime and anywhere. Traditional Web applications relied on a "thin client" whereby all interaction with the Web application required data to be sent to the server, the server had to respond and the page had to be regenerated from the server-side and sent to the client. RIAs circumvent this for many transactions, as processing can be performed on the client-side rather than the server-side. There are a number of important client-side technologies, some of the most popular of which include the following:

- *JavaScript*. JavaScript is an important scripting language most commonly used client-side scripting, although it can be used on the server-side [108]. It was influenced by many languages and was designed to have a similar look to Java, but be easier for non-programmers to work with. The primary use of JavaScript is to write functions that are embedded in or included from HTML pages and interact with the Document Object Model (DOM) of the page. Pillutla [109] gives an example of a Web-based simulation gaming exercise which uses JavaScript.
- *Asynchronous JavaScript and XML (AJAX)*. AJAX is a group of techniques for Web development used for creating RIAs. Using AJAX, a combination of JavaScript and Document Object Model (DOM) manipulation takes place along with asynchronous server communication achieving a high level of user activity [110].
- *Java applets*. Java applets are programs written in Java that execute on the client-side through the use of a plug-in in a browser, and run by executing the browser's Java Virtual Machine (JVM). Bodner et al. [100] gives an example of a Web-based simulation application developed using a Java applet.
- *Microsoft Silverlight*. Silverlight was first released in 2007 and is a Web browser plug-in used for creating RIAs. It can interact with .NET server-side technologies.
- *Adobe Flash*. Flash is a technology that can be used for creating RIAs, and is in existence since 1996. It runs in a Web browser through a plug-in, and because it has existed so long the penetration of this plug-in on the client-side is high in Web browsers. It supports the streaming of audio and video, and can manipulate vector and raster graphics, making it a very good solution for animation. The scripting language Actionscript is included with Flash.
- *Adobe Flex*. Flex is a collection of technologies for the development of RIAs based on Adobe Flash.
- *Adobe Shockwave*. Shockwave is a client-side technology for multimedia files, and can be viewed in a Web browser by anyone who has the Shockwave plug-in installed.
- *JavaFX*. JavaFX is a family of products for creating RIAs, and is in existence since 2007. It runs using a browser plug-in on the client-side, and it is anticipated that it will compete with Microsoft Silverlight and Adobe Flash.
- *Google Gears*. Gears is developed by Google (http://www.google.com) as free and open source software. It is a plug-in designed to run on the client-side to extend the Web browser to include support for RIAs.
- *OpenLaszlo*. OpenLaszlo is an open source platform created by Laszlo Systems (http://www.laszlosystems.com) for creating RIAs.

As can be seen, there are a number of different server and client-side technologies in existence that can be used in the creation of an RIA Web-based simulation application on the client-side.

Another key technology in relation to Web-based simulation applications is extensible mark-up language (XML). XML is a free open standard and is used for creating custom mark-up languages. It is the dominant standard for representing data on the Internet. XML tags describe data, and it is possible for programs to interpret this data, in multiple ways, such as filtering the document based upon its content or restructuring it to suit the application's needs [111].

In addition to (and including some of) the technologies mentioned above, two platforms/development frameworks have emerged as being most important for enterprise-level application development in recent years, namely Microsoft .NET and Sun Java Platform, Enterprise Edition (Java EE) (previously known as Java 2 Enterprise Edition (J2EE)) [112–114].

Microsoft .NET Framework is an API for programming on Windows platforms [115]. It includes the .NET Framework, which is a development and implementation platform. The .NET Framework was developed with the Internet in mind from the very beginning and fits into the overall .NET goal which is of allowing software as a service [116].

According to Robinson et al. [115], central to the .NET Framework is its runtime execution environment, known as the Common Language Runtime (CLR) or the .NET runtime. Before code can be executed by the CLR, any code needs to be compiled first, and this compilation occurs in two steps in .NET:

- Compilation of source code to the Microsoft Intermediate Language (IL), which is known as managed code.
- Compilation of IL to platform-specific code by the CLR.

By using a two-stage compilation process, .NET provides platform independence and language interoperability. By compiling to IL, platform independence is obtained for .NET, although at present a complete implementation of .NET is only available for Windows. The use of the IL enables the code to be compiled to IL from one language, and this code should then be interoperable with code that has been compiled from another language. Languages that at present are interoperable with .NET include: Visual Basic .NET, C# and Visual J# .NET, as well as the scripting language Jscript.NET. There is also a version for Python as well as several others. COM and COM+ are also interoperable with .NET interfaces. Visual Studio .NET is the most popular IDE for .NET.

Java EE (or J2EE) is a set of specifications (and a platform framework) created by the Java Community Process (JCP), for developing enterprise-level applications [112,114]. It defines an architecture for developing complex, distributed enterprise Java applications [117]. J2EE was modelled after the Java 2 Platform, Standard Edition (J2SE), and includes extended APIs to support enterprise systems [118]. The task of developing enterprise-level multi-tier applications is simplified through the provision of containers, which provide certain complex functionality so that software developers can concentrate on writing the business logic [112]. As Java EE is a set of specifications, they are implemented by a number of vendors, examples of which include IBM Websphere, BEA WebLogic, JBoss, and many more [119].

Java EE source code is compiled to Java byte code and runs on a Java Virtual Machine (JVM) that converts Java byte code to machine code, therefore multiple operating systems are supported through installation of the appropriate JVM, giving platform independence [112,118].

According to Kachru and Gehringer [112], both technologies are similar, with arguments in favour of Java EE including platform independence, multiple vendor support and the large number of tools and resources from which to choose, while arguments in favour of .NET include support for multiple languages and integrated (rather than add-on) support for Web Services. The disadvantage of single-vendor support in .NET must be weighed against Java EE's single-language support.

In summary, through reviewing the literature it was apparent that there is a large number of tools and technologies available for the development and deployment of Web-based simulation applications, on both the server-side and client-side, and a sample of some of the most popular of these have been given in this section. The next section presents database management systems, in relation to the different technologies available and different application integration options.

### 4.4. Database management systems

When developing Web-based simulation applications, a database is typically a necessity, in order to store information for example on users, application data and for storage of simulation models, structure and data. By far the most common form of database used on the Web is the relational database. The basic data structure of the relational model can be thought of as a table, where information about a particular entity is represented in columns and rows. Users (or programs) request data from a relational database by sending it a query that is written in a special language, usually a dialect of Structured Query Language (SQL). There are a number of different relational database management systems (RDBMSs) vendors in the marketplace and include the following:

- *MySQL*. MySQL is a very popular and powerful open source relational RDBMS. McGrath et al. [105] used MySQL in the development of their distributed simulation for support of emergency response services.
- *Microsoft Office Access*. Access is part of the Office suite and is designed to be installed on a client machine rather than being used for the Web. However, because it is relatively easy to use, it has been known to be used in Web-based simulation applications, in particular prototypes. For example, in their paper presenting a Web-based virtual factory and simulator for industrial statistics, Chi et al. [120] use Microsoft Access as the database.
- *Microsoft SQL Server*. SQL Server is Microsoft's database designed for use on the Web. Its primary query language is Transact-SQL (T-SQL), an implementation of SQL. Kumara et al. [121] gives an example of the use of SQL Server in a Web-based simulation application.
- *PostgreSQL*. PostgreSQL is an object RDBMS, and is a free open source tool in the same vein as MySQL.
- *Sybase*. Sybase is a RDBMS very like Microsoft SQL Server due to a deal being struck in the past for them to share source code for it with Microsoft, who marketed under their own name.
- *Oracle Database*. The Oracle Database is a proprietary RDBMS developed by Oracle Corporation. Guru et al. [34] give an example of a Web-based simulation application developed using Oracle.
- *IBM DB2*. DB2 is a RDBMS developed by IBM and was the first major database product to use SQL.

In designing a Web-based simulation application which uses a database management system, it is important to consider technologies that exist in order to carry this out. Typically the simulation tools outlined in Section 4.2 all have either one or multiple software APIs in order to read from and write to a DBMS. These types of technologies include:

- *Open Database Connectivity (ODBC)*. Simulation tools such as eM-Plant and Flexsim provide the ability to connect to a DBMS using ODBC.

- *Object Linking and Embedding, Database (OLEDB)*. OLEDB was designed by Microsoft for accessing different types of data and is implemented in COM.
- *ActiveX Data Objects (ADO.NET)*. ADO.NET is a set of software components, which are part of the base class library included with Microsoft .NET. They are used to access data and data services.
- *Java Database Connectivity (JDBC)*. JDBC is an API for Java defining how a database can be accessed.

