Proceedings of the 2012 Winter Simulation Conference
C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A.M. Uhrmacher, eds

APPLYING MODEL-RECONSTRUCTION BY EXPLORING MES AND PLC DATA FOR SIMULATION SUPPORT OF PRODUCTION SYSTEMS

András Pfeiffer, Botond Kádár
Gergely Popovics, Csaba Kardos
Zoltán Vén, Lőrinc Kemény, László Monostori

Fraunhofer Project Center on Production Management and Informatics at
Computer and Automation Research Institute, Hungarian Academy of Sciences
13-17 Kende str., Budapest, H-1111, Hungary

ABSTRACT

The paper introduces a discrete-event simulation-based decision supporting system aiming at automatically mirroring the current state of a large-scale material handling system of a production system in a digital model and supporting the analysis of diverse control settings and rules. The discrete-event digital model is built in an automated way and all the data necessary for the model are taken from a manufacturing execution system (MES) and additionally directly from programmable logic controllers (PLC). Main focus is given to present the results of the PLC program code processing method (code mapping) generating a structured dataset, as well as the model-reconstruction method for the simulation software. The easy-of-use support tool is intended to be used both in planning and operation phases of an automotive manufacturing company, thus the capabilities of model reconstruction and simulation are tested on real-world data.

1 INTRODUCTION

1.1 Simulation-based Support of Production Systems

Simulation technologies are often used in supporting production control decisions and this is also particular for large-scale manufacturing systems. Several different applications of discrete-event simulation (DES) models in the control of manufacturing systems were presented in the two well-known text books Banks (1998) and Law and Kelton (2000). The simulation models used for supporting or evaluating production control decisions, generally, represent the flow of materials to and from processing machines and also the operations of machines themselves (Rabelo et al. 2003).

Simulation models may capture the relevant aspects of a production system which cannot be represented in a deterministic optimization model. The most important topics in this respect are the uncertain availability of resources, uncertain processing times, quality of the raw material, unpredictable human behavior and insertion of conditional operations into technological routings (e.g. rework). Moreover, in case of a simulation model is used for supporting short-term or real-time decisions, it is particularly important to have quick response time. For instance, regarding production control, the closer we are to the short-term plans the more detailed simulation model is needed to accurately support factory control and planning decisions (Scherer 1998). However, detailed models will have lengthy, unacceptable response times. Consequently, special modeling logics are required to resolve this contradiction.

A further issue related to the usage of the simulation in real-time control decision-making is the ability to map the exact state of the physical system into the digital world. This means that before each simulation run, the simulation model has to be parameterized at runtime in a way to represent the state of the manufacturing system under analysis (i.e., the products under process on each machine tool, the state of each machine tool, the buffers queued with the products waiting for the process, batch sizes, allocated operators to machine tools and their calendar settings, etc.)

Summing up the related issues of closely connecting simulation with the physical world in large-scale manufacturing systems, it can be stated that the three key issues of simulation are as follows

1. Data acquisition and validation for simulation input,
2. Quick response time of simulation runs and analysis,
3. The ability of creating the snapshot of the physical system status in the simulation model by instantaneous feedback (Monostori at al. 2007).

1.2 Automated Simulation Model Building

Discrete event simulation is one of the most widely spread techniques to evaluate various aspects of a manufacturing or logistics system (O'Rielly and Lilegdon 1999). However, the design phase of a simulation project needs great resource expenditures. On the other hand, simulation is applied to long-term planning, design and analysis of manufacturing systems. These models are termed "throw away" or "stand-alone" models because they are seldom used after the initial plans or designs have been finalized. As stated by Ryan and Heavey (2006) the most commonly used rule of a simulation project is the so called "40-20-40 rule". The rule states that time spent developing a simulation project can be divided as follows: 40% to requirements and data gathering, 20% to model translation and 40% to experimentation.

Time-consuming requirements gathering phase contains input data collection and preparation. Significant planning time reduction can be achieved by automating data gathering and preparation.

Several approaches have been used for automating simulation model buildup by automatic input data gathering and processing. As opposed to the "traditional" use of simulation, Son and Wysk (2001) proposed that once the system design has been finalized, the simulation that was used for evaluation could be used as the basis for system control. In their concept simulation is created by using neutral system components, i.e., they made efforts to build simulation models for shop floor control system, generated automatically. Park et al. (2010) suggest a naming rule in programmable logic controllers (PLC) program codes to automatically identify objects and control logic in code giving a basic data set to build simulation model. This approach needs a renaming process on PLC codes if naming rule suggested is not applied. Bagchi et al. (2008) describe a discrete event simulator developed for daily prediction of WIP position in an operational wafer fabrication factory to support tactical decision-making. Model parameters are automatically updated using statistical analysis performed on historical event logs generated by the factory, while "snapshot" of current status of production is generated by using the manufacturing execution system (i.e., aggregated info of PLC).

The most widely spread applications of using PLC codes for generating simulation models aims of verifying PLC codes themselves. Han et al. (2010) propose a prototyping to improve limitations of existing control logic verification methods and ladder programming. The technique proposed by them supports functionality verification of PLC code on low control level. Contrarily, PLC code process method proposed by the authors is for evaluating the effects of changing PLC codes on the overall system.

