Proceedings of the 2006 Winter Simulation Conference
L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds.

SIMULATION-BASED EARLY WARNING SYSTEMS AS A PRACTICAL APPROACH FOR THE AUTOMOTIVE INDUSTRY

Ingo Hotz

DaimlerChrysler AG
Gaggenau Plant, WG/PZP-DPM
Hauptstrasse 107
76571 Gaggenau, GERMANY

André Hanisch

Fraunhofer Institute for Factory
Operation and Automation IFF
Sandtorstrasse 22
39106 Magdeburg, GERMANY

Thomas Schulze

School of Computer Science
University of Magdeburg
Universitätsplatz 1
39106 Magdeburg, GERMANY

ABSTRACT

Simulation-based Early Warning Systems (SEWS) support proactive control of real material flow systems. In consequence of real or potential state changes, proactive control (unlike reactive control) makes foresighted and target-oriented acting possible. Starting from the definition of SEWS their architecture and the requirements for design of SEWS are discussed. The compliance with simulator-independency is one important facet. Basically, this is achieved by the use of Web Services, XML and XSD (XML Schema Definition). One key component of SEWS are online simulation models which are initialized with the current system state of a real system. The utilization of RFID technology to generate information about current system states improves the quality of simulation-based forecasting. Example applications from the automotive industry show the benefits of SEWS.

1 INTRODUCTION

Decision support with long time horizon is one classic application of simulation models. Different parameters of planned systems have to be evaluated based on simulation results and planners endeavour to optimize them. In this standard application simulation models are offline, i.e. the models are not directly linked with the real system. Usually, the developed simulation models are not used upon completion of the decision-making process.

By contrast, short time decisions have to be made constantly during controlling and managing processes of existing systems. Nowadays, the complexity of such systems is increasing as well as the need of their efficient control and management. There are two main classes of control strategies: simulation-based strategies and strategies based on heuristic rules and mathematical equations. Comparing both the simulation is based on the current state of the real system and provides decision makers higher quality support.

Simulation-based Early Warning Systems (SEWS) support proactive control of real material flow systems. In consequence of real or potential state changes, proactive control (unlike reactive control) makes foresighted and target-oriented acting possible (Banks 1998).

The usage of early warning systems to control complex systems like material handling systems makes grand demands on simulation models and system environment. One requirement is that potential users of SEWS do not have to parameterize, start or analyze simulation runs. That means the simulation model has to be embedded invisibly into a SEWS and special programming constructs are required that allow simulation models to be integrated in a production control or operating system (Banks 2000).

The main objective of this paper is to describe the design and architecture of SEWS with regard to the independency of integrated simulators. The following section gives a description of Simulation-based Early Warning Systems including a suggested architecture. After that, requirements for the design are pointed out. Communication principles between SEWS-components and possibilities to guarantee simulator independency are discussed in the next section. Also, methodologies for collecting state data about the real system and their influence on the initialization process is pointed out. Finally, the paper presents applications from the automotive industry and gives a conclusion.

2 ARCHITECTURE OF SIMULATION-BASED EARLY WARNING SYSTEMS

Generally, early-warning systems are mechanisms deployed to inform persons about risk of imminent danger at an early stage. So the purpose is to enable the user or the deployer of the early-warning system to prepare for the danger and act accordingly to mitigate the effects or to even avoid them (Greulich and Barnert 2003).

Simulation models are used to predict the behavior of real and complex systems which are mostly stochastically

influenced. Typically in this case, simulation models help to analyze the effects of various alternatives without real implementation and possible negative effects to the real system. Usually, simulation models are not directly initialized with the state of the real system. Typically, in classical simulations the interest lies on parameters that converge or adjust after a long period of time.

Simulation-based Early Warning Systems combine the functionalities of simulation and early-warning systems. The simulation-based forecasting of future system states is the significant difference to classic early-warning systems that are exclusively based on historical and current measuring data. According to (Hotz and Schulze 2006a) the following definition of SEWS may be used:

"A Simulation-Based Early Warning System (SEWS) is a mechanism which is based on a simulation model of a complex real system and that points out negative effects or positive potentials as soon as possible. It also conveniently informs the user of the complete system by forecasting and analyzing different action alternatives."

The potential users of SEWS are decision makers who are instructed to control a real system. In practice the acceptance of SEWS is strongly affected by the capability to detect exceptions, the generation of reasonable action alternatives and the user-friendly presentation of these events, exceptions and problem-solving strategies. For example, the use of ambiguous thresholds in exception detection might lead to a higher chance of false alarms or no alarms at all and thereby SEWS could lose their credibility and people would stop using them. In the context of SEWS exceptions are defined as unwanted system states or specific system values that the user of SEWS wants to be informed about - e.g. schedule variances, machine failures or inadequate stock.

