New Features in the ProC/B toolset

M. Arns, M. Eickhoff, M. Fischer, C. Tepper and M. Völker
Dep. of Computer Science, University of Dortmund
D-44221 Dortmund, Germany
e-mail: {arns,eickhoff,fischer,tepper,voelker}@ls4.cs.uni-dortmund.de

***Abstract*— This paper presents the ProC/B toolset for modeling and analysis of logistics networks. The toolset includes a graphical user interface (GUI) which supports the specification of logistics models and controls subsequent model analysis including simulative, algebraic and numerical techniques. We briefly sketch recent advances and extensions that found their way into the ProC/B toolset. In addition to several improvements of the ProC/B paradigm, a module for detection of steady-state in simulative models is integrated. Another extension is aggregation of ProC/B sub-models into equivalent ProC/B models that are much more simpler than the originals. These so-called aggregates can be calculated through simulative, algebraic and numerical analysis.**

I. Introduction

Process-oriented modeling has attracted attention by the management of business processes, workflow, supply chains, logistics networks etc, since enclosed processes are being re-designed and fine-tuned in ever shorter time-cycles in order to maintain considerable performance and flexibility by means of adapting to dynamic environments. Several authors [10] have pointed out, how model-based performance analysis using formal techniques may contribute to this challenging task, in particular, how models can be used to answer what-if questions. Aforementioned application areas share a common view of systems and fall into the class of discrete event-driven dynamic systems (DEDSs) [6] providing a rich repertoire of techniques for determining characteristics of dynamic systems.

Within the last 6 years, the collaborative research center ‘Modeling of Large Logistics Networks’ at the University of Dortmund came up with the ProC/B toolset [2], [1] with capabilities for modeling and simulative, algebraic and numerical analysis of process-oriented systems. The toolset provides a graphical user-interface for the generation of ProC/B models and controls subsequent model analysis. At present, the HIT modeling environment [7] and the APNN-toolbox [5] are available and render access to discrete event simulation (→HIT), to familiar Queueing Network techniques (→HIT) and to Generalized Stochastic Petri Nets (GSPNs) with related Kronecker-based numerical techniques for underlying Markov chains (→APNN-toolbox).

In the following we highlight certain recent extensions with respect to improvements of the ProC/B paradigm, a module for detection of steady-state behavior in simulative models. Another extensions is aggregation of ProC/B sub-models in equivalent ProC/B models. The main idea of aggregation is to replace suitable sub-models with simplified/aggregated descriptions in order to reduce complexity substantially and/or to eliminate properties of the model that prevent from certain analysis techniques.

This research is supported by DFG, collaborative research center 559 ’Modeling of Large Logistics Networks’

II. Model Specification

In the logistics context there is a paradigm for modeling networks (see [8]) with the aspects of physical layout, used resources, control rules, management policies etc. This paradigm describes the sequence of activities of a logistics system by using so-called process chains, also including influencing parameters. However, the major purpose of such models is to offer a description of logistics networks. So, to get a paradigm that allows for network analysis by using an automatic mapping to several analysis techniques, one needs to develop a subset of the complete process chain paradigm that includes all relevant information for evaluating technical and economical measures. This precise paradigm will be depicted in brief in the following and new features will be introduced.

A. The Process Chain Paradigm

The main structuring elements of the Process Chain Paradigm (see [2]) are *Functional Units* (FUs), that are built of other FUs and *Process Chains* (PCs) resp. *Process Chain Elements* (PCEs). FUs describe the behavior and the components of a logistics entity, i.e. departments of a factory with their sub-departments, controlled resources, the sequence of activities in their work and which work could be provided resp. given to other elements in the model. As the sub-departments as well as the super-departments have the same general structure, the Process Chain Paradigm is based on a hierarchical FU structure with every layer built with a single set of elements for all of them.