This section presented a number of different DBMS alternatives, and the types of technologies in existence in order to read from them and write to them. In order to understand the nature of the Web in terms of its effect on Web-based simulation, it is important to know how it is evolving from a technical viewpoint. The next section presents the evolution of the Web from Web 1.0 to Web 3.0 and its relationship to WBS is discussed.

## 5. Evolution of the Web and its relationship to WBS

### 5.1. Web 1.0 and Web 2.0

According to Schroth [122], the term Web 2.0 has been coined by Tim O'Reilly in 2005 to describe a quickly growing set of Web-based applications that share a common philosophy of "mutually maximising collective intelligence and added value for each participant by formalized and dynamic information sharing" [123]. There is no precise definition of Web 2.0 [60]. The early World-Wide-Web has been retrospectively named "Web 1.0" after "Web 2.0" was coined, and generally covers the period of the Web up until 2003, or "pre-Web 2.0" [124]. Due to its strong impact on the way the Web is perceived by users and also due to its relevance for businesses, Web 2.0 has attracted the attention of both mass media and the scientific community [122], and many developers are currently migrating their applications to Web 2.0 by using RIA technologies [125].

According to Murugesan [124], Web 2.0 is both a usage and a technology paradigm which is different from Web 1.0 in several ways, for example it:

- facilitates flexible Web design, creative reuse, and updates;
- provides a rich, responsive user interface;
- facilitates collaborative content creation and modification;
- enables the creation of new applications by reusing and combining different applications on the Web or by combining data and information from different sources;
- establishes social networks of people with common interests; and
- supports collaboration and helps gather collective intelligence.

Schroth [122] and Hoegg [126] also attempt to classify Web 2.0 applications as follows:

- First, *communities* aim at unifying their users by means of a common ideal such as social networking, knowledge sharing, or participation in a special interest group.
- Second, certain *platforms or tools* enable users to create and share content or even application functionality with a broad audience.
- *On-line* collaboration tools support users in collaboratively performing certain tasks such as performing on-line collaborative brainstorming.

The following concepts can be attributed to being Web 2.0 concepts, amongst others:

- *Blogs* [60,124,127]. The term "Blog" is short for "Web-log" and users have individual pages with regular updates of, for example, commentary or other multimedia material.
- *Wikis* [60,124,127]. A wiki is a page or collection of Web pages designed to enable anyone who accesses it to contribute or modify content, using a simplified mark-up language.
- *Mash-ups* [60,124,128]. Mash-ups is the "mixing" of different sets of data to create a new Web application.
- *Rich Internet Applications* [128]. As noted in Section 4.3, RIAs are Web applications that run on the client-side and have features and functionality akin to traditional desktop-based applications.
- *RSS* [124,128]. This technology is used to enable RIAs, and is known as Really Simple Syndication (RSS). RSS is currently being widely used in media companies on a subscription basis to deliver real-time news as feeds to users.
- *Software-as-a-service (SaaS) and cloud computing*, [60,128]. SaaS is a business model to deliver software to the end user as a service instead of packaged software which requires local installation. SaaS is often referred to as Software-on-Demand or Application-on-Demand. In cloud computing information is permanently stored in servers on the Internet and cached temporarily on clients
- *Collective intelligence/on-line communities/forums*, [124,127,128]. Collective intelligence is which is the generation of knowledge based on the wisdom of crowds rather than the expert few.

Technologies driving and enabling the growth of Web 2.0 include AJAX and RIAs, as well as other Web technologies as described in Section 4.3. Taking the examples above, it is useful to think about how they can be exploited by the area of

Web-based simulation. There were not many identified examples in literature of Web 2.0 concepts being exploited in the area of Web-based simulation. However, in examining the different areas, the following paragraphs give examples of the above in the context of Web-based simulation.

Table 4 gives current Web 2.0 concepts as applied to WBS, and some possible uses of Web 2.0 for WBS. Referring to this table, blogs currently give users the possibility of writing articles and personal opinions about Web-based simulation, see [129] and [130] for examples of their use at the time of writing. One possible further use is in Web-based simulation documentation, including text, images, audio and video, and in disseminating results, see Section 3.4.

Wikis are currently in use in the area of Web-based simulation. One use is to describe the area of Web-based simulation, as seen in Web-based simulation Wikipedia article, see [131]. The team behind the open source SimPy simulation package provide a wiki for tool documentation and support, see [66]. Suggs and Lewis [132] give an example of the use of a Wiki-style knowledge portal in giving an organisation the ability for users to share simulation models with others and apply them to all levels of an organisation. This is a form of model repository, see Section 3.5. Users can find pre-built simulation models using the wiki for use in their work. It is further used to aid collaborative modelling.

There are a number of examples of the use of *Mash-ups* in relation to Web-based simulation identified in literature. Li et al. [133] propose the use of services mash-up technologies in relation to collaborative model development within the "Enterprise Interoperability Roadmap". In the same manner, Jones et al. [134] propose the use of mash-up technologies when dealing with complex information-based networked industrial (CINI) systems as defined by the same roadmap. Wainer et al. [135] present the design and implementation of a distributed simulation engine with Web service components which paves the way for discrete event system specification (DEVS) standardisation, providing a mash-up approach allowing for integration with environments such as Google maps. One possible use for mash-ups is to create a customised Web-based simulation application from a number of Web-based simulation sources.

RSS is currently used in a number of ways in the area of WBS. The Website for Simio (see Table 4 and [136]) gives RSS feeds providing news of such things as updates to their software and events. Similarly, the Website for downloading SimPy (see Table 3 and [137]) gives an option to view package updates as RSS feeds. In addition to this use, journals in the area of WBS provide such things as new abstracts as RSS feeds.

A number of papers give application service providing (ASP) as a possibility for deploying Web-based simulation applications, see [31,138,139]. Application service providing is a form of SaaS, whereby the Web-based simulation software is hosted by a third party. The SaaS software vendor may also host the application on its own Web server. This model of software deployment is becoming more popular for deploying applications.

On-line communities and forums provide for a method of collaborative intelligence on the Web. For example, many simulation vendors now have support forums for their software, forming on-line communities, for example Simul8 provide users with a PHPBB (PHP Bulletin Board) forum for discussion, see [140], and Simulation123 provide a general simulation discussion forum, see [141]. Another possible use for these could be in the area of collaborative simulation model development, or used as model repositories.

In conclusion, it can be seen that there are many aspects of Web 2.0 concepts that are and can be utilised in the development of Web-based simulation applications. The next section presents another current trend/characteristic of the Web, namely service-oriented architectures in relation to Web-based simulation.

### 5.2. Service-oriented architectures

Howerton [142] describes the next generation after Web 2.0 as service-oriented architectures (SOA). In describing SOA in relation to WBS, it is useful to describe how simulation modelling styles have evolved. In the past, simulation languages such

**Table 4**
Current and possible Web 2.0 concepts applied to WBS.

| Web 2.0 concept | Current WBS use | Examples of current WBS use | Other possible WBS use |
|---|---|---|---|
| Blog | WBS News. article publishing, personal opinions | See [129–130] | WBS documentation (see Section 3.4) |
| Wiki | Simulation tool description and support, WBS description, WBS documentation, Model repository | See [66,131,132] | System model description for simulation |
| Mash-up | Integration of existing distributed simulation system with other environments | [135] | Customising a Web-based simulation application |
| RSS | News delivery for Web-based simulation, including general news, WBS journal updates, WBS software updates | See [136,137] | No possible further uses identified |
| RIA | Providing "desktop-like" applications on the Web, see Section 4.3 | See Section 4.3 for examples | Providing more interactive and usable WBS applications |
| SaaS and cloud computing | Deploying WBS applications | See [31,138,139] | No possible further uses identified |
| Collective intelligence/ on-line communities and forums | Simulation software forums | See [140,141] | Collaborative simulation model development, model repository |

as GPSS, SLAM, and SIMAN used a procedural based style of programming, whereby a level of abstraction was provided through the use of simulation libraries, and were augmented by general programming code such as C and Fortran [143]. Two of the main problems with this style of programming for simulation are that procedures do not correspond to real world components (instead they correspond to methods and algorithms), and the lack of extensibility [143]. The only way to adapt simulations developed in this way was through functional extension, because procedural changes were the only approach to model changes, and vendors had no way to hide implementation details, either being forced to give access to source code or restrict access to these features [143]. An example of this is highlighted by Legato and Mazza [144], whereby they choose an object-oriented approach over a procedural one, due to what they perceive to be a higher level of complexity and programming effort involved in taking a procedural approach to the development of their model.

