A Tool for the Simulation of π-Calculus Systems

Anja Bog, Frank Puhlmann
{Anja.Bog,Frank.Puhlmann}@hpi.uni-potsdam.de
Hasso-Plattner Institute for IT Systems Engineering
at the University of Potsdam

Abstract: This paper presents a tool, called PiVizTool, for the simulation of business processes based on the π-calculus. The target of the tool is to simulate the evolution behavior of π-calculus systems and to enable a user to interactively influence this evolution by selecting the execution steps. The simulation of π-calculus systems in this case includes the presentation of the linking structure of the system in each step, utilizing a suitable visual representation.

1 Introduction

The π-calculus is a formal algebra for concurrent, communicating and mobile systems. It provides a framework for the representation, analysis, simulation and verification of these systems. Business processes, as a collection of consecutive, alternative and parallel units of work, with the objective to create value to customers or the organization they are part of, communicate with other processes, which may belong to the same organization or to another one, and therefore are also part of communicating systems. Nowadays with regard to the development of service oriented architectures (SOA)[8], the communication links between business processes are by no means permanent, but may change continuously, for example as the effect of utilizing services offered by a service registry. In this case the links between the business process using a service and the service are established dynamically. On this account, especially for showing the mobile aspect of business processes, the π-calculus is proposed as a representation for business processes.

Different implementations exist addressing the π-calculus. Some of them are for example Another Bisimulation Checker (ABC)[4], Open Bisimulation Checker (OBC)[6] and Mobility Workbench (MWB)[17]. To some extent these tools provide simulation functionality, but only on a basic level as command line output, without user-friendly representation and interaction capabilities. Additionally no tool support exists for creating π-calculus processes. This is currently done by writing the π-agents in text files, which becomes very confusing with an increasing number of agents and interactions. As the complexity of π-calculus process definitions grows with the size of the systems described and since business processes, usually contain a certain amount of tasks and events, modeling them in the notation of the π-calculus is a hard piece of work.

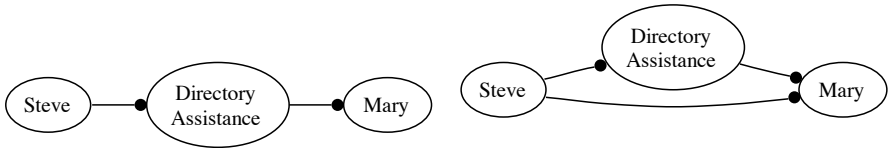
This paper focuses on the presentation of a tool providing an environment for enacting and monitoring π-calculus systems. The tool enables the simulation of their evolution behavior

and offers interactive means to exert influence on their course of execution. The contents of this paper is structured as follows. Firstly ... todo ... Thereafter the elaborated concepts will be illustrated by an example and finally the paper concludes with a discussion of related work and further developments.

2 Motivation

As already mentioned above, by using the π-calculus as a formal representation of business processes, the dynamic linking behavior of processes within service oriented architectures can be modeled. Other formal approaches for modeling business processes are provided by Carl Adam Petris Petri nets[12] and notations based on them like workflow nets[1]. However the problem of these approaches is that they represent systems with static structures, which means that all links for communications and actors within the system have to be known beforehand. Already a simple example, found in everyday life, illustrates the problem.

Lets assume $Steve$ wants to communicate with $Mary$ using a telephone, but does not know $Marys$ telephone number. Without her number, no communication can be established. To resolve this problem $Steve$ calls the directory assistance, whose telephone number is publicly known, and asks for $Marys$ number. The directory assistance can provide him with her telephone number, since $Mary$ has signed up in the telephone book. The systems state is depicted in figure 1(a). After receiving the number, $Steve$ is able to call $Mary$. He thereby establishes a new communication link between himself and $Mary$, which did not exist before. The new systems state of the linking structure is shown in figure 1(b). Such evolution can not be modeled by Petri nets and approaches based on them.