Each PC starts at a source and ends at a sink. Sources and sinks build an interface to the model layers above resp. define the load given to a PC and the point of destroying a process. The graphical representation of the activity flow uses arrow-like hexagons called *Process Chain Elements* (PCEs) each one denoting one activity. Horizontal connections between the PCEs indicate the logical sequence of their executions. Branches into and merges from alternative sub-chains as well as

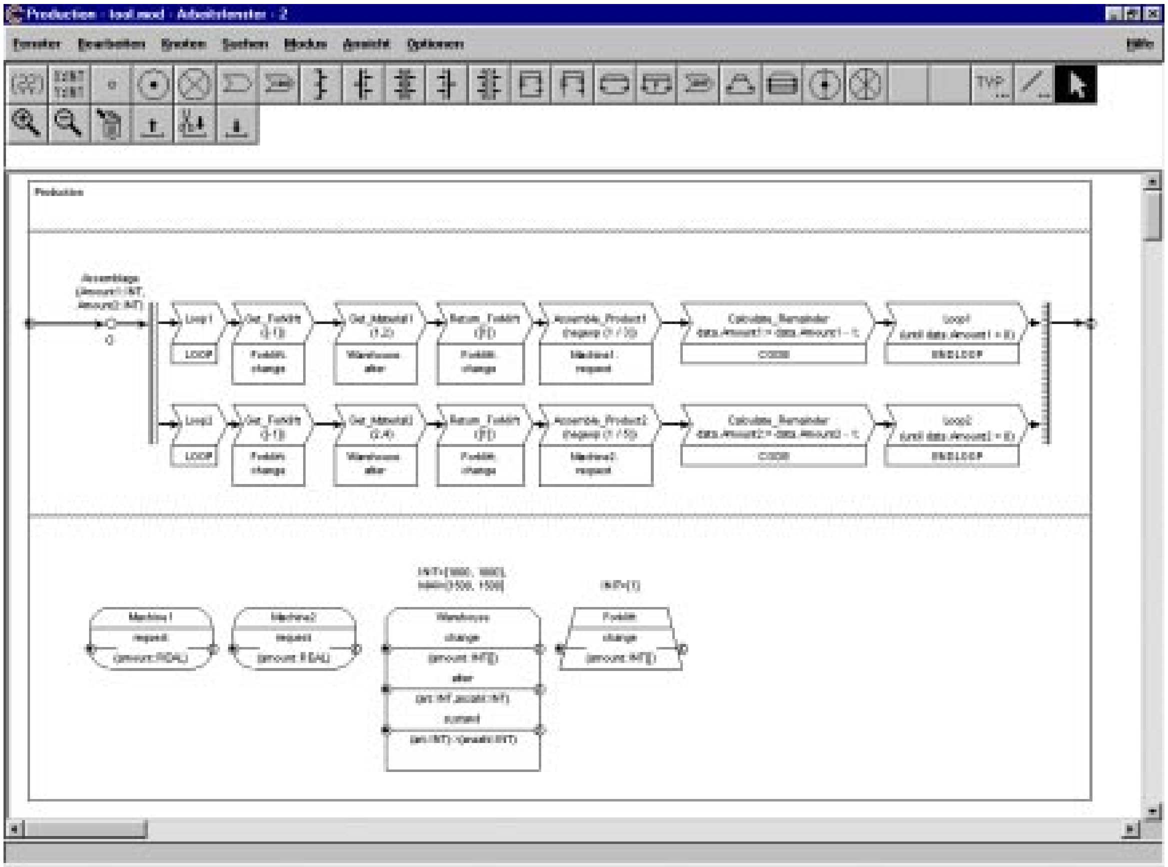


Fig. 1. A screen-shot of a Process Chain using LOOP Elements.

concurrent executions of sub-chains can be realized by various forms of connectors (graphically represented by vertical bars).

As mentioned FUs may contain other FUs. These sub-FUs may be self-defined as well as predefined FUs, i.e. counters, servers or stock modules that close the hierarchy from the bottom and have a predefined behavior and predefined services and mainly manage basic time and space consumption. Sub-FUs can be used by causing the execution of the activities inside the Sub-FUs. The hierarchy is closed at the topside with FUs that do not provide services to the outside but contain sources that generate processes at certain times or recurrently after time intervals. By this means the user is able to define the load that has to be managed by the modeled system.