According to Joines and Roberts [143], object-orientation for simulation originated with SIMULA, a programming language with simulation functionality built into the language grammar and syntax. An object-oriented simulation consists of a set of objects that interact with each other over time, and has great intuitive appeal in applications because it is very easy to view the real world as being composed of objects [143,145]. In developing an object-oriented simulation model, a simulation language provides the user with a set of pre-defined object classes (i.e. resources, activities, etc.) from which the modeller can create the necessary objects, declare them and specify their behaviour through the parameters available [143]. The integration of all the objects into a single bundle provides the simulation model. The properties of each object are encapsulated, and classes are used to define the objects, and objects are formed through constructors, composition or inheritance. Many simulation languages and packages offer pre-specified functionality produced in another language, so that users have only limited opportunity to extend their features, using an object-based approach, as opposed to an object-oriented one [143]. In this case, extensibility is only allowed through composition (new objects can only be created out of existing objects) [143]. Many of the COTS VIMS tools presented in Table 2 use this approach. Many of the languages presented in Table 3 utilise a fully object-oriented approach.

The concept of service-oriented computing (SOC) based on service-oriented architectures first came about in the late 1990s [122], the idea behind being to provide a basic set of services that each application can access to provide a common functionality, in essence, to write code once and then use it everywhere [142]. The major components of a basic SOA and their possible interactions are: a service provider publishes his service directly via a service registry where a service requester can find the service and subsequently may bind to the service provider [122]. SOA shares similarities with object-oriented architecture/design (OOA/OOD) in that data is encapsulated, interfaces are defined and both provide levels of abstraction. One of the key differences is that OOA provides abstraction at a class level (defining a system's architecture as a composition of objects), whereas SOA provides it at a business level (defining a system's architecture as a composition of services). Another key difference is a focus on the loose-coupling of services. The central concept of the SOA reference model is the existence of services which provide access to capabilities by well-defined interfaces to be exercised following a service contract with certain policies, which enables this loose-coupling of services [122]. This allows for more agility than if the architecture/systems were tightly coupled, and the failure of a single component should not take down other components in the system if the SOA is designed correctly, leading to more resiliency. Regarding the implementation of SOA, the most common method is via Web Services, which use XML for data and platform neutral communications [142].

In terms of utilising SOA for modelling and simulation (M&S), Sarjoughian et al. [146] note that there has been a focus on the use of Web Services for distributed simulation. It has been noted in Section 3.7 that Grids have been closely aligned with Web Services and SOA [60]. Other examples of this (see [146]) are of the core capabilities of HLA (see Section 3.7) being extended with SOA concepts [147] or Web Services used for distributed simulation [148]. An example of distributed simulation with SOA is given by Lendermann et al. [149], who use a combination of HLA and SOA in addition to COTS simulation software and Grid Computing techniques in the development of their proposed system.

There are a number of other examples of the use of SOA in relation to simulation. An example of SOA for Web-based simulation is given by Meyer zu Eissen and Stein [30]. In their paper they describe Web-based simulation services, using a Web Services with SOAP approach to implement them to be used within a SOA environment. Fitzgerald and Harper [150] reference an example whereby ExtendSim (see Table 2) was utilised in the creation of SOA simulation models for the Department of Defence. Hotz et al. [151] designed a decision support system following an SOA in which simulation models are dynamically configured, using two types of simulation engine: WBI Modeler (IBM Corporation) and Vensim (Ventana Systems Inc.). In terms of simulation application reuse, Balci et al. [152] give an example whereby SOA can be employed for developing a network-centric M&S application by way of reuse of simulation models, submodels, components, and services over a network. Kanacilo and Verbraeck [153] use SOA to support the control design of rail infrastructures, proposing an architecture consisting of a library of simulation components organised in a service-oriented way. Pitchitlamken [154] developed a spreadsheet simulation system that utilises the power of parallel computing on a Windows-based desktop grid, through the use of Web Services and SOA. Sarjoughian et al. [146] propose an SOA-compliant DEVS (SOAD) simulation framework for modelling service-oriented computing systems. Zhang et al. [155] give an example in which, based on SOA, they propose *Service Oriented Simulation Development* (SOSD), in which simulation resources are encapsulated as Web Services. They utilise the Semantic Web (see Section 5.3) in doing so. Lacy and Gerber [156] present a possible future scenario whereby if simulation Web Services were described and represented using Semantic Web technology, software agents could scour the Web for available Web Services to compose a simulation model.

The next section presents the next phase of the Web, known as Web 3.0, and introduces the evolution of the Web towards the Semantic Web and 3D spaces.

### 5.3. Web 3.0 (Semantic Web)

Subsequent to the establishment of Web 2.0, authors are now describing the "Semantic Web" as Web 3.0, see [157,158], or "next generation Web", see [159]. The Semantic Web is an evolving extension of the World-Wide-Web in which the semantics of information and services on the Web is defined, making it possible for the Web to understand and satisfy the requests of people and machines to use the Web content [160]. According to Miller and Baramidze [159], one of the missions of the Semantic Web is to put more knowledge on the Web in an organised fashion and link it to other information and data sources.

Much of the work of creating meaningful descriptions for Web resources involves the use of ontologies, and creating ontologies for modelling and simulation is harder than for other areas for two reasons [159]:

1. It is not domain limited since models may simulate biological, chemical, physical, clerical, transportation, military, services or manufacturing, etc.
2. Modelling and simulation methodology is founded in mathematics, probability and statistics and hence to be rigorous about it, ontologies for these fields should serve as foundations (as so called mid-level ontologies).

This is not to say it cannot be done. In a paper by [155] they attempt to use Semantic Web Services and Ontology to realize a new modelling method for rapidly constructing simulation systems through simulation services customisation and composition. Holmes et al. [161] state that the future of Web-based simulation may be found in the Semantic Web.

In their paper, Araujo et al. [162] present a differing vision of Web 3.0. They (along with the Web 3D Consortium, see [163]) envisage the evolution of Web browsing from 2D forms towards 3D spaces, which in the case of the Web go from the VRML (Virtual Reality Modelling Language) file format created in 1994 to the most recent X3D (Extensible 3D) standard. An example of this applied to Web-based simulation is where Salisbury et al. [164] present a method of coupling applet technologies with Java3D in developing 3D visualisation for Web-based simulation. An example of X3D used in a Web-based simulation environment is given by Kim et al. [165], who constructed an XML-based framework and application called "rube", a key contribution of which allows for the ability to customise simulation model presentation using X3D.

## 6. Conclusions

The area of Web-based simulation is growing, with the future holding great promise. In this paper, it has been seen that utilising the Web for the area of WBS brings with it a number of advantages, including ease of use, collaboration features, license and deployment models, the ability to reuse models, the ability to control access, and the advantages in terms of versioning, customisation and maintenance. However, it can be seen that there are also a number of disadvantages, including loss in speed, GUI limitations, security vulnerability, application stability and licensing restrictions. However, a number of these disadvantages are being addressed as Web technologies evolve, such as Rich Internet Applications which have features and functionality akin to desktop-based applications and can be used to alleviate the limitations of the graphical user interface provided by the Web, and loss in speed being negated by faster networks and faster servers.

A number of different categories of WBS have been established and outlined. Local, remote and hybrid WBS architectures have been described and their advantages and disadvantages over each other presented. WBS documentation and model repositories are also given as categories of WBS. Related categories to WBS have also been presented, namely component-based and distributed simulation.

Technologies in existence for enabling WBS have been presented in a modular fashion, and include middleware, simulation tools, Web-related tools and database management systems. Middleware is important as it enables different modules in a Web-based simulation application to interoperate. Simulation tools provide the engine for a simulation module in a WBS application, and possible tools, languages and libraries for this are described. Web-related tools include languages and technologies that can be used to create WBS modules on both the server- and client-sides in a WBS application. Database management systems are presented as when developing a Web-based simulation application, a database is typically a necessity in order to store information for example on users, application data and for storage of simulation models, structure and data.

Finally, the current evolution of the Web is given in terms of its applicability to the area of WBS. Web 1.0 and Web 2.0 have been reviewed, along with service-oriented architectures, the Semantic Web, and the evolution of the Web towards 3D spaces. It can be seen that all of these evolutionary areas have applications in the area of WBS.