(a) Initial state (b) New link established

Figure 1: Change of linking structure during evolution

Since the π-calculus also supports the formalization of the workflow patterns, shown in [11, 14] and the formalization of the service interaction patterns [2][5], it provides a considerable approach as a formal representation of business processes, especially regarding the service oriented architectures, constantly gaining in importance these days. These arguments are strongly speaking for using the π-calculus in the business process domain. However, due to lacking tool support working with it, regarding modeling and simulation, further development in this area is desired.

The issue of modeling π-calculus processes is currently targeted by the development of a

tool chain, further described in [13]. This tool chain exports business processes modeled in BPMN[3] to an intermediate XML format, checks the business process diagram (BPD) for structural soundness and converts the diagram to π-calculus agents. The algorithm used for the conversion can be found in [15].

Solving the second problem of lacking tool support for π-calculus simulation is the aim of the tool presented in this paper. To ease the modeling of π-systems for simulation as input for this tool, the ASCII output produced by the converter tool mentioned above, can be used.

3 π-Calculus Simulation

Besides the functionalities provided by the MWB and ABC tools, a functionality for advanced simulation of the evolution of π-calculus systems is desirable. Advanced simulation in this context means to be presented with a visual representation of the π-calculus system, being able to interactively select reductions of the monitored π-calculus system to take place in the next step and being presented with an updated snapshot of the linking structure of the system after each step with the possibility to select the next one. Such simulation functionality is implemented by the PiVizTool. Its architecture is depicted as a block diagram of the Fundamental Modeling Concepts (FMC)[7] notation in figure 2. Rectangle shapes in this notation represent actors, being able to communicate with each other and rounded shapes represent storages, that can be read or written to by actors.

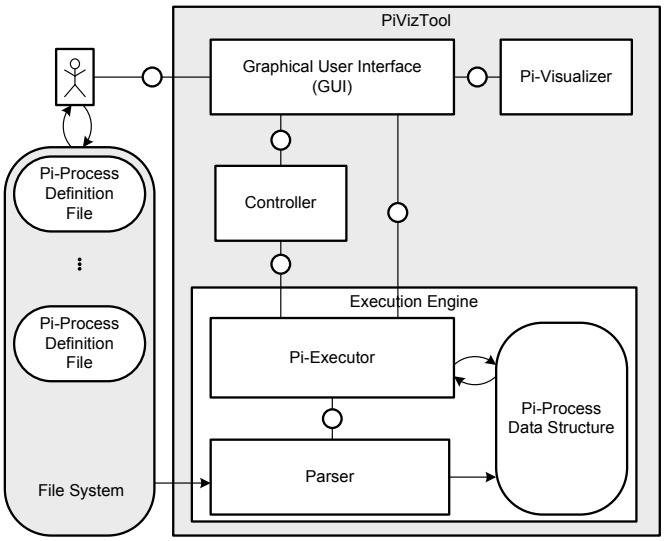


Figure 2: Architecture of the PiVizTool

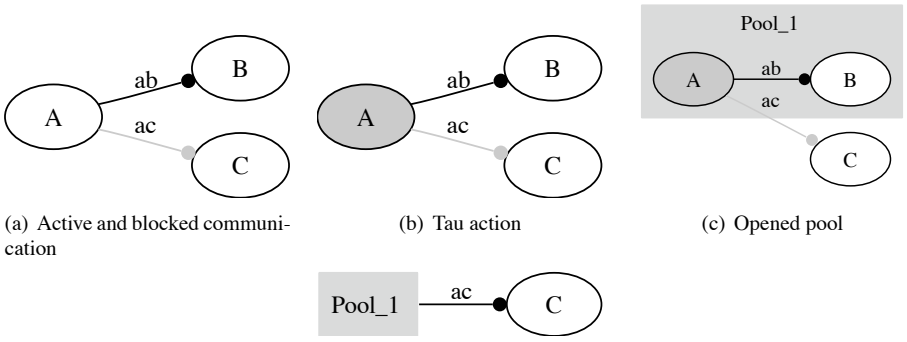
A user can interact with the PiVizTool via a graphical user interface (GUI). He can select the process definitions of the π-calculus system he wants to simulate from his file system. The process definitions have to be conform to the ASCII notation created by the converter. The π-execution engine loads the process definitions specified in the selected file into the π-executor with the help of a parsing component. The π-executor uses an intermediate data structure to simulate the behavior of the π-system according to the π-calculus reduction semantics[16] and extracts information about the active agents and the linking structure between them, which is needed for a visual representation.

