Simulation Modelling Practice and Theory 27 (2012) 47–64

Contents lists available at SciVerse ScienceDirect

Simulation Modelling Practice and Theory

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

# Using collaborative computing technologies to enable the sharing and integration of simulation services for product design

Hongwei Wang [a,c], Heming Zhang [b,*]

[a] School of Engineering, University of Portsmouth, Portsmouth PO1 3DJ, UK
[b] National CIMS Engineering Research Centre, Tsinghua University, Beijing 100084, PR China
[c] Engineering Design Centre, Cambridge University Engineering Department, Cambridge CB2 1PZ, UK

ARTICLE INFO

*Article history:*
Received 26 February 2011
Received in revised form 26 April 2012
Accepted 2 May 2012

*Keywords:*
Collaborative computing
Simulation services
Multidisciplinary simulation
Product design

ABSTRACT

Nowadays simulation is commonly used in engineering design for verifying design concepts before physical prototypes are produced. The simulation of complex products such as mechatronics in general involves a synergy of multiple traditional disciplinary areas and entails the collaborative work of a multidisciplinary team. A need thus arises for supporting the effective and efficient integration of subsystem models at simulation runtime and in a distributed environment. These models are generally created using different simulation tools and depend on the inputs from each other to perform numerical integration. As such, many issues need to be addressed, e.g. system modeling, the use of computing technologies, and the runtime interaction between models. In this paper, a service-oriented paradigm is presented which is underpinned by collaborative computing technologies to enable the provision of simulation models as services as well as the integration of these services for performing simulation tasks in product design. As well as the implementation of such a paradigm, a method for the interaction between models is in particular developed to achieve high accuracy for the simulation of design problems involving the solving of system equations. Preliminary evaluation work shows that the proposed paradigm underpinned by collaborative computing technologies is viable and have great potential in supporting collaborative simulation development in industry and the method for interaction control successfully achieves better accuracy compared with traditional methods.



## 1. Introduction

The design and development of complex engineering systems such as mechatronics generally requires a synergy of several traditional disciplinary areas, e.g. multi-body dynamics, system control, and hydraulics. A number of characteristics can be identified for the development process of such systems: firstly, the 'divide and conquer' philosophy is often recommended and a multidisciplinary development team is required; secondly, various Computer-Aided Engineering (CAE) tools are used to improve the effectiveness and efficiency of design and simulation technology is widely applied to verify and validate design solutions at an early stage of design; and thirdly the development is increasingly accomplished by outsourcing a significant number of components [1]. Thereby, the collaborative work of the members in a development team either within an organization or across several organizations and the integration of subsystems for different disciplines should be supported. An integrated and collaborative approach to the design and development of complex engineering systems is thus desirable for addressing these characteristics [2]. When applying CAE simulations in complex products development, Multidisciplinary

* Corresponding author. Address: Department of Automation, Tsinghua University, Beijing 100084, PR China.
*E-mail address:* hmz@mail.tsinghua.edu.cn (H. Zhang).

1569-190X/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.simpat.2012.05.002

Collaborative Simulation (MCS) is often used to integrate multiple computational models and simulation tools, as well as to support the collaboration between members of a development team.

MCS is a systematic approach to developing and performing accurate, effective, and efficient simulation, which emphasizes that system design should be optimized as a whole by taking into account the objectives and constraints from multiple disciplinary areas. For instance, simulations for mechanical components design and control system design are generally run together to identify the issues involved in the overall design [3]. Early work started in the 1990s and was focused on using advanced virtual prototyping simulation for design of mechanical systems [4]. In the context of mechatronic product design, MCS essentially involves constructing the mathematical models for a complex system and solving them by using one or more numerical solvers. Commercial CAE tools and bespoke simulation packages are widely applied in industry and as such pave the way for MCS implementation. There are two main ways for implementing MCS, namely the modeling using common language and the modular approach. The former focuses on describing the subsystems in a MCS problem using a common language from which system equations can be generated and solved using a single solver, see for example [3,5]. The latter, on the other hand, offers much flexibility by utilizing multiple tools to create models and integrating these models at simulation runtime, see for example [1,6–9]. The advantages of the former mainly include no need for system integration, high accuracy achieved by using a single solver. The latter has advantages of modularity, flexibility, and better support for collaboration. Although system integration needs to be addressed for the modular approach, it is still a good choice for the MCS for product design. First, modular modeling has been identified as one of the requirements for modern simulation environments [10]. Second, it well supports distributed collaboration and integration of models created using different languages or tools. Last but not least, designers working on different disciplines tend to have diverse preferences (e.g. 3D modeling, diagrams, and human-in-the-loop) on the simulation tools and this diversity is difficult to be addressed by a single tool.

It is therefore useful to develop an effective and efficient collaborative environment in which MCS can be undertaken by integrating models created for different disciplines and distributed on the Internet. The development of network and communication technologies opens the opportunity for supporting system integration and group collaboration for engineering applications [11]. For instance, many middleware technologies such as the Web, Web Services, the High Level Architecture (HLA), and semantic Web have been used for developing Web-based simulation environments [12]. These standards and technologies supply good support for the development of MCS environments. In this work, we focus on developing a MCS environment for product design and as such there are two main requirements raised in the first place. Firstly, this environment is aimed at supporting the collaborative development of distributed teams. This kind of collaboration can be either between members of a same organization or between teams from different organizations. In the latter case, source codes and model files may need to be kept confidential and only simulation results are allowed to be shared. Secondly, this environment should support the solving of equations in parallel and thus the runtime interaction between computational models also holds the key to system design and implementation. This requirement is overlooked in previous research [1,6,13,14] as the focus was on enabling the transfer of simulation data. This research is motivated by these issues and aims to go a step further beyond traditional Web-based simulation by developing a service-oriented paradigm in which simulation services for product design can be shared, found, and integrated.

The remainder of this paper is organized as follows. In Section 2, relevant research work is reviewed. In Section 3, MCS is analyzed in detail to achieve an understanding of the problem, as well as to provide insights into the development of simulation environments. In Section 4, the development of MCS system, in particular how collaborative computing technology is employed, is discussed. In Section 5, different ways for the runtime interaction between simulation models are discussed and an effective approach for interaction control is described. In Section 6, an engineering example is discussed to undertake preliminary evaluation on the proposed solution and the prototype system implemented. Finally the conclusions of this work are given in Section 7.

## 2. Related work

This research is focused on developing a MCS environment to support the collaborative and integrated design and development of complex engineering systems. Therefore previous work of interest includes collaborative product development, distributed simulation, Web-based simulation, application of collaborative computing in engineering, and the runtime interaction between models. Collaborative product development emphasizes the distribution of resources and the collaboration between people, aiming to address various issues, e.g. resources management, manufacturability, and maintenance, at an early stage of the design process [15]. Nowadays, simulation has become an important technique widely applied in product development. The distribution and integration of simulation models is an essential part of collaborative product development. Research in distributed simulation is actually as early as the research on networking standards, driven by requirements from the military sector [4]. Although Web technologies and distributed simulation have grown up largely independently, it is predicted that they will become synonymous in the future due to the former's influence on the latter [12]. An important and heavily researched standard for distributed simulation is the HLA which is a generalization and extension of the Distributed Interactive Simulation (DIS) protocols and the Aggregate Level Simulation Protocol (ALSP) [12].

Web-based simulation is a broad research area and mainly refers to the use of Web technologies to support simulation development and running. Kuljis and Paul stated that the pressure imposed by the proliferation of Web uses is high that it has forced the simulation community to migrate to the Web to remain "alive" [16]. Byrne et al. identified a number of advantages for Web-based simulation, namely ease of use, collaboration, licence and deployment models, model reuse, cross

platform capability, controlled access, wide availability, integration and interoperability. Meanwhile, it also has some disadvantages, e.g. loss in speed, Graphical User Interface (GUI) limitation, security vulnerability, Web-based simulation application stability [12]. They further classified Web-based simulation systems into three main categories, namely local simulation & visualization, remote simulation & visualization, and hybrid simulation & visualization, in terms of where simulation & visualization is performed [12]. Kuljis and Paul identified the main application domains of Web-based simulation, namely military applications, scientific applications, education and training, and manufacturing [16]. In particular, the first and last domains are related to engineering. Due to the requirement for distributed collaboration, Web-based simulation has been applied in engineering and all the applications fall into the three categories identified in [12]. For example, an ontology-based Web simulation system is developed for hydrodynamic modeling, which wraps legacy codes in the server side and provides simulation functionality to the clients [17]. Using a similar framework, a problem solving environment was developed by Cheng and Fen to perform computing in the server side and transfer results to the clients [18]. Han developed a Web-based simulation system for multi-body systems which enables users to create models and run simulations through the GUI implemented using Java Applet [19].

With the rapid advancement of information and communication technologies, technologies for system integration and group collaboration have been developed and applied in different application domains, including architecture, engineering, construction, and facilities management. Specifically, system integration approaches include Web-based systems, distributed objects/components, software agents, Web Services and Semantic Web, and collaboration can be supported by using Web-based technology, agent-based technology, collaborative virtual environments, and virtual organizations [11]. These enabling middleware technologies for Web-based simulation also have impact on the development of MCS. As summarized in Byrne et al. [12], the most commonly used technologies include Web, Web Services, the HLA, Semantic Web, Enterprise Java Beans (EJB), Common Object Request Broker Architecture (CORBA), and Distributed Component Object Model (COM/DCOM). For instance, Ryu developed a Web-based distributed simulation system [9]. Senin et al. envisioned a scenario of undertaking design simulations by integrating services published via CORBA [6]. Zhang developed a solution which uses the HLA to manage multiple simulation agents [20].