2.1 Main Components of Simulation-based Early Warning Systems

The main components of SEWS are Framework, Data Sources, Simulation Models and User.

Relevant system data has to be collected from different data sources. In the automotive industry this could be databases of operating systems, production control centers or planning and documentation databases. Other specific scenarios might demand to collect data from CNC (computer numeric controls), storage programmable logic controllers (SPC) of conveyors and machines or even directly from OPC-servers.

The ability of forecast and proactive control is provided by simulation models for the real system. A SEWS should not be specific to a special simulation model nor a particular simulator. Of course, there are certain simulators that are more suitable because of their performance, their communication abilities and their extensibility.

The control center user is especially important. He or she is a supplier of data as well as a beneficiary of the system. With a given user interface he or she must be able to configure the different functions of the framework regarding to the very specific problems of the real system.

XML is used to exchange data between components. A dedicated XML Schema Definition (XSD) has to be developed and used efficiently. Figure 1 shows the main components of SEWS including their coherences.

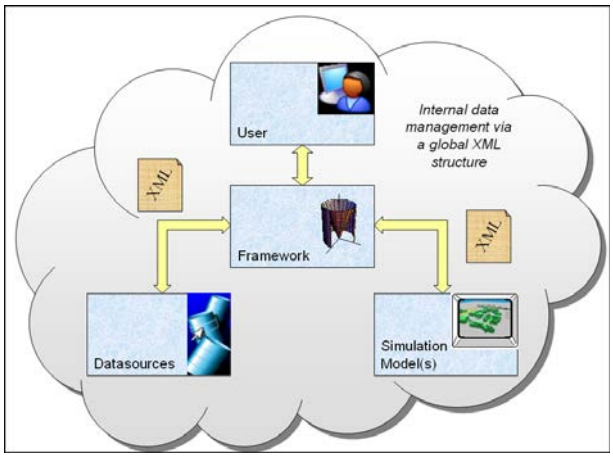

Figure 1: System Architecture of SEWS

2.2 The Kernel Component Framework

The kernel component inside SEWS is a modular structured framework which provides the user with the functions of a SEWS. Furthermore, the kernel manages and controls all the other main components. The components shown in Figure 2 are described as following:

The component **Data Listener** collects data from the real system and provides it to the SEWS. Collection activities are periodically conducted or can be started depending on defined events. Raw data is aggregated and transformed into XML and provided to other components.

Integrated simulation models and their simulation runs are controlled by the **Simulation Control** component. Input data have to be transferred to the simulation models and output data have to be received. Simulation results have to be processed and analyzed.

Both simulation results and system data have to be checked for exceptions by the component **Exception Interpreter**. Stored rules from a knowledge base support this process. User defined exceptions have to be processed too.

Herein already mentioned knowledge base is hidden in the **Learned Rules** component. Predefined rules are managed for interpretation of simulated or measured data. On the one hand rules can be defined by the user and on the other hand SEWS are able to "learn" rules from executed alternatives.

The **Generator for Alternatives** implements the alternative generation of control instructions depending on

the recognized exceptions. There exist strong connections to the component Learning Rules.

The historical results of SEWS are stored using a **Logging Function** which not only includes simple tracing functions but in addition also logs detected exceptions, used alternatives and rules. Historical data allow the reuse of collected knowledge and learned experiences in the future.

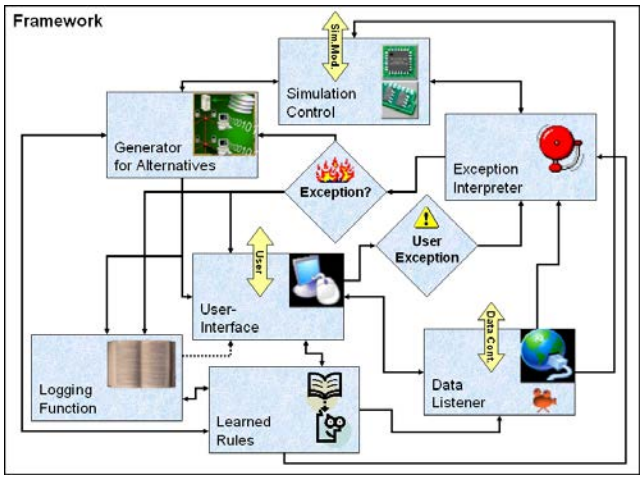

Figure 2: Components of the Framework of SEWS

User Interface controls the framework, the visualization and the management of data, the definition of rules and the parameterization of the entire SEWS. The communication between SEWS-components is based on XML. Dedicated technologies such as XSLT and Web Services are applied which supports the requirements for reusability, standardization and usability (Lovell 2002).