## References

[1] Y.-C. Luo, C.-H. Chen, E. Yücesan, I. Lee, Distributed web-based simulation optimization, in: J.A. Joines, R.R. Barton, K. Kang, P.A. Fishwick (Eds.), Proceedings of the 32nd Conference on Winter Simulation, Orlando, Florida, USA, 2000, pp. 1785–1793.
[2] V. Paxson, S. Floyd. Why We Don't Know How to Simulate The Internet, in: S. Andradóttir, K.J. Healy, D.H. Withers, B.L. Nelson (Eds.), Proceedings of the 29th Conference on Winter Simulation, Atlanta, Georgia, USA, 1997, pp. 1037–1044.
[3] S. Straßburger, T. Schulze, U. Klein, J.O. Henriksen, Internet-based Simulation using off-the-shelf simulation tools and HLA, in: D.J. Medeiros, E.F. Watson, D. Carson, M.S. Manivannan (Eds.), Proceedings of the 30th Conference on Winter Simulation, Washington, DC, USA, 1998, pp. 1669–1676.
[4] M. Alfonseca, J. de Lara, H. Vangheluwe, Web-based simulation of systems described by partial differential equations, in: B.A. Peters, J.S. Smith, D.J. Medeiros, M.W. Rohrer (Eds.) Proceedings of the 33rd Conference on Winter Simulation, Arlington, Virginia, USA, 2001, pp. 629–636.

[5] E. Yücesan, Y.-C. Luo, C.-H. Chen, I. Lee, Distributed web-based simulation experiments for optimization, Simulation Practice and Theory 9 (1–2) (2001) 73–90.
[6] H.M. Deitel, P.J. Deitel, T.R. Nieto, Internet & World Wide Web How to Program, Prentice-Hall Inc., New Jersey, 2000.
[7] E.H. Page, Beyond speedup: PADS, the HLA and web-based simulation, in: R. Fujimoto, S.J. Turner (Eds.), Proceedings of the Thirteenth Workshop on Parallel and Distributed Simulation, Atlanta, Georgia, USA, 1999, pp. 2–9.
[8] P.A. Fishwick, Web-based simulation, in: S. Andradóttir, K.J. Healy, D.H. Withers, B.L. Nelson (Eds.), Proceedings of the 29th Conference on Winter Simulation, Atlanta, Georgia, USA, 1997, pp. 100–102.
[9] J. Kuljis, R.J. Paul, An appraisal of web-based simulation: whither we wander?, Simulation Practice and Theory 9 (1–2) (2001) 37–54
[10] E.H. Page, A. Buss, P.A. Fishwick, K.J. Healy, R.E. Nance, R.J. Paul, Web-based simulation: revolution or evolution?, ACM Transactions on Modelling and Computer Simulation (TOMACS) 10 (1) (2000) 3–17
[11] Y. Huang, G. Madey. Autonomic web-based simulation, in: Proceedings of the 38th Annual Symposium on Simulation (ANSS'05), IEEE Computer Society, Washington, DC, USA, 2005, pp. 160–167.
[12] S.D. Bencomo, Control learning: present and future, Annual Reviews in Control 28 (1) (2004) 115–136.
[13] S.W. Reichenthal, Re-introducing web-based simulation, in: E. Yücesan, C.-H. Chen, J.L. Snowdon, C.J. M. (Eds.), Proceedings of the 34th Conference on Winter Simulation: Exploring New Frontiers, San Diego, California, USA, 2002, pp. 847–852.
[14] J.A. Miller, A.F. Seila, J. Tao, Finding a substrate for federated components on the web, in: J.A. Joines, R.R. Barton, K. Kang, P.A. Fishwick (Eds.), Proceedings of the 32nd Conference on Winter simulation, San Diego, California, USA, 2000, pp. 1849–1854.
[15] J.A. Miller, P.A. Fishwick, S.J.E. Taylor, P. Benjamin, B. Szymanski, Research and commercial opportunities in web-based simulation, Simulation Practice and Theory 9 (1–2) (2001) 55–72.
[16] P.A. Fishwick, Web-based simulation: some personal observations, in: J.M. Charnes, D.J. Morris, D.T. Brunner, J.J. Swain (Eds.), Proceedings of the 28th Conference on Winter Simulation, Coronado, California, USA, 1996, pp. 772–779.
[17] E.H. Page, The rise of web-based simulation: implications for the high level architecture, in: D.J. Medeiros, E.F. Watson, J.S. Carson, M.S. Manivannan (Eds.), Proceedings of the 30th Conference on Winter Simulation, IEEE Computer Society Press, Washington, DC, USA, 1998, pp. 1663–1668.
[18] T.L. Veith, J.E. Kobza, C.P. Koelling, Netsim: Java(TM)-based simulation for the World Wide Web, Computers & Operations Research 26 (6) (1999) 607–621.
[19] J.A. Miller, A.F. Seila, X. Xiang, The JSIM web-based simulation environment, Future Generation Computer Systems 17 (2) (2000) 119–133.
[20] P.A. Fishwick, Using XML for simulation modeling, in: E. Yücesan, C.-H. Chen, J.L. Snowdon, J.M. Charnes (Eds.), Proceedings of the 34th Conference on Winter Simulation: Exploring New Frontiers, San Diego, California, USA, 2002, pp. 616–622.
[21] T. Wiedemann, Simulation application service providing (SIM-ASP), in: B.A. Peters, J.S. Smith, D.J. Medeiros, M.W. Rohrer (Eds.), Proceedings of the 33rd Conference on Winter Simulation, Arlington, Virginia, USA, 2001, pp. 623–628.
[22] T.K. Leong, B.M. Ali, V. Prakash, N.K. Nordin, A prototype of web-based simulation environment: using CGI and Javascript, in: Proceedings of IEEE TENCON 2000, Kuala Lumpur, Malaysia, 2000, pp. 357–360.
[23] D.M. Rao, P.A. Wisley, Dynamic component substitution in web-based simulation. in: J.A. Joines, R.R. Barton, K. Kang, P.A. Fishwick (Eds.), Proceedings of the 32nd Conference On Winter Simulation, Orlando, Florida, USA, 2000, pp. 1840–1848.
[24] L. Whitman, B. Huff, S. Palaniswamy, Commercial simulation over the web, in: D.J. Medeiros, E.F. Watson, J.S. Carson, M.S. Manivannan (Eds.), Proceedings of the 30th Conference on Winter simulation, Washington, DC, USA, 1998, pp. 335–340.
