Proceedings of the 2011 Winter Simulation Conference
S. Jain, R.R. Creasey, J. Himmelspach, K.P. White, and M. Fu, eds.

FACTORY FLOW DESIGN AND ANALYSIS USING INTERNET-ENABLED SIMULATION-BASED OPTIMIZATION AND AUTOMATIC MODEL GENERATION

Amos H.C. Ng
Jacob Bernedixen
Matias Urenda Moris
Mats Jägstam

Virtual Systems Research Centre
University of Skövde
PO Box 408, 54128 Skövde, Sweden

ABSTRACT

Despite simulation offers tremendous promise for designing and analyzing complex production systems, manufacturing industry has been less successful in using it as a decision support tool, especially in the early conceptual phase of factory flow design. If simulation is used today for system design, it is more often used in later phases when important design decisions have already been made and costs are locked. With an aim to advocate the use of simulation in early phases of factory design and analysis, this paper introduces FACTS Analyzer, a toolset developed based on the concept of integrating model abstraction, automatic model generation and simulation-based optimization under an innovative Internet-based platform. Specifically, it addresses a novel model aggregation and generation method, which when combined together with other system components, like optimization engines, can synthetically enable simulation to become much easier to use and speed up the time-consuming model building, experimentation and optimization processes, in order to support optimal decision making.

1 INTRODUCTION

Real-world systems in manufacturing, supply chains and public services are too complex to be modeled by analytical techniques. Therefore, discrete event simulation (DES) are very useful for performing modeling and analysis on these systems. However, DES models are by nature evaluative – instead of suggesting any optimal solutions, a DES model evaluates a given set of design variables and generates the required performance measures. For a decision maker, the process of finding a sufficiently good design setting could be too time-consuming and in many cases impossible if the search space is huge. Simulation-based optimization (SBO) is a relatively new technique applied to seek the “optimal” setting for a complex system based on one or multiple performance measures generated from simulation by using various searching methodologies. SBO is a technology that offers huge potentials to solve real-world problems and have been successfully applied in many different domains (April et al. 2004). Nevertheless, until now, virtually all of today’s commercial SBO packages still possess several major limitations: (1) they work in a deterministic mode, without taking into account the stochastic outputs from DES; (2) they do not explicitly address multi-objective problems, and (3) similar to most of the DES packages, the majority of SBO tools available is traditional software that need to be installed and run locally on the users’ computers. With the vision that Internet and Web technologies could enable the explosive growth in research and commercial opportunities (Fu et al. 2000; Boesel et al. 2001; Miller et al. 2001), many efforts have been paid on Web-based simulation (WBS) since the 1990s. However, as summarized recently by Byrne et al. (2010), the number of real applications and efficient tools for WBS is still very small. As will be discussed with more details in Section 2, an Internet-based SBO system for real-world applications is yet to be seen.

This paper introduces an Internet-based DES and SBO software system, called FACTS (Factory Analyses in ConcepTual phase using Simulation) Analyzer, which is specifically developed for supporting the design, analysis and improvement of production systems within a truly multi-objective context. Implemented as a client-server system over the Internet, FACTS Analyzer (or hereafter FACTS) is a parallel and distributed SBO software which supports multiple DES experiments and SBO processes to run concurrently. In the FACTS server, various optimization algorithms, artificial neural network (ANN) based metamodels, stochastic simulation systems and a SQL database management system are integrated and made available to multiple users to access through the Web Services technology.

In addition to covering the system architecture of FACTS Analyzer (Section 3), this paper will also present the unique capabilities provided by FACTS Analyzer that are beyond those found in conventional DES packages, for example, the automatic generation of complicated models based on optimization parameters (Section 4). Through a simple case study, this paper will also briefly introduce how FACTS Analyser can support the generation of Pareto-optimal (best trade-off) solutions for the decision making in the improvement of production systems (Section 5).

2 LITERATURE REVIEW

Parallel and distributed simulation (PADS) represents the computing technology that enables a simulation program to execute on a computing platform containing multiple processors, interconnected by a communication network. It can be used to reduce execution time and/or addressing problems like geographical distribution (e.g. multiple participants), heterogeneous simulators from different manufacturers and fault tolerance (Fujimoto 2000). In recent years, the ability to connect multiple distributed simulation models/sub-models into a larger, complex simulation has gained more attention from domains like military, telecommunication and education. Nevertheless, in many simulation applications, the primary benefit offered by PADS is the execution speedup of running many replications on parallel processors. Early work can be found in Biles et al. (1985) in which different computer architectures for carrying out a large number of simulation replications in a parallel computing environment were examined. Subsequent work was done by Heidelberger (1988), who proposed a parallel replications environment equipped with more advanced statistical methods for supporting replication analysis. In this approach, several replications of a sequential simulation are run to completion independently on different processors. In the jargon of parallel computing, this kind of applications belong to the so-called embarrassingly parallel problem because no particular effort is needed to segment the problem into a number of parallel tasks, and requires no essential communication between those parallel tasks. Embarrassingly parallel problems are ideally suited to large-scale problems over the Internet. Public applications include *climatepredication.net* (Stainforth et al. 2002), BOINC (Berkley Online Internet Computing) and probably the most well-known *SETI@home* project in which 3 millions PCs distributed all over the world are donating their unused computing power for searching extra-terrestrial intelligence.