Web services have attracted much attention due to the support it offers for cross-platform and language-independent interoperability. For example, Johansson and Krus studied the integration of computational models that are accessible as Web services [13]. Wainer et al. encapsulated resources for Discrete Event System specification (DEVS) as Web services in the CD++ toolkit, aiming at interoperating different DEVS implementations [21]. The provision of models based on Modelica as Web services is also studied and implemented in a prototype system [22]. Gyimesi proposed to publish generic simulation models as Web services to support discrete event simulation [23]. Research work has also been done to assembly computational models provided as services to perform analysis tasks for evaluating design solutions [1,6]. Apart from the middleware technologies mentioned above, the emerging Web computing paradigms, e.g. Web 2.0 (e.g. blog and wiki), Web 3.0 (Semantic Web), Grid, and Service-Oriented Architecture (SOA), also hold great potential for Web-based simulation [12]. In terms of utilizing SOA for Web-based simulation, the focus is on the use of Web Services for distributed simulation and in this sense SOA, Grid, and Web Service have been closely aligned [12].

Research work on the application of SOA and Service Oriented Computing (SOC) has been identified as a promising area albeit publications in this area are still not many. Tsai et al. developed a service-oriented distributed modeling and simulation framework to support the rapid development and deployment of large-scale distributed systems such as network-centric and system-of-systems applications [24]. This work can be viewed as early endeavors in this direction and the framework has features such as a modeling and specification language, dynamic model checking, automatic code generation from specification, a platform builder, and multi-agent based simulation for easy re-configuration and re-composition. Systems with such a framework can provide all-round support for users from specification to running [24]. Tsai et al. proposed an ontology-based service-oriented simulation with Microsoft Robotics Studio (MRS) which was aimed at offering support for applying SOA to embedded systems [14]. Specifically, simulation services are offered by MRS and ontology model is used for the effective composition of these services. There are two important concepts in SOC applications, namely dependability and Quality of Service (QoS) [25]. The latter is more a perspective from the user's point of view while dependability attributes are more a perspective from the designer's point of view [25]. Research work on SOC systems includes system validation based on service composition languages, QoS ontology and ranking algorithms for the evaluation of Web services, model-based monitoring and policy enhancement, and applications in the business and defence sections [25]. This is a relatively new area and further work is required to explore the concepts, methods, theories, and technologies.

The middleware technologies discussed above offer good support to resolve the issues such as data distribution, calling of remote methods, and modularity of system architecture. The HLA, as a specific standard for distributed simulation, provides further support for the synchronization of the subsystems involved in a simulation. On the basis of these technologies and standards, other issues still need to be addressed for the implementation of an effective MCS environment for product design, e.g. the analysis of factors influencing simulation performance [1], the high-level modeling method for complex systems [26]. Moreover, the runtime interaction between computational models also has paramount importance as it directly influences simulation performance as well as the way in which these models should be made accessible on the Internet. Research in this area is much less compared with the utilization of middleware technologies for MCS implementation. Kübler and Schiehlen proposed two methods for simulators coupling, namely the iterative and the non-iterative way, and pointed out that the stability of the non-iterative approach for problems with algebraic loops cannot be guaranteed [7,27]. Ryu proposed an enhanced glue algorithm for data exchanging and applied it to a Web-based distributed simulation

system [9]. The modular approach to MCS has been studied by many researchers [1,5,6,8,13]. Nonetheless, much further work is required to research the interaction between models at runtime to achieve improved accuracy especially for complex systems such as those in product design.

In summary, the middleware technologies discussed above have different advantages and the development of MCS environments requires further work to be done once one or more technologies have been chosen for the implementation. The simulation of complex systems such as mechatronics often requires the involvement of human operators or devices, making the capability of managing and scheduling discrete events an advantage. The HLA is thus suitable for applications with such requirements, but methods for interaction control still need to be developed to achieve high accuracy for complex systems modeled using differential equations and solved using numerical integration. The effective and efficient interaction between subsystems is very important for the performance of a simulation especially for applications running on the Internet and as such it requires the utilization of enabling technologies such as Web, Web Services and CORBA. In addition to being easy to develop, Web Services also support flexible system integration (fits naturally with the modular MCS method) whilst supporting cross-platform and language-independent interoperability for various applications running on the Internet. Semantic Web is a very powerful and promising technology for applications that require complex and intelligent integration methods so that it is more appropriate to be applied at a later stage when MCS systems are developed and applied in industry. Research on using SOA in constructing MCS environments is still handful. The modularity, flexibility, interoperability, and support for collaboration offered by SOA makes it fits well with the purpose of development a MCS environment for product design. A service-oriented framework has been developed in this work and will be described in detail in the following sections together with a runtime interaction method.

## 3. Understanding the multidisciplinary collaborative simulation problem

### 3.1. Structure of a collaborative simulation problem

Market needs are transformed to specific information that can be used to manufacture a product during an engineering design process. A lot of issues, e.g. reliability, manufacturability, application of new technologies, attention to new legislations, etc., need to be taken into account throughout this process. The integrated and collaborative design paradigm is therefore useful for such a complex process, as discussed in [2,15]. From the perspective of a systematic approach to design, function, behavior and form are three important factors for design artifacts [28]. Market needs are transformed into the detailed functional requirements that will be fulfilled by the form of a specific design. The behavior of design solutions needs to be studied and evaluated by either experiments or simulations to predict the performance of a real product. Simulation technology is widely used in product design to achieve improved efficiency in terms of both time and cost. The criteria for evaluating simulation methods and paradigms include accuracy, efficiency of development and execution, and complexity.

The design and development of complex products requires a systematic approach and involves an iterative process which entails multidisciplinary collaboration for both the development of design solutions and the running of simulations. The simulation for complex products such as mechatronics generally involves multi-body dynamics models, control system models, and drive or servo system models. The assembly of these models is necessary to accurately predict the performance of a design solution. Furthermore, the involvement of either a human operator or a physical component in a simulation may also be desirable in some applications. Therefore, it is helpful to first analyze the structure of a MCS problem so that the elements and their interactions can be identified. As shown in Fig. 1, a MCS problem generally consists of computational models in different locations (e.g. sites A–C), engineers who work on the development of the models and may also interact with the running process of a simulation, and instruments which represent physical prototypes. Specifically, engineers, when necessary, obtain data from the simulation process and give control signals to the models during runtime. Instruments process the data obtained from the simulation process and perform operations based on the data. The computational models are created based on the physical laws and principles of specific disciplines and work as a whole to simulate the operation process of a real product. The exchange of data between these models is underpinned by the simulation system.

For a computational model $i$, vector $X_i$ denotes its design variables, vector $G_i$ denotes the output variables, and $Y_{ij}$ represents the transfer of data from it to model $j$. Computational models are created either by using off-the-shelf Modeling and Simulation (M&S) tools or by developing bespoke packages. The accuracy of simulation is actually determined by a number of factors, e.g. the accuracy of the models, the accuracy of the integration algorithms used for the models, as well as the accuracy of the interactions between models. The complexity and efficiency of a MCS problem often depends on the coupling between the computational models involved. Moreover, there are some circumstances under which the interactions between the models and engineers (and instruments) also play an important role. Consequently, the runtime interaction of a simulation running on the Internet is a major issue for improving the performance of MCS problems, and raises the need of studying the interaction mechanisms in MCS. Specifically, the interactions between models and engineers (instruments) are data-centric where the interactions between models are more complex and have a big influence on the numerical calculation process.

### 3.2. Supporting MCS development in a distributed environment

As analyzed in the last section, MCS essentially involves human operator, physical components, and virtual components (models), and the interactions between models hold the key to MCS. In the context of distributed and collaborative product

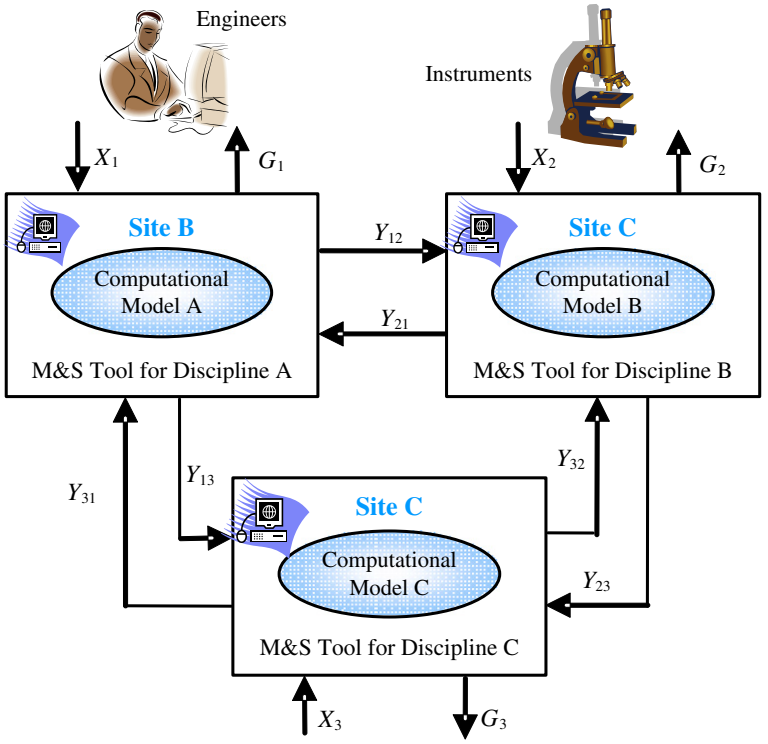


**Fig. 1.** The structure of a MCS problem.

development, the models, instruments, and operators can be distributed on the Internet and the effective interactions between them should be well supported by a MCS environment. Such an environment can be offered by developing a computer tool using collaborative computing technologies. A number of requirements have been identified for such a computer tool [1]:

- To enable distributed design teams to decompose complex design problems and evaluate simulation results in an integrated environment.
- To support the high-level modeling of a simulated system and generate the necessary codes for the running of simulations which could be built by reusing legacy simulation models and codes.
- To provide interfaces for the post-processing of simulation results, and the generation of simulation reports.
- To effectively synchronize the simulation advancement so as to guarantee accurate results, with different events influencing each other in the correct sequence.
- To signpost the simulation development by the provision of a process management facility.
- To enhance the scalability of the platform and address security issues for effective operation.