The π-visualizer component uses this information to construct a visual graph structure, which represents the linking structure and the active agents in the current state of the π-system. The graphical representation is an extended version of the flow graphs introduced in [9] and adapted to the π-calculus in [10]. The extension is needed since the original flow graphs do not differentiate between blocked and active communications, whereas blocked communications are those where a link in the structure exists between two agents, but a reduction is currently not available. This is needed though, because reductions selectable for execution in the next step need to be visually distinguishable from the other ones, to aid the user. Additionally, since the user should be able to select available reductions for execution, τ reductions also need to presented.

Another extension to the flow graphs are pools. The π-calculus has no notion of pools as they exist in BPMN, but since they are helpful in structuring processes and grouping together tasks or activities, that belong to a certain group, e.g. an organization as a communication partner, this concept will be introduced to the graphical representation for π-calculus systems. Pools will be displayed as boxes with the agents belonging to that pool placed within. Along with pools structuring the visual representation comes the possibility to hide parts of a π-calculus system, whose actions are not of interest for monitoring, but are needed for the execution, by grouping them in a pool and showing the pool as a black box. In the case of a pool hiding its internal actions, for example communications taking place between agents, that are both located within this pool, an automatic execution of those actions has to be implemented, so the evolution of the entire π-system does not get stuck, because the user will not be able to see or select those capabilities for execution.

Figure 3 shows the elements, that have been added to the flow graphs or have been slightly changed. Three nodes representing the π-calculus agents A, B, C are depicted in figure 3(a). Agent A has two capabilities currently visible, which are sending a message using channel ab to B and sending a message using channel ac to C. Agents B and C have only one capability, that is currently visible, which is receiving a message on channel ab, or ac respectively. The communication link between A and C is shaded, in contrast to the communication link between A and B, which means that this communication is momentarily not executable, since it may be blocked by another action, that has to be executed beforehand. Figure 3(b) shows the same system as previously, with an additional executable τ action of agent A, illustrated by shading the according node. A pool $Pool_1$ is defined in the system depicted in figure 3(c), that contains the agents A and B. In this case the pool is open, meaning that its internals are not hidden, so the user may still select executable actions within the pool. The last figure shows the same system as in 3(c), but with the pool being closed this time, which means that its internal agents are hidden and

the actions within have to be executed automatically, so the systems, does not lock.



(a) Active and blocked communication

(b) Tau action

(c) Opened pool

(d) Black box pool

Figure 3: Flow graph extensions

The way of arranging the user interaction of selecting reductions from the system to be executed, is a very important aspect in addition to implementing the simulation features regarding π-calculus systems. This interaction is supposed to be intuitive to the user and exactly such an intuitively easy interaction can be realized by making the visual representation interactive itself. Thereby the user can directly select the edge representing the communication, or the node representing the τ action he wants to execute with the mouse pointer by clicking on it in the graph.

4 Example

An illustrating example will deepen the understanding of the concepts mentioned above and show the extensions to the flow graphs utilized in the PiVizTool. Before executing a π-calculus system, its process definitions have to be created, which will be done be using the converter. As input the business process shown in figure 7 is used.

The business process is composed of four communication partners, that are a customer, a reseller, manufacturer and payment organization. The customer wants to order an item and sends the appropriate request to a reseller, who triggers the manufacturer to produce such an item and sends a message to the payment organization to take care of the payment process. In addition to the order the customer tells the reseller on which addresses he wants to receive the product and invoice. This information needs to be forwarded by the reseller accordingly, so the manufacturer and the payment organization can communicate with the customer using the desired addresses.