Several previous studies aimed at reducing the time needed by the development phase of a simulation project of a manufacturing system highlights the importance of this topic. Wya et al. (2011) proposed a generic simulation modeling framework to reduce the simulation model building time. The proposed framework composed several software that contained information of layout and control logic of the modeled objects. According to this approach layout and control logic of the manufacturing system must be designed by the appropriate software.

Data needed to build simulation model of a manufacturing system are available in production database or can be gathered. Nowadays majority of the enterprises are installing automated manufacturing system consisting of PLCs. Subsequently, the topology and the control logic of the manufacturing system needed for a simulation model are inherently kept in these PLCs. Consequently, building of simulation models can be supported by data and control logic extracted from PLC codes.

The paper introduces an ongoing research of PLC program code and historical data processing method that generates a structured dataset that can be used by manufacturing simulation software to automatically create and parameterize a model.

2 NOVEL SOLUTION FOR SIMULATION-BASED SUPPORT OF PRODUCTION SYSTEMS

2.1 Self-Building Simulation

The main goal of development is the enhancement of the simulation-based analysis and control system by eliminating the manual data collection through automatic interfaces, creating a more realistic model of a real production system. Furthermore, the self-building production simulation should provide both, prospective (e.g. locate anticipated disturbances, identify trends of designated performance measures), and retrospective (e.g. gathering statistics on resources) simulation functionalities. Self-building simulation means that the simulation model is built up by means of the combination of the MES data as well as the knowledge extracted from the MES data (e.g. resource and execution model). In addition to the automatic model building feature, main requirement of the solution is to minimize the response time of the experiments and to enable the quasi "real-time" applicability of the simulation.

Regarding the main operation modes of the simulator in the proposed architecture (Figure 1) are as follows:

- Off-line validation, sensitivity analysis of the system. Evaluation of the robustness of system against uncertainties (e.g., different control settings, thresholds and system load levels). Consequently, this scenario analysis can point out the resources or settings which can endanger the normal operation conditions.
- On-line, anticipatory recognition of deviations from the planned operation conditions by running the simulation parallel to the plant activities; and by using a look ahead function, support of situation recognition (proactive operation mode, Figure 1).
- On-line analysis of the possible actions and minimization of the losses after a disturbance already occurred (reactive operation mode, Figure 1), e.g., what-if scenario analysis.

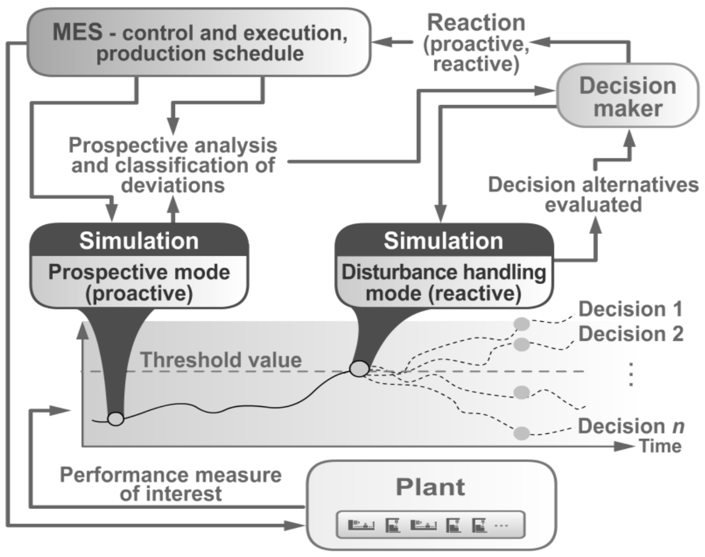

MES - control and execution, production schedule
Reaction (proactive, reactive)
Decision maker
Prospective analysis and classification of deviations
Decision alternatives evaluated
Simulation
Prospective mode (proactive)
Simulation
Disturbance handling mode (reactive)
Decision 1
Decision 2
Threshold value
Decision n
Time
Performance measure of interest
Plant

Figure 1: Plant-level active disturbance handling realized by using reactive/proactive operation modes for simulation (Monostori et al. 2007).

The model structure in the simulator is the same for the three operation modes; however, the granulation (level of modeling details), time horizon, applied failure models and considered outputs depend on the purpose of the experiments.

In the on-line modes the simulation models represent various virtual mirrors of the plants (or logistics systems) and run parallel to the real manufacturing (or logistics) environment, simulating also the future processes for a predefined short period. The performances of the predicted and the so far executed system are compared (highlighted as "Performance measure of interest" in Figure 1).

The off-line operation mode refers to either the factory or individual plants, while in the on-line modes the work of a plant-level decision maker is supported (Figure 1). The main goal of the decision maker is to keep normal operational conditions as far as possible (e.g., daily schedule, or buffer levels) and if it is not possible to minimize the deviances. In case of occurred or predicted disturbances, a decision has to be made, whether to intervene, or not. In the former case an action has to be performed with a limited scope (in space and time) in correspondence to the sphere of authority of the decision maker (e.g., rescheduling or reallocate material flow to different buffers). The control action made in this point incorporates the selection of the appropriate control policy and method, which might be supported by the simulation-based analysis.

2.2 Proposed System Structure

In order to be able to realize the self-building concept regarding the simulation of a production system, relevant input data and model elements have to be formalized and presented, e.g., in a relational database.

Figure 2 represents the dataflow of the (self-building) simulation architecture. The core of the architecture is the main simulation relevant database, in which relevant data are gathered and maintained ("Simulation DB" in Figure 2).

