SELECTED ENGINEERING PROBLEMS
NUMBER 5
INSTITUTE OF ENGINEERING PROCESSES AUTOMATION
AND INTEGRATED MANUFACTURING SYSTEMS

Damian KRENCZYK[1*]

[1] Institute Of Engineering Processes Automation And Integrated Manufacturing Systems, Faculty of Mechanical Engineering, Silesian University of Technology, Gliwice, Poland
[*]damian.krenczyk@polsl.pl

THE AUTOMATED PROCESS OF CREATING SIMULATION MODELS

Abstract: In the paper the method of data transformation to a scripting language code creating the simulation model objects used in the implementation of the methodology of production planning and control with discrete simulation systems integration has been presented. For this purpose the so-called transformation templates containing the code of the templates, without a specified value, but together with the placement of the input data have been used. The examples of transformation template notation and the result code generated by the XSLT processor have been shown.

1. Introduction

Increasing the efficiency of the production planning and scheduling process, which in today's enterprises is one of the main areas where on a large scale computer support applications are used, is made possible by the integration of production planning and control (PPC) systems and systems of computer modeling, simulation and visualization of production processes flow [1, 2, 6, 7].

In the previous papers [2, 3, 4, 5], the methodology of integration of production planning and control systems with discrete event simulation systems (DES), based on the methods of data mapping and data transformation has been presented. The main advantage of the proposed approach is the ability to generate the simulation models automatically regardless of the production system structure, which depends on the actual type of production flow.

The structure of the production flow can be changed depending on the produced assortment in the production system, resulting from a package of orders for the planning period. Another distinguishing feature of the proposed method from the solutions available on the market is the flexibility in adapting the proposed solutions to computer aided planning systems and discrete-event simulation systems through the use of the neutral data formats and the neutral intermediate data model. Generated simulation models are given the opportunity to use the data obtained from conducted simulation experiments to support decision-making process in the planning area and should be a very effective tool for verification of production plans and the possibility of order execution.

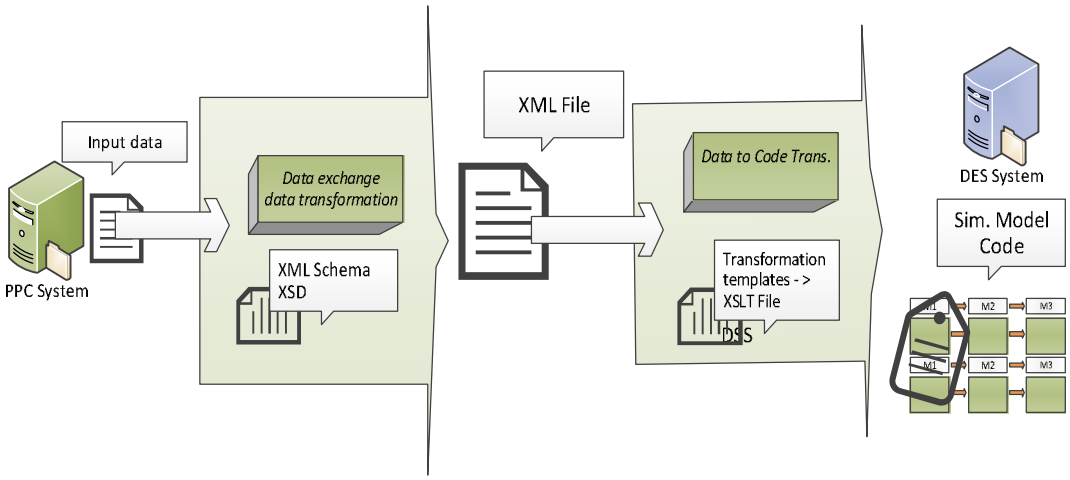

Fig.1. The method of integration of production planning and control systems with discrete event simulation systems

The implementation of the proposed methodology (Fig. 1) requires the identification of data mapping and transformation methods for specific, chosen production planning and simulation systems. The implementation process requires to carry out the process of the data schema mapping between source and target schema for the intermediate data model. Then the process of data transforming between the intermediate model and the target model is performed for the data required in the transformation template process to script code of simulation system internal language. The results of the mapping in the form of logical formulas, so called depending formulas, are the basis to generate a transformation code in extensible style sheet language transformations (XSLT). In this paper the stage of data transformation to script code using a *transformation template* for the two commercial simulation systems has been presented.

2. Transformation template

Generating a script code in the internal simulation system language is a sequential process and must be performed in the order of the individual layers to a defined simulation model. It is not possible, for example, to generate the parameterization code of the cycle-times for manufacturing resources without previous generating the code to create the appropriate amount of production resources resulting from the input data. Data transformation process to the internal code of the simulation system programming language is implemented based on the defined *transformation template* - templates containing code with a specific location of the input data (either directly or using a transformation function). The developed functions correspond to individual layers of code of creation of a simulation model.

The transformation template is defined as three [4]: TSi <[tmp(#1...#n)]; Aj; m>:,

where:

TSi – i-th numer of the transformation template; this number determines the location of the transformation in the sequence,

tmp(#1...#n) – the template which is a text string containing the lines of the code of internal simulation system programming language with tags #1…#n, specifying

the place of entry to a data template; tags can be used to implement arithmetical operations on the data, which correspond to (e.g. #2-#1, #2*#11),

Aj – data corresponding to the tags in the template, j = 1…n,

m – the number of repetitions in the template code.