For the automatic conversion to create a valid π-system, additional annotations for the message flows have to be created by the modeler, which include the channel names used for sending and the information sent. The other extra annotations, shaded in the image, will be created by the converter and are only shown, so the connections between the modeled

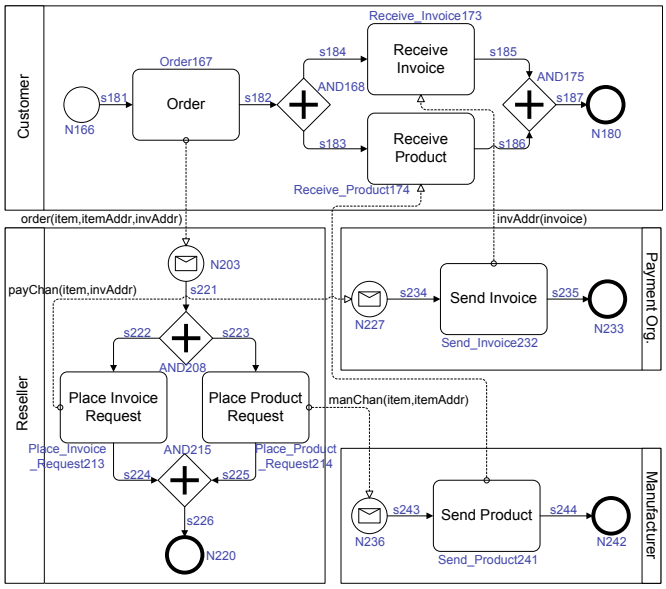
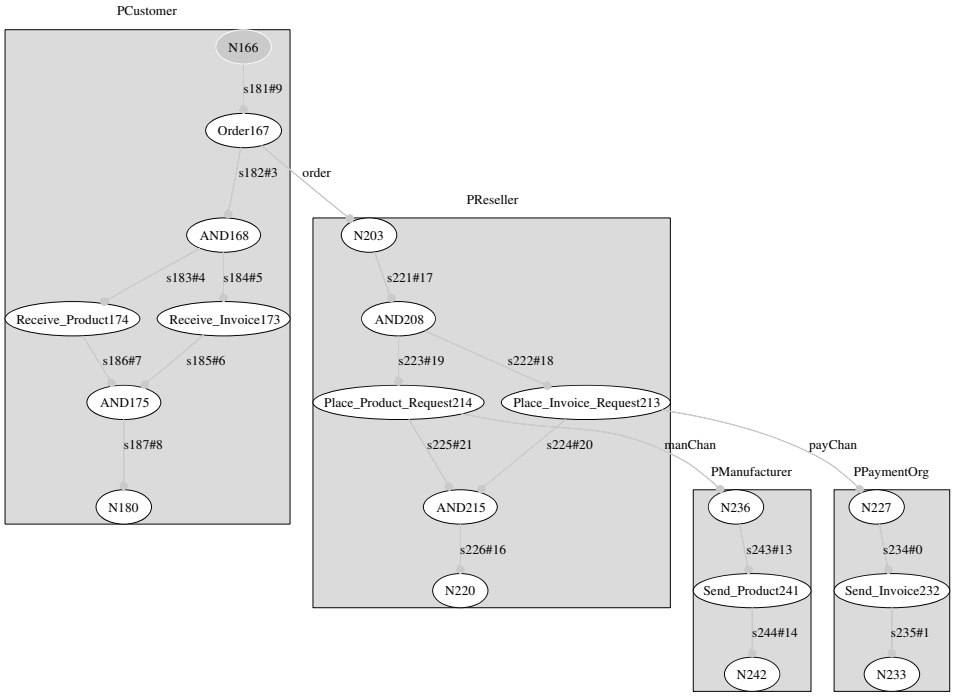