With the advent of Internet technologies, many efforts in PADS have been made for developing simulation languages and building model libraries that can be assembled and executed over the Internet (Yoo et al. 2006). In this sense, Web-based parallel and distributed simulation (WPADS) is understood as the research area where PADS methodologies and Web-based technologies are conjoined. As noticed by zu Eissen and Stein (2006), the term WBS is used collectively for describing various applications and may have very different meanings. However, in general, it refers to the use of Web-based technologies that enable users to remotely access and execute simulation languages and simulation programs over the Web.

With the vision that the Internet and Web technologies can facilitate numerous research and commercial opportunities for cost-effective distributed simulations, many efforts have been made in WBS since late 1990s. A Java-based Simulation Manager (SimManager) is described in (Marr et al. 2000; Biles and Kleijnen 2005) which is essentially a parallel-replication approach in which the *SimManager* identifies and controls a number of slave processors (simulation engines) that run the actual simulation trials. Through the concept of *Alliance*, computer owners can make their processors "available" to the *SimManager* by entering into an agreement to participate in a simulation consortium dynamically. A simulation engine is run as a low-priority "background" task if the slave processor is used for some other application in the front end. This bears similarity with other public WPADS system such as *climateprediction.net*. Later, their work has extended to integrate commercial simulation packages such as Arena, applied to compare alternative material handling configurations for automated manufacturing (Biles and Casebier 2004).

Kumara et al. (2002) depicted a Web-based three-tier client/server framework for allowing multiple users to access a simulation program for evaluating and predicting Order-to-Delivery systems. The simulation program, developed in GM Enterprise Systems Laboratory (GMESL), was originally designed as a standalone program accessed in a single-user mode. The framework separates the functions of presentation, data management and analysis into three tiers: (1) Web client; (2) relational database server, and (3) multi-agent based virtual executor server. Focus of their work was on the scalability and user responsiveness of the system, enabled by the information model in the database server and the multi-agent execution model.

With Web Services as the enabling Internet technology, WPADS is now seen as a more viable simulation option than ever before and many researchers are aware of the benefits it can offer. There are many answers to the question "What is a Web Service?". Within the context of this paper, a Web Service is defined as a remotely accessible application component that listens and reacts for certain requests made over HTTP. When compared to the other standard object architecture for distributed applications (i.e. DCOM, Java and CORBA), Web Services technology is the only one that truly enables heterogeneous platform interoperability. Different realization alternatives for WBS services are explained and discussed with respect to their advantages and disadvantages in Eissen and Stein (2006). The authors also implemented a prototype Web Service which allows the analysis and execution of technical models for DES, continuous time or hybrid simulation described in the modeling language Modelica. Their focus was on fast model building and quick experimentation using Modelica model libraries. Gyimesi (2008) proposed a Web Services based framework for generic DES. More recently, using Web Services based SBO through the distribution of simulation replications across different servers was presented by Yoo et al. (2009). Their focus was on using an Optimal Computing Budget Allocation (OCBA) to allocate different number of simulation replications to different servers to improve the overall execution efficiency.

Byrne et al. (2010) reported that research on WBS is still in its infancy and that the cases that have been tried have not been carried out against any real customers, but only as test scenarios. For WBS tool to have a greater impact in solving real-world problems, it requires significant work to be put into finding out how these tools can be designed to not only support simulation, but also optimization, in a truly distributed and multi-user environment. As a short conclusion, based on the huge potential of Internet-based solutions but lack of research on how these solutions can be used for SBO, the ultimate goal of this work is to explore the opportunities of Internet-enabled SBO through the design and implementation of FACTS Analyzer for solving real-world industrial-scale problems.

3 FACTS ANALYZER

3.1 System Architecture

The major system components and communication protocols in the system architecture of FACTS Analyzer, which can support multiple users on the Internet, are schematically illustrated in Figure 1. At the heart of this system architecture is the server-side components which spread across four distributed sub-systems: (1) Web server; (2) optimization server; (3) database server, and (4) simulation clusters. For a SBO process running with FACTS, the optimization engine (*OptEngine*) in the optimization server is the most important component because it provides the core functionality for major algorithmic processing and acts as the hub for coordinating other functions, including interacting with the user, sending simulation and optimization data to the simulation clusters and database server as well as metamodeling. In principle, the server components can be accessed by any client applications through the Web Services hosted by the Web server. In the current implementation, FACTS Analyzer client (or FACTS client) is the main client-side application that consumes the FACTS server functionalities by sending XML requests and model specifications in form of XML files, launching/controlling SBO processes (through *OptManager*) and retrieving optimization data from the optimization database (*OptDB*).