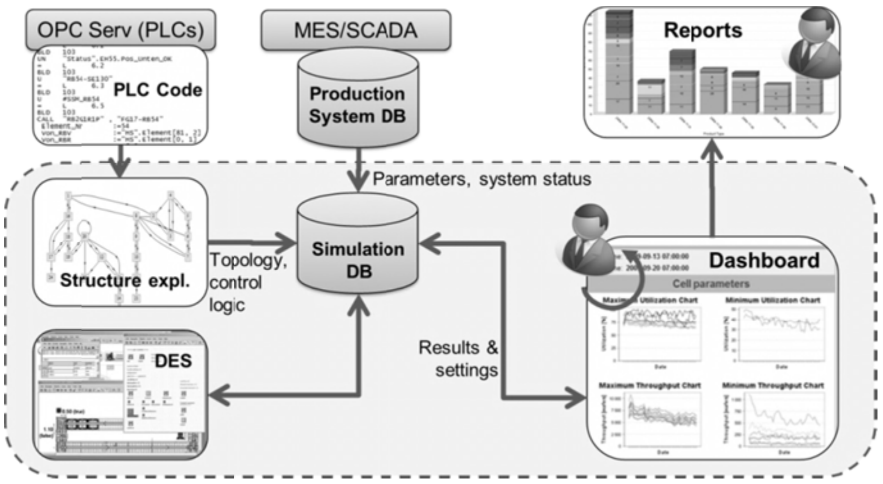


Figure 2: Simplified structure of the proposed system

The physical structure of the system, basically the layout, identifies their dimensions and internal distances, as well as the relations of these elements can be extracted from PLC codes. Even it is not easy to reverse engineer them the topology of controlled system is incorporated in the PLC code (highlighted as "Structure Exploration" in Figure 2). Second essential information of manufacturing systems is the control logic of their elements. Control logic describes the response to be given on PLC's output depending on the input. The control logic consists of structured methods so variables and object's relationship can be transformed to the language of the simulation software.

It is also necessary to parameterize elements of the simulation model. Most of PLC codes do not operate with these kinds of parameters of controlled elements however, (automated) manufacturing systems usually apply MES and/or SCADA system that stores status changes of controlled elements and timestamp of state change events ("Production system DB" in Figure 2). Possible states and parameters of elements, as well as global settings can be retrieved by statistical evaluation accomplished on these data as highlighted by "parameters, system status" in Figure 2.

Data preparation is carried out before the overall simulation. MES production system data is refreshed on a real-time, formalizing the factory snap-shot, while PLC program codes are refreshed only if changes are made on the real physical system. The redundant data (applying "Simulation DB", Figure 2) storage in the simulation model is compensated by the advantage of the shorter response time. Modeling real production systems frequently brings up the problem of handling hundreds of resources in a simulation model. Having the modeling objects in hand, which were created on the base of the conceptual model, in our architecture the simulation model is created automatically based on the pre-processed data (highlighted as "DES" in Figure 2).

The automatic generation of the model is followed by initialization. There, besides classical parameter settings, the procedure involves the generation of input parameter specific model components (entities such as products, tools, machines and the snapshot of the system to be modeled, more detailed description can be found in (Pfeiffer et al. 2009)). Contrary to the previous phase, this one is carried out for each replication. The simulation model incorporates a number of control settings (e.g., thresholds values for buffer elements or dispatching rules for alternative machines) with which the simulation runs can be manually initialized by simulation experts ("Dashboard" and "Results&settings" in Figure 2). The simulation is started on the base of these statistics by generating random production orders which cover the product type distribution calculated from the MES database. Naturally, instead of randomly generated orders, the users of the simulation can also provide the input for the simulation model on the base of e.g., real customer order data. The simulation runs are repeated until the required number of replications is obtained. Each replication is a terminating, non-transient simulation run.

In the last phase, the results are evaluated (critical values for defined KPI-s) and the results of the evaluation process are interpreted by the decision-maker (e.g. in form of reports, highlighted as "Dashboard" in Figure 2) who is responsible for taking the necessary actions. Several simulation results and statistics are calculated inside the simulation model and a graphical user interface (GUI) is provided for the visualization via a web browser of both, input settings, and statistical results ("Reports" in Figure 2).

2.3 Model-Reconstruction Method using Program Code Exploration

As mentioned above, several types of data are needed to build up a simulation model. In this section the PLC program code mapping procedure is explained more in the details, as the most important part of the automated input data preparation method, described previously in section 2.2.

2.3.1 Variable and Value Identification of the PLC Code

The PLC based data acquisition was carried out after a detailed inspection of the PLC program code, in order to determine the blocks of the code, which are essential for the model building. During the exportation of these blocks a suitable data format has to be chosen, which offers high level of data consistency and simple accessibility to the data stored. Considering these requirements Instruction List (or AWL) programming language was chosen as a textual export format. Since it is a low level programming language, it has a strict syntax which allows less difference in the code, which is desirable for further processing of the code.

However, the file format of the exported data is plain text, which contains no markup information about the structure of the code and therefore it has to be included. This was accomplished by applying grammar analysis. The grammar analysis is based on a grammar which comprises the rules and class definitions concerning with the PLC code. An appropriate grammar is closely related to the IEC 61131-3 standard (IEC) and the analyzed PLC code therefore, it can be used to parse the PLC code and to create a structured set of data, which highlights the desired information for model building. Grammar analysis can be applied on different PLC codes and to be extended dynamically, which makes it able to be a part of automated model generation. A software called *ProGrammar* was used to develop the grammar.