Figure 4: Example business process in BPMN

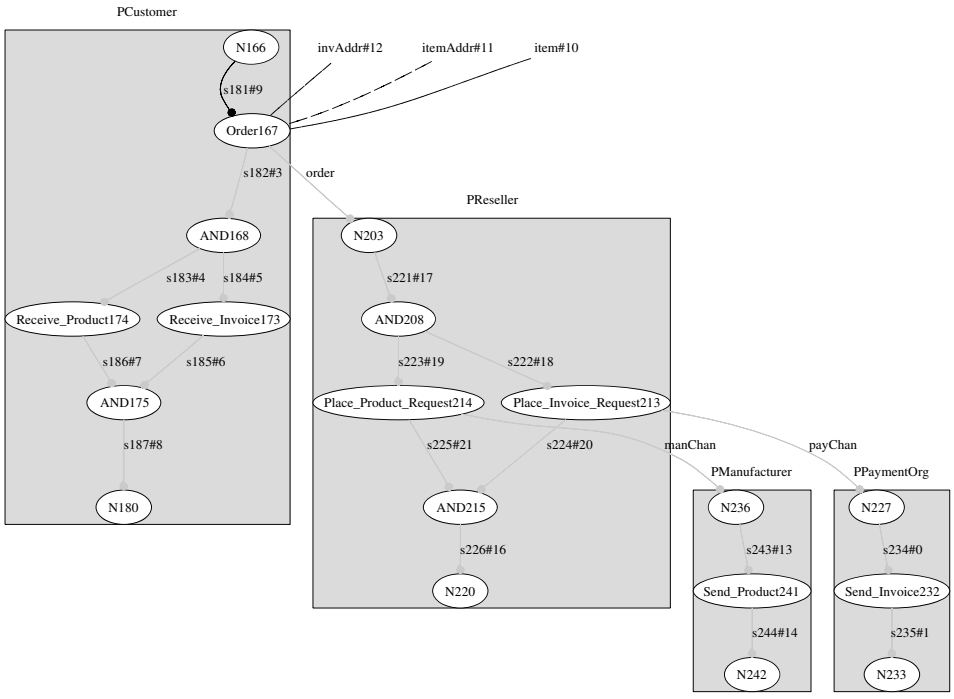
process and the resulting π-calculus agents can be followed. After exporting the model into the intermediate XML format and checking for structural soundness, which in this case returns the result, that the model is structurally sound, the business process will be converted to π-calculus process definitions.

The produced process definitions can directly be used as input for the π-calculus simulator PiVizTool. As soon as they are loaded into the PiVizTool, the initial linking state of the example, as depicted in figure 5(a), is presented. As can be seen from the image, the only active capability in this view is the customers initial τ reduction, which is displayed by shading the node representing the according agent. Lets assume the user has selected this τ action for execution, then the communication between the agent representing the customer processes start event and the agent representing the customers parallel split gateway is activated, shown in figure 5(b). An additional option to be selected by the user is the visualization of scoped names. The names, that are scoped in the system, meaning that they are restricted to certain agents are presented in a list, where the user may select the ones, whose scopes he wants to monitor. In the this figure, the scopes of the names, representing the ordered item and the addresses, the customer chooses for sending the invoice and product to, are visualized by drawing a dotted line from the name to all nodes, that belong to its scope. In this case the names are solely known by the customers order task, but this will change upon sending the information to the reseller, because after this communication the reseller has access to these names, too. The new state is depicted in figure 6(a).

Figure 6(b) shows the state of the example π-system during execution, where the reseller is



(a) Initial state



(b) State after executing the τ action

Figure 5: Snapshots at the beginning of the simulation

ready to forward the item information to the manufacturer and the payment organization. Additionally the display mode of pools hiding their internals is illustrated. The closed pools are the $Customer$, $Manufacturer$, and $PaymentOrg$, with the communication links ending at the borders of the pools. Any capabilities, that are active within these pools have to be executed automatically, since they are unobservable.

In 6(c) the manufacturer and payment organization finally establishing new links to the customer are presented. These links are based on the address information transmitted by the reseller, that has been created by the customer originally and sent to the reseller together with the order request.

Seeing this π-system evolve and complete until no further capabilities are active points out a problem, that has to be the target of further research. This problem is garbage collection. In the example figure 6(c) the reseller processes has finished its execution and none of the agents left will ever be active again in the particular system. The question is if such agents can be detected and be removed.