2.3 Requirements for Design of SEWS

The following requirements for design of SEWS have to be fulfilled to build reasonable applications:

- **Performance:** Short response times for simulation based forecasting and the derivation and evaluation of alternative strategies are required. The time-consuming of this process is strongly influenced by simulation tool performance, level of detail in simulation models and data transfer times.
- **Reactivity and Proactivity**: SEWS have to support reactive and proactive control by simulation-based analysis of different control strategies. The analysis starts at decision points. Reactive control decision points emerge from current state value changes exceeding a certain threshold whereas proactive control is based on predicted state values. Thus, exceptions are recognizable ere their actual occurrence by what suitable actions are implementable to mitigate against or avoid it.
- **Scalability and Adaptability**: Scalability indicates the capability of SEWS to continue to function well when it is changed in size or volume to meet modified needs. Increasing level of detail in simulation models has to be compensated by parallel execution of simulation runs on different processors. Adaptability is the ability to change or to be changed to fit the evolution of an underlying real system. SEWS have to recognize such changes and have to prepare adaptations.
- **Extensibility**: Extensibility indicates the capability of a system to consider future growth. SEWS have to adopt the extensions of a real system. New components have to be mapped to the SEWS in a simple way. An open architecture and the use of standards are necessary for the design of SEWS.
- **Platform Independence:** This requirement is not only limited to computers and different operating systems. Varying simulation models, based on different simulation systems and data either stored in heterogeneous information systems or available as online-data have to be integrated. SEWS must not focus exclusively on one simulation system.
- **Usability**: SEWS are used by different kinds of users. User views have to be flexible, configurable, robust and plain.
- **Standardization and Reusability:** Standardization indicates the use of guidelines for interoperability between system components. XML is used as a data exchange standard. Reusability characterizes the capability of components in different systems. Components of SEWS are reusable in other applications and in other components inside SEWS.

3 COMMUNICATION PRINCIPLES AND DATA HANDLING

After the description of components of SEWS in the last section the communication principles between these components are explained (syntactical interoperability). To fulfil the requirement of simulator independency semantical interoperability is needed to equip components with the "knowledge" which information they need to exchange. One possibility to implement semantical interoperability is the use of XSD which is described in the second part of this section.

3.1 Communication between Components

The SEWS components have different functions and tasks. An efficient and conclusive communication between components has to be implemented because performance is one key requirement for the design of SEWS and is strongly influenced by the data transfer times between and inside the components.

Generated alternatives have to be evaluated by simulation runs which are very time-consuming. So SEWS need a distribution systematic of tasks and components over a network to fulfill the request for efficiency. At this point Web Services come into play.

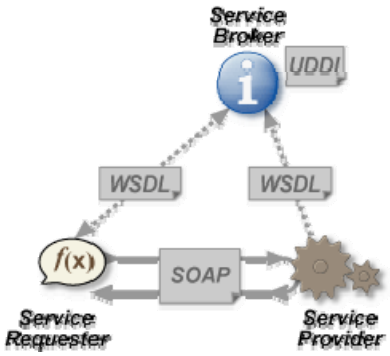

Figure 3: Principle Architecture of Web Services

A Web Service is a piece of software designed to support interoperability of interactions between various software applications running on disparate platforms over a network (Graham et al. 2002). They allow software and services from different locations to be combined easily to an integrated service and to be reused within an infrastructure. Web Services have an interface that is described in a machine-readable format such as WSDL (Web Services Description Language). A Web Service has to be published at a service broker where a service requester can find dedicated services for specific tasks (Figure 3). The service broker returns information about suitable service provider. Systems that interact with Web Services use SOAP (Simple Object Access Protocol). SOAP is a protocol to exchange XML-based messages over a computer network using HTTP (Livingston 2002).

One disadvantage of Web Services and their text-based format compared to communication approaches like RMI (Remove Method Invocation) or CORBA (Object Request Broker Architecture) is that they suffer from poor performance (Brose et al. 2001). This is the reason why from the very first the communication effort caused by requests and responses has to be minimized. That is the drawback of using an open and easy to implement standard.

However, using these Web Services in SEWS makes sense because time-consuming simulation runs and the collection of data can be done on different machines whereas the framework is working on a separate web server. Figure 4 shows the distributed organization of SEWS components. The undistributed components of the framework (generator for alternatives, user interface, learned rules component, logging function) are running on a master web server where the initialization data is generated and provided.

The machines that run the simulation models and the data collection have to be equipped with a Web Service to handle the task of simulation control and exception interpretation. Finally, the collected data and the results of simulation runs and potential exceptions are returned to the master web server.