The test PLC code as all the PLC codes consists of blocks. One block describes the behavior of one actuator (e.g. electric engine actuating the conveyor). Each block consists of two main parts, as highlighted in Figure 3:

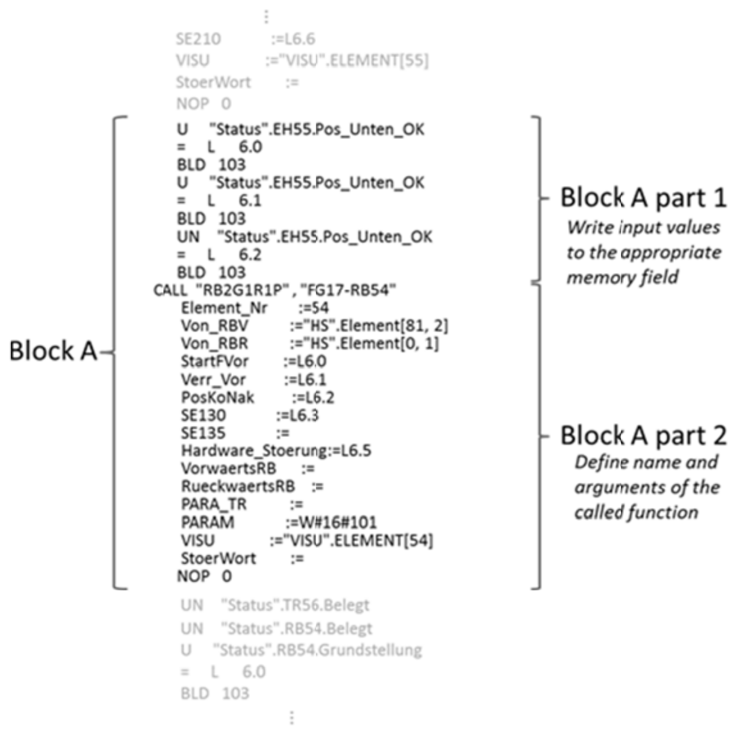
- In the first part the logical conditions and the corresponding data operations are stored.
- The second part of the block is for function block call. The arguments belonging to the call are the input parameters which are needed to instantiate a block.

The grammar analysis results a structure called parse tree, which is a graph object that is composed of the elements defined in the grammar. According to the scope of the analysis these elements can reference various type of information: from the logical rules of operation to the logical or physical connection between the elements.

In order to extract the required information from the parse tree its further processing is needed. This can be achieved by a program, which is able to handle the parse tree and supports transferring the processed results to a simulation environment. A technical computing software (*Wolfram Mathematica*) is applied to process the parse tree, which is able to parse the input text by using a parsing engine of the *ProGrammar* by an API.

In the paper a method is introduced which is purposed to reconstruct the physical topology of a conveyor system. Therefore, the processing of the parse tree was executed in order to acquire the information which is essential to reconstruct the connections between the elements of the conveyor system,

Since the connections of the conveyor system can be represented by a directed graph, the data which describe the connections can be stored in a matrix. Identifying all the elements of the system and their connections makes it possible to create the connection reference matrix. However, this can be difficult and requires general purpose methods, which are able to handle the code-specific differences of the input data and that is the point where the flexibility of the created grammar is fundamental.



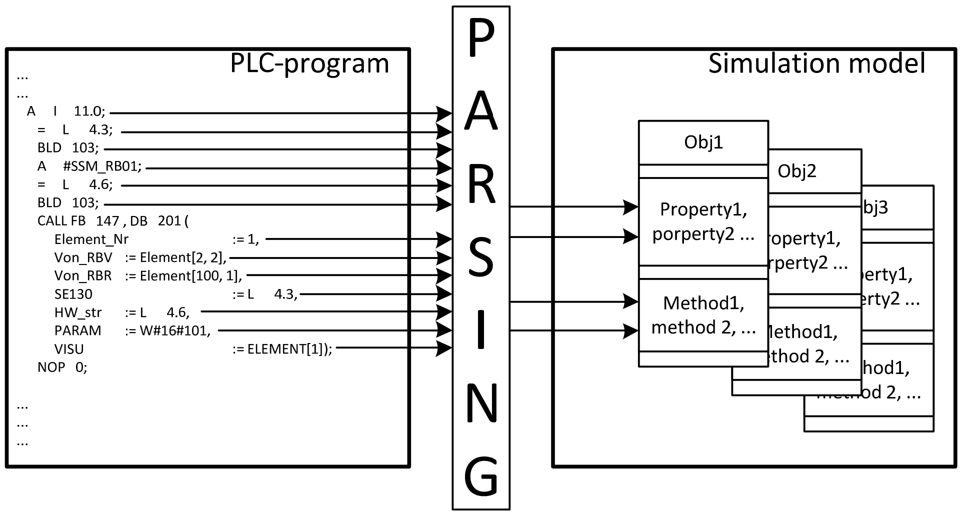
```
                ⋮
SE210         :=L6.6
VISU         :="VISU".ELEMENT[55]
StoerWort      :=
NOP  0
U    "Status".EH55.Pos_Unten_OK
=    L    6.0
BLD  103
U    "Status".EH55.Pos_Unten_OK
=    L    6.1
BLD  103
UN   "Status".EH55.Pos_Unten_OK
=    L    6.2
BLD  103
CALL "RB2G1R1P" , "FG17-RB54"
  Element_Nr      :=54
  Von_RBV        :="HS".Element[81, 2]
  Von_RBR        :="HS".Element[0, 1]
  StartFVor      :=L6.0
  Verr_Vor       :=L6.1
  PosKoNak        :=L6.2
  SE130         :=L6.3
  SE135         :=
  Hardware_Stoerung:=L6.5
  VorwaertsRB     :=
  RueckwaertsRB   :=
  PARA_TR        :=
  PARAM         :=W#16#101
  VISU         :="VISU".ELEMENT[54]
  StoerWort      :=
  NOP  0
  UN   "Status".TR56.Belegt
  UN   "Status".RB54.Belegt
  U    "Status".RB54.Grundstellung
  =    L    6.0
  BLD  103
                ⋮
```