B. New Model Features

In order to easily allow modeling repetitions with variable numbers of cycles the *LOOP Element* (cf. Fig. 1) was introduced to the paradigm. It consists of two parts, one denoting the beginning of the loop and one denoting the end of the loop as well as testing if the activity flow shall leave the cycle. The leaving is controlled by means of a condition consisting of a boolean expression which allows to determine the number of cycles through the loop with the actual value of parameters. Needing loops was typically caused through describing logistics processes in detail that join or split identical parts appearing with a variable number.

In order to support a wide variety of cost accountings it has been made possible to use so-called *Rewards*. *Rewards* are self-defined measurements allowing the user to get e.g. values of the throughput at any point of the model, of the population at any sector of the *Process Chains* or of the amount of object flow at any point of the model. The observable values may be the same as for standardized measurements i.e. mean values, standard deviations, confidence levels.

The service offers of the *Stock Modules* were extended. Typical situations in logistics systems are out-of-stock-situations, i.e. empties are taken back on the return journey of a delivery only if there are more empties ready for return than a lower threshold because it would be too expensive to load just a few of them or to let a transport means wait for returnable empties. On the other hand, it cannot be carried more than the transport means is able to load. So the amount of the return load is between a lower and a higher level or zero. These situations are supported by a new service of the *Stock Modules* that also offer standardized measurements i.e. of the difference between the optimal level and the reached level of objects.

Also the capabilities of the server have been extended to offer the possibility to use priorities for the service demands. Also preemptive and non-preemptive strategies are supported.

III. Analysis Techniques

The ProC/B toolset has been developed for rendering access to formal techniques for both modeling of process-oriented systems (supply chains, logistics networks etc.) and their quantitative/qualitative analysis. An important observation is that many of the aforementioned application areas share a common view of systems and fall into the class of discrete event-driven dynamic systems (DEDSs) [6]. DEDS provide a rich repertoire of formal techniques for determining quantitative as well as qualitative characteristics of dynamic systems. For ProC/B model analysis, the toolset is connected with several independent analysis engines. This is accomplished with the help of transformers that translate ProC/B models onto the input language of selected analysis engines. At present, HIT [7] and the APNN-toolbox [5] are available rendering access to discrete event simulation (→HIT), to familiar Queueing Network techniques (→HIT) and to generalized stochastic Petri nets (GSPNs) with related functional and quantitative techniques based on the underlying Markov chains (→APNN-toolbox).

In comparison with the expressiveness of general ProC/B models, the familiar restrictions of QN techniques appear fairly strong. Their advantage in modeling systems is that in the early design phases, they can efficiently assess gross initial models with a large number of parameter variations. The general scheme of mapping ProC/B models onto QNs is fairly natural. A standard QN is characterized by a set of queues and a set of routing chains that capture system structure and behavior, respectively. System dynamics are explained in terms of customers moving along chain descriptions, for open chains initiated via particular arrival processes. `FUs` of the `Server` type map naturally onto queues, and `PCs` onto routing chains. Customers mirror the progressing ProC/B processes. `Unconditional Sources` determine stochastic arrival processes that determine the load of a QN.

The hierarchical structure of ProC/B-models due to self-defined FUs is preserved in the corresponding QN-models by assigning each self-defined FU a dedicated QN. Thus QN-models again can consist of a hierarchy of QN-models whereas the load of lower-level QNs depends on the requests of QN-models on higher levels.

The aspect of synchronization which cannot be treated by queueing networks, can be realized with Petri nets. Therefore a translation from ProC/B to a C++ interface for Petri nets is implemented. Analysis techniques for Petri nets are provided in the APNN-Toolbox [3], a collection of functional and quantitative analyzers.

The main idea of the translation is that every *Process Chain Element* has a Petri Net description with an input place and an output transition. The process chain can then be modeled by connecting every input place with the output transition of the previous PCE.