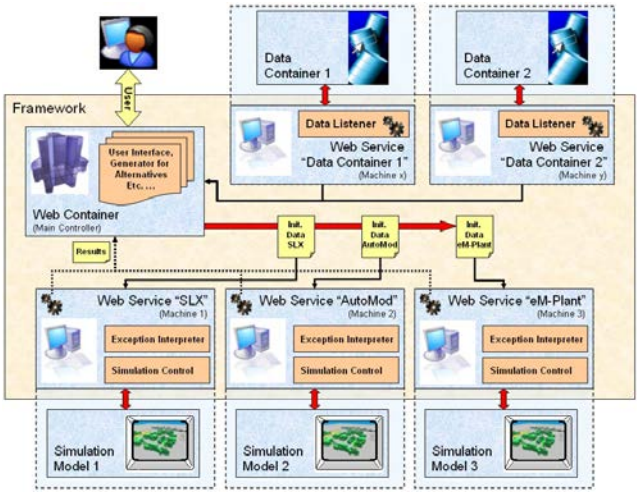

Figure 4: Distributed Organization of SEWS Components

The usage of a Service-Oriented Architecture (SOA) makes the integration of distributed simulation models possible. HLA-based distributed simulation is wrapped in a Web Service which represent the whole HLA-federation as one embeddable feature. Towards the other components of SEWS the distributed simulation appears as a common Web Service (Raape, Simonis and Schulze 2005).

3.2 Consideration of Simulator Independency

The main component framework has to be designed independently from the simulation tool used. For this reason a standardized communication strategy between the simulation tool and the framework is needed. XML can help to solve this problem because of its capability to be used in different technologies and presentability in a variety of display formats.

Recommended by the W3C (W3C 2006) the eXtensible Markup Language (XML) is a general-purpose markup language for creating special-purpose markup languages capable of describing many different kinds of data. XML plays an important role in facilitating the exchange of data across different systems, especially systems connected via the Web.

An XML document has to be well-formed and valid. Valid XML documents contain data that is conform to an XML schema that describes correct data values and locations. Schema definition formats for XML are the Document Type Definition (DTD) and the XML Schema Definition (XSD). XSD is the more powerful successor of DTD in describing XML languages because its rich data typing system allow more detailed constraints. It is based on XML

format and it can be processed by ordinary XML tools. In this case the semantical interoperability is assured by XSD which is binding for all components.

There exist many approaches to utilize the advantages of XML for simulation (Lee and Luo 2005, Röhl and Uhrmacher 2005) but the problem of any approach is that there is no standard XML schema definition for simulation models. Therefore, a global XML schema definition for SEWS was designed that helps to describe production systems (Figure 5).

In this context one sub problem is the mapping of simulator-independent XML data to simulator-specific data structures. The method of solution is to map this data in this article includes the following steps:

- Generation of simulator-specific code from XML data via XSLT stylesheets and an XSTL processor.
- Integration of generated code into the simulation model.

The process of code generation is done with an XSLT transformation where two things are necessary. An XSL document describes the transformation output depending on the utilized simulator and an XSLT processor is doing the actual transformation. The programming efforts to integrate new simulation tools are reduced to the creation of new simulator-specific XSLT stylesheets.

There are different XSLT processor distributions available. The most common distribution is Xalan-Java. It can be used from the command line, as an applet, as a servlet or as a module in other programs.

There are a variety of potential communication methods for the generated source code. Usually, company networks provide slow transfer rates and should not unnecessarily burden with additional network traffic. Making importable data files available to simulation models reduces communication costs to the required amount of data. These files are read or included at the beginning of simulation runs depending on the simulator used.

Furthermore, this method of solution supports the enabling of SEWS with functions to generate flexible reports independently from the utilized simulator. Variables are freely configurable by the control center user of SEWS. He or she can flexibly define own reports as requested and thus independence is further supported. The simulation model can process the calculation rules of these variables and return the results after or during the simulation run. Such points of time could be the end of simulation runs or the emergence of heavy exceptions in the simulated material flow system. The functionality of flexible report generation is described in (Hotz and Schulze 2006b).