These are actually high-level requirements specific to the MCS system. Moreover, a number of detailed issues also need to be addressed to implement a useful MCS system. A map of these issues is shown at the bottom of Fig. 2 where a service-oriented paradigm is depicted. Specifically, these issues are listed and explained as follows:

- *Visualization of design solutions and simulation results.* In the context of MCS for product design, visualization should be used to enable designers and analysts to view 3D models, create diagrams for system model, monitor simulation configuration and running, and perform post-processing.
- *A high-level modeling scheme.* As discussed in the previous sections, MCS is used for the simulation of complex systems which generally involve several subsystems. Therefore, a scheme is needed to describe the system model in detail by identifying the subsystems and their relationships. Moreover, details about available services and the accessing (related to collaborative computing technologies used) of these services should also be included in the scheme as the MCS discussed in this work is aimed at integrating simulation services for product design.
- *Utilization of collaborative computing technologies.* MCS in a distributed environment needs to be underpinned by these technologies and as such it is important to choose the ones that are useful and easy to implement.
- *The scheduling of discrete events to make them happen in the right sequence.* Uncertainty is inevitable in network communication. In distributed simulation, any operation (e.g. a simulation command and a request of data exchange) is an event and a simulation process depends on that the events take place in a logical order.

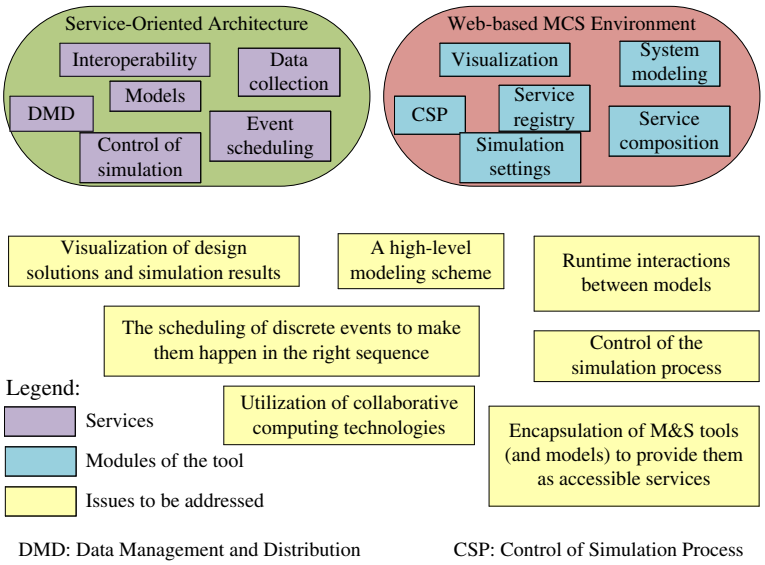


**Fig. 2.** A service-oriented paradigm for the implementation of MCS system for product design.

- *Runtime interactions between models.* The MCS problems in product design essentially involve the numerical integration of subsystem models in parallel and the integration of any one model is dependent on the data received from other models at intervals. The accuracy of simulation is thus greatly influenced by the intervals at which data exchange takes place. Therefore, a study of the runtime interactions holds the key to achieving high accuracy.
- *Control of the simulation process.* Even though a MCS system involves multiple models running in parallel, the functionality for starting, pausing, and stopping a simulation is still desirable. Moreover, designers and analysts should also be able to send commands/data to the simulation process to implement human-in-the-loop simulation.
- *Encapsulation of M&S tools (and models) to provide them as accessible services.* A key feature of the service-oriented paradigm is that subsystem models can be developed using any kind of language and run in anywhere on the Internet. Therefore, a method must be developed to encapsulate these models as accessible services to which behaviors (e.g. advancement of simulation time, and obtain/generate data) of the models are delegated.

In summary, the design and development of a MCS environment is complicated and involves many issues. These issues can be again analyzed on the basis of the participants and components in a MCS problem. The user of such an environment in general works on creating system model, finding and composing services to perform simulation, controlling the simulation process, processing results, and updating models. The models in a MCS environment run separately and strongly depend on the inputs from others. The inputs should be received at the correct time and outputs also need to be received by subscribers at the correct time as the MCS problems in product design mainly involve time integration in parallel. Therefore, a computing paradigm is desirable to develop a modular, flexible, and effective solution for MCS. In this paper, the focus is on the development of a service-oriented paradigm in particular on the collaborative computing technologies used and the structure employed by the paradigm. Moreover, a method for runtime interaction is also presented, which holds the key to achieving good simulation accuracy. Solutions for other issues are beyond the scope this paper and have been published elsewhere, e.g. the high-level modeling [26], the development of software components for the implementation of a prototype MCS system [1].

## 4. Using collaborative computing technology to support collaborative simulation

### *4.1. A service-oriented paradigm for the implementation of MCS systems*

In this work, a service-oriented paradigm is developed to fulfill the requirements and address the issues discussed in Section 3.2. Service-Oriented Architecture (SOA) is a new concept for the development of flexible and extensible software systems based on specific businesses, which has great capacity in supporting distributed computing. A procedural style of programming in simulation was mainly used prior to the emergence of Object-Oriented Architecture (OOA) based simulation and SOA-based simulation, which has a big drawback. That is, procedural changes are the only approach for models changes and vendors have no way to hide implementation details, either being forced to give access to source codes or restrict access to these features [12]. A key difference between OOA and SOA is that the former provides an abstraction of data at a class level whereas the latter provides an abstraction of data at a business level. Another difference is that SOA has a focus on

the loose-coupling of services and thus allows for more agility [12]. Attention has been paid to the application of SOA to simulation although real applications are still handful as this is a relatively new concept [14]. Nonetheless, the study of encapsulating legacy models/codes as Web services have been done by many researchers, see for example [1,6,13,21–23], which paves the way for developing a SOA-based simulation environment. The proposed service-oriented paradigm is shown in Fig. 2.

There are three parts in such a paradigm, namely a SOA-based environment underpinning distributed computing and interoperability, a Web-based simulation environment supporting the interactions between users and the MCS system, and the specific issues to be addressed by the other two parts. Specifically, the SOA-based environment offers a set of services to address the issues such as data collection/management/distribution, controlling the simulation process, interoperability, and encapsulation of models. A Web-based simulation offers interfaces through which users can create system model, find services from a registry, perform simulation via service composition, specify simulation setting, and control the simulation process. Discussion on these issues has been done in Section 3.2 and these issues will be addressed by the other two parts in the paradigm. Clearly, the key issue in this paradigm is the development, provision, and integration of various services, with the support of collaborative computing technologies. Detailed solutions for this issue are discussed in Section 4.2 through to Section 4.4.

### 4.2. Synchronizing the simulation advancement

As discussed above, a MCS system generally includes elements such as computational models, human operators, and instruments. These elements interact with each other during the running of a simulation, and perform different tasks such as performing calculation or processing data. The interactions between them can be regarded as a process with different events influencing each other and a prerequisite of accurate simulation is that the events take place in the correct sequence. A mechanism is therefore desirable to manage the execution of such a process, i.e. synchronizing the simulation advancement. The services for data collection/management/distribution and controlling the simulation process are specifically developed to address these issues. In a distributed simulation, the updating of data, the receiving of updated data, and the sending of commands are all events. Hence, a technology that can effectively schedule these events is required. In this work, the High-Level Architecture (HLA) is employed to implement and provide these services. The HLA is a well-established standard for distributed simulation on the basis of a combination of the advantages of its predecessors such as the DIS protocol and ALSP. It offers management capabilities for various distributed simulation issues such as time advancement, data distribution, ownership of data, and data management. In particular, it supports both time-step based simulation and the simulation based on discrete events, and thus can be utilized to synchronize the simulation advancement for MCS problems. The implementation of the services for scheduling events and distributing data is done by using a framework based on the HLA, as shown in Fig. 3.

The core of such a framework is the HLA-based simulation management and the programming for this part is based on the Runtime Infrastructure (RTI) which is an enabling software tool for the HLA and implements the specific interfaces. A simulation running with such a framework is called a 'federation' which consists of a number of interoperating components called 'federates'. The roles of external components, i.e. computational models, human operators, instruments, and data collector, are delegated to these 'federates'. They do not actually exchange data directly with each other, but instead communicate with the RTI that serves as the communication bus as shown in Fig. 3. At runtime, the RTI is responsible for the data/time management for the whole 'federation' and mainly performs three tasks: (1) getting information about the current logical time of all the 'federates' and calculating which logical time they can go to next; (2) receiving updates of data from, and

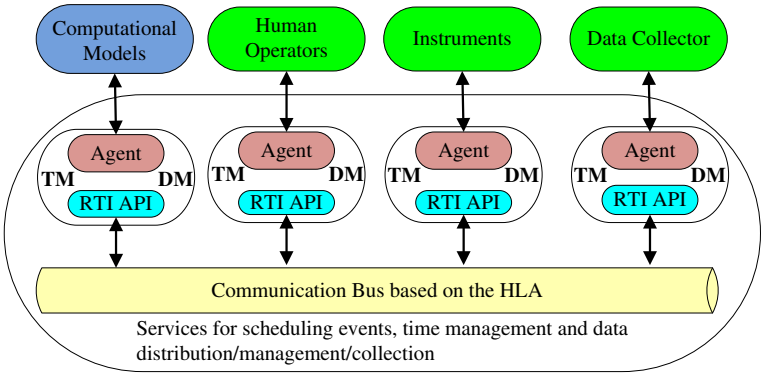


**Fig. 3.** Implementation of services for TM and DM based on the HLA.

**Table 1**
Time management strategies used and data exchanged in HLA-based simulation for MCS.

| Element | Time Mgt. strategy | Data exchanged |
|---|---|---|
| Computational models | Time control and time constrained | Subscribing input data needed by the numerical integration process and publish output data |
| Human operators | No | Sending control signals and getting data from the Graphical User Interface (GUI) |
| Instruments | Time control and time constrained | Subscribing input signal and returning the data measured or generated |
| Data collector | Time control and time constrained | Subscribing and storing all the data exchanged within a HLA-based simulation federation |

then forwarding them to, the 'federates' that express interests by explicitly subscribing and publishing the specific data objects; and (3) deciding whether a component has the right to update a data object.