[25] E.H. Page, J.M. Opper, Observations on the complexity of composable simulation, in: P.A. Farrington, H.B. Nembhard, D.T. Sturrock, G.W. Evans (Eds.), Proceedings of the 31st Conference on Winter Simulation: Simulation – A Bridge to the Future, Phoenix, Arizona, USA, 1999, pp. 553–560.
[26] Y.-H. Wang, Y.-C. Liao. Implementation of a collaborative web-based simulation modeling environment, in: Proceedings of the Seventh IEEE International Symposium on Distributed Simulation and Real-Time Applications (DS-RT'03), Delft, The Netherlands, 2003, pp. 150–157.
[27] Y.-H. Tao, S.-M. Guo, The design of a web-based training system for simulation analysis, in: B.A. Peters, J.S. Smith, D.J. Medeiros, M.W. Rohrer (Eds.), Proceedings of the 33rd Conference on Winter Simulation, Arlington, Virginia, USA, 2001, pp. 645–652.
[28] Y.J. Son, A.T. Jones, R.A. Wysk, Component based simulation modeling from neutral component libraries, Computers & Industrial Engineering 45 (1) (2003) 141–165.
[29] M.M.G. DeRico, R.B. Byrnes Jr., J.H. Schafer, J.A. Martin, M.D. McNett, G.F. Stone III, Using intelligent agents to combine heterogeneous distributed data, in: IEEE International Conference on Systems, Man, and Cybernetics, San Diego, California, USA, 1998, pp. 2831–2835.
[30] S. Meyer zu Eissen, B. Stein, Realization of web-based simulation services, Computers in Industry 57 (3) (2006) 261–271.
[31] C. Marr, C. Storey, W.E. Biles, J.P.C. Kleijnen, A Java-based simulation manager for web-based simulation, in: R.R.B. J. A. Joines, K. Kang, P. S. Fishwick (Eds.), Proceedings of the 32nd Conference on Winter Simulation, Orlando, Florida, USA, 2000, pp. 1815–1822.
[32] R.J. Paul, S.J.E. Taylor, What use is model reuse: is there a crook at the end of the rainbow? in: E. Yücesan, C.-H. Chen, J.L. Snowdon, J.M. Charnes (Eds.), Proceedings of the 34th Conference on Winter Simulation: Exploring New Frontiers, San Diego, California, USA, 2002, pp. 648–652.
[33] W. Suh, Web application development methodologies, in: J. Travers (Ed.), Web Engineering: Principles and Techniques, Idea Group Publishing (an imprint of Idea Group Inc.), London, 2005, pp. 76–93.
[34] A. Guru, P. Savory, R. Williams, A web-based interface for storing and executing simulation models, in: J.A. Joines, R.R. Barton, K. Kang, P.A. Fishwick (Eds.), Proceedings of the 32nd Conference on Winter Simulation, Orlando, Florida, 2000, pp. 1810–1814.
[35] T.-D. Graupner, H. Richter, W. Sihn. Configuration, simulation and animation of manufacturing systems via the internet, in: E. Yücesan, C.-H. Chen, J.L. Snowdon, J.M. Charnes (Eds.), Proceedings of the 34th Conference on Winter Simulation: Exploring New Frontiers, San Diego, California, USA, 2002, pp. 825–831.
[36] R.M. Cubert, P.A. Fishwick, A framework for distributed object-oriented multimodeling and simulation, in: S. Andradóttir, K.J. Healy, D.H. Withers, B.L. Nelson (Eds.), Proceedings of the 29th Conference on Winter Simulation, Atlanta, Georgia, USA, 1997, pp. 1315–1322.
[37] S. Narayanan, Web-based modeling and simulation, in: J.A. Joines, R.R. Barton, K. Kang, P.A. Fishwick (Eds.), Proceedings of the 32nd Conference on Winter Simulation, Orlando, Florida, USA, 2000, pp. 60–62.
[38] P. Lorenz, H. Dorwarth, K.-C. Ritter, T.J. Schriber, Towards a web based simulation environment, in: S. Andradóttir, K.J. Healy, D.H. Withers, B.L. Nelson Atlanta, Georgia (Eds.), Proceedings of the 29th Conference on Winter Simulation, USA, 1997, pp. 1338–1344.
[39] D.S. Myers, An Extensible Component-Based Architecture for Web-Based Simulation Using Standards-Based Web Browsers, Department of Computer Science, Virginia Polytechnic Institute and State University, 2004.
[40] W.J. Davis, X. Chen, A. Brook, Implementing on-line simulation upon the world-wide web, in: D.J. Medeiros, E.F. Watson, J.S. Carson, M.S. Manivannan (Eds.), Proceedings of the 30th Conference on Winter Simulation, Washington, DC, USA, 1998, pp. 87–96.
[41] W. Yurcik, J. Vila, L. Brumbaugh, An interactive web-based simulation of a general computer architecture, in: Proceedings of the IEEE International Conference on Engineering and Computer Education (ICECE 2000), San Paulo, Brazil, 2000.
[42] A. Holzinger, W. Emberger, S. Wassertheurer, L. Neal, Design, development and evaluation of online interactive simulation software for learning human genetics, Elektrotechnik & Informationstechnik 125 (5) (2008) 190–196.
[43] H.B. Chen, O. Bimber, C. Chintamani, E. Poole, S.J. Buckley, eSCA: a thin-client/server/web-enabled system for distributed supply chain simulation, in: P.A. Farrington, H.B. Nembhard, D.T. Sturrock, G.W. Evans (Eds.), Proceedings of the 31st Conference on Winter Simulation: Simulation – A Bridge to the Future – Volume 2, Phoenix, Arizona, USA, 1999, pp. 1371–1377.
[44] W.E. Lulay, G. Reinhart, Coordination order processing in decentralized production units using hierarchical simulation models and Web-technologies, in: D.J. Medeiros, E.F. Watson, J.S. Carson, M.S. Manivannan (Eds.), Proceedings of the 30th Conference on Winter Simulation, Washington, DC, USA, 1998, pp. 1655–1661.