Logistic systems are typically open systems containing an unlimited number of processes. Numerical analysis of the underlying Markov chain requires finiteness of the state space. For Petri net modeling we therefore have to restrict artificially the number of processes, whereby the number should be sufficient large.

For an analysis of the Petri net it is necessary that the state space is bounded. This will be achieved by connecting the output transition of the sink with the environment place of the source, i.e. the Petri net has a short circuit.

Standard FEs of type server can be realized only with service discipline random. A counter may only consist of a one-dimensional array. The hierarchy of the Process Chain model is mapped into the Petri net using so-called subplaces and subtransitions, whereby a bijective correspondence between sub-FUs and subplaces exists.

In addition to the previously mentioned analysis techniques the tool offers aggregation of model entities to speed up performance of analysis. The main idea is to replace parts of a model by aggregates that have the same structure as FUs and thus transform a ProC/B model into another (nearly) equivalent ProC/B model that is much more simpler than the original.

Aggregation is general in that sense that the user can model self-defined types of aggregates that are modeled just as FUs but specific parameters are left variable. The precise characteristics of the aggregate is determined by appropriate solvers that calculate the parameters. Thus types of aggregates can be reused in different models having different characteristics.

The currently implemented solvers use the above analysis techniques and support aggregation with Flow-Equivalent-Servers.

IV. Steady-state detection in simulative models

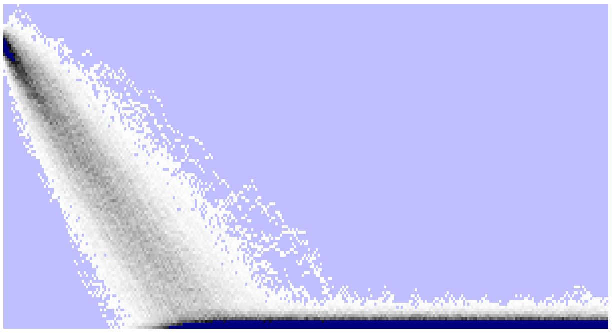
Simulation is a very generous approach for system analysis, because there are nearly no restrictions on the derivated model, even though there are some traps in output analysis of a simulation. One common problem is the initialization bias of the result measures caused by the warm-up interval in the beginning of a simulation. During this transient phase, the influence of the initial state is decreasing to a negligible limit, if the model is ergodic.

When using steady-state simulation, only the ”long-run behavior” of the system is of interest. So two questions have to be asked:

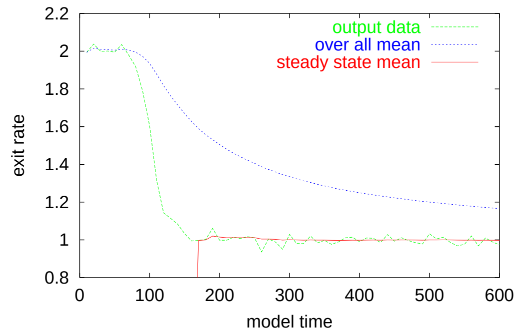
- Is there a steady-state behavior?
- At which model time is the influence of the initial state negligible?

The ProC/B framework includes some methods and algorithms to answer this two questions and to analyze output data of the steady-state phase.

In general a simulation run is very time consuming. So the ProC/B framework provides, beside the analysis of a single simulation run, the analysis of multiple replications in parallel (MRIP). But the use of MRIP do not only have an advantage in execution time, they



(a) Density distributions of the M/M/1-server‘s population over model time.



(b) Exit rate: a great difference between the mean of all collected observations and the mean of the observations of the steady-state phase.

Fig. 2. M/M/1-server initialized with a very high population.

give a strategic advantage in output analysis. Starting k replications, the algorithms can use k observations of one point in model time. An additional ensemble analysis is possible. The state distributions over time (cf. Fig. 2(a)) can be used to check the ergodicity and to find a proper truncation point of the warm-up interval.