Finally some screenshots of the entire application are depicted in figures 7(a), 7(b). The first screenshot shows the segmentation of the application into the parts for showing the visual representation of the π-calculus system (right side) and the list, where the scopes of restricted names can be chosen for visualization (left side). In the example three names are chosen from the list. The second screenshot shows another view, where the process definitions of the π-system in the visualized state are given. In a third view, the process definition of the initially specified system can be viewed.

Additional options selectable in the application are auto-execution of a randomly selected active step and the dumping of the underlying data structure of the π-calculus execution engine.

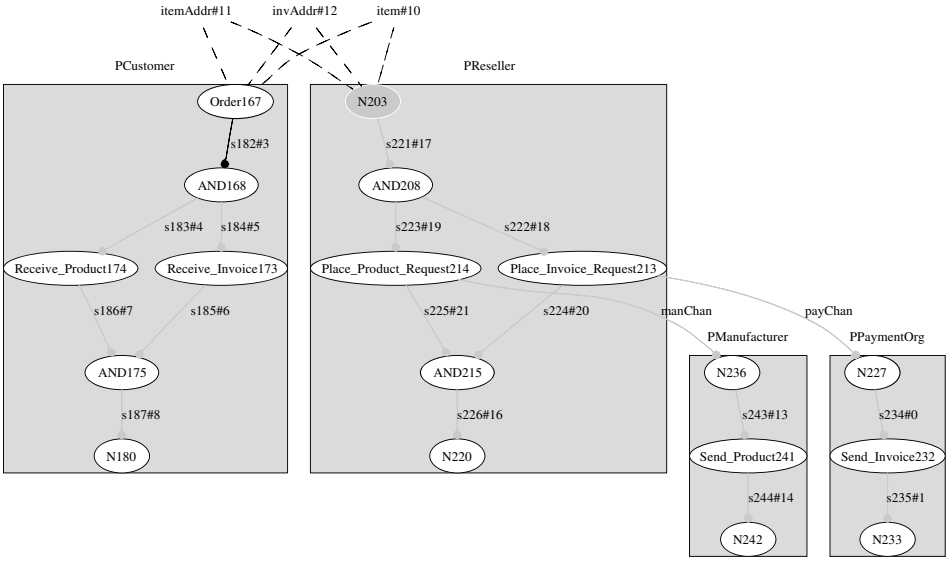
5 Conclusion

This paper has introduced a prototypical tool for the advanced simulation of business processes based on the π-calculus. This tool has the functionality of visualizing the linking structure of a π-calculus system and giving a step by step simulation of the behavior of the system during its evolution, whereas the single execution steps can be interactively selected by a user.

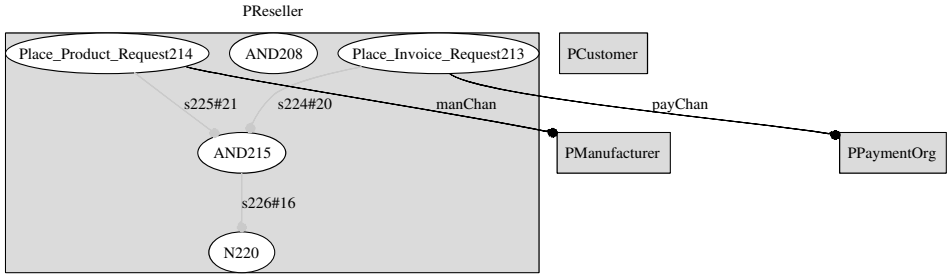
Further development in this field, besides research about garbage collection in π-calculus systems as mentioned above, can take place regarding a workflow engine based on the π-calculus, that is actually able to assign responsibilities according to roles and trigger automatic execution of work units.