[45] M. Pidd, N. Oses, R.J. Brooks, Component-based simulation on the web? in: P.A. Farrington, H.B. Nembhard, D.T. Sturrock, G.W. Evans (Eds.), Proceedings of the 31st Conference on Winter Simulation: Simulation – A Bridge to the Future, Phoenix, Arizona, USA, 1999, pp. 1438–1444.
[46] A. Brogi, C. Canal, E. Pimentel, On the semantics of software adaptation, Science of Computer Programming 61 (2) (2006) 136–151.
[47] K.-K. Lau, Component-Based Software Development – Case Studies, World Scientific Publishing Co. Pte. Ltd., London, 2004.
[48] M.F. Bertoa, J.M. Troya, A. Vallecillo, Measuring the usability of software components, Journal of Systems and Software 79 (3) (2006) 427–439.
[49] C. Syzyperski, Component technology: what, where, and how? in: Proceedings of the 25th International Conference on Software Engineering, IEEE Computer Society, Portland, Oregan, 2003, pp. 684–693.
[50] M. Fukunari, Y.-L. Chi, P.M. Wolfe, JavaBean-based simulation with operational procedure table (OPT), Future Generation Computer Systems 17 (5) (2001) 513–523.
[51] E.H. Page, S.P. Griffen, S.L. Rother, Providing conceptual framework support for distributed web-based simulation within the higher level architecture, in: Proceedings of the SPIE Conference on Enabling Technologies for Simulation Science II, Orlando, Florida, USA, SPIE, Bellingham, WA, USA, 1998, pp. 287–292.
[52] E.H. Page, R.L. Moose Jr., S.P. Griffen, Web-based simulation in SIMJAVA using remote method invocation, in: S. Andradóttir, K.J. Healy, D.H. Withers, B.L. Nelson (Eds.), Proceedings of the 29th Conference on Winter Simulation, Atlanta, Georgia, USA, 1997, pp. 468–474.
[53] H.M. Soliman, A.S. Elmaghraby, M.A. El-Sharkawy, Parallel and distributed simulation: an overview, in: Proceedings of the IEEE Symposium on Computers and Communications (ISCC'95), IEEE Computer Society, Alexandria, Egypt, 1995, pp. 270–276.
[54] W.E. Biles, B.J. Casabier, Web-based evaluation of material handling alternatives for automated manufacturing: a parallel replications approach, in: R.G. Ingalls, M.D. Rossetti, J.S. Smith, B.A. Peters (Eds.), Proceedings of the 36th Conference on Winter Simulation, Washington, DC, USA, 2004, pp. 1527–1532.
[55] W.E. Biles, J.P.C. Kleijnen, International collaborations in web-based simulation: a focus on experimental design and optimization, in: M.E. Kuhl, N.M. Steiger, F.B. Armstrong, J.A. Joines (Eds.), Proceedings of the 37th Conference on Winter Simulation, Orlando, Florida, USA, 2005, pp. 218–222.
[56] B.P. Gan, L. Liu, S. Jain, S.J. Turner, W. Cai, W.-J. Hsu, Distributed supply chain simulation across enterprise boundaries, in: J.A. Joines, R.R. Barton, K. Kang, P.A. Fishwick (Eds.), Proceedings of the 32nd Conference on Winter Simulation, Orlando, Florida, 2000, pp. 1245–1251.
[57] U. Klein, T. Schulze, S. Straßburger, Traffic simulation based on the high level architecture, in: D.J. Medeiros, E.F. Watson, J.S. Carson, M.S. Manivannan (Eds.), Proceedings of the 30th Conference on Winter Simulation, Washington, DC, USA, 1998, pp. 1095–1104.
[58] K.L. Morse, M.D. Petty, High level architecture data distribution management migration from DoD 1.3 to IEEE 1516: research articles, Concurrency and Computation: Practice and Experience 16 (15) (2004) 1527–1543.
[59] A. Elkins, J.W. Wilson, D. Gracanin, Security issues in high level architecture-based distributed simulation, in: B.A. Peters, J.S. Smith, D.J. Medeiros, M.W. Rohrer (Eds.), Proceedings of the 33nd Conference on Winter Simulation, Washington, DC, USA, 2001, pp. 818–826.
[60] G. Fox, M. Pierce, Grids challenged by a Web 2.0 and multicore sandwich, Concurrency and Computation: Practice and Experience 21 (3) (2009) 265–280.
[61] N. Mustafee, S.J.E. Taylor, Supporting simulation in industry through the application of grid computing, in: S.J. Mason, R.R. Hill, L. Monch, O. Rose, T. Jefferson, J.W. Fowler (Eds.), Proceedings of the 40th Conference on Winter Simulation: Global Gateway to Discovery, Miami, Florida, USA, 2008, pp. 1077–1085.
[62] S. Phatanapherom, P. Uthayopas, V. Kachitvichyanukul. Fast simulation model for grid scheduling using hypersim, in: S. Chick, P.J. Sánchez, D. Ferrin, D.J. Morrice (Eds.), Proceedings of the 35th Conference on Winter Simulation: Driving Innovation, New Orleans, Louisiana, USA, 2003, pp. 1494–1500.
[63] I. Foster, C. Kesselman, The Grid 2: Blueprint for a New Computing Infrastructure. The Elsevier Series in Grid Computing, Morgan Kaufmann, 2004.
[64] H. Xi, H. Cao, L. Berman, D. Jensen, Distributed supply chain simulation using a generic job running framework, in: S. Chick, P.J. Sánchez, D. Ferrin, D.J. Morrice (Eds.), Proceedings of the 35th Conference on Winter Simulation: Driving Innovation, New Orleans, Louisiana, USA, 2003, pp. 1305–1312.
[65] K.S. Perumalla, R.M. Fujimoto, P.J. Thakare, S. Pande, H. Karimabadi, Y. Omelchenko, J. Dirscoll, Performance prediction of large-scale parallel discrete event models of physical systems, in: M.E. Kuhl, N.M. Steiger, F.B. Armstrong, J.A. Joines (Eds.), Proceedings of the 37th Conference on Winter Simulation, Orlando, Florida, USA, 2005, pp. 356–364.
[66] SimPy, Simpy Wiki, 2008a [cited 2008 20th June]; Available from: <http://www.mcs.vuw.ac.nz/cgi-bin/wiki/SimPy>.
[67] J.O. Henriksen, A. Hanish, S. Osterburg, P. Lorenz, T.J. Schriber, Web-based simulation center: professional support for simulation projects, in: E. Yücesan, C.-H. Chen, J.L. Snowdon, J.M. Charnes (Eds.), Proceedings of the 34th Conference on Winter Simulation: Exploring New Frontiers, San Diego, California, USA, 2002, pp. 807–815.
[68] P. Liston, J. Byrne, C. Heavey, P.J. Byrne, Discrete-event simulation for virtual organisation creation, International Journal of Production Research 46 (5) (2008) 1335–1356.
[69] J. Byrne, P. Liston, C. Heavey, P.J. Byrne, Towards a web-based simulation application for contract costing utilizing a hybrid S&V approach, in: H.R. Arabnia (Ed.), The 2007 International Conference on Modelling, Simulation and Visualisation (MSV'07), Part of the 2007 World Congress in Computer Science, Computer Engineering and Applied Computing (WORLDCOMP'07), CSREA Press, Las Vegas, Nevada, USA, 1997, pp. 171–177.
[70] B. Clegg, S. Wingrove, I. Alexander, Use of the internet to support management flight simulation in the extended enterprise, in: Web Applications in Aerospace (Ref. No. 1999/079), IEE Seminar, 1999, pp. 2/1–2/4.
[71] G.-J. Ahn, Role-based access control in DCOM, Journal of Systems Architecture 46 (13) (2000) 1175–1184.
[72] A. King, R. Hunt, Protocols and architecture for managing TCP/IP network infrastructures, Computer Communications 23 (16) (2002) 1558–1572.
[73] C.C. Chiang, The use of adapters to support interoperability of components for reusability, Information and Software Technology 45 (3) (2003) 149–156.
[74] W. Gilani, N.H. Naqvi, O. Spinczyk. On adaptable middleware product lines, in: Proceedings of the 3rd Workshop on Adaptive and Reflective Middleware, Toronto, Ontario, Canada, 2004, pp. 207–213.
[75] A. Patel, CORBA: protocols, applications, process models and standards, Computer Standards & Interfaces 25 (4) (2003) 299–301.
[76] S. Lankes, A. Jabs, T. Bemmerl, Design and performance of a CAN-based connection-oriented protocol for real-time CORBA, Journal of Systems and Software 77 (1) (2005) 37–45.
[77] G.T. Edwards, D.C. Schmidt, A. Gokhale, Integrating publisher/subscriber services in component middleware for distributed real-time and embedded systems, in: Proceedings of the 42nd Annual Southeast Regional Conference, Huntsville, Alabama, USA, 2004, pp. 171–176.
[78] P. Papajorgji, H.W. Beck, J.L. Braga, An architecture for developing service-oriented and component-based environmental models, Ecological Modelling 79 (1) (2004) 61–76.
[79] J. Hurwitz, Sorting out middleware, DBMS 11 (1) (1998) 10–12.
[80] G. Gousios, E. Aivaloglou, S. Gritzalis, Distributed component architectures security issues, Computer Standards & Interfaces 27 (3) (2005) 269–284.
[81] M. Verwijmeren, Software component architecture in supply chain management, Computers in Industry 53 (2) (2004) 165–178.
[82] S. Chen, Y. Liu, I. Gorton, A. Liu, Performance prediction of component-based applications, Journal of Systems and Software 74 (1) (2005) 35–43.
[83] D.C. Schmidt, F. Buschmann, Patterns, frameworks, and middleware: their synergistic relationships, in: Proceedings of the 25th International Conference on Software Engineering. Portland, Oregan, USA, 2003, pp. 694–704.
[84] C. Zhang, H.-A. Jacobsen, Quantifying aspects in middleware platforms, in: Proceedings of the 2nd International Conference on Aspect-Oriented Software Development, Boston, Massachusetts, USA, 2003, pp. 130–139.
[85] N. Medvidovic, On the role of middleware in architecture-based software development, in: Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering, Ishia, Italy, 2002, pp. 229–306.
[86] W. Emmerich, Software engineering and middleware: a roadmap, in: Proceedings of the Conference on The Future of Software Engineering, Limerick, Ireland, 2000, pp. 117–129.
[87] N. Darwish, COPS: cooperative problem solving using DCOM, Journal of Systems and Software 63 (2) (2002) 79–90.