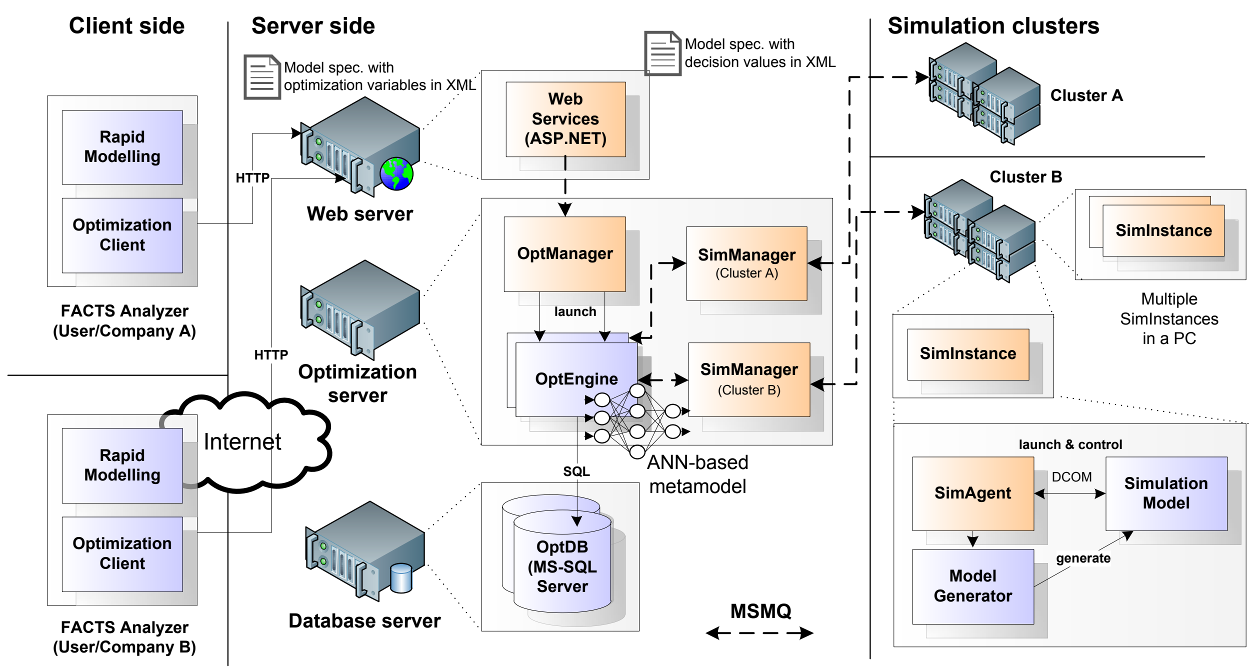

Figure 1: System architecture of FACTS Analyzer in a multi-user environment.

The optimization manager (*OptManager*) is a Windows service that listens to the request from the Web server to launch different OptEngines according to the preferences and parameters specified by the users through the FACTS Client application. Data that are required to start a SBO procedure may include: (1) simulation settings such as warm-up time, simulation time and number of replications; (2) multiple objective functions; (3) list of input variables; (4) list of output variables; (5) constraints; (6) which optimization algorithm, and (7) optimization parameters, e.g. population size, crossover rate if an evolutionary algorithm (EA) is selected. The last two options are particularly useful for SBO researchers to select and compare the performance of different algorithms or different optimization parameters. Currently, the FACTS server supports several versions of metamodel-assisted EA but new algorithms can be added easily by compiling the modified algorithm core into the platform with the object-oriented (OO) libraries developed.

By allowing all active *OptEngines* to save their optimization trajectories and other experiment results into a central database, i.e. *OptDB*, FACTS supports the following advanced features:

- The quality and diversity of the initial solutions play a crucial role in the performance of an optimization process, especially when a population-based search method is used. By saving all experimental results into *OptDB*, FACTS enables the user to choose the set of initial solutions from previous experimental records when starting a new optimization process. This is usually used in combination with the Design of Experiments (DoE) functions provided in the FACTS client.

- Dynamically changing the metaheuristic algorithms in an optimization process. This is especially useful when a global search method like EA is used for exploration first and then followed by a local search method like Hill Climbing for exploitation to further improve the optimization result.
- Fault tolerance – while faults in a simulation can be easily detected and recovered by re-starting the run by a *SimAgent*, software faults occurred in an *OptEngine* may cause a single point of failure and waste the time spent for all previous simulation runs if the optimization data are not stored. The FACTS system architecture indirectly facilitates error-recovery by allowing the user to start a new *OptEngine* and re-load the previous SBO records saved in *OptDB* as the initial solutions and training data set for the metamodel when the search process is re-started.

3.2 Simulation components

Simulation components are decoupled from the core server components because they can be highly distributed in the computer clusters to support parallel runs of the computationally expensive simulations to speed up the optimization processes. The design of the platform can actually support various types of simulations to connect to *OptEngine* through the *SimAgent* technology, which faciltates heterogeneous simulation systems to be connected to *SimManager* in an unified protocol using Microsoft Message Queues (MSMQ), as illustrated in Figure 1. Depending on the application interfacing methods supported by the target simulation system, *SimAgent* can start the corresponding *BackEnd* object to launch, interact and control the simulation software. In the current implementation, FACTS supports two types of model generation: (1) DES models generated in commercial software, using some customized model generators, or (2) binary models compiled with dynamic linked libraries. The former option allows a DES model to be generated based on the FACTS model specification and subsequently be modified to include some specific logic or details. For the latter option, optimal running speed is the major concern of the user. For the former case, a *BackEnd* object to communicate with the target DES software, e.g., through DCOM. The output data from the simulation evaluations are then "assembled" and sent back in a standard format, in form of MSMQ, via *SimManager*, and returned to the corresponding *OptEngine* for further processing and data logging.

More than one single *SimInstance*, which consists of a *SimAgent* controlling a single simulation run, can be started on a single node with multi-core processors in the simulation cluster. When a *SimAgent* is launched, it will register to the *SimManager* to announce its existence. By knowing the number of available *SimAgents*, the *SimManager* can dispatch multiple jobs received from the *OptEngines* to multiple simulations running in parallel.

3.3 FACTS Analyzer Client

To enable rapid modeling of production systems, especially in the conceptual design phase, FACTS Analyzer supplies a limited number of standard DES objects, combined with a list of objects dealing with production control mechanisms (PCMs), as listed in Table 1. The PCM objects have all been developed based on modelling experiences in industrial case studies and their functionalities have been made as generic as possible without losing their ease of use.

Table 1: The standard DES objects (left-hand side) and PCM objects (right-hand side) in FACTS.

