Analysis of resource-constrained processes with Colored Petri Nets

Mariska Netjes, Wil M.P. van der Aalst, Hajo A. Reijers

Eindhoven University of Technology, Faculty of Technology and Management,
(PAV J5,) PO Box 513, NL-5600 MB Eindhoven, The Netherlands
Contact: m.netjes@tm.tue.nl

Abstract. Formal models of business processes support the performance analysis of processes and the evaluation of redesign alternatives. This paper presents a formal model to analyze the behavior of resource-constrained processes. The model is developed using Colored Petri Nets (CPN or CP nets) and the supporting software package CPN Tools. In our approach, a business process consists of tasks and resources able to perform one or more tasks in the process. We developed a task building block to model tasks and a resource module to model the allocation of resources with different allocation methods. The opening of a bank account is used as an example process to investigate two so-called "best practices" while using the simulation facility of CPN Tools. First, we explore the specialist-generalist trade-off, i.e., finding the optimal ratio of specialists and generalists. Then we explore the flexible assignment policy, i.e., a strategy to deploy specialists first to preserve operational flexibility.

1 Introduction

Organizations are constantly looking for ways to improve their performance. By emphasizing business processes rather than hierarchies and by putting a focus on customer satisfaction, organizations tend to attain better overall performance, a better *esprit de corps* and less interfunctional conflicts [16]. A survey among 90 Swedish organizations, ranging from small consulting firms, hospitals and state-owned companies to multinational companies such as ABB, Ericsson, Saab and Volvo, pointed out that organizations seem to be quite satisfied with the effects of such a process orientation and that they were planning to allocate more resources to their process initiatives [7].

A process initiative can take on different forms, for example, as a project to rethink the underlying process structure, to change resource allocation strategies, or by the introduction of new technology such as workflow management systems. In this paper, we focus on a number of *best practices* in this area. A best practice prescribes a historical solution that seems worthwhile to replicate in another situation or setting, although it may need to be adapted in skilful ways in response to prevailing conditions. A collection of best practices for business process improvement is given in [18]. It should be noted here that many of the best practices lack adequate (quantitative) support. It is our objective to investigate two of these best practices in more detail in this paper, i.e., the so-called *specialist-generalist trade-off* and the *flexible assignment policy*. We would like to obtain insights in the underlying mechanisms and see under which conditions the best practices indeed provide improvements.

The *specialist-generalist trade-off* aims at *finding the optimal ratio of specialists and generalists* in a process. Given a fixed number of resources, the question is how many of the resources should be specialized resources and how many of them should be generic resources. We define a specialist as a resource able to perform exactly one task. Routine is built up in doing this task and as a result he or she performs the task faster. A generalist is able to perform more tasks and adds flexibility to the process leading to a better utilization of resources [18].

Using a *flexible assignment policy* means that resources are *assigned to the work in such a way that maximal flexibility is preserved for the near future*. When both specialists and generalists are available to perform a certain task the most specialized resource is assigned to the task. By doing this, more generic resources are 'saved' for other tasks for which this specialized resource might

not be suitable. In this way, the waiting time for a suitable resource may be reduced. Another advantage is that the specialized resource is more skilled and will need less time to perform the task [18].

In this paper, we focus on an abstraction of an actual business process as a collection of a number of inter-related *tasks*, where a limited number of *resources* (people, systems, machines, etc.) is available to execute the process. To be more precise, a task is an "atomic" and logical unit of work, which is carried out in full or not at all. A task which is just about to be executed for a specific case (or process instance) is called a *work item*. Each work item should be performed by one resource suited for its execution. This implies that, while there may be several resources able to perform a given work item, exactly one resource should be assigned to perform the work item. Note that we reserve the term *activity* for the actual execution of a work item [3]. Resources are grouped into roles to facilitate the mapping of resources on work items. A role is a group of resources with similar characteristics. A resource has one or more roles and each role may be performed by many resources [12]. The objective of this paper is to give an approach for analyzing the performance of such a *resource-constrained process* and to show how a choice between alternative designs for a process using best practices can be supported with this kind of analysis.

For the modeling and analysis of resource-constrained processes, CPN Tools is used. This tool provides support for the construction, simulation and performance analysis of high-level Petri nets [11]. CPN Tools is chosen, because it combines the strength of Petri nets with the strength of programming languages. The reasons for using Petri nets to model business processes are stated in [1]. The basic behavior of a process is modeled and visualized with Petri nets and more sophisticated behavior is added through ML functions. It is possible to debug the model and validate the correct behavior of the model with a step-by-step simulation of the model. Alternatively, the functional correctness of the system can be validated and verified with state space analysis. Once the model is validated, it is easy to make adaptations to it and create alternatives for the original model. The simulation environment in CPN Tools also has the capability to perform an automatic sequence of firings to examine the behavior of a model in the long run [20]. The combination of time and simulation in CPN Tools provides the possibility to analyze the performance of the modeled process. For an introduction into high-level Petri nets and CPN Tools the reader is referred to [8,9,11,20].