Block A
Block A part 1
Write input values to the appropriate memory field
Block A part 2
Define name and arguments of the called function

Figure 3: PLC program code part in instruction list format

2.3.2 Program Code Mapping

Automatic model reconstruction, in this context, can be considered as a mapping, which assigns the corresponding information – extracted from the PLC program – to properties of the simulation object instances. Properties such as identifier, type, physical and logical connections and other additional parameters of the instances are extracted from the PLC program during the parsing process (Figure 4). However, not all of the required information is accessible for the parser. This is mainly because of the large number of device level connections, which are therefore not represented explicitly in the relatively higher level PLC program. Including this missing information into the simulation model requests the application of alternative methods.



```
...
...
 A    I    11.0;
   =    L     4.3;
   BLD   103;
   A    #SSM_RB01;
   =    L     4.6;
   BLD   103;
   CALL FB   147 , DB   201 (
      Element_Nr                        := 1,
      Von_RBV    := Element[2, 2],
      Von_RBR    := Element[100, 1],
      SE130                             := L     4.3,
      HW_str     := L      4.6,
      PARAM      := W#16#101,
      VISU                              := ELEMENT[1]);
   NOP   0;

...
...
...
```

Figure 4: Basic concept of PLC program code mapping

The most important property of a model element instance is its type, because it essentially determines the behavior of the object. In the PLC program each element is referenced by a unique identifier number. In the declaration part of the program, the identifiers and the element types are matched. Based on this assignment a map structure can be created, where the type of each element can be looked up. However, there are certain references in the PLC program, where the elements cannot be identified unequivocally. The lack of the identifier refuses using the map structure to obtain the type of the element and causes the appearance of disconnected sequences of elements in the model.

Analyzing the system in consideration in details shows that the occurrence of these "unidentified elements" are closely related to their neighbor elements, which together form a straight sequence of elements, a so called pattern. These patterns also contain the previously unidentified elements hence, they can provide the missing links between separated sequences of the model. Since the recognition of these patterns is based on a priori knowledge about the system, defining them is an input parameter of the reconstruction.

Extending the model with the data gained from patterns has disadvantages as a pattern contains no information regarding the direction of the connections between the elements comprised. To avoid collision with already reconstructed sequences, each pattern is assumed as bidirectional. Hence, except the case when a pattern is in fact bidirectional, it is leading to including non-existing connections, which can be confusing. Therefore, removing these unused connections is desirable.

Analyzing the graph of connections it turned out that in most cases one end of the pattern is connected to a "regular" sequence of elements, which contains proper directional data. Therefore, it is possible to determine the valid direction inside a pattern of elements by using the available directional data of the surrounding elements. Figure 5 shows the basic idea of removing unnecessary connections.

Figure 5: Considering the directional data between E2 and E1 (or E4 and E5) it can be pointed out that the direction inside the pattern has to match the direction of the elements outside the pattern.

Creating an algorithm, which is able to clean the graph from the non-existing connections, requires proper identification of elements on the border of each pattern. These elements can be characterized as follows: an element, which has only two neighbor elements, but has two inputs and one output or has two outputs and one input. As it is shown in Figure 5 E2 and E4 fulfill these criterions, respectively. The basic idea is that these elements have three connections, but one can be thrown away. Applying the following steps the connections to be eliminated can be defined .

1. Search for an element with 2 neighbors and 3 connections (2 in + 1 out or 1 in + 2 out)
2. Create two sets of connections through the element: one that points from one neighbor element to the other and one that points backwards.
3. Evaluate the intersection of each set and the set of original connections.
4. The intersection with less member equals the connection to be removed.

Iteratively applying the above described algorithm effects that each unnecessary connection (according to the previously described criterions) could be removed, thus enabling the proper graph representing the network of the connected objects.

2.3.3 Topology Graph Generation

The resulted matrix of the above introduced method is visualized by using the graph plotting features of the *Mathematica* software. The nodes of the directed graph represent the elements of the conveyor system, while the directed edges represent the connections between them (Figure 6). Since the gathered data does not contain information about the layout of the system, the arrangement of the graph was used instead. Nevertheless, by setting the parameters of the elements properly the real operation of the system can be modeled.

Therefore, by gathering the relative coordinate data of each element, the graph can be used to provide element arrangement data to the simulation model. Processing the parse tree results three different data structure, which are the following.