[88] A. Davis, D. Zhang, A comparative study of SOAP and DCOM, Journal of Systems and Software 76 (2) (2005) 157–169.
[89] S. Wang, J. Xie, Integrating building management system and facilities management on the internet, Automation in Construction 11 (6) (2002) 707–715.
[90] S. Purao, H.K. Jain, D.L. Nazareth, ODE: a tool for distributing object-oriented applications, Information & Management 39 (8) (2002) 689–703.
[91] A. Bechini, P. Foglia, C.A. Prete, Use of a CORBA/RMI gateway: characterization of communication overhead, in: Proceedings of the Third International Workshop on Software and Performance, Rome, Italy, 2002, pp. 150–157.
[92] M.A. de Miguel, Solutions to make Java-RMI time predictable, in: Proceedings of the Fourth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC'01), Magdeburg, Germany, 2001, pp. 379–386.
[93] R. Schwarzkopf, M. Mathes, S. Heinzl, B. Freisleben, H. Dohmann, Java RMI versus .NET Remoting architectural comparison and performance evaluation, in: Proceedings of the Seventh International Conference on Networking (ICN 2008), 2008, pp. 398–407.
[94] C. Britton, P. Bye, IT Architectures and Middleware, second ed., Addison-Wesley, Boston, 2004.
[95] J. Cao, M. Cao, A.S.T. Chan, G. Wu, S.K.S.K. Das, A framework for architecting and high-level programming support of CORBA applications, Journal of Parallel and Distributed Computing 64 (6) (2004) 725–739.
[96] D.J. Ram, A.V. Srinivas, P.M. Rani, A model for parallel programming over CORBA, Journal of Parallel and Distributed Computing 64 (11) (2004) 1256–1269.
[97] Y.-I. Cho, T.G. Kim, DEVS framework for component-based modelling/simulation of discrete event systems, in Proceedings of the 2002 Summer Simulation Conference, San Diego, California, USA, 2002.
[98] A. Cholkar, P. Koopman, A widely deployable web-based network simulation framework using CORBA IDL-based APIs, in: P.A. Farrington, H.B. Nembhard, D.T. Sturrock, G.W. Evans (Eds.), Proceedings of the 31st Conference on Winter Simulation: Simulation – A Bridge to the Future, Phoenix, Arizona, USA, 1999, pp. 1587–1594.
[99] M. Pidd, Simulation worldviews: so what? in: R.G. Ingalls, M.D. Rossetti, J.S. Smith, B.A. Peters (Eds.), Proceedings of the 36th Conference on Winter Simulation, Washington, DC, USA, 2004, pp. 288–292.
[100] D.A. Bodner, T. Giovindaraj, E. Kemahlioglu, E.E. Blanco, M. Goetschalckx, L.F. Mcginnis, G.P. Sharp, Using internet technology for design of facilities and material handling systems, in: Proceedings of the 2000 International Material Handling Research Colloquium, York, Pennsylvania, USA, 2000.
[101] A.G. Bruzzone, Web-based simulation: best of Websim99, Future Generation Computer Systems 17 (5) (2001) 501–502.
[102] Witness, Witness Server, 2008 [cited 2008 10th August]; Available from: <http://www.lanner.com/en/media/witness/server.cfm>.
[103] P.H.M. Jacobs, A.L. Niels, A. Verbraeck. D-SOL; a distributed Java based discrete event simulation architecture, in: E. Yücesan, C.-H. Chen, J.L. Snowdon, J.M. Charnes (Eds.), Proceedings of the 34th Conference on Winter Simulation: Exploring New Frontiers, San Diego, California, USA, 2002, pp. 783–800.
[104] G. Merkuryeva, Y. Merkuryev, J. Tolujev, Computer simulation and metamodelling of logistics processes at a container terminal, Studies in Informatics and Control 9 (1) (2000).
[105] D. McGrath, A. Hunt, M. Bates. A simple distributed simulation architecture for emergency response exercises, in: Proceedings of the 2005 Ninth IEEE International Symposium on Distributed Simulation and Real-Time Applications (DS-RT 2005), 2005, pp. 221–226.
[106] J.-X. Chu, Z.-F. Sun, K.-M. Du, Q. Jia, Y.-C. Wang, S. Liu, A universal web-based simulation system for greenhouse crops, in: Second International Symposium on Plant Growth Modeling, Simulation, Visualization and Applications (PMA'06), 2006, pp. 185–187.
[107] M. Held, W. Blochinger. Collaborative BPEL design with a rich internet application, in: 8th IEEE International Symposium on Cluster Computing and the Grid, 2008, pp. 202–209.
[108] P.H. Chang, A unit test architecture that traces variables of JavaScript programming, in: Proceedings of the IEEE Electro/Information Technology Conference, 2004 (EIT 2004), 2004, pp. 96–100.
[109] S. Pillutla, Creating a web-based simulation gaming exercise using PERL and JavaScript, Simulation and Gaming 34 (1) (2003) 112–130.
[110] A. Mesbah, E. Bozdag, A. van Deursen. Crawling AJAX by inferring user interface state changes, in: Eighth International Conference on Web Engineering (ICWE'08), 2008. pp. 122–134.
[111] J. Shanmugasundaram, K. Tufte, C. Zhang, G. He, D.J. DeWitt, J.F. Naughton, in: Relational databases for querying XML documents: limitations and opportunities, in: Proceedings of the 25th International Conference on Very Large Data Bases, Edinburgh, Scotland, 1999, pp. 302–314.
[112] S. Kachru, E.F. Gehringer, A comparison of J2EE and .NET as platforms for teaching web services, in: 34th ASEE/IEEE Frontiers in Education Conference (FIE 2004), Savannah, GA, USA, 2004, pp. 12–17.
[113] M. Fowler, D. Box, A. Hejlsberg, A. Knight, R. High, J. Crupi. The great J2EE vs. Microsoft.NET shootout, in: Conference on Object Oriented Programming Systems Languages and Applications, Companion to the 19th Annual ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages and Applications, ACM, Vancouver, BC, Canada, 2004, pp. 143–144.
[114] T.C. Shan, W.W. Hua, Taxonomy of Java web application frameworks, in: Proceedings of the IEEE International Conference on e-Business Engineering, 2006, pp. 378–385.
[115] S. Robinson, C. Nagel, K. Watson, J. Glynn, M. Skinner, B. Evjen, Professional C# Third Edition, third ed., Wiley Publishing, Inc, Indianapolis, 2004.
[116] J. Bentrum, J. Whatley, Building e-Commerce Sites with the .NET Framework, SAMS, Indiana, 2002.
[117] K.Z. Ahmed, C.E. Umrysh, Developing Enterprise Java Applications with J2EE and UML, first ed., Addison-Wesley, Boston, 2002.
[118] J. Woo, The comparison of J2EE and .NET for e-Business, in: H.R. Arabnia (Ed.), Proceedings of the 2005 International Conference on e-Business, Enterprise Information Systems, e-Government and Outsourcing (EEE 2005), CSREA Press, Las Vegas, Nevada, USA, 2005, pp. 53–59.
[119] S.P. Ahuja, R. Clark. Comparison of web services technologies from a developer's perspective, in: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05), 2005, pp. 791–792.
[120] X. Chi, M.P.J. Pepper, S.T. A., A web-based virtual factory and simulator for industrial statistics, in: R.G. Ingalls, M.D. Rossetti, J.S. Smith, B.A. Peters (Eds.), Proceedings of the 36th Conference on Winter Simulation, Washington, DC, USA, 2004, pp. 2103–2106.
[121] S.R.T. Kumara, Y.-H. Lee, K. Tang, C. Dodd, J. Tew, S.-T. Yee, Simulation anywhere any time: web-based simulation implementation for evaluating order-to-delivery systems and processes, in: E. Yücesan, C.-H. Chen, J.L. Snowdon, J.M. Charnes (Eds.), Proceedings of the 34th Conference on Winter Simulation: Exploring New Frontiers, San Diego, California, USA, 2002, pp. 1251–1259.
[122] C. Schroth, Web 2.0 versus SOA: converging concepts enabling seamless cross organizational collaboration, in: Proceedings of the 9th IEEE International Conference on E-Commerce Technology and the 4th IEEE International Conference on Enterprise Computing, E-Commerce and E-Services (CEC/EEE 2007), 2007, pp. 47–54.
[123] T. O'Reilly, What Is Web 2.0? Design Patterns and Business Models for the Next Generation of Software, 2005 [cited 2008 5th July]; Available from: <http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html>.
[124] S. Murugesan, Understanding Web 2.0, IT Professional 9 (4) (2007) 34–41.
[125] M. Linaje, J.C. Preciado, F. Sánchez-Figueroa, Engineering rich internet application user interfaces over legacy web models, IEEE Internet Computing 11 (7) (2007) 53–59.
[126] R. Hoegg, R., Martignoni Robert, Meckel Miriam, K. Stanoevska-Slabeva. Overview of business models for Web 2.0 communities, in: Proceedings of GeNeMe (Gemeinschaften in Neuen Medien) 2006, Dresden, Germany, 2006, pp. 23–37.
[127] S. Parise, P.J. Guinan, Marketing using Web 2.0, in: Proceedings of the 41st Annual Hawaii International Conference on System Sciences, 2008, pp. 281–281.
[128] G. Chong Minsk, L. Siew Poh, H. Wei, T. Puay Siew, Web 2.0 concepts and technologies for dynamic B2B integration, in: IEEE Conference on Emerging Technologies & Factory Automation, 2007. ETFA, 2007, pp. 315–321.
[129] T. Blevins, "Modeling and Control" Blog, "Process Simulation Archives" Section, 2008 [cited 2008 20th June]; Available from: <http://www.modelingandcontrol.com/process_simulation/>.