In this way, the decisions are all made by the RTI automatically and all the 'federates' involved only need to exchange data with, and send time advancement requests to, the RTI. Therefore, the time management strategy and data subscription/publishing of each component should be specified before a simulation starts. There are two time management strategies in the HLA, namely time constrained and time control. The former means that the time advancement of a 'federate' is constrained by that of others, i.e. it must wait for others before moving to the next step. The latter means that the current status of a 'federate' will influence the advancement of others' time, i.e. other components need to wait for this component before moving to the next step. Actually, a 'federate' can have both of the two strategies and the choices on the strategies are dependent on the roles of different 'federates'. Table 1 shows the time management strategies used for the four components, together with data exchanged by them during a simulation. In particular, the human operator only sends commands to other components and this can happen any time during a simulation, so it is not constrained by, and has no control over, the logical time of others.

On the 'federate' side, each 'federate' has a life-cycle that consists of the initialization, joining, registration, running, and resignation stages. Specifically, at the initialization stage the RTI creates a 'federation' and 'federates' complete the construction of data and objects and the establishment of connection with the RTI. At the joining stage, a 'federate' sends the request of joining a 'federation' to the RTI which will wait until having got all the requests from the expected 'federates' and proceed to start next stage. The registration stage involves the setting of time management strategies and the declaration to the RTI about these strategies together with the requests for publishing/subscribing specific data objects. The running stage specifically refers to the running of a simulation, which mainly deals with time advancement and data exchange. In the resignation stage, a 'federate' sends a request to inform the RTI that it is ready to leave a 'federation' and all the connections will be closed once this request is approved. The provision of simulation management based on the HLA as services can be done in two ways. The first way is to use the RTI which offers Web Services based API, e.g. the recent version of pRTI [29]. The second way is the encapsulate legacy RTIs as Web services by doing a bit extra programming. In this work, the second method is used as codes in our previous work on HLA-based simulation can be reused. The programming for 'federates' is complicated and in practice it is done by partially generating codes based on a template. In Fig. 3, 'agents' play the role of 'federates' in a HLA-based simulation and at the same time they also communicate with external computational models to complete the task of system integration. Detailed discussion on the 'agents' will be given in Section 4.4. Human operators mainly work with the Web-based simulation tool (the second part of the service-oriented paradigm) which also plays the role of a data collector to interface with the HLA-based simulation for collecting data, sending commands, and receiving simulation status. As highlighted in the figure, models are encapsulated as Web services, which can be developed by any provider and integrated with the services for simulation management. Details about the encapsulation will be discussed in the next section.

### *4.3. Provision of simulation models as services*

There are a number of advantages to provide computational models as services accessible on the Internet. Firstly, the M&S capacities can be shared and utilized by authorized clients. Secondly, details about a model can be kept confidential with only data transferred during a simulation process. Thirdly, the integration between computational models in a distributed environment is enabled, and thus provides support for the collaborative development of simulations. The interoperability offered by HLA-based collaborative simulation is not enough for MCS running on the Internet, though the synchronization of simulations running separately can be achieved. This is due to a few reasons as follows: (1) the HLA has a particular focus on the interoperability between its components and thus overlooks sharing these components with other systems; and (2) the HLA is not a widely used standard so it is hard for its components to be accessed by various clients. Nevertheless, components in a HLA-based simulation can communicate with external computational resources during a simulation. For example, in Fig. 3 the simulation of computational models can actually be performed by external services. As well as having open interfaces and various implementation techniques, Web Services technology offers good support for distributed computing on the Internet and thus is able to improve the interoperability of MCS. By using Web services, remote methods and data can be accessed by authorized clients anywhere on the Internet, which achieves good interoperability. A method for providing models as Web services is shown in Fig. 4.

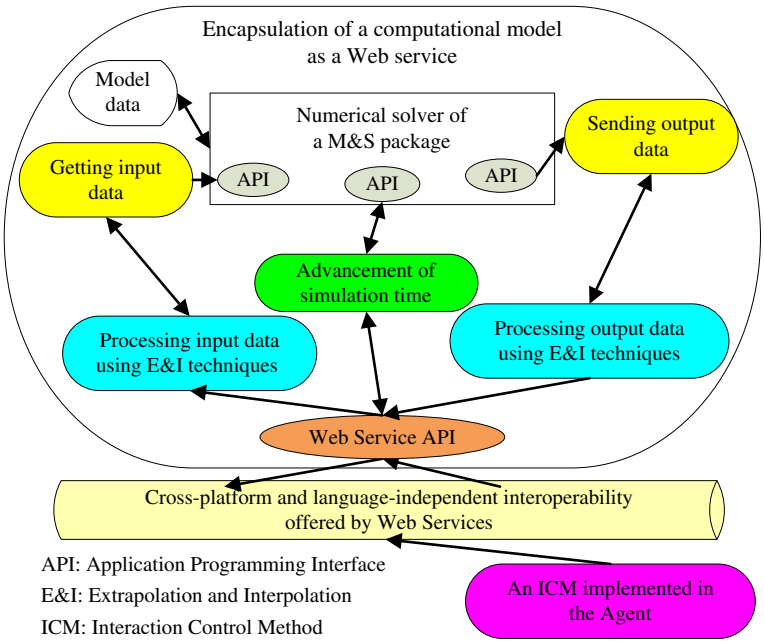


**Fig. 4.** A method for the encapsulation of computational models as Web services accessible on the Internet.

This encapsulation structure mainly has two functions, namely the processing of Input/Output (I/O) data and the advancement of simulation time. To implement these functions, the Application Programming Interfaces (APIs) of M&S packages need to be utilized for developing programs for controlling the simulation advancement and accessing simulation data. The I/O data are generally processed by using extrapolation and interpolation methods because data exchange only takes place at some intervals in a distributed simulation but data that are not available at the time between these intervals are often required by the numerical integration processes of the M&S packages. When these functions are realized, the API of the specific Web Service implementation technique can be used to interface the models/solvers so that authorized remote clients can interoperate with them. As discussed in Section 2, some work has been done on the encapsulation of legacy models/codes as Web services and as such there are a number of methods for the encapsulation [1,6,13,21–23]. The choice of these methods is dependent on the specific M&S packages involved. Moreover, Web Services is a popular research area and both commercial and open-source middleware technologies are now available for its design and development. In this work, the programming work for model encapsulation is done using Java through the Java Native Interface (JNI) API. The middleware used for Web Services programming is Axis which is an open-source platform managed by the Apache Software Foundation [30]. Four main functions are developed in the Web Services encapsulation program:

- *A function for initialization*. This function initializes the simulation engine in response to a request (including initial values of variables, starting and ending time of simulation, time step) from a remote client.
- *A function for running simulation*. This function receives input data from remote service clients, executes the simulation for a while, and stores the data as well as the simulation results for their usage at a later stage in the simulation process.
- *A function for getting simulation results*. This function is defined to get the current values of variables and send them back to service clients as output data.
- *A function for running the simulation to a specific time*. This function is used to start the numerical integration process for a specified period (the advancement of simulation time), get the current simulation time that will be sent to remote clients, and send commands to start, pause, and stop the simulation process.

As shown in Fig. 4, Web Services technology offers cross-platform and language-independent interoperability which enables the flexibility and modularity of this solution. In this sense, models developed using any M&S tools or languages can be assessed by remote clients developed using any techniques or languages and running on any platforms. The remote clients do not necessarily have to know the technical details of a service, but instead only need to interpret the Web Service Description Language (WSDL) document which describes details about the specific functions to call and the specific data involved in these functions. In the Axis platform, WSDL is well supported and the bi-directional transformation between WSDL and Java code is enabled. That is, A WSDL schema can be generated on the basis of a piece of Java code and vice versa. The WSDL schema for the model encapsulation services discussed in this section is shown in Fig. 5 (only core information is shown for the sake of brevity). There are totally five parts in the schema, namely definition of data types, definition of data for interaction,