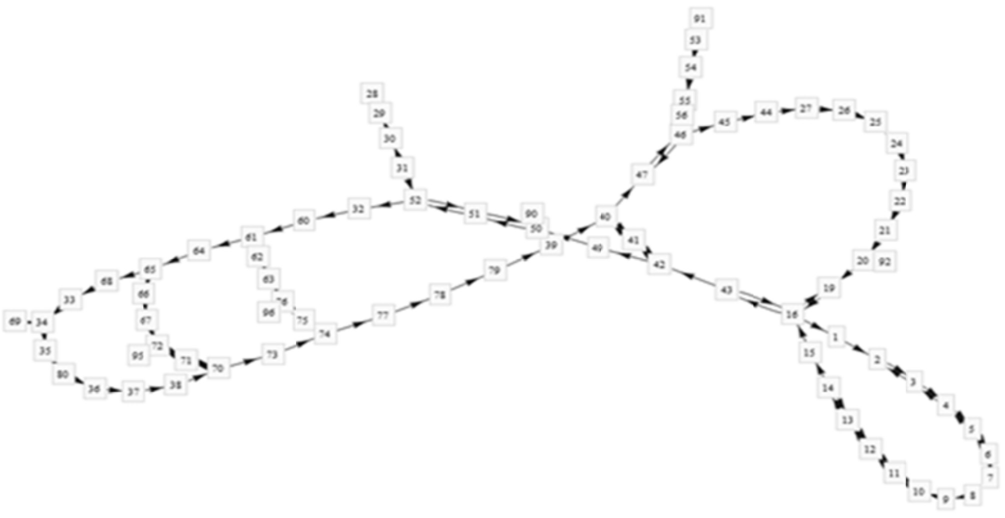


Figure 6: Part of the reconstructed topology graph of a production logistics system.

- A list which contains all the existing elements of the conveyor system. These are the elements to be created in the simulation model.
- A connection reference matrix, which describe the connection between the elements.
- An element arrangement table, which contains the relative coordinates of the elements.

These data have to be transferred to the simulation software, where the simulation model can be built up. The data transfer was carried out by using an SQL-server, which can be accessed by the built-in SQL-client of the software applied. After a conversion of data, the topology of the examined system can be generated by a graph creating command. The data received from parser can also be converted to a connection table form. The simulation software (*Tecnomatix Plant Simulation*) and its internal connection tables and methods are applied to generate the simulation object instances and run the simulation.

2.3.4 Parameterization of the Simulation Elements Generated

The "Production system database" is used by providing parameters of the elements and input data for the simulation model (Figure 2). Directly and indirectly usable data are gathered from MES database, and transformed as well as processed to the format that simulation software is able to apply (Kádár et al. 2010, Pfeiffer et al. 2009.). The supposed simulation element parameterization method is demonstrated by an example.

As test model elements were generated and connected based on the information stored in the low level control devices of a conveyor system it is obvious to calculate the transportation time as the most relevant information of a conveyor element. Production database of the test system stores data of the number of pallets located on each conveyor sections as the function of time (Figure 7).

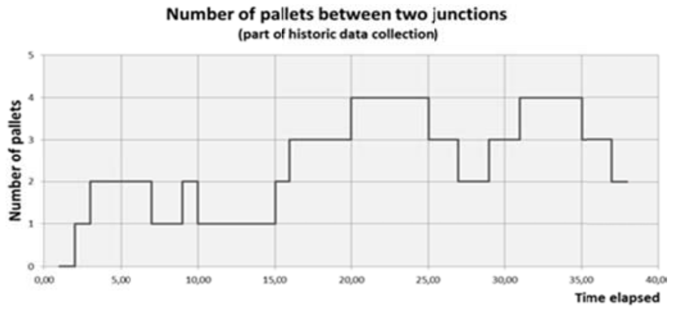


Figure 7: Number of pallets located on a section of the conveyor system as the function of time

In Figure 7 each rising edge represents the event when a pallet enters the observed section and falling edge is dedicated to the exiting event of a pallet. As the sequence of the pallets does not change between two junctions on the conveyor system the entering and exiting time stamp of every pallet can be calculated by a built in method of the simulation software. It also calculates the distribution of the transportation time of every section of the test area and sets it as a parameter of the appropriate simulation element.

3 PRELIMINARY SIMULATION RESULTS – CASE-STUDY

The proposed solution introduced in the previous section is tested on real data of complex conveyor system in a large-scale manufacturing factory by using a part of the real PLC code of the running system. The systems contains approx. 1500 elements and more than 20 PLCs formalizing a hierarchical control structure. Main drawbacks of currently applied simulation methods enlisted by decision makers for testing changes of either physical system or control system are as follows:

- specific building blocks, models of processes are redundantly implemented in separated systems, which results in more development time and costs;

- there is no integrated design and analysis environment or system where the overall system and processes can be analyzed;
- there is no factory-standard, uniform user interface;
- results of current simulations cannot be automatically adapted in real system;
- real-time daily use is not possible (manual changes required).

On the base of the above assumptions, the target of the research is to build a new simulation system in which the simulation model is an inherent part of the control system hence, input comes automatically from real execution system and support is provided to evaluation of simulation results.