Icon	Name	Description	Icon	Name	Description
	Source	For controlling how materials enter the model.		Timetable	For modeling shifts, i.e. for controlling when production is allowed in parts of or the entire production flow.
	Sink	For controlling how materials exit the model.		Takt	Synchronizes production of a serial line; can be used for parts or the entire serial flow.
	Operation	Operation for standard processing of material.		Demand	For modeling demand. The demand could be applied/satisfied at one or more locations in a flow. Important statistics include backlog and tardiness.
	Assembly	Operation in which two or more parts are joined together, according to an assembly description.		Batch	For grouping variants together at selected operations in the production flow; limits the number of setups required. Sizes and sequence of batches could be based on safety stock levels of desired buffers and stores in the model.
	Disassembly	Operation in which two or more parts are split.		Kanban	Pull mechanism that authorizes production at an operation based on the Kanban cards received in the succeeding buffer/operation.
	Buffer	Place that holds one or more parts for a minimum time, sequence of parts are preserved.		MaxWIP	Mechanism that limits the total amount of work in process (WIP) in a part of or the entire production flow.
	Store	Same as buffer, but sequence of parts is allowed to change.		CriticalWIP	Mechanism that limits the amount of WIP in a part of or the entire production flow based on the inventory level of a certain buffer or store.
	Component	Object that allows the modeling of custom components that could be used in one or more locations in the production flow.		Selection	Selection provides an easy way to model design alternatives in which different production settings could be compared and evaluated against each other.

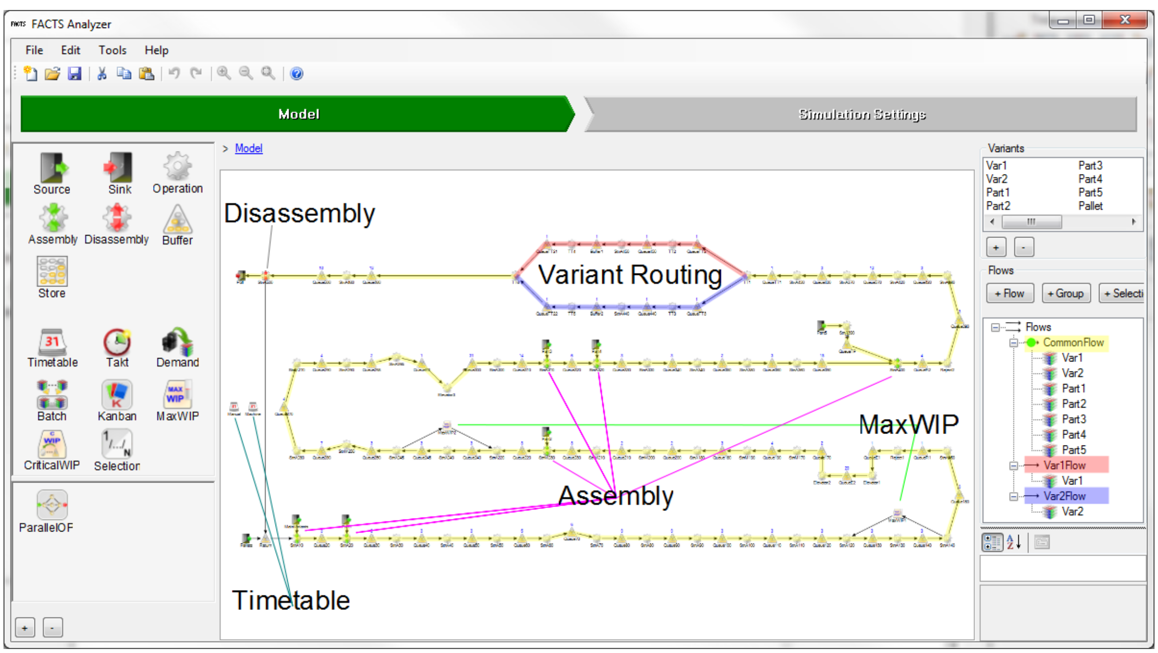

Figure 2: Using FACTS Analyzer objects to model an automotive machining line.

The PCM objects allow the modeling of complex production flows commonly found in industry today without the need of customized programming. As an example, Figure 2 shows the use of FACTS DES and PCM objects to model a real-world automotive engines machining line. FACTS supports a novel product variants handling for modeling the flows of material for different variants or groups of variants. These material flows are created similarly to how standard production flows traditionally are created in DES software, i.e. by connecting objects in the model with connectors/arrows. However, the major difference is that FACTS allows the creation of multiple flows collected in a tree structure (see right-hand side of Figure 2). Within this tree structure, there is also the possibility to have selections similar to the PCM object selection but for material flows instead of for the standard DES objects. The concept of the selection object is a novel concept to support the user to use the optimization to automatically evaluate different design alternatives, which will be illustrated with an example in the coming section.

4 MODEL GENERATION BASED ON OPTIMIZATION PARAMETERS: AN EXAMPLE

In the conceptual design phase of a production system, it is very common that there exist multiple scenarios of how to configure the production system. Having one model per scenario and hence one analysis per scenario will easily limit the amount of scenarios that could be evaluated with respect to the limited time for decision making in industry. It would be ideal to be able to analyze all scenarios, representing different design alternatives, in an efficient way. These types of scenarios are likely to have the same production flow but only differ at some locations. *Selection* (both *selection object* and *flow selection*) can enable these design alternatives to be built into a FACTS model and then switched in the simulation evaluation based on the values of the corresponding decision variables. The analysis of such a model could then be made in an efficient way by letting the optimization algorithm to seek the "best" combination of design alternatives with their optimal settings together with the optimal values of other decision variables in a single SBO run.