References

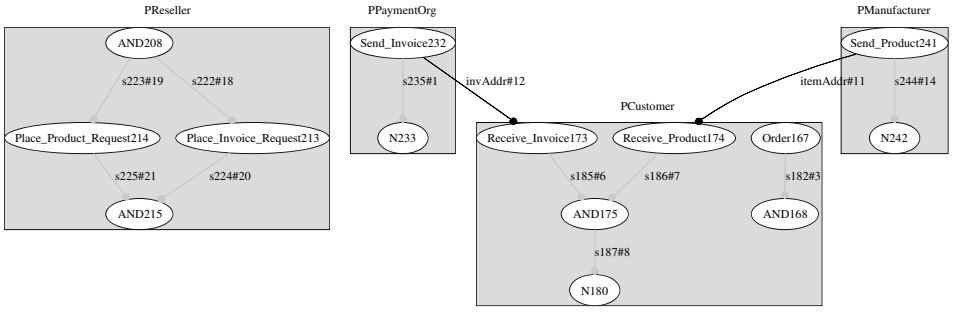
[1] W.M.P. van der Aalst. *The Application of Petri Nets to Workflow Management*. PhD thesis,



(a) Scopes after sending the information to the reseller

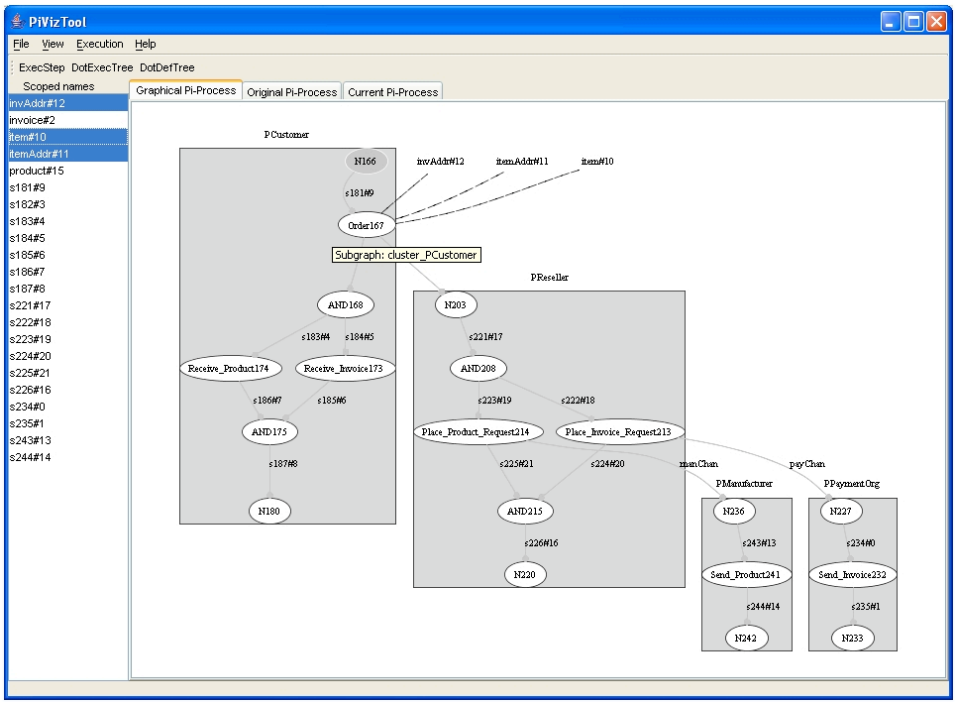


(b) Reseller forwarding the order

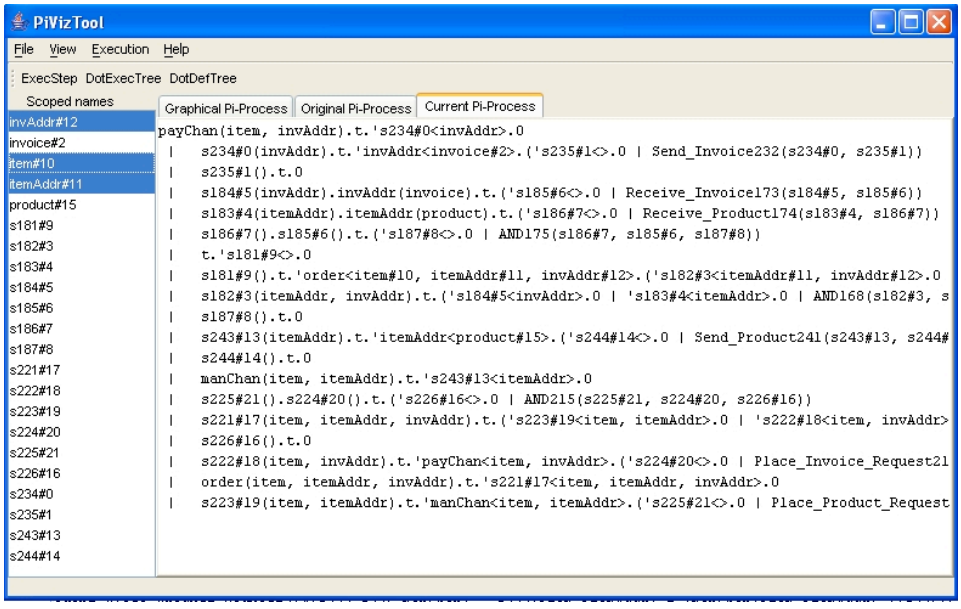


(c) Manufacturer and payment organization ready to send product and invoice

Figure 6: Snapshots during the simulation of the example



(a)



(b)

Figure 7: Screenshots of the PiVizTool

1998.

[2] Alistair P. Barros, Marlon Dumas, and Arthur H. M. ter Hofstede. Service interaction patterns. In *Business Process Management*, pages 302–318, 2005.

[3] Business Process Management Initiative (BPMI). Business Process Modeling Notation (BPMN), May 2004.

[4] Sebastien Briais. The ABC's User Guide, May 2005.

[5] Gero Decker and Frank Puhlmann. Formalizing service interactions extended version. to be published at the 4th International Conference on Business Process Management (BPM'2006), Vienna, Austria, September 2006, 2006.

[6] Ulrik Frendrup and Jesper Nyholm Jensen. Checking for Open Bisimilarity in the π-Calculus. Febuary 2001.

[7] Andreas Knöpfel, Bernhard Gröne, and Peter Tabeling. *Fundamental Modeling Concepts: Effective Communication of IT Systems*. Wiley, March 2006.

[8] Dirk Krafzig, Karl Banke, and Dirk Slama. *Enterprise SOA : Service-Oriented Architecture Best Practices (The Coad Series)*. Prentice Hall PTR, November 2004.

[9] Robin Milner. Flowgraphs and flow algebras. *J. ACM*, 26(4):794–818, 1979.