This means the collection of data from PLC level, enabling reliable, automatically generated DES models, furthermore, manual intervention regarding simulation (data collection, modeling, parameterizing) can be reduced to a definitely low level. Two main operation modes (detailed in section 2.1) has been identified by the users of the proposed system:

- off-line, i.e., planning level: Simulation is used for analyzing effects of changes in system, and fed with historical input data and relevant production data for parameterization.
- on-line, i.e., execution level: Simulation is applied for short-term "what-if" analysis. In case of certain deviation or disturbance has occurred decision maker is able to test several predefined methods, scenarios for resolving problem.

Preliminary test runs were taken on a simulation model that was built automatically based on the data on the PLC codes of a test area of the above mentioned conveyor system. The generated simulation elements were parameterized based on the historical data of this section by the suggested parameterizing method. Input data was also generated based on historical data and time dependent capacity limitations of the output of the system were implemented in the simulation model as well.

The aim of the test run was to validate the simulation model by comparing historical data and simulated data. The compared value was the number of pallets located in test area. This value is plotted in the gear of time (Figure 8). Ramp-up phase of the simulation run can be recognized at the start phase of observed time interval. In this phase there is significant difference between the real and simulated values because at the initial moment of the simulation run the model does not contain any pallet. After the ramp up phase the trend of the curves are similar that shows the simulated system behavior is similar to the behavior of the real system. As uncertainty raises as the observed time interval of the forecast, this phenomenon can be observed on this graph as well. On the right side of the diagram the curves are deviating, because the uncertainty rises as the function of time.

As the test run revealed it is possible to represent the behavior of a manufacturing system by a simulation model generated and parameterized automatically based on the information stored in low level control devices of the system.

Consequently, these results of simulation experiments can be easily adapted in real factory, thus enabling fast, flexible and smooth changes in factory, e.g., analyzing possible effects of changes in some part of the PLC program code of the control system.

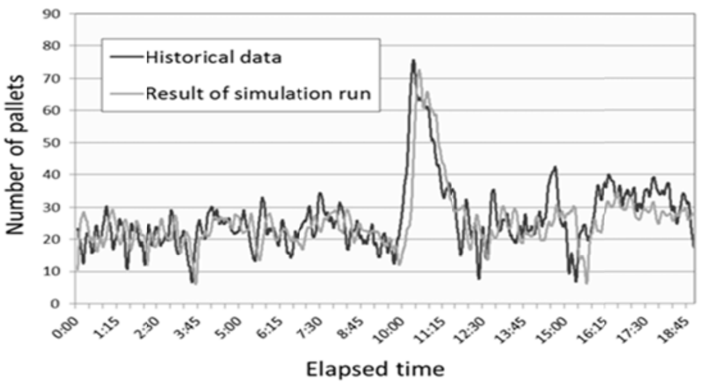


Figure 8: Number pallets located in the observed test area

4 CONCLUSIONS

The paper revealed a discrete-event simulation approach applied for decision support of control related production applications. Design phase is a significant part of a simulation activity; hence reducing time of it significantly affects the effectiveness of the whole simulation. Automated data gathering supporting the buildup of simulation models is a possible solution to achieve this goal and is also a solution to create reusable models. Thus, self-building simulation has been introduced with three main operation modes to be applied for decision support. Moreover, several approaches were studied in the topic and revealed that PLC codes store information needed to build up a simulation model. A new procedure for extracting topology and control logic data of system from PLC codes has been introduced. Data stored in production database were used to parameterize objects of the model and generating input for simulation experiments.

The results introduced in the paper is intended to be applied on a real-world, automated intra-plant logistics system of a manufacturing company in the near future. Two levels of application are considered, planning and execution level.

At the planning level the proposed simulation-based system supports the decisions related to a planning activities, by analyzing possible effects of changes in some part of PLC code of the control system, by critical situation analysis of the overall material handling system, as well as by the evaluation of throughput, bottlenecks, waiting times, number of pallets needed. Contrary, at the execution level the main goal is to support decisions related to the anticipatory recognition of disturbances and to estimate their influences (runtime simulation, monitoring KPI-s). This means testing predefined methods or scenarios for resolving problems and deviations (e.g., buffer settings, routing of pallets).

ACKNOWLEDGMENTS

The research has been partially supported by the Hungarian Scientific Research Fund (OTKA) grant “Production Structures as Complex Adaptive Systems” T-73376 and by National Office for Research and Technology (NKTH) grant "Digital, real-time enterprises and networks", OMFB-01638/2009. The research reported in this paper has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement No: NMP2 2010-228595, Virtual Factory Framework (VFF).

REFERENCES

Bagchi, S., Chen-Ritzo, C., Shikalgar, S.T., Toner, M., 2008, A full-factory simulator as a daily decision-support tool for 300mm wafer fabrication productivity, in Proceedings of the 2008 Winter Simulation Conference edited by S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler, pp. 2021-2029

Banks, J., 1998, Handbook of Simulation, Principles, Methodology, Advances, Application and Practice. John Wiley & Sons Inc.

IEC. Iec 61131-3: Programmable controllers – part 3: Programming languages. Technical report, IEC.

Kádár, B., Lengyel, A., Monostori, L., Suginishi, Y., Pfeiffer, A., Nonaka, Y., 2010, Enhanced control of complex production structures by tight coupling of the digital and the physical worlds, CIRP Annals - Manufacturing Technology, 59, pp. 437–440. (http://dx.doi.org/10.1016/j.cirp.2010.03.123)

Kwan Hee Han, Seock Kyu Yoo, Bohyun Kim, Geon Lee, 2010, Rapid Virtual Prototyping of PLC-Based Control System. ICAI'10 Proceedings of the 11th WSEAS international conference on Automation & information

Law, A., Kelton, D., 2000, Simulation modeling and analysis, McGraw-Hill, New York.