[130] D. Sturrock, Success in Simulation Blog, 2008 [cited 2008 13th September]; Available from: <http://simio.biz/blog/>.
[131] Wikipedia, Web based simulation – Wikipedia, The Free Encyclopedia, 2008 [cited 2008 20th August]; Available from: <http://en.wikipedia.org/w/index.php?title=Web_based_simulation&oldid=239574210>.
[132] R. Suggs, B. Lewis. Enterprise simulation: a practical application in business planning, in: S.G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J.D. Tew, R.R. Barton (Eds.), Proceedings of the 39th Conference on Winter Simulation: 40 Years! The Best is yet to Come, IEEE Press, Washington, DC, USA, 2007, pp. 205–209.
[133] M.-S. Li, A. Deshmukh, A. Jones, An infrastructure to support performance analysis in complex systems, in: Proceedings of the PERMIS Conference, PerMIS, Gaitersburg, MD, USA, 2006.
[134] A. Jones, A. Deshmukh, S. Kumara, M.-S. Li, Engineering complex, information-based, networked industrial systems: a research roadmap, Journal of Computing and Information Science in Engineering 8 (1) (2008) 011005 (13 pages).
[135] G.A. Wainer, R. Madhoun, K. Al-Zoubi, Distributed simulation of DEVS and cell-DEVS models in CD++ using web services, Simulation Modelling Practice and Theory 16 (9) (2008) 1266–1292.
[136] Simio, Simio Website displaying an example of the use of RSS, 2008 [cited 2008 August 12th]; Available from: <http://www.simio.biz/>.
[137] SimPy, SimPy RSS feeds on download page, 2008b [cited 2008 20th June]; Available from: <http://sourceforge.net/project/showfiles.php?group_id=62366>.
[138] S. Chen, F.Y. Lu, Web-based simulations of power systems, IEEE Computer Applications in Power 15 (1) (2002) 35–40.
[139] Y. Huang, X. Xiang, G. Madey, A self manageable infrastructure for supporting web-based simulations, in: Proceedings of the 37th Annual Symposium on Simulation (ANSS-37 2004), IEEE Computer Society, 2004, pp. 149–156.
[140] Simul8, Simul8 Café (support forum), 2008 [cited 2008 July 14th]; Available from: <http://www.simul8.com/phpBB2/viewforum.php?f=1>.
[141] Simulation123, Simulation123 – a "not-for-profit" site for simulation users, 2009 [cited 2009 February]; Available from: <http://www.simulation123.com/>.
[142] J.T. Howerton, Service-oriented architecture and Web 2.0, IT Professional 9 (3) (2007) 62–64.
[143] J.A. Joines, S.D. Roberts. Fundamentals of object-oriented simulation, in: D.J. Medeiros, E.F. Watson, J.S. Carson, M.S. Manivannan (Eds.), Proceedings of the 30th Conference on Winter Simulation, Washington, DC, USA, 1998, pp. 141–150.
[144] P. Legato, R.M. Mazza, Berth planning and resources optimisation at a container terminal via discrete event simulation, European Journal of Operational Research 133 (3) (2001) 537–547.
[145] J.A. Joines, S.D. Roberts, An introduction to object-oriented simulation in C++, in: S. Andradóttir, K.J. Healy, D.H. Withers, B.L. Nelson (Eds.), Proceedings of the 29th Conference on Winter Simulation, Atlanta, Georgia, USA, 1997, pp. 78–85.
[146] H. Sarjoughian, K. Sungung, M. Ramaswamy, S. Yau, A simulation framework for service-oriented computing systems, in: S.J. Mason, R.R. Hill, L. Monch, O. Rose, T. Jefferson, J.W. Fowler (Eds.), Proceedings of the 40th Conference on Winter Simulation, Miami, Florida, USA, 2008, pp. 845–853.
[147] X, Chen, W. Cai, S.J. Turner, Y. Wang, SOAr-DSGrid: service-oriented architecture for distributed simulation on the grid, in: Proceedings of the 20th Workshop on Principles of Advanced and Distributed Simulation, Washington, DC, USA, 2006, pp. 65–73.
[148] X. Hu, B. Zeigler, M.H. Hwang, E. Mak, DEVS systems-theory framework for reusable testing of I/O behaviours in service oriented architectures, in: Proceedings of the 2007 IEEE International Conference on Information Reuse and Integration, Las Vegas, NV, USA, 2007, pp. 394–399.
[149] P. Lendermann, M.Y.H. Low, B.P. Gan, N. Julka, L.P. Chan, L.H. Lee, S.J.E. Taylor, S.J. Turner, W. Cai, X. Wang, T. Hung, L.F. McGinnis, S. Buckley, An integrated and adaptive decision-support framework for high-tech manufacturing and service networks, in: M.E. Kuhl, N.M. Steiger, F.B. Armstrong, J.A. Joines (Eds.), Proceedings of the 37th Conference on Winter Simulation, Orlando, Florida, USA, 2005, pp. 2052–2062.
[150] L.M. Fitzgerald, T.J. Harper, Application of simulation modeling for air force enterprise IT transformation initiatives, in: S.J. Mason, R.R. Hill, L. Monch, O. Rose, T. Jefferson, J.W. Fowler (Eds.), Proceedings of the 40th Conference on Winter Simulation, Miami, Florida, USA, 2008, pp. 1173–1178.
[151] P. Huang, Y.M. Lee, L. An, M. Ettl, S. Buckley, K. Sourirajan, Utilizing simulation to evaluate business decisions in sense-and-respond systems, in: R.G. Ingalls, M.D. Rossetti, J.S. Smith, B.A. Peters (Eds.), Proceedings of the 36th Conference on Winter Simulation, Washington, DC., USA, 2004, pp. 1863–1870.
[152] O. Balci, J.D. Arthur, R.E. Nance, Accomplishing reuse with a simulation conceptual model, in: S.J. Mason, R.R. Hill, L. Monch, O. Rose, T. Jefferson, J.W. Fowler (Eds.), Proceedings of the 40th Conference on Winter Simulation, Miami, Florida, USA, 2008, pp. 959–965.
[153] E.M. Kanacilo, A. Verbraeck. Simulation services to support the control design of rail infrastructures, in: L.F. Perrone, F.P. Wieland, J. Liu, B.G. Lawson, D.M. Nicol, R.M. Fujimoto (Eds.), Proceedings of the 38th Conference on Winter Simulation, Monterey, California, 2006, pp. 1372–1379.
[154] J. Pichitlamken, S. Kajkamhaeng, P. Uthayopas. High performance spreadsheet simulation on a desktop grid, in: S.J. Mason, R.R. Hill, L. Monch, O. Rose, T. Jefferson, J.W. Fowler (Eds.), Proceedings of the 40th Conference on Winter Simulation, Miami, Florida, 2008, pp. 663–670.
[155] T. Zhang, L. Yun-sheng, Z. Ya-bing, Semantic web based simulation service customization and composition, in: Proceedings of the 7th IEEE International Conference on Computer and Information Technology (CIT 2007), IEEE Computer Society, 2007, pp. 235–240.
[156] L. Lacy, W. Gerber. Potential modeling and simulation applications of the web ontology language – OWL, in: R.G. Ingalls, M.D. Rossetti, J.S. Smith, B.A. Peters (Eds.), Proceedings of the 36th Conference on Winter Simulation, Washington, DC, USA, 2004, pp. 265–270.
[157] O. Lassila, J. Hendler, Embracing "Web 3.0", IEEE Internet Computing 11 (3) (2007) 90–93.
[158] J. Cardoso, The semantic web vision: where are we?, IEEE Intelligent Systems 22 (5) (2007) 84–88
[159] J.A. Miller, G. Baramidze, Simulation and the semantic web, in: M.E. Kuhl, N.M. Steiger, F.B. Armstrong, J.A. Joines (Eds.), Proceedings of the 37th Conference on Winter Simulation, Orlando, Florida, USA, 2005, pp. 2371–2377.
[160] T. Berners-Lee, J. Hendler, O. Lassila, The Semantic Web, Scientific American Magazine, 2001; Available from: <http://www.sciam.com/article.cfm?id=the-semantic-web&print=true>.
[161] V.P. Holmes, W.R. Johnson, D.J. Miller, Integrating metadata tools with the data services archive to provide web-based management of large-scale scientific simulation data, in: Proceedings of the 37th Annual Simulation Symposium, IEEE Computer Society, 2004, pp. 72–79.
[162] R.B. Araujo, F.M. Iwasaki, E.B. Pizzolato, A. Boukerche, Framework for 3D web-based visualization of HLA-compliant simulations, in: 3D Technologies for the World Wide Web Archive: Proceedings of the 13th International Symposium on 3D Web Technology, Los Angeles, California, USA, 2008, pp. 83–90.
[163] Web3D, The Web3D Consortium, 2008 [cited 2008 14th August]; Available from: <http://www.web3d.org/>.
[164] C.F. Salisbury, S.D. Farr, J.A. Moore, Web-based simulation visualization using Java3D, in: P.A. Farrington, H.B. Nembhard, D.T. Sturrock, G.W. Evans (Eds.), Proceedings of the 31st Conference on Winter Simulation: Simulation – A Bridge to the Future, Phoenix, Arizona, USA, 1999, p. 1425–1429.
[165] T. Kim, J. Lee, P. Fishwick, A two-stage modeling and simulation process for web-based modeling and simulation, ACM Transactions on Modeling and Computer Simulation (TOMACS) 12 (3) (2002) 230–248.