[10] Robin Milner. *Communicating and mobile systems: the π-calculus*. Cambridge University Press, New York, NY, USA, 1999.

[11] Hagen Overdick, Frank Puhlmann, and Mathias Weske. Towards a Formal Model for Agile Service Discovery and Integration. In Kunal Verma, Amith Sheth, Michal Zaremba, and Christoph Bussler, editors, *Proceedings of the International Workshop in Dynamic Web Processes (DWP 2005)*, volume International Business Machines, 2005 IBM RC 23822, pages 25–36, 2005.

[12] C.A. Petri. *Kommunikation mit Automaten*. PhD thesis, Institut für instrumentelle Mathematik, 1962.

[13] Frank Puhlmann. A Tool Chain for Lazy Soundness. Demo Session of the 4th International Conference on Business Process Management, Vienna, Austria. 2006.

[14] Frank Puhlmann and Mathias Weske. Using the π-calculus for formalizing workflow patterns. In *Business Process Management*, pages 153–168, 2005.

[15] Frank Puhlmann and Mathias Weske. Investigations on soundness regarding lazy activities. In S. Dustdar, J.L. Fiadeiro, and A. Sheth, editors, *Proceedings of the 4th International Conference on Business Process Management (BPM 2006), volume 4102 of LNCS, Vienna, Austria*. Springer Verlag (2006) 145-160, 2006.

[16] Davide Sangiorgi and David Walker. *The π-Calculus: A Theory of Mobile Processes*. Cambridge University Press, 2001.

[17] Björn Victor and Faron Moller. The Mobility Workbench — a tool for the π-calculus. In David Dill, editor, *CAV'94: Computer Aided Verification*, volume 818 of *Lecture Notes in Computer Science*, pages 428–440. Springer-Verlag, 1994.