Figure 3 is an illustration of how alternative designs can be easily built in FACTS with both a selection object as well as a flow selection object. The *selection object*, *OPSelection*, constitutes a choice of having one fast machine instead of two slow machines in parallel, which is a common decision-making problem faced in industry when considering the replacement of old slow machines with new fast machines. For the three operations, OP1, OP2 and OP3, which are connected to *Buffer2*, there is a *flow selection* in which two alternatives are being evaluated. The first one being an option in which all three product variants (VarA, VarB and VarC) have their own flow (denoted in color red, green and blue) and in the second one the three variants are allowed to be flexibly processed in any of the three machines but with the cost in increasing setup times during variant switch. Therefore, apart from the decision on whether to replace the old machine in the first stage, the decision maker also needs to evaluate whether it is cost-effective to invest flexible machines in the second stage.

The basic model in Figure 3 will be used to illustrate the need and usefulness of applying optimization with the selection modeling. The original settings of this model can be found on the left-hand side of Table 2. The alternative selections along with some possible additional improvements are all modeled as investments with associated costs, shown on the right-hand side of Table 2.

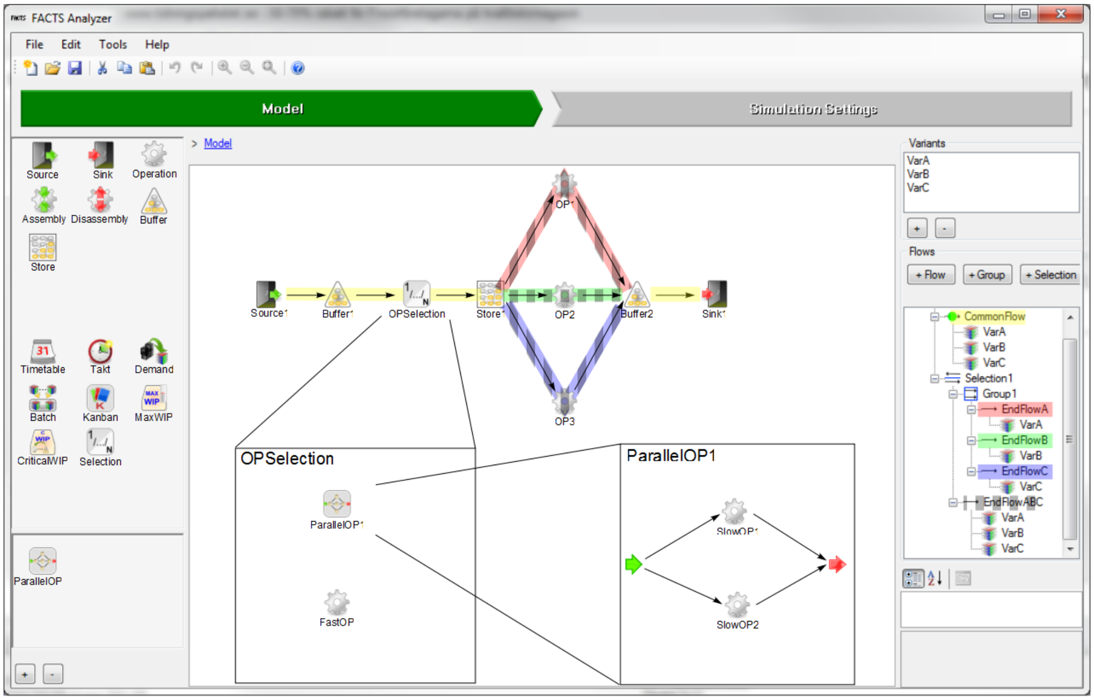

Figure 3: Illustration of selection object and flow selection in FACTS Analyzer.

Table 2: Initial setup (left-hand side) and investment costs (right-hand side) of the basic model.

Objects	**Settings**	**Objects**	**Investments and costs**
Model	The model is run with continuous production (without shifts) for 31 days of which one is used as warm-up period and with 15 replications to deal with the stochastic elements in the model.	Buffer1, Store1	Possibility of buying 1 to 5 extra places each costs $10,000.
Source1	Infinite random supply of the three variants (VarA – 40 %, VarB – 30 %, VarC – 30 %).	OPSelection	Investment in new machine (FastOP) costs $100,000.
Buffer1, Store1, Buffer2	Transportation time of 60 seconds and a capacity of 5 variants.	SlowOP1, SlowOP2, OP1, OP2, OP3	Improvement of availability from 90 % to 95% in steps of 1%. Cost $10,000 per step.
SlowOP1, SlowOP2	Constant processing time of 7,000 seconds with an availability of 90% and a mean time to repair (MTTR) of 5 minutes.	SlowOP1, SlowOP2, OP1, OP2, OP3	Improvement of MTTR from 5 minutes to 3 minutes in steps of 1 minute. Cost $10,000 per minute.
FastOP	Constant processing time of 3,600 seconds with an availability of 95 % and a MTTR of 3 minutes.	Selection1 (Flow selection)	Improve machines to multi-purpose/flexible machines that can process all variants. Cost $90,000.
OP1, OP2, OP3	Constant processing time of 9,000 seconds for VarA and 1,2000 seconds for VarB and VarC and with an availability of 90 % and a MTTR of 5 minutes. Setup time of 10 minutes for moving to/or leaving production of VarC.		
Sink1	Infinite demand.		