The structure of the paper is now as follows. Section 2 describes the developed CPN model and explains the task building block, the resource module and the different allocation methods in more detail. Section 3 describes the application of the developed CPN model to investigate the mechanisms of the specialist-generalist trade-off and the flexible assignment policy. Section 4 discusses the related work and section 5 gives the conclusions and proposes future work.

2 Development of the CPN model

Before investigating the *specialist-generalist trade-off* and the *flexible assignment policy*, we describe the developed CPN model. Both best practices aim at the improvement of resource-constrained processes. The purpose of the model is to provide an easy way to model such a resource-constrained process and change it to evaluate the use of the best practices under different circumstances. In this section we first give an overview of the top level of the developed model to show its main parts. Further, we explain the sub models which represent the different elements of a general resource-constrained process. These elements are the generic task building block, the generic resource module and different resource allocation methods.

2.1 Overview of CPN model

In Figure 1, an example of the top level or main page of the CPN model (for a process consisting of two sequential tasks) is shown. The different parts of the CPN model are the generator, the process (a number of tasks in some relation to each other, e.g. sequential) and the generic resource module for the allocation of resources to the process.

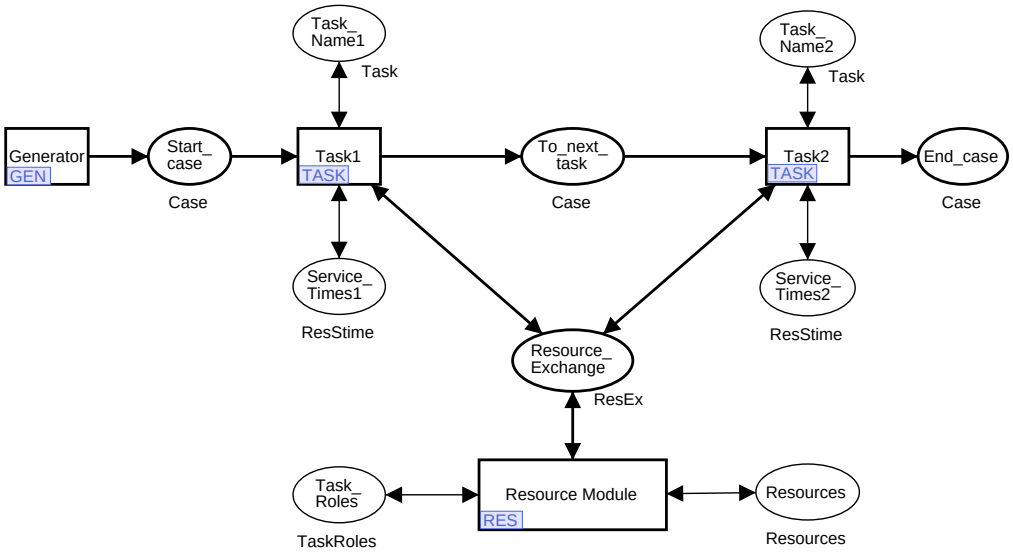

Fig. 1. Top level

We give a global description of each of the parts to explain how the model works. The `Generator` generates cases which flow through the process. The process consists of `tasks` placed in some configuration, allowing for both sequential and parallel configurations. To ensure rapid and straightforward modeling of (large) processes a generic task sub model is developed. This sub model gives each task the same layout and functionality, because for each additional task an instance of the generic task sub model is used. At the top level the specific task variables need to be configured, because these specific variables distinguish one task instance from another. For each task the task name needs to be added to the place `Task_Name` and the service times (one for each role that may perform the task) need to be added to the place `Service_Times`. The `Task` and `Resource Module` substitution transitions communicate and exchange resources through the place `Resource_Exchange`. The `resource module` arranges the allocation of resources to work items in a generic way. The resource module is generic, i.e., the number of tasks and resources in the process is not predefined. Only a specification of the roles able to perform a specific task and the available resources is required. For each task, the associated roles need to be specified in the place `Task_Roles` and the available resources need to be placed in the place `Resources`. Available resources are named according to their role, because the role of the resource is the only attribute required for the mapping of resources to tasks. According to the standard definition of a role [3], it is possible that resources have more than one role, but in this model the assumption is made that every resource fulfills exactly one role. This is not necessarily a limitation, because resources with more roles can be linked to a 'meta'-role enclosing these roles. For example, if a resource has roles r1 and r2, we could add a new role r12 which is attached to all tasks that require r1 or r2.