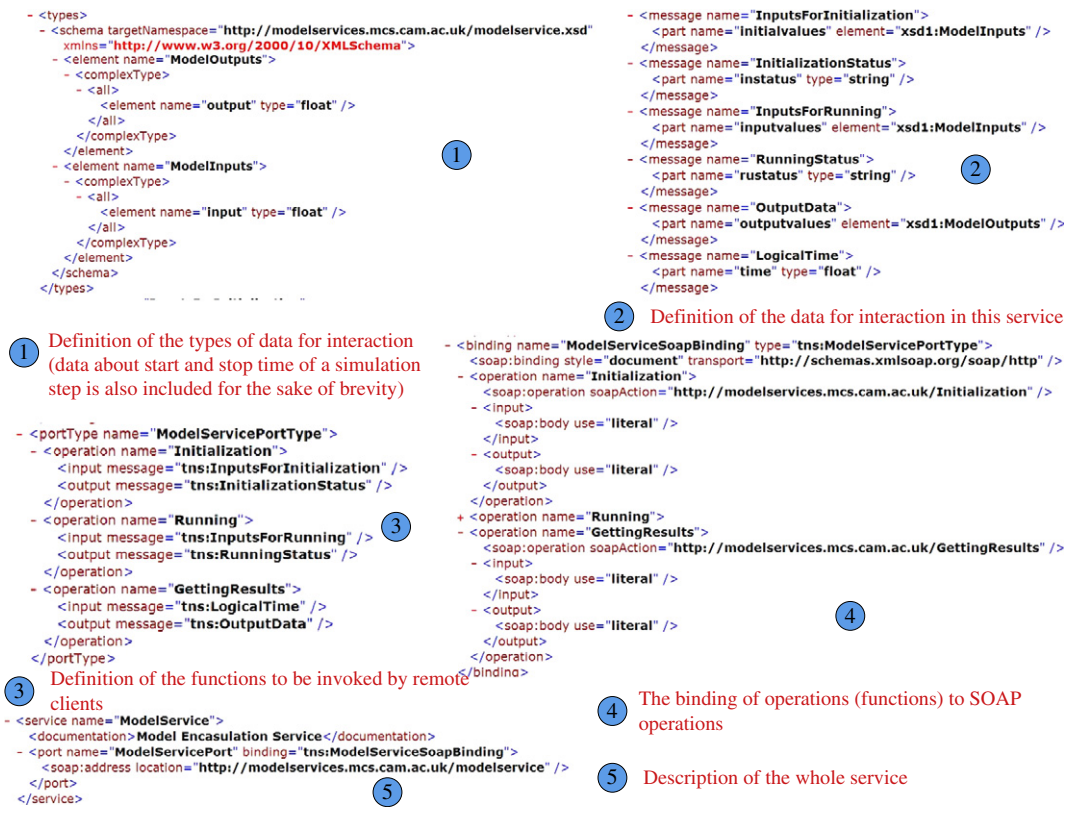


**Fig. 5.** The WSDL schema for the model encapsulation Web service.

definition of operations (functions to be called), binding of operations to SOAP operations (SOAP is a protocol for data transfer in Web Services), and the description of the service. Through this schema, the remote clients are informed how to assess a service by calling its operations which correspond to the functions of a Java Class and where the four functions discussed in the last paragraph are implemented. It is shown in Fig. 4 that an interaction control method is implemented in the agent to communicate with a service to initiate the running of the encapsulated model and thus coordinate the simulation advancement for all the models.

### *4.4. System integration based on agent technology*

As discussed above, both the HLA and Web Services are utilized in this work to implement a service-oriented paradigm in which designers, analysts, IT engineers can work together to build up simulations for product design by composing services. Hence, it is necessary to develop programs to integrate a HLA simulation federation and Web services on the Internet. This programming work is not trivial as the simulations in product design essentially involve many numbers of iteration and design changes always happen, although a simple connection between the two is not difficult. This work also causes problem for the target users of a MCS system, i.e. design engineers, who are in general not familiar with distributed computing topics. As such a MCS system should hide the technical details of the collaborative computing technologies as much as possible. Referring back to the discussions in the previous sections, the agent shown in Figs. 3 and 4 is developed to integrate the collaborative computing technologies used, as well as to mitigate the effect of the changes in these technologies. Agent-based systems have been widely studied recently, which have advantages such as a modular structure, intelligent information processing, and the autonomous way of working [11]. It has three main roles, namely a 'federate' in a HLA-based simulation, a remote client of a Web service that encapsulates a model, and a component implementing the interaction control method.

Each agent actually plays the role of a simulation model in a HLA-based simulation and thus it needs to exchange data and coordinate simulation advancement with the RTI. In this way, it can get data from other models via the RTI and decide to which time the next simulation step can go. Once this information is available, it plays the role of a remote client of a Web service by sending the data from other models together with information about simulation time to the service and obtaining simulation results. These results are then sent to other agents, again via the RTI. Thus the interoperability between models is implemented. This process explains how the agent works in a MCS system. As to the programming for these agents, another solution is developed to generate codes based on a template. As the programs for both Web Services and

HLA 'federates' are very structured, a template can therefore be used to define the information about the interaction between models. With that piece of information, Java codes can be generated. As discussed in Section 4.3, the programming for Web Services encapsulation mainly involves adding details to the functions of a Java Class. This customization fits better with the programming of the HLA 'federates' as they only deal with the exchange of data rather than generating data even though they are more complicated. The code generation is beyond the scope of this paper and more details have been published elsewhere [26]. However, the interaction control method is not implemented by the agent implemented in the previous work. The interaction control method holds the key to simulation accuracy in MCS and has been developed and added to the agent.

## 5. An effective runtime interaction approach to integrating simulation services

### 5.1. Solving of collaborative simulation problems using multiple solvers

Generally, the simulation of electro-mechanical system involves the construction of Differential Equations (DEs) and Differential Algebraic Equations (DAEs) as the subsystems such as multi-body dynamics, control system, and hydraulic system, can all be described using these equations [5]. The first assumption made in this research is that most MCS problems can be modeled using a set of DEs or DAEs. As discussion in Section 1, A MCS problem can be solved in two ways, namely the use of common language and the modular MCS. The first method utilizes unified theory to construct system model which then will be transformed to a set of DEs or DAEs. The latter, on the other hand, aims to solve a MCS problem by combining the functionalities of M&S tools for different disciplines, as well as to support the divisions of tasks so that better collaboration can be achieved. As a scenario of sharing and integrating distributed simulation services to perform MCS tasks is envisioned in this research, the focus is mainly on how to improve the performance of solving MCS problems using multiple solvers. A second assumption is that there are no algebraic loops in the MCS model studied because special treatment, e.g. using iterative algorithms [7], is needed for problems that have algebraic loops and is out of the scope of this paper.

The HLA can help ensure that discrete events take place in the right sequence so that the interaction between components is accurate in terms of simulation time, i.e. synchronization of all components. Moreover, Web Services enables the interoperability of computational models and the modularity of the structure of a MCS problem. All these collaborative computing technologies work together to achieve dynamic and flexible integration of computational models during runtime so that a complex simulation problem can be solved by a synergy of multiple disciplinary areas effectively. However, the synchronization enabled using the HLA can only ensure the rightness of interaction at a macro level, i.e. interaction received at time $T_1$ is not mistakenly regarded as that of time $T_2$ where $T_1$ and $T_2$ are two time points updated by the RTI during a simulation. Actually, different commercial M&S packages have different numerical integration methods for solving DEs and DAEs, and the numerical integration process of any model can largely affect the accuracy of MCS. Therefore, the accuracy of simulation should also be ensured in terms of numerical calculation. In this section, the mechanism of solving MCS using multiple solvers will be analyzed.

As shown in Fig. 6, assume that two models A and B are created using different M&S packages (say PA and PB) and exchange data with each other during a simulation. The data exchange happens N times during the simulation and thus simulation time is divided into N portions each of which is called a macro step. The solving (numerical integration) of model A and B is done separately by the solvers of PA and PB using different methods (either fixed step or variable step). Using the APIs of PA and PB, an external routine is able to make the simulation to run for a while with a step specified as a reference value. When data exchange happens at macro steps, the numerical integration of each model actually has continued for a few micro steps (the number of micro steps is determined by the specific M&S package used). At macro steps, each model will update its input data and process the data using interpolation before next macro step starts. It can be proved that if the macro step is small enough, the numerical calculation as shown in Fig. 6 can find solutions for the differential equations of A and B with acceptable accuracy [31]. Two methods, namely the Parallel Interaction Method (PIM) and the Staggered Interaction Method (SIM), have been developed for the model interaction of such a process [32,33], as shown in Fig. 6. The only difference between the two methods is that whether one of them is allowed to do time integration one macro step faster than the other. However, there are still some drawbacks in these two methods and an effective method is needed for MCS.

### 5.2. An effective interaction method

There are a number of drawbacks in the PIM and SIM. Firstly, a macro step needs to be specified for the simulation, which is error-prone and requires much expertise about the numerical integration of differential equations. Secondly, different simulation problems may have different requirements in terms of accuracy, speed, and numerical stability, and thus it is hard to find out a macro step that fits well with most applications. Thirdly, the integration processes of commercial M&S packages tend to be complex for achieving good performance. For instance, MSC Adams uses the variable-step BDF method for some simulations [34]. In variable-step numerical integration methods, an initial big step is often chosen first and the integration errors will subsequently be evaluated: if the error is too big, a smaller integration step will be used instead until convergent calculation is achieved. Therefore, it is difficult to inform the solver what step to use for time integration. Thirdly, different

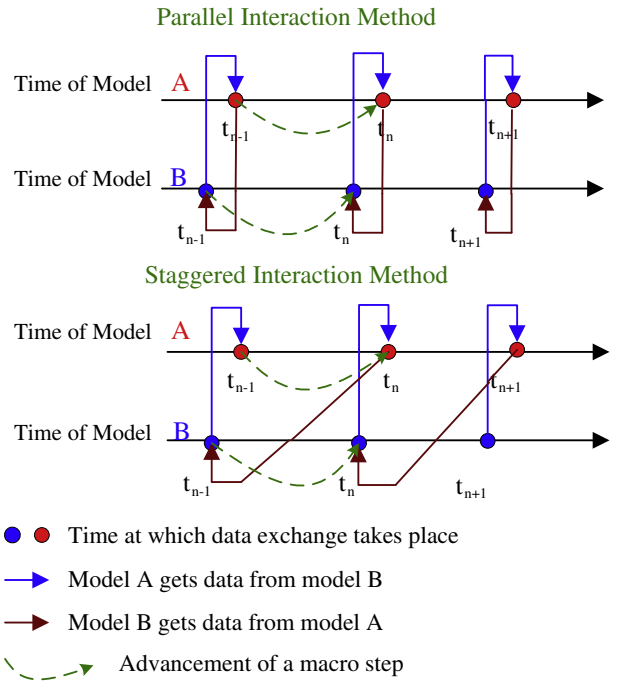


**Fig. 6.** Two interaction methods with data exchanged at macro steps.

solvers may use different step sizes and the use of improper step size for any model will significantly affect the accuracy of a simulation.