As mentioned, models built using FACTS Analyzer are stored and sent to the server using XML. These models (XML files) could then be interpreted by a model generator (see Figure 1) in each *SimIn-*

stance in the distributed simulation cluster. However, before the model is sent directly to the model generators, it is first processed by the *OptEngine* which sets the values of the decision variables. Each model generator then interprets the XML file and generates the model according to the decision vector set by the optimization algorithm. The highlighted sections in Figure 4 and Figure 5 have illustrated how *selections* are denoted in a FACTS XML file. For *selection objects* and *flow selections* a selection element encircles all available options (objects/flows), with an *ActiveID* attribute referencing the currently selected object/flow. Modeling and storing selections in this way provides an easy way of switching between the possible selection alternatives, which is a key aspect when incorporating selections into a SBO process.

```xml
<n:node id="source1" name="Source1" xpos="60" ypos="260">
<n:node id="buffer1" name="Buffer1" xpos="140" ypos="260">
<n:node id="opselection" name="OPSelection" xpos="220" ypos="260">
    <n:selection activeID="parallelop1">
        <n:nodes>
            <n:node id="parallelop1" name="ParallelOP1" xpos="200" ypos="260">
                <n:component>
                    <v:variants>
                    <n:nodes>
                        <n:node name="EntInt" id="entint" xpos="60" ypos="260">
                        <n:node id="slowop1" name="SlowOP1" xpos="440" ypos="140">
                        <n:node id="slowop2" name="SlowOP2" xpos="440" ypos="340">
                        <n:node name="ExitInt" id="exitint" xpos="660" ypos="260">
                    </n:nodes>
                    <f:flows>
                </n:component>
            </n:node>
            <n:node id="fastop" name="FastOP" xpos="200" ypos="360">
        </n:nodes>
    </n:selection>
</n:node>
<n:node id="store1" name="Store1" xpos="320" ypos="260">
<n:node id="op3" name="OP3" xpos="440" ypos="380">
<n:node id="op2" name="OP2" xpos="440" ypos="260">
<n:node id="op1" name="OP1" xpos="440" ypos="140">
<n:node id="buffer2" name="Buffer2" xpos="560" ypos="260">
<n:node id="sink1" name="Sink1" xpos="660" ypos="260">
```

Figure 4: A snippet of the XML file with the feature of selection object highlighted.

```xml
<f:flow name="CommonFlow" id="commonflow">
<f:flow name="Selection1" id="selection">
    <f:selection activeID="group1">
        <f:flows>
            <f:flow name="Group1" id="group1">
                <f:group>
                    <f:flows>
                        <f:flow name="EndFlowA" id="endflowa">
                        <f:flow name="EndFlowB" id="endflowb">
                        <f:flow name="EndFlowC" id="endflowc">
                    </f:flows>
                </f:group>
            </f:flow>
            <f:flow name="EndFlowABC" id="endflowabc">
        </f:flows>
    </f:selection>
</f:flow>
```

Figure 5: Another snippet of the XML file with the feature of flow selection highlighted.

5 MULTI-OBJECTIVE OPTIMIZATION

Multi-objective optimizations (MOO) for the hypothetical case study described in Section 4 have been run. The Pareto Frontier generated, in this case using a variant of NSGA-II (Deb et al. 2002), can help the decision maker to select one of the best Investment-Throughput trade-off solutions. FACTS Analyzer provides the functions for users to plot the best/medium/worst attainment surface (Knowles 2005) of the Pareto-optimal solutions generated from several MOO runs. The snapshots in Figure 6 illustrate how the user can group the results from 3 SBO replications to plot the best attainment surface for the example problem (the left-hand path). For a particular SBO run, the user can browse the optimization results and select any solution for checking the performance measures and their statistics in details (the right-hand path in Figure 6). For this example, the highlighted Pareto-optimal solutions at investment \$90,000 has shown a sudden “leap” of the throughput (8.6% improvement) and can be regarded as a “knee” point in MOO literature. It represents the best trade-off solution because significant improvement of throughput can be achieved with an additional investment of \$10,000, which cannot be achieved with any other Pareto-optimal solutions. The Pareto-optimal solution with investment \$90,000 is solely attributed to the investment of the flexible machines, without making any other changes in the original model. Further improvement of the throughput can be made with some minor additional changes but not by replacing the old/slow machine with a new/fast one. Subject to the budget on hand, these are valuable information for the decision maker in the improvement of the production line. For this simple problem, it is easy to compare the attributes of the Pareto-optimal solutions. But for a complex problem which involves tens or even hundreds of decision variables with their complex correlations, finding the attributes of the decision vectors that constituent the Pareto-optimal solutions is a very challenging task. To implement efficient methods for such kind of so-called post-optimality analysis is a very important and interesting research now underway in the development of the next generation of FACTS Analyzer.

6 SUMMARY AND OUTLOOK

Up to now, there are only very few real-world applications and efficient tools for running simulation and optimization on the Internet. This paper has depicted an Internet-based DES and SBO system called FACTS Analyzer, which is specifically developed for general conceptual factory flow design, analysis and optimization. Using FACTS Analyzer, parallel and distributed simulation experiments and SBO can be run and controlled remotely by multiple users anytime anywhere via the Internet. FACTS Analyzer also inherently supports MOO so that Pareto-optimal solutions can be generated efficiently for the decision maker to choose a configuration that is the “best” trade-off among the conflicting performance objectives in designing/improving a production system. It is currently the key focus of our work in extending FACTS with post-optimality analysis techniques for discovering the important attributes of the Pareto-optimal solutions to support decision making in real-world production systems design.