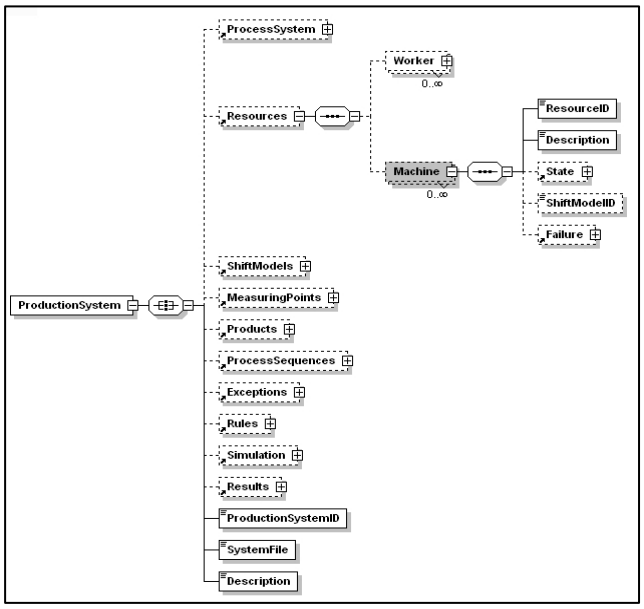

Figure 5: Global XML Schema Definition for SEWS

4 INITIALIZATION OF SIMULATION MODELS

Simulation models used in SEWS are very similar to online-simulation models. Generally, online-simulation is the simulation of an existing real system based on a simulation model of this system. The simulation model has to be initialized with real data of the current system state. This initialization process has to be done as exactly as possible which requires the mapping between representative system variables in the simulation model and available system data from the real system (Hanisch et al. 2005). The initialization strategy depends on the density of measuring points in the real system, data availability, data inaccuracies and system inconsistencies which are described in the following.

4.1 Measuring Points and Initialization Strategies

The abilities and characteristics of material flow systems can be described by a specific range of data. Basically, this data can be separated in two main groups.

The first group contains data which is deterministic and independent from the current state of the system. This data is called master data and includes the description of resources (workers, machines, operating facilities, etc. …), shift models, confirmation and respective measuring points. Furthermore, information about the process system and the process sequence of products are part of this group. In this context a process system describes all operations that are achievable by the material flow system.

The second group is dynamic data which describe the current system state, e.g. the position and status of specific products as well as the state of specific resources and the degree of performance of production orders.

This data is provided by different production planning systems, production control centers or planning and documentation databases. Usually, material flow systems or other systems which are provided with SEWS have to be connected with information systems that retrieve data by direct measurement. This can be done by means of sensors or manual confirmation points where the system state changes are perceivable and the underlying data bases are updated with new information. In the following these points are called measuring points.

Activities and procedures between measuring points are not traceable which profoundly influences the factor for the required level of detail of the simulation models. The denser the measuring points in a real system are the more precisely the simulation model can be initialized. Each product or good has a unique identifier of some sort. When it passes a measuring point it is identified by that unique id. Usually, other information regarding the state of the product and a timestamp are registered too. The development of simulation models has to consider the existence of measuring points in the way that it is possible to create and infiltrate products (simulation objects) in the right order and at the right time corresponding to the measuring points' information which is always combined with a timestamp. Figure 6 clearly shows this coherences of measuring points which are interlaced in a process system.

Generally, at the initialization of online simulation models no transient phase can be used to adjust statistics compared to the classical simulation approach where the simulation models are "initialized" with an empty state.

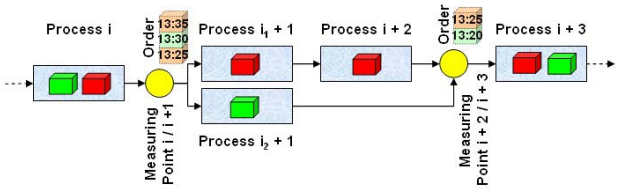

Figure 6: Schematic Diagram of Measuring Points in a Process System

Two basic initialization methods are explained in (Hanisch et al. 2005). First, an existing parent simulation model of the real system is running synchronic to real-time and is permanently updated. At specific points of time a child model of the parent model is created and starts a simulation run with maximum speed to forecast future system states. This initialization strategy is useful in the case of an insufficient data quality but it also demands high requirements for the simulation software. The disadvantages are very complex program routines to update the parent simulation model. Furthermore, only a few simulation tools are capable of making copies of a current running model and executing this copy with a different time advance mechanism. This problem contradicts to the requirement of simulator independency at the development phase of SEWS. In the case of decision the effects of different alternative actions have to be evaluated via simulation runs. These decision points emerge if state value changes exceed a defined threshold. For this reason even more child models of the parent model have to be created which makes this method much more complicated to implement. The question of license costs is not even touched.

The second basic initialization method mentioned in (Hanisch et al. 2005) takes existing simulation models and initializes these models with relevant system state data. There are two main advantages of this method over the first approach mentioned above. It is relatively simple to implement and it does not require a specific simulation tool because almost every simulation tool provides communication functionalities to read and process initialization data. The requirements on data quality are higher but the application of SEWS in the automotive industry without a specific standard of data quality is not reasonable.

4.2 Data Inaccuracies and System Inconsistencies

The choice of the right initialization strategy is strongly influenced by the different characteristics of the initialization data. These characteristics are availability and quality (Hanisch et al. 2005). The availability specifies whether the data from material systems is measurable or deterministic. Data quality describes the correctness of data which is obviously negatively effected by measuring and acquisition errors. Both characteristics are not given for all applications and clearly the problem is the availability of on-time and correct data. Data availability is given when all necessary data is present and computable at the right time.