Monostori, L., Kádár, B., Pfeiffer, A., Karnok, D., 2007, Solution approaches to real-time control of customized mass production. CIRP Annals Manufacturing Technology, 56(1), pp. 431–434.

O’Reilly, J.J., Lilegdon, W.R., 1999, Introduction to FACTOR/AIM. In: Proc. of the 1999 Winter Simulation Conference, pp. 201-207.

Park, Hyeong-Tae, Kwak, Jong-Geun,Wang, Gi-Nam, Park, Sang C., 2010, Plant model generation for PLC simulation. International Journal of Production Research 48(5), pp. 1517-1529.

Pfeiffer, A., Kádár, B., Szathmári, M., Popovics, G., Vén, Z., Monostori, L., 2009, Self-building simulation tool for daily decision support in production control, In proc. of the 7th International Workshop on Modelling & Applied Simulation, MAS 2009, 23-25 September, 2009, Tenerife, Spain, pp.: 246-254.

Rabelo, L., Helal, M., Jones, A., Min, J., Son, Y.J., Deshmukh, A., 2003, A hybrid approach to manufacturing enterprise simulation. In: Proc. of the 2003 Winter Simulation Conference, 1125-1133.

Ryan, J., Heavey, C., 2006, Process modeling for simulation, Computers in Industry 57, pp. 437–450

Scherer, E., 1998, The reality of shop floor control – approaches to system innovation, In Shop Floor Control – A System Perspective, Ed. Scherer E., Springer, 3-26.

Son, Y.J., Wysk, R.A., 2001, Automatic simulation model generation for simulation-based, real-time shop floor control, Computers in Industry 45. pp 291-308.

Wya, J., Jeong, S., Kim, B., Park, J., Shin, J., Yoon, H., Lee, S., 2011, A data-driven generic simulation model for logistics-embedded assembly manufacturing lines. Computers & Industrial Engineering. 60(1) pp. 138-147.

AUTHOR BIOGRAPHIES

ANDRÁS PFEIFFER earned his PhD in 2008 at the Budapest University of Technology and Economics. Currently he is a senior research fellow at Engineering and Management Intelligence Laboratory of the Computer and Automation Research Institute, Hungarian Academy of Sciences (EMI, MTA SZTAKI), project manager at the Fraunhofer Project Center at SZTAKI. His current interest includes decision support in production planning and control, as well as the simulation and emulation modeling of complex production systems, self-building simulation systems. Email address: pfeiffer@sztaki.hu

BOTOND KÁDÁR is a senior researcher at EMI, manager of the Fraunhofer Project Center at SZTAKI. He obtained his MSc and Ph.D. degrees at the Budapest University of Technology and Economics, Hungary, in 1993 and 2002, respectively. His current interest includes production control, simulation and multi-agent approaches for production engineering and manufacturing systems and he is involved in several research and development projects from these fields. Dr. Botond Kádár is author or co-author of 70 publications with over 120 citations. Email address: kadar@sztaki.hu

GERGELY POPOVICS graduated in 2008 at the Budapest University of Technology and Economics. Currently he is a research associate at EMI, systems engineer at the Fraunhofer Project Center at SZTAKI. His current interest includes the simulation modeling of complex production systems and automatic identification technologies. Email address: popovics@sztaki.hu

CSABA KARDOS graduated in 2012 at the Budapest University of Technology and Economics. Currently he is a systems engineer at the Fraunhofer Project Center at SZTAKI. The fields of his interests cover simulation of complex manufacturing systems and self-building simulation. Email address: kardos@sztaki.hu

ZOLTÁN VÉN graduated in 2006 at the Budapest University of Technology and Economics. Currently he is a research associate at EMI, project manager at the Fraunhofer Project Center at SZTAKI. His current interest includes reconfigurable manufacturing systems, as well as the simulation and emulation modeling of complex production systems, self-building simulation systems. Email address: ven.zoltan@sztaki.hu

LŐRINC KEMÉNY graduated in 2010 at the Budapest University of Technology and Economics. Currently he is a software engineer at the Fraunhofer Project Center at SZTAKI. Email address: lorinc.kemeny@sztaki.hu

PROF. LÁSZLÓ MONOSTORI acts as Deputy Director Research of SZTAKI, Head of the Engineering and Management Intelligence Laboratory, and Director of the FhG Project Center. He is also full time professor at the Dept. of Manuf. Sc. &Techn. Budapest Univ. of Techn. and Econ. He is a Fellow and Council Member of the International Academy for Production Engineering (CIRP); Full Member of the European Academy of Industrial Management (AIM), International Federation of Automatic Control (IFAC). He is Editor-in-Chief of the CIRP Journal of Manufacturing Science and Technology; Associate Editor of Computers in Industry, as well as the Measurement, and member of the editorial boards of other international scientific periodicals. For his research achievements published in more than 370 publications resulted in about 1800 independent citations and for his development activities – among others – the Dennis Gabor Prize was given to him in 2004. Prof. Monostori is a corresponding member of the Hungarian Academy of Sciences and member of the Hungarian Academy of Engineering. Email address: laszlo.monostori@sztaki.hu