ACKNOWLEDGMENTS

This work was initiated in the FACTS project between 2006-2008 and is now continued in the FFI-HSO project (2009-2012), both are financed by VINNOVA, Sweden. The authors gratefully acknowledge their provision of research funding over the years. The authors are also thankful for our partners from the Swedish automotive manufacturers, including Scania AB, Volvo Car Corporation and Volvo AB, for their supports in the development, testing and evaluation of FACTS Analyzer.

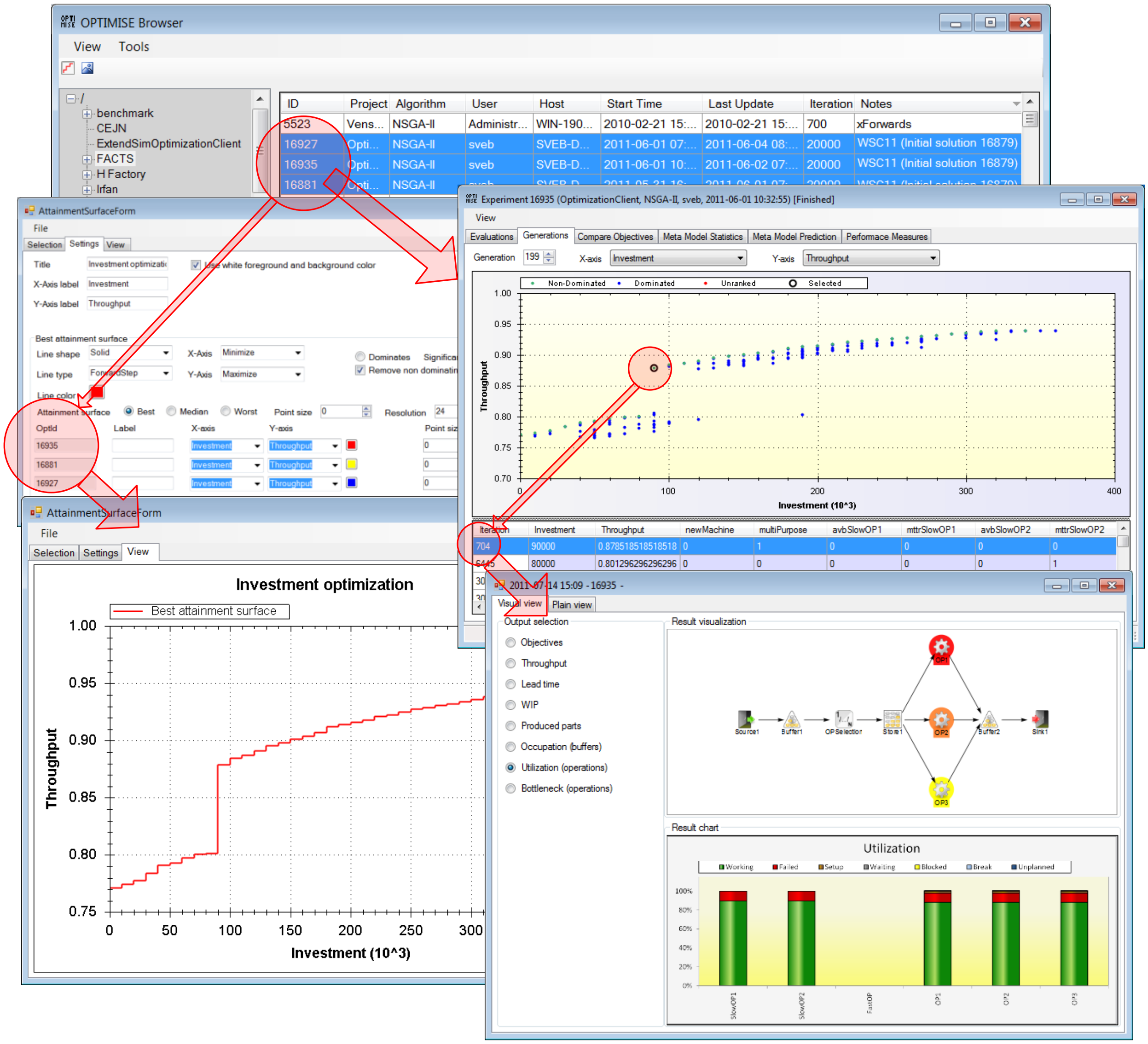

Figure 6: Snapshots of using FACTS tools to browse and analyze MOO results.

REFERENCES

April, J, M. Better, F. Glover, and J. Kelly. 2004. "New advances for marrying simulation and optimization." In *Proceedings of the 2004 Winter Simulation Conference*, edited by R. G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A. Peters, 80-86. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Biles, W. E., C. M. Daniels, and T. J. O' Donnell. 1985. "Statistical considerations in simulation on a network of microcomputers." In *Proceedings of* the *1985 Winter Simulation Conference*, edited by Blais, G. C., Soloman, S. L. & Gantz, D. T., San Francisco, California.