The system state is traced via measuring points at specific confirmation points. One problem in the day-to-day business is the human factor. We do not really know what happens between measuring points e.g. without allegation that someone has dishonest motives - workers of an assembly line could take products off the line without acknowledging the effects on the systems. The result might be obsolete data records in factory databases which would have to be localized and separated out at the initialization phase of the simulation models. Measuring points are predetermined by the real system. SEWS have to regard all available measuring points. The denser measuring points are established in the real world the more accurate the simulation models can be initialized.

A second problem arises from divergent system consistency. Every system is updated at specific points in time. During these time periods the model represents a past system state. In reality, more then just one system provides relevant data to the initialization strategy. These different data synchronization points and time intervals lead to phases when the condition of chronological synchronization is not fulfilled anymore.

The schematic diagram in Figure 7 shows the problem of unfulfilled chronological synchronization of two data sources *i* and *j*. Every data source has its own synchronization point where the real system is synchronized with the data source. If these points are not coordinated the phases of chronological synchronized data sources are scattered arbitrary. Dark areas in the diagram shows phases of an unfulfilled condition of chronological synchronization. In contrary the yellow areas show that the condition is fulfilled.

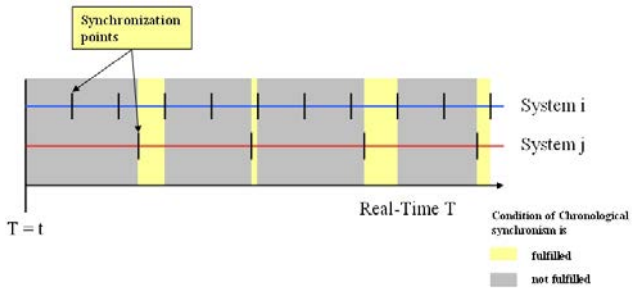

Figure 7: Phases of Unfulfilled Chronological Synchronization of Data Sources

To co-ordinate the synchronization points of involved systems during the implementation of SEWS would be one method of solution. Another method can be RFID.

4.3 Using RFID to fill the Gap

As mentioned before there need to be some kinds of measuring points in the manufacturing system in order to retrieve data from the real system and to initialize the simulation model. Obviously, the density of the measuring points and the type of sensor or measuring technology have a strong influence on data availability and data correctness.

One of the main tasks during initialization of manufacturing system is to match the distribution of goods over the complete production line, machines etc. with the distributions of for example entities in the simulation model. There are many different ways to identify a product when it passes a measuring point. The long-established technology of barcodes is widely accepted and used in today's industry. Besides the unbeatable production costs of such a barcode there are several disadvantages that favored the development of a new technology. Just to mention a few inadequacies: a barcode reader needs to be in line of sight with the barcode to read and decode it. Naturally, dirt constricts that process. Also, a barcode is static and cannot change once it is applied to an object. The new and promising technology to overcome these shortcomings is called Radio Frequency Identification (RFID) and shall be shortly introduced in the following.

A RFID system typically consists of two main components: RFID tag or transponder and RFID transceiver as a antenna reader combination. Similar to a barcode the tag is a small object that can be attached to an object. RFID tags contain silicon chips and antennas to enable them to receive and respond to radio-frequency queries from an RFID transceiver.

Typically, a tag has a unique identifier and often some memory that a user can write data to. There exist many different kinds of RFID tags differing in form, look, memory size, working frequency and range of functions.

In comparison to barcodes the tag does not need to be in line-of-sight with the reader. As long as it is in the reading range there can be obstacle between the two. Dirt usually does not hinder that process. Often, a tag is not only readable but also writeable and therefore not static anymore. Depending on the memory size it can store and deliver any kind of information

Passive tags require no internal power source, whereas active tags require a power source. The electrical current induced in the antenna by the incoming radio frequency signal provides just enough power for the CMOS integrated circuit (IC) in the tag to power up and transmit a response. Depending on the standard or frequency they work and the material they are attached to the distance between a passive tag and reader can range from a few millimeters (Low Frequency 125 KHz) over centimeters (High Frequency 13,56 MHz) up to several meters (UHF 915 MHz) (Finzenkeller 2002).

Considering that, the passive tags need to be quite close to the reader in order to function properly this would surely eliminate some of the shortcomings of identifying items with barcode. But the main problem of only being able to measure at certain checkpoints and not being able to see what happens between the measuring points would still remain. Here the active RFID tags come into play because they cannot only be identified but also offer the functionality to be localized - much farther away from the antenna than a passive tag.