The transformations templates are defined at the stage of implementation of the methodology, because the templates necessarily correspond with syntax of the internal programming language of the simulation system.

a. Practical example

An example of the transformation which uses 4dScript language of Enterprise Dynamics system that creates an instance of the production resource in the manufacturing simulation model is shown below:

```
  TS10 <[Sets(CreateAtom(AtomByName([Server], Library), Model, [M#1])),
setloc(_(4+7*(#1 - round((#1-1)/#2))*#2)_, _(round((#1-1)/#2+1)*3)_ ,0),
     SetChannels (_(count(#3))_, _(count(#3))_,s)], Id, y_pos,
No_of_Processes, 1>
```

The example of the transformation template, using FlexScript language of FlexSim simulation system, which, like the above, it creates an instance of the manufacturing resource in the simulation model is:

```
  TS10 <[createinstance(node("/Processor",library()),model());
setname(last(model()),"M#1"); setloc(last(model()),_(4+7*(#1 - round((#1-
1)/#2))*#2)_, _(round((#1-1)/#2+1)*3)_ ,0);], Id, y_pos, No_of_Processes,
1>
```

For prepared transformation template may then be generated the resulting code in the data transformation language selected by the user: XSLT, SQL, XQuery, Java, or C++. Generated XSLT code for the above example (for Enterprise Dynamics system) is as follows:

```
  <xsl:for-each select="Production_System/Resources/Resource">
     Sets(CreateAtom(AtomByName([Server], Library), Model, [M<xsl:value-
of select="Id"/>])),
     setloc(<xsl:value-of select="4 + 7 * (Id - ((round((Id - 1) div
$rzad)) * $rzad))"/>,<xsl:value-of select="(round((Id - 1) div $rzad) + 1)
* 3"/>,0),
     SetChannels (<xsl:value-of
select="count(/Production_System/Processes/Process)"/>,<xsl:value-of
select="count(/Production_System/Processes/Process)"/>,s),
  </xsl:for-each>
```

For the other simulation systems creation of transformation templates is carried out in a similar manner, and its content is also dependent on the adopted neutral data model.

b. Simulation model objects code

By using data, describing the production system and manufacturing processes, stored in the intermediate data model and above transformations templates and generated on the basis of their XSLT code, program code containing instructions for creating a fully functional

simulation model can be generated. For this purpose, it can be used on any XSLT processor software, eg. XMLSpy, Sablotron for C++, XSLT for PHP, etc. The following is a fragment of the generated 4DScript code for creating objects in the simulation model:

```
      Sets(CreateAtom(AtomByName([Server], Library),  Model,  [M2])),
      setloc(18,3,0),
      SetChannels (5,5,s),

SetExprAtt(1,[czasy(Value(StripString(Name(First(c)),[P])),Value(StripStrin
g(Name(c),[M])))],s),
   SetExprAtt(2,[Value(StripString(Name(First(c)),[P]))],s),

SetExprAtt(18,[openic(lrrkz(Value(StripString(Name(c),[M])),+(mod(input(c),
lrrkz(Value(StripString(Name(c),[M])),299)),1)),c)],s),
   OnReset(s):=[Do(Inherit,CloseAllIc(c),InStrategy)]
```

The full versions of the models and files presented in this article, can be found on the website: imms.home.pl/RapidSim.

3. Conclusion

Approach and software tools presented in this paper make it possible to directly generate the simulation models of production systems. The implementation of the methodology of PPC and DES systems integration, which uses the presented methods is applicable to data acquired from PPC systems, regardless of the structures of the production system, the topology of the processes flow in the system or the amount of resources and production orders. Further research on the methodology presented in the article addresses issues related to the area of the development of the concept of virtual enterprises and dynamic manufacturing networks.

References

1. Drake G. R., Smith J. S., Peters B. A.: Simulation as a planning and scheduling tool for flexible manufacturing systems. Proceedings of the WSC 1995, pp. 805–812.
2. Krenczyk D.: Data-driven modelling and simulation for integration of production planning and simulation systems. Selected Engineering Problems, no. 3, 2012, pp. 119–122.
3. Krenczyk D.: RapidSim – software for supporting PPC and DES systems integration. Selected Engineering Problems, no. 4, 2013, pp. 119–122.
4. Krenczyk D.: Integration of production planning and discrete event simulation systems (in polish), Wydawnictwo Politechniki Śląskiej, Gliwice, 2013.
5. Krenczyk D., Skołud B.: Production preparation and order verification systems integration using method based on data transformation and data mapping. Lecture Notes in Computer Science, vol. 6697 2011, part II,. 297-404.
6. Lee S., Son Y.-J., Wysk R. A.: Simulation-based planning and control: From shop floor to top floor. Journal of Manufacturing Systems, vol. 26, no. 2, 2007, pp. 85–98.
7. Sihn W.: Simulation-based configuration, animation and simulation of manufacturing systems. Progress in Virtual Manufacturing Systems, pp. 215–218, 2003.