Biles, W. E. and J. B. Casebier. 2004. "Web based evaluation of material handling alternatives for automated manufacturing: a parallel replications approach." In *Proceedings of the 2004 Winter Simulation Conference*, edited by R. G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A. Peters. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Biles, W. E. and J. P. C. Kleijnen. 2005. "International collaborations in Web-based simulation: a focus on experimental design and optimization." In *Proceedings of the 2005 Winter Simulation Conference*, edited by M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Boesel, J., R.O. Jr. Bowden, J.P. Kelly, and F. Westwig. 2001. "Future of simulation optimization." In *Proceedings of the 2001 Winter Simulation Conference*, edited by B. A. Peters, J. S. Smith, D. J. Medeiros, and M. W. Rohrer, 1466-1469. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Byrne, J., C. Heavey, P.J. Byrne. 2010. "A review of Web-based simulation and supporting tools." *Simulation Modelling Practice and Theory*, 18(3): 253-276.

Deb, K., A. Pratap, S. Agarwal, and T. Meyarivan. 2002. "A fast and elitist multi-objective genetic algorithm: NSGA-II." *IEEE Transaction on Evolutionary Computation*, 6(2): 181-197.

Fu, M.C., S. Andradóttir, J. S. Carson, F. Glover, C. R. Harell, Y.-C. Ho, J. P. Kelly and S. M. Robinson. 2000. "Integrating optimization and simulation: research and practice." In *Proceedings of the 2000 Winter Simulation Conference*, edited by J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, 610-616. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Fujimoto, R. M. 2000, *Parallel and distributed simulation systems*. New York, Wiley, 2000.

Gyimesi, M. 2008. "Web Services with generic simulation models for discrete event simulation." *Mathematics and Computers in Simulation*, 79: 964-971.

Heidelberger, P. 1988. "Discrete-event simulation and parallel replications: statistical properties," *Scientific and Statistical Computing*, 9:1114–1132.

Knowles, J. 2005. "A summary-attainment-surface plotting method for visualizing the performance of stochastic multiobjective optimizers." In *Proceedings of the 5th International Conference on Intelligent Systems Design and Applications* (ISDA), 8-10 September 2005, Wroclaw, Poland, IEEE Computer Society, 552-557.

Kumara, S. R. T., Y.-H. Lee, K. Tang, C. Dodd, J. Tew and S.-T. Yee. 2002. "Simulation anywhere any time: Web-based simulation implementation for evaluating order-to-delivery systems and processes." In *Proceedings in 2002 Winter Simulation Conference*, edited by E. Yücesan, C. H. Chen, J. L. Snowdon, and J. M. Charnes, 1251-1259. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Madan, M., Y.-J. Son, H. Cho, and B. Kulvatunyou. 2005. "Determination of efficient simulation model fidelity for flexible manufacturing systems." *International Journal of Computer Integrated Manufacturing*, 18, (2–3), 236–250.

Marr, C., C. Storey, W. E. Biles, and J. P. C. Kleijnen. 2000. "A Java-based simulation manager for Web-based simulation." In *Proceedings of the 2000 Winter Simulation Conference*, edited by J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Miller, J.A., P.A. Fishwick, S.J.E. Taylor, P., Benjamin, B. Szymanski. 2001. "Research and commercial opportunities in Web-Based Simulation." *Simulation Practice and Theory*, 9(1-2): 55-72.

Stainforth, D., J. Kettleborough, A. Martin, A. Simpson, R. Gillis, A. Akkas, A., R. Gault, M. Collins, D. Gavaghan, and M. Allen. 2002. "Climateprediction.net: design principles for public-resource modeling research." *The 14th IASTED International Conference Parallel and Distributed Computing and Systems*, Cambridge, USA.

SETI@home, Search for Extraterrestrial Intelligence [online]. http://setiathome.ssl.berkeley.edu/

Yoo, T, H. Cho H and E. Yücesan. 2006. "Web service based parallel and distributed simulation experience." *WSEAS Transactions on Systems*. Vol 5, No 5, 973–980.

Yoo, T., Cho, H. and Yücesan, E. 2009. "Web Services-Based Parallel Replicated Discrete Event Simulation for Large-Scale Simulation Optimization." *Simulation*, 85 (7), 461-475.

zu Eissen, S. M. and B. Stein. 2006. "Realization of Web-based simulation services." *Computers in Industry*, 57: 261–271.

AUTHOR BIOGRAPHIES

AMOS H.C. NG is an Associate Professor at the University of Skövde, Sweden. He holds a B.Eng. degree and a M.Phil. degree, both in Manufacturing Engineering from the City University of Hong Kong and a Ph.D. degree in Computing Sciences and Engineering from De Montfort University, Leicester, UK. He is a member of the IET and a Chartered Engineer in the UK. His research interests include modeling, simulation and multi-objective optimization for production systems design and analysis. His e-mail is amos.ng@his.se.

JACOB BERNEDIXEN has a M.Sc. degree in Industrial Engineering and Management from the Institute of Technology at Linköping University, Sweden. Since his graduation in 2007, he has been working in industrial-based SBO research projects at the University of Skövde, Sweden. His e-mail is jocab.bernedixen@his.se.

MATÍAS URENDA MORIS is a Senior Lecturer at the University of Skövde, Sweden. He holds a B.Sc. degree in Automation Engineering, a M.Sc. degree in Manufacturing Management and a Ph.D. degree in Manufacturing Engineering from the University of Skövde, Loughborough University, UK, and De Montfort University, UK, respectively . His main research area is DES for manufacturing and healthcare systems with emphasis on system modeling and analysis. His e-mail is matias.urenda.moris@his.se.

MATS JÄGSTAM is a Pro-Vice Chancellor (External Relations) of the University of Skövde, Sweden, since 2009. He holds a B.Sc. degree and a M.Sc. degree in Automation Engineering at the University of Skövde and a Ph.D. degree in Manufacturing Engineering at De Montfort University, UK. His e-mail is mats.jagstam@his.se.