Starting from this point, an effective interaction method, namely the Crossed Interaction Method (CIM), is proposed to overcome the difficulties met by the PIM and SIM. The basic idea is that no matter how many times it has tried to find a proper step size for a solvable problem, a solver will ultimately achieve convergent calculation with a satisfactory step size. If data exchange happens at this step, then numerical integration is more accurate. The time step at which numerical convergence is achieved can be provided by commercial M&S packages, e.g. using the 'ssGetTimeOfLastOutput' function of Matlab Simulink [35] and the 'TIMGET' function of MSC. Adams [34]. The interaction between two models through the CIM is shown in Fig. 7. In the CIM, Model A reaches time $t_n$ first and updates output data, then Model B initiates time integration for consecutive two times (each of which achieves convergence of numerical calculation) until reaching time $t_n$. When Model is B is performing time integration, it uses the data updated by Model A at time $t_n$. A flowchart for such a process is shown in Fig. 8. To make all the models start numerical integration at the right order, the model having the smallest current simulation time always starts simulation first until numerical convergence is achieved, and then data exchange is performed. During the simulation, the method shown in the flowchart is implemented in the agents. Specifically, an agent accesses the service and

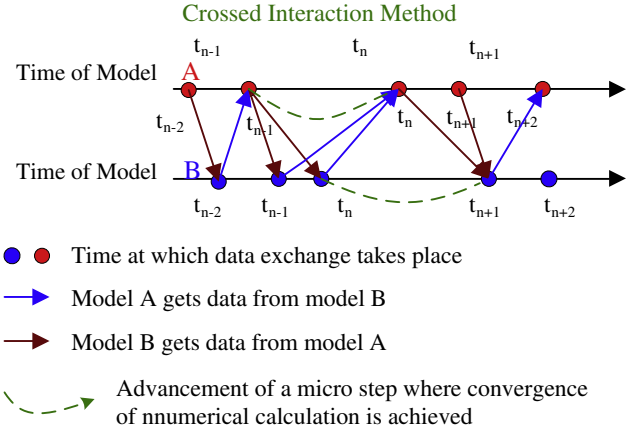


**Fig. 7.** An interaction method with data exchanged at micro steps.

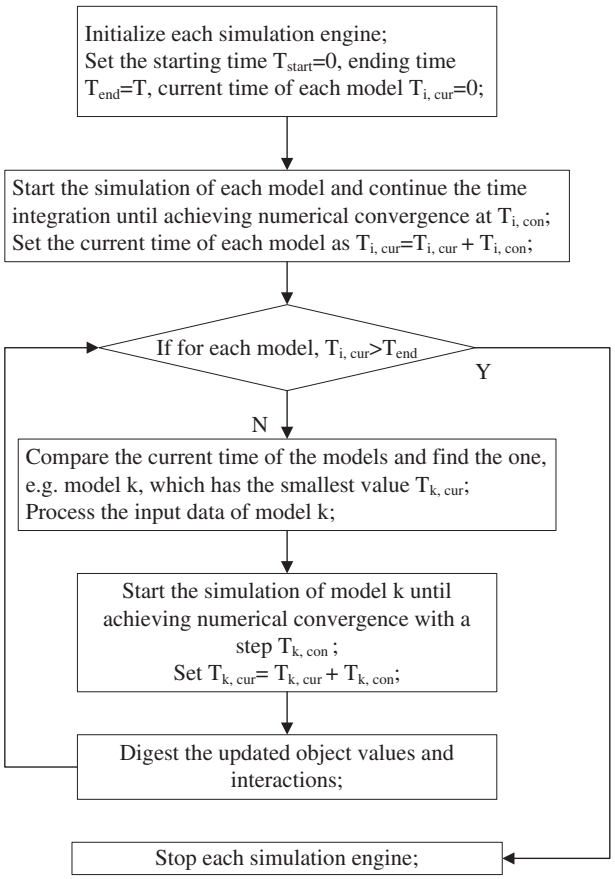


**Fig. 8.** A flowchart for the crossed interaction method.

gets the current time at which the encapsulated model achieves numerical convergence, and then updates its Current Time (CT). After that, the agent sends a request the RTI to grant it to perform simulation advancement to its CT. As all the agents are both time constrained and time controlled as discussed in Section 4.2, the RTI will only approve the request from the agent whose CT is the smallest. Before an agent's simulation advancement request is approved, it can keep getting updated data from the RTI and thus the updated data can be used for next simulation step. Such a process is repeated until all the models have completed the simulation.

## 6. Evaluation and discussion

### *6.1. Prototype system and an engineering example*

A prototype system has now been developed and further development and evaluation is still ongoing. A preliminary evaluation is done to check: (1) the viability and feasibility of the solution developed in this work; (2) whether the system can support distributed MCS; (3) the performance of the system in terms of both simulation development and simulation running; and (4) how the runtime interaction method works and whether it can achieve good accuracy. The prototype system uses Web-based interfaces and supports different users with different roles to simultaneously create models and develop simulations. A snapshot of the GUI for specifying interactions between models is shown in Fig. 9. Currently, the system can offer functionality such as high-level modeling of a MCS problem, configuration of simulation settings, controlling the simulation process using the GUI, displaying simple 3D models, and simple post-processing. Java codes can be partially generated to reduce the amount of programming work for the middleware technologies such as the HLA and Web Services. As evidenced in the prototype system, users with different roles, e.g. design engineers, simulation experts, and software engineers, can work together in the virtual environment and models can be integrated at runtime to perform the simulation tasks. The system has been tested in the lab of the first author where it is shown the speed of the Web-based system is acceptable with an average response time of 100 ms and it is found that the updating of the graphical interfaces based on Java Applet takes a lot of time. As the focus of this paper is on evaluating the viability of the solution as well as the

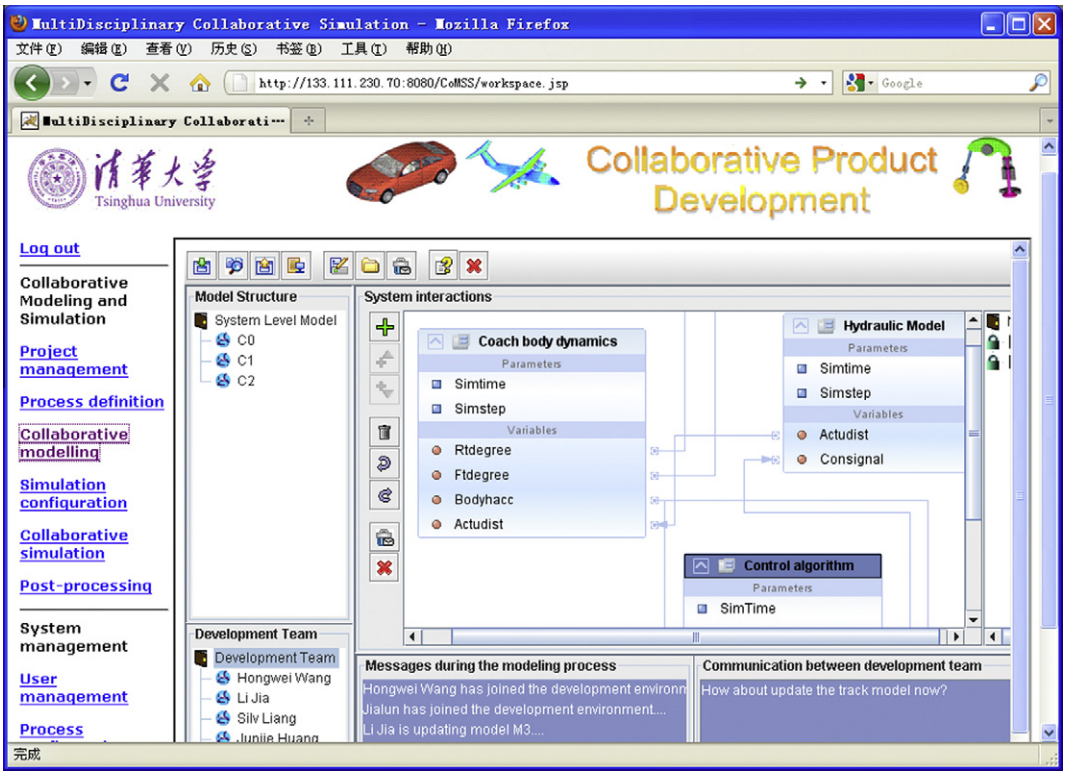


**Fig. 9.** A snapshot of GUI for specifying the interactions between sub-system models.

performance of the interaction method, the operation performance of the prototype system has not been thoroughly evaluated. Further development and evaluation is still required to test the system in real design problems and on the Internet.