There exist different methods to do the localization such as signal strength or signal travel time measurement. One that is similar to GPS (Global Positioning System) functionality is the latter method.

A tag is programmed to send out a signal (beacon) in certain intervals e.g. every five second. This signal is now picked up by antennas that are installed in an specific area. Due to the different relative position of the tag to each of the antennas the signal needs different amounts of time to travel to each of these antennas. Once the signal is picked up a timestamp is put together with the *tagID* and this information is passed on to a server which then calculates the tag's position. The feasible accuracy of this method depends on a series of factors and can go from a few centimeters up to some meters.

With such kind of technological support it is now possible to track every single item in a real manufacturing system almost continuously. At any given point in time the

information about the current position and status of a good is available. Things such as unobserved disappearance of an item or product from the production line and therefore an erroneous initialization of the simulation model cannot happen anymore. Every abnormality and plan variance regarding the position of an item can be detected.

Furthermore, each item can keep track of its processing history with the help of the RFID tag's memory. This information can also be used to initialize the simulation model correctly.

5 APPLICATIONS IN AUTOMOTIVE INDUSTRY

At the transmission plant Rastatt of the DaimlerChrysler AG several branches of production shall be provided with a SEWS as an experiment to verify the usability, benefits and potentials of SEWS in the automotive industry.

The manufacturing process in transmission production basically consists of production areas, heat treating areas and assembly lines. Production areas manufacture important components such as gears and shafts. These production areas are divided into the divisions machining before and after heat treating. The last step in the process is the assembling of manufactured parts (Figure 8).

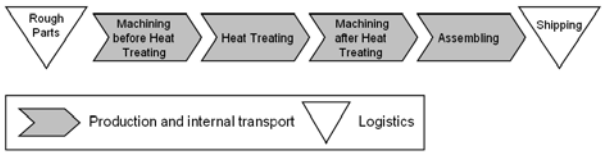

Figure 8: Manufacturing Process in Transmission Production

For every production area significant case studies have been identified. Representative for machining areas the heavy duty machining precursor of heat treating was chosen. The current system state is evaluable over the connection to a control center database.

The case study heat treating analyzes the effects of different part distributions on an hardening stove and the respective downstream processes. Figure 9 shows the simulation model of the hardening stove A28 which is developed in AutoMod.

The FSG assembly (FSG: Frontschaltgetriebe) analyzes the integration of measuring points to initialize simulation models.

Generally, simulation models which are implemented in SEWS have to reproduce the process system of the real material flow system. They have to be able to import or to read initialization data to assign these to relevant system variables. The Simulation objects representing the products have to be created and infiltrated into the process system. Corresponding to the last measuring point they passed in the real world system. After that, simulation runs forecast future system states for a specific forecasting horizon. Finally, at the end of the simulation run or at specific points in time the results have to be exported as XML-files.

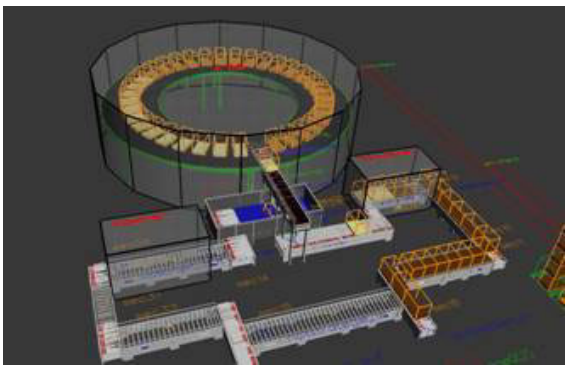

Figure 9: AutoMod Simulation Model of the Hardening Stove A28

The common ground of every representative case study is that the SEWS are supported with data from production planning and production control data bases. An efficient and integrated data handling is a very important point and must not be neglected.

6 CONCLUSION AND FUTURE WORK

Simulation-based Early Warning Systems support decision-maker at the management of complex systems. The introduced architecture and the interconnection mechanism of components provide a simulator-independent integration of simulation models. One important advantage of this approach is the usability and the insertion of already existing simulation models.

In contrary to SEWS simulation-based monitoring systems detect incurred exceptions in real systems and evaluate predefined action alternatives via simulation runs. It is just a reactive behaviour which tries to mitigate or to smooth out exceptions that have occurred already. In addition SEWS have to generate and evaluate action alternatives in an intelligent manner e.g. for this optimization algorithms can be combined with already evaluated action alternatives deployed in similar cases and exception classes respectively. To fulfil the requirement of performance an efficient limitation of simulation runs must not be neglected at the generation of alternatives. The better and the more realistic alternative proposals are made the more SEWS are accepted.

Future work will focus on the development of such dedicated components and the usage of SEWS in the industrial every day life.