The truncation of the output data of the warm-up interval is a common known approach ([9]) and is necessary to estimate unbiased result values (cf. Fig. 2(b)). The ProC/B framework uses some new algorithms to detect this truncation point. These algorithms, described in [4], require the choice of the ratio between the lengths of the transient phase and the observed part of the steady-state phase, thus avoiding that the independent choice of both parameters leads to poor results.

Most detection rules are based on the convergence of the mean to its steady-state value. But what, if only the mean converges and for example other quantiles do not?! The algorithm implemented in the ProC/B framework checks the random sample distribution and is therefore a better test for the ergodicity of the system. Even non-ergodic models can be detected more easily [4].

If a proper truncation point is found, the steady-state phase can be analyzed in two ways, depending whether one simulation is executed or MRIP are used. Using one simulation run, the output data is separated into batches with independent means. These independent means can be analyzed as usual. Using MRIP, a mean for each replication is calculated (replication/deletion approach: [9]). Their independence is assured by a proper choice of the seed values for the random number generator. This independent means can be analyzed as usual again.

With the algorithms described before, the ProC/B framework provides a complete environment to execute and analyze one simulation run or multiple replications. Not only result measures are calculated, the hole process of analysis can be inspected with the help of many graphs and figures.

V. Conclusions

We briefly introduced the ProC/B toolset for modeling and analysis of logistics networks. We presented a graphical user interface that allows for the specification of ProC/B models and controls subsequent analysis through simulative, algebraic and numerical techniques. New enhancements consider improvements of the ProC/B paradigm, a module for detection of steady-state in simulative models. Also integrated in the ProC/B toolset is aggregation of ProC/B submodels into equivalent ProC/B models that are simpler than the originals. Aggregation can be done by simulative, algebraic and numerical analysis.

References

[1] M. Arns, M. Fischer, H. Tatlıtürk, C. Tepper, and M. Völker. Modeling and Analysis Framework of Logistic Process Chains. In *Proc. of Joint Tool Session at PNPM/MMB/PAPM Conferences*, pages 56–61, Aachen, Germany, Sept. 2001.

[2] F. Bause, H. Beilner, M. Fischer, P. Kemper, and M. Völker. The ProC/B Toolset for the Modelling and Analysis of Process Chains. *In T. Field, P. G. Harrison, J. Bradley, U. Harder (eds.) Computer Performance Evaluation Modelling Techniques and Tools (Proc. Performance TOOLS 2002)*, Springer, LNCS 2324:51–70, 2002.

[3] F. Bause, P. Buchholz, and P. Kemper. A toolbox for functional and quantitative analysis of DEDS. *Quantitative Evaluation of Comp. and Comm. Sys.*, pages 356–359, Springer LNCS 1469, 1998.

[4] F. Bause and M. Eickhoff. Initial transient period detection using parallel replications. *Proc. of the 14th European Simulation Symposium*, pages 85–92, 2002.

[5] P. Buchholz, M. Fischer, P. Kemper, and C. Tepper. New features in the APNN-Toolbox. In P. Kemper (ed.), editor, *Tools at Int. Multiconference on Measurement, Modelling and Evaluation of Computer-Communication Systems*, Technical Report 760, University of Dortmund, pages 62–68, 2001.

[6] Ch. G. Cassandras and St. Lafortune. *Introduction to Discrete Event Systems*. Kluwer Academic Publishers, 1999.

[7] H. Beilner, J. Mäter, C. Wysocki. The Hierarchical Evaluation Tool HIT. 1994.

[8] A. Kuhn. *Prozessketten in der Logistik, Entwicklungstrends und Umsetzungsstrategien (in German)*. Verlag Praxiswissen, Dortmund, 1995.

[9] A. M. Law and W. D. Kelton. *Simulation Modeling and Analysis*. McGraw-Hill Higher Education, 2000.

[10] W.M.P. van der Aalst, J. Desel, and A. Oberweis. *Business Process Management*. Springer LNCS 1806, 2000.