An engineering example, which is used to study the landing process of undercarriage, is utilized to evaluate the viability and performance of the methods developed. Three models are developed for this simulation, namely the multi-body dynamics subsystem, the hydraulic subsystem, and the control model. Other two modules are also included, namely the data collector and the human operator. The former provides data which will be displayed on the control panel of the user interface and the latter is mainly used for practicing steering operations during the landing process. The dynamics subsystem is modeled using MSC Adams [34], which consists of a tyre, a post, and a hydraulic cylinder. The hydraulic model is created using EASY5 [36]. The M&S tool used to create the control model is Matlab Simulink [35]. The 3D multi-body dynamics model is

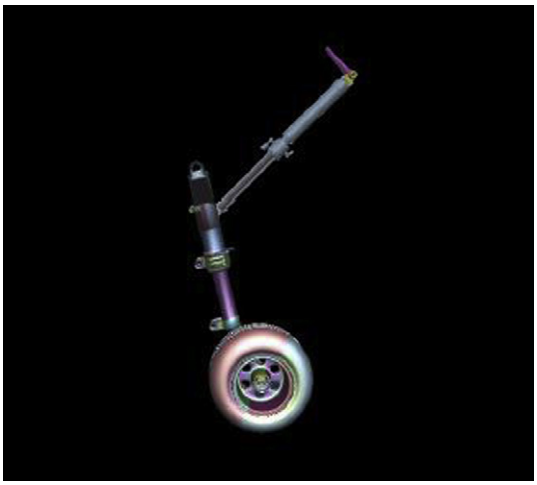

**Fig. 10.** Multi-body dynamics model of the undercarriage.

**Table 2**
Simulation configuration for the engineering example.

| Item name (unit) | Configuration |
|---|---|
| Integration method | ODE45 |
| Simulation time (/s) | 0–2 |
| Macro step used (/s) | 0.005 |
| Interpolation methods | None, one-order, and two-order methods |
| Order of starting the models | Control model, hydraulic model, dynamics model |

shown in Fig. 10. Once the three models are created by design engineers from different disciplinary areas, they are encapsulated as Web services by IT engineers and are deployed in a Web server. The setting of this simulation is shown in Table 2. Specifically, the total simulation is 2 s (logical time) with a 0.005 s (logical time) macro step. In each second, tens of updates can be done by the system, which means a total of 400 data exchange will take a few minutes to complete. To evaluate the performance of the interaction control methods, a number of different simulations are run to obtain results for comparison. Numerical results obtained from these simulations are shown in Fig. 11 where the abscissa is time (/s) and the ordinate is angular velocity in $X$ (rad/s).

Specifically, the first picture in Fig. 11 shows the reference results obtained by running the simulation for which three M&S packages are installed in a single computer and models interact with each other through the interfaces provided by the vendors of these tools. The other nine pictures show the results obtained from simulations using the three different interaction methods discussed in Section 5. Among these nine pictures, the first three pictures display the results using the CIM for three cases where no interpolation method is used, one-order interpolation method is used, and two-order interpolation method is used, respectively. Likely, the following six pictures are the results obtained from simulations using the SIM and the PIM respectively. As shown in the figure, the results obtained from CIM with two-order interpolation method have the best results which are comparable with the reference results. Interpolation methods are very useful for the CIM as a model's next time step (at which numerical convergence can be achieved) can be quite different from that of another model due to the different integration methods used by different M&S packages. The accuracy of the SIM is moderate whilst no prominent improvement is observed when interpolation methods are used. The results obtained using the PIM are least accurate and cannot successfully perform the simulation task. In summary, the CIM is feasible and achieves much better accuracy compared with traditional methods such as the PIM and SIM. As evidenced in the comparison, the CIM can resolve simulation problems that other methods fail to resolve (e.g. PIM cannot resolve the simulation problem of the engineering example). As well as achieving improved accuracy, the CIM also does not require specifying a fixed macro step before the simulation starts, which makes users' work a lot easier.

### 6.2. Discussion

MCS is a complex process which requires engineering designers, simulation analysts, software packages, and even hardware to work together to evaluate design solutions at an early stage of the design process. Collaborative computing technologies are used in this work to implement a service-oriented paradigm due to their capacities for enabling both the system integration and the collaborative work required by MCS. As evidenced in the prototype implementation, designers and simulation experts can focus on their own work whilst computational models can be well integrated at runtime. The runtime interaction method proposed achieves better accuracy than traditional methods. Broadly, Web-based simulation technologies include the methods that utilize distributed computing standards such as Web Services and the HLA [12]. Therefore, the proposed solution can also be categorized as a kind of Web-based simulation. Nevertheless, compared with the traditional Web-based simulation methods in which models are created and stored in the Web Server, this solution can offer better support for MCS.

Firstly, models for various disciplinary areas need to be created for MCS and it is very difficult to utilize the modeling functionality of a single Web-based simulation tool to resolve all the MCS problems for product design. For instance, control engineers would like to use block components for designing a control system whereas engineers working on multi-body dynamics may prefer using a software tool with 3D modeling functionality. In this solution, models created using multiple tools can be integrated. Secondly, in a traditional Web-based simulation, M&S tasks are all done on a server and all the details about a model are stored on the server. This scheme may not work well for some circumstances, e.g. cross organization collaboration, under which data and technical details are confidential and can only be kept within an organization. In this solution, computational models are encapsulated as Web services which make their computing functions accessible while their details are kept confidential by only offering interfaces for calling the services. Thirdly, such a solution offers better support for collaborative work as only simulation data are transferred throughout a simulation process. In this way, engineers can only focus on their portion of work, and when design changes have to be made, they only need to take into account the changes of input data while having no concerns about the coupling between different models (this is hard to achieve when only a single simulation tool is used).

Since it can be viewed as a special case of Web-based simulation, the main advantages (e.g. cross-platform and language-independent interoperability and model reuse) of Web-based simulation can also be observed in this solution. Compared

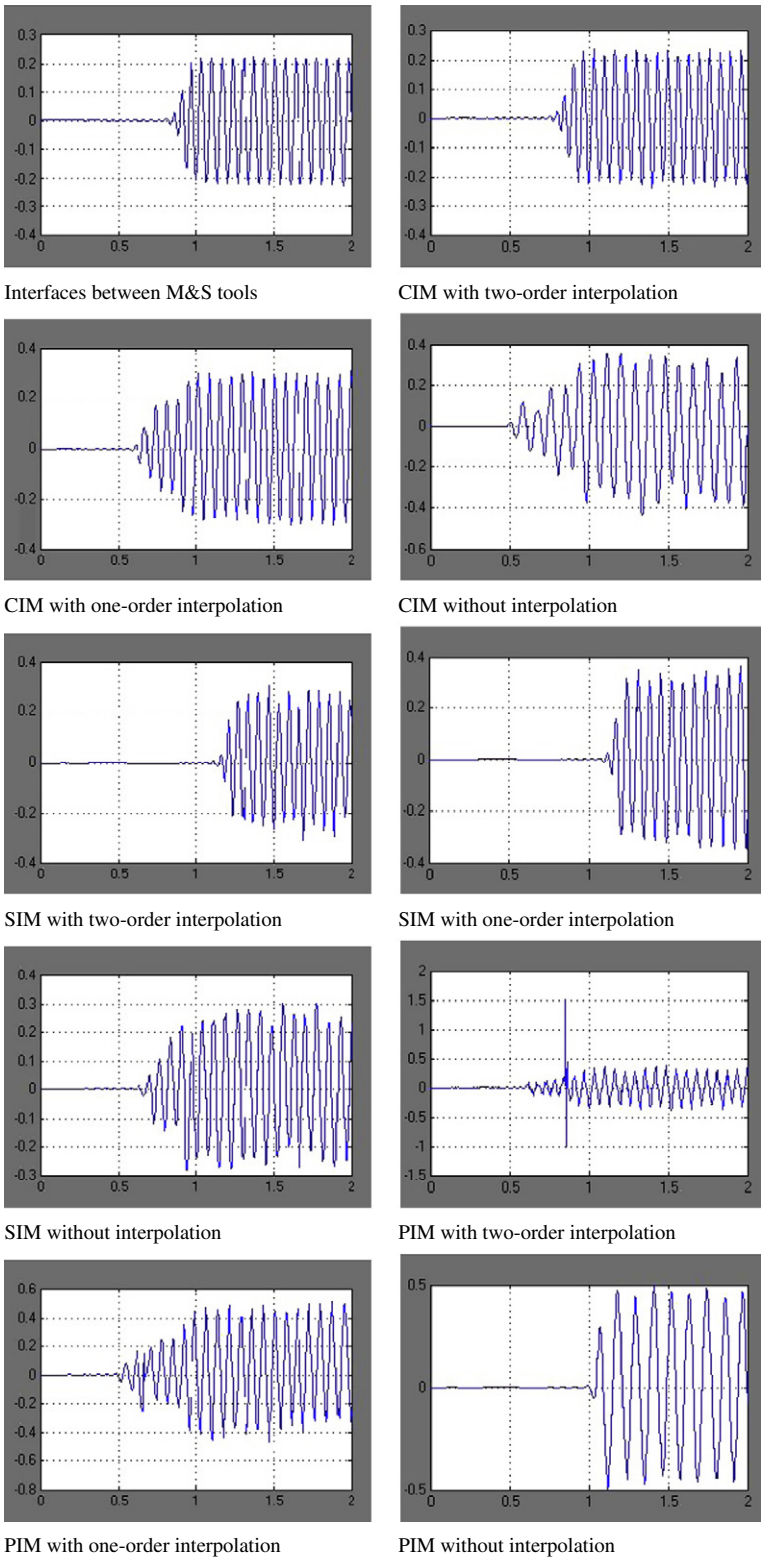


**Fig. 11.** Numerical solutions obtained from simulations using different interaction methods.

with some other solutions [6,13] that also utilize distributed computing technologies, this solution also has advantages such as high efficiency of simulation development and the support for more types of simulations. A more detailed comparison

between this solution and others that also utilize distributed computing technologies was given in [26]. Compared with the work by Ryu [9], the issue of runtime interaction is studied with more details and a method is developed to bridge the high-level description of a simulation and the underlying collaborative computing technologies. Early novel work on service-oriented simulation reported in [14,24,25] lays the foundation for this area. Compared with these pieces of work, this research is more focused on addressing simulation problems in product design by utilizing advanced distributing technologies, and involves the solving of models in parallel and their integration at run-time effectively and efficiently. The common features in those pieces of work and this work is that a modeling scheme and code generation are used to hide technical details and achieve rapid application development. The difference is that systematic validation and evaluation of simulation deployment is not addressed in our preliminary work, opening opportunities for further research in this area. Nonetheless, this solution also has a number of drawbacks. Firstly, like any other Web-based simulation technique, issues such as the lost of running speed and security vulnerability are still difficult to address. Secondly, the encapsulation of models is a difficult task and depends upon specific M&S packages involved. Thirdly, this solution utilized both HLA and Web Services to construct a service-oriented paradigm. This inevitably increases the complexity of the solution even though solutions such as automatic code generation are provided.