REFERENCES

Banks, J. 1998. *Handbook of simulation: principles, methodology, advances, applications and practice*. New York: John Wiley & Sons.

Banks, J. 2000. Simulation in the future. In *Proceedings of the 2000 Winter Simulation Conference*. Eds. J.A. Joines, R.R. Barton, K. Kang, P.A. Fishwick, Picataway, NJ: IEEE, pp.1568-1576.

Brose, G., A. Vogel and K. Duddy. 2001. *Java programming with CORBA. advanced techniques for building distributed applications*. New York: John Wiley & Sons.

Finzenkeller, K. 2002. *RFID-Handbuch: Grundlagen und praktische Anwendungen induktiver funkanlagen, transponder und kontaktloser chipkarten*. Munich: Carl Hanser Verlag.

Graham, S., S. Simeonov, T. Boubez, D. Davis, G. Daniels, Y. Nakamura and R. Neyama. 2002. *Building web services with java: making sense of XML, SOAP, WSDL and UDDI*. Indianapolis: Sams Publishing.

Greulich, W. and S. Barnert. 2003. *Der brockhaus computer und informationstechnologie: hardware, software, multimedia, internet, telekommunikation*. Mannheim, Leipzig: Brockhaus.

Hanisch, A., J. Tolujew, and T. Schulze.2005. Initialization of online simulation models. In *Proceedings of the 2005 Winter Simulation Conference*. Eds. M. Kuhl, N. Steiger, F. Armstrong, J. Joines, Picataway, NJ: IEEE, pp. 1795-1803.

Hotz, I. and T. Schulze. 2006a. Simulationsbasierte frühwarnsysteme – definition, anforderungen, architektur. In *Simulation und Visualisierung 2006*. Eds. T. Schulze, S. Schlechtweg, V. Hinz, SCS European Publishing House, Erlangen 2006, pp. 63-77.

Hotz, I. and T. Schulze. 2006b. Flexible generation of reports for simulation-based early warning systems using XML. In *ECMS 2006 Bonn*.

Lee, T.Y. and Y. Luo. Data exchange for machine shop simulation. In *Proceedings of the 2005 Winter Simulation Conference*. Eds. M. Kuhl, N. Steiger, F. Armstrong, J. Joines, IEEE, New York, pp. 1446-1452.

Livingston, D. 2002. *Advanced SOAP for web development*. Upper Saddle River, NJ: Prentice Hall PTR.

Lovell, D. 2002. *XSL Formatting objects*. Indianapolis: Sams Publishing.

Raape, U., I. Simonis I. and T. Schulze.2005. Concepts and applications of spatiotemporal interoperability in environmental and emergency management. In *Proceedings ITEE'2005*. Eds. Filho,L. , Marx Gomez J. und C. Rautenstrauch., Shaker Verlag, Aachen 2005, pp. 535-551.

Röhl, M. and A.M. Uhrmacher. 2005. Flexible integration of XML into modelling and simulation systems. In *Proceedings of the 2005 Winter Simulation Conference*. Eds. M. Kuhl, N. Steiger, F. Armstrong, J. Joines, Picataway, NJ: IEEE, pp. 1813-1820.

W3C World Wide Web. 2006. XML [online]. Available via <www.w3.org> [accessed March 10, 2006].

AUTHOR BIOGRAPHIES

INGO HOTZ is working as a Ph.D. student in the production planning department of the transmission plant Rastatt, which belongs to the plant Gaggenau of the DaimlerChrysler AG. After the university degree in industrial engineering and management at the University of Karlsruhe in 2004 he is preparing his Ph.D. thesis at the University of Magdeburg. His main tasks are material flow simulation as well as developing digital planning methods. His e-mail address is <ingo.hotz@daimlerchrysler.com>.

ANDRÉ HANISCH is a research engineer in the Department for Logistics Systems and Networks at the Fraunhofer Institute for Factory Operation and Automation in Magdeburg, Germany. His Diplom thesis completed in January, 2002 dealt with "Groupware Components for Collaborative Work in a Web-based Mining Simulation Center". He started to work with RFID technology in 2002 and was strongly involved in setting-up the Fraunhofer IFF Logmotion Lab (www.logmotionlab.de) which is one of the biggest RFID test and evaluation centers in Germany. His e-mail address is <andre.hanisch@iff.fraunhofer.de>.

THOMAS SCHULZE is a Professor at the School of Computer Science at the Otto-von-Guericke-University, Magdeburg, Germany. He received the Ph.D. degree in civil engineering in 1979 and his habil. degree for computer science in 1991 from the University of Magdeburg. His research interests include modeling methodology, public systems modeling, manufacturing- and distributed-simulation with HLA. He is an active member in the ASIM, the German organization of simulation. His e-mail address is <tom@iti.cs.uni-magdeburg.de>.