In summary, performing MCS in a distributed environment is very useful especially for complex products which involve multiple development teams working at different sites. The provision of computational capabilities as Web services can support collaborative MCS development which uses multiple M&S tools. The utilization of advanced collaborative computing technologies is promising for developing a successful solution for such a scenario though more work needs to be done to make the system easier and more robust. The drawbacks mentioned can be addressed in a number of ways. The running speed can be improved by using better communication techniques and developing faster algorithms for the advancement of simulation time. The security problem can be alleviated as Web Services technology provides a set of solutions for this. Once the encapsulation of models created by using an M&S tool is completed, the work for other models developed using the same tool will not be difficult. For the third drawback, a model-driven approach has been developed to make the system less complex to users. As evidenced in the prototype implementation, such a software tool is implementable and achieves better performance though it is a bit more complex than other previous systems.

## 7. Conclusion remarks

This paper presents our research work on supporting MCS in a distributed environment by using collaborative computing technologies. A scenario of the sharing and integration of simulation services on the Internet for product design is envisioned to achieve efficiency, accuracy, as well as collaboration in both simulation development and simulation execution. An understanding of the MCS problem helps identify the key requirements for a computational environment. To achieve flexible and effective integration of computational models at runtime, a synergy of two collaborative computing technologies is studied and a service-oriented paradigm is developed. Furthermore, the application of these technologies to the implementation of MCS environments has been discussed.

Accuracy is an important criterion for any simulation and therefore becomes a focus of this research. The integration process of each model and the data exchanged during this process is the key to achieving accurate simulation, though the HLA can help synchronize all the components involved in a simulation. The mechanism of runtime interaction between computational models is analyzed, together with different methods developed for this purpose. Based on the drawbacks of methods analyzed, an effective interaction method is proposed to find out the steps when numerical calculation converges and perform data exchange at these steps. As evidenced in the evaluation of a prototype system and the running of an engineering example, the solution proposed is viable and feasible, and meanwhile achieves better accuracy than traditional methods.

Currently, further development and evaluation of the system is still ongoing. In our future work, we will work out the reason why the CIM can achieves better accuracy and develop other methods to improve the performance of simulations. In particular, we are interested in how to speed up the simulation. Furthermore, more work will be done to improve the prototype system in terms of both functionality and performance.

## Acknowledgments

This work is supported by the National Natural Science Foundation of China (Grant No. 61074110). The authors are grateful to Mr. Silv Liang and Ms. Junjie Huang for the work they have done in developing the prototype system and the engineering example.

## References

[1] H. Wang, A. Johnson, H. Zhang, S. Liang, Towards a collaborative modeling and simulation platform on the Internet, Advanced Engineering Informatics 24 (2) (2010) 208–218.
[2] H.W. Wang, H.M. Zhang, An integrated and collaborative approach for complex product development in distributed heterogeneous environment, International Journal of Production Research 46 (9) (2008) 2345.
[3] J. van Amerongen, P. Breedveld, Modelling of physical systems for the design and control of mechatronic systems, Annual Reviews in Control 27 (1) (2003) 87–117.

[4] E.J. Haug, K.K. Choi, J.G. Kuhl, J.D. Wargo, Virtual prototyping simulation for design of mechanical systems, Journal of Mechanical Design 117 (B) (1995) 63–70.
[5] J. Samin, O. Brüls, J. Collard, L. Sass, P. Fisette, Multiphysics modeling and optimization of mechatronic multibody systems, Multibody System Dynamics 18 (3) (2007) 345–373.
[6] N. Senin, D.R. Wallace, N. Borland, Distributed object-based modeling in design simulation marketplace, Journal of Mechanical Design 125 (1) (2003) 2–13.
[7] R. Kübler, W. Schiehlen, Modular simulation in multibody system dynamics, Multibody System Dynamics 4 (2) (2000) 107–127.
[8] W. Xiang, S.C. Fok, G. Thimm, Agent-based composable simulation for virtual prototyping of fluid power system, Computers in Industry 54 (3) (2004) 237–251.
[9] Geun Soo Ryu, Integration of Heterogeneous Simulation Models for Network-Distributed Simulation, University of Michigan, 2009.
[10] G. Ferretti, G. Magnani, P. Rocco, Virtual prototyping of mechatronic systems, Annual Reviews in Control 28 (2) (2004) 193–206.
[11] W. Shen, Q. Hao, H. Mak, J. Neelamkavil, H. Xie, J. Dickinson, R. Thomas, A. Pardasani, H. Xue, Systems integration and collaboration in architecture, engineering, construction, and facilities management: a review, Advanced Engineering Informatics 24 (2) (2010) 196–207.
[12] J. Byrne, C. Heavey, P.J. Byrne, A review of web-based simulation and supporting tools, Simulation Modelling Practice and Theory 18 (3) (2010) 253–276.
[13] B. Johansson, P. Krus, A web service approach for model integration in computational design, in 23rd Computers and Information in Engineering Conference, Parts A and B, vol. 1, Chicago, Illinois, USA, 2003, pp. 247–254.
[14] W.T. Tsai, X. Sun, Q. Huang, H. Karatza, An ontology-based collaborative service-oriented simulation framework with Microsoft Robotics Studio®, Simulation Modelling Practice and Theory 16 (9) (2008) 1392–1414.
[15] H. Wang, H. Zhang, A distributed and interactive system to integrated design and simulation for collaborative product development, Robotics and Computer-Integrated Manufacturing 26 (6) (2010) 778–789.
[16] J. Kuljis, R.J. Paul, An appraisal of web-based simulation: whither we wander?, Simulation Practice and Theory 9 (1–2) (2001) 37–54
[17] A.S. Islam, M. Piasecki, Ontology based web simulation system for hydrodynamic modeling, Simulation Modelling Practice and Theory 16 (7) (2008) 754–767.
[18] H.-C. Cheng, C.-S. Fen, A web-based distributed problem-solving environment for engineering applications, Advances in Engineering Software 37 (2) (2006) 112–128.
[19] H.S. Han, Web-based dynamic simulation system for multi-body systems, Advances in Engineering Software 35 (2) (2004) 75–84.
[20] H. Zhang, A solution of multidisciplinary collaborative simulation for complex engineering systems in a distributed heterogeneous environment, Science in China Series F: Information Sciences 52 (10) (2009) 1848–1862.
[21] 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.
[22] S. Meyer zu Eissen, B. Stein, Realization of web-based simulation services, Computers in Industry 57 (3) (2006) 261–271.
[23] M. Gyimesi, Web Services with generic simulation models for discrete event simulation, Mathematics and Computers in Simulation 79 (4) (2008) 964–971.
[24] W.T. Tsai, C. Fan, Y. Chen, R. Paul, A service-oriented modeling and simulation framework for rapid development of distributed applications, Simulation Modelling Practice and Theory 14 (6) (2006) 725–739.
[25] Y. Chen, W.T. Tsai, Towards dependable service-orientated computing systems, Simulation Modelling Practice and Theory 17 (8) (2009) 1361–1366.
[26] H. Zhang, H. Wang, D. Chen, G. Zacharewicz, A model-driven approach to multidisciplinary collaborative simulation for virtual product development, Advanced Engineering Informatics 24 (2) (2010) 167–179.
[27] A. Rükgauer, W. Schiehlen, Simulation of modular mechatronic systems with application to vehicle dynamics, Acta Mechanica 125 (1–4) (1997) 183–196.
[28] G. Pahl, W. Beitz, K. Wallace, J. Feldhusen, L. Blessing, Engineering Design: A Systematic Approach, Springer, 2007.
[29] Pitch, "pRTI". <http://www.pitch.se/products/prti>.
[30] Apache Software Foundation, "Axis". <http://ws.apache.org/axis/>.
[31] Hongwei Wang, Heming Zhang, A study on the run-time interaction between distributed computational models in multidisciplinary collaborative simulation, in: The 2010 IEEE International Conference on Systems, Man, and Cybernetics, Istanbul, Turkey, 2010, pp. 3702–3709.
[32] A. Bonelli, O.S. Bursi, L. He, G. Magonette, P. Pegon, Convergence analysis of a parallel interfield method for heterogeneous simulations with dynamic substructuring, International Journal for Numerical Methods in Engineering 75 (7) (2008) 800–825.
[33] C.A. Felippa, K.C. Park, C. Farhat, Partitioned analysis of coupled mechanical systems, Computer Methods in Applied Mechanics and Engineering 190 (24–25) (2001) 3247–3270.
[34] MSC Software, MSC Adams Software. <http://www.mscsoftware.com/Products/CAE-Tools/Adams.aspx>.
[35] Mathworks, Inc., Matlab Simulink. <http://www.mathworks.co.uk/products/simulink/>.
[36] MSC Software, MSC Easy5 Software. <http://www.mscsoftware.com/Products/CAE-Tools/Easy5.aspx>.