2.2 Task building block

A process model consists of a number of tasks placed in a certain order. When a work item flows through the process, service times could differ per task and depend on the type of resource performing the task. Each task has the same basic functionality with a different value for the service time and different resource types. A generic task sub model enclosing the required functionality is developed. This model is called a task building block, because a process can easily be built with these blocks. It is possible to place the building blocks in a sequential order or to put them in parallel. The task building block is depicted in Figure 2. The main functions of the task block are requesting a suitable resource and putting a time delay on the case and the involved resource. An explanation of the color sets used in Figure 2 is given in Table 1.

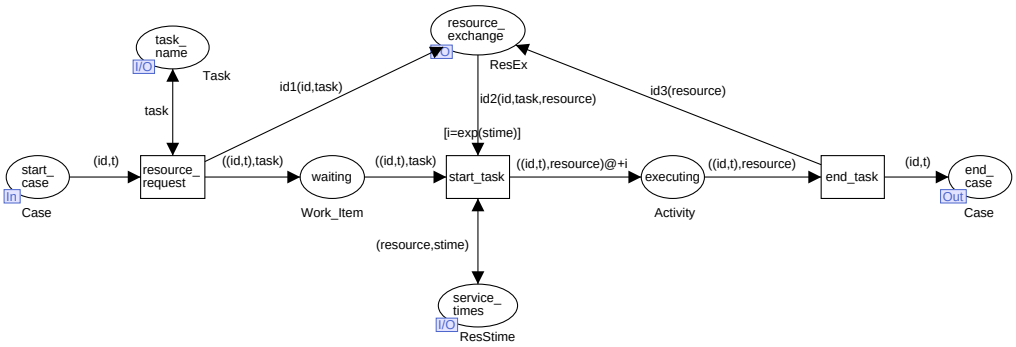

Fig. 2. Task building block

Table 1. Color sets

Color sets task building block	**Color sets resource module**
color I = int	color ID = int timed
color ID = int timed	color Task = string
color T = int	color Prerequest = product ID * Task
color Case = product ID * T timed	color Role = string
color Task = string	color Roles = list Role
color Work_Item = product Case * Task	color TaskRoles = product Task * Roles
color Stime = int	color Request = product ID * Task * Roles
color Role = string	color Requests = list Request
color Resource = Role	color Resource = Role
color ResStime = product Resource * Stime	color Resources = list Resource
color Activity = product Case * Resource timed	color Match = product ID * Task * Resource
var i:I	color ResEx = union id1:Prerequest+id2:Match
var id:ID	+id3:Resource
var t:T	var id:ID
var task:Task	var task:Task
var stime:Stime	var roles:Roles
var resource:Resource	var request:Request
	var requests:Requests
	var resource:Resource
	var resources:Resources
	var match:Match

Before we describe the details of the task block we will look at the interaction between a task building block and the resource module. A task building block communicates with the resource module through the place `resource_exchange`. The purpose of the communication is obtaining a suitable resource from the resource module to perform a specific work item. After completion of the task the resource is returned to the resource module. In Figure 2 we see two arcs going to and one coming from the place `resource_exchange`. A token travelling the left arc going to the place `resource_exchange` represents the request for a resource and for requesting two attributes, the case id and the task name, are required. The middle arc returns a token representing the request and the resource attached to it. The attributes on this arc are the case id, the task name and the resource. With the right arc the resource returns to the resource module. We see that the sets of attributes, or color sets, on the inputs and output of the place `resource_exchange` differ. However, the place `resource_exchange` should have a color set consistent with each of these input and output colors. We make a *union* place from the place `resource_exchange` uniting the different color sets to have one consistent color set. The color set of the place is a union of the different input and output color sets. Each arc has an identifier with the required attributes to specify the color set related to the arc. With the place `resource_exchange` specified as a *union* place different tokens with different color sets may pass it.

Let us now consider the task building block in more detail. A case enters the task block via the place `start_case`. A case has attributes specifying a unique case id and the time the case was generated. The task name is stored in the place `task_name`. The task name is added to the case attributes, because it is necessary for the resource request. By adding the task name the case is changed to a work item, which needs to be executed. A resource request stating the case id and the task name is sent to the resource module via place `resource_exchange`. The work item waits in the place `waiting` until a work item with the case id and the same task name (together with the allocated resource) is available in the place `resource_exchange`. The mapping on the waiting work item is done based on both the case id and the task name to allow for parallel execution of tasks for the same case. Recall that the term activity [3] refers to the actual execution of a work item. The average service time needed by the resource to complete the activity is received from the place `service_times`. The duration or actual service time is modeled as a negative exponentially distributed delay. When the delay time of the activity is passed, the resource is returned to the place `resource_exchange` and the case leaves the task block via the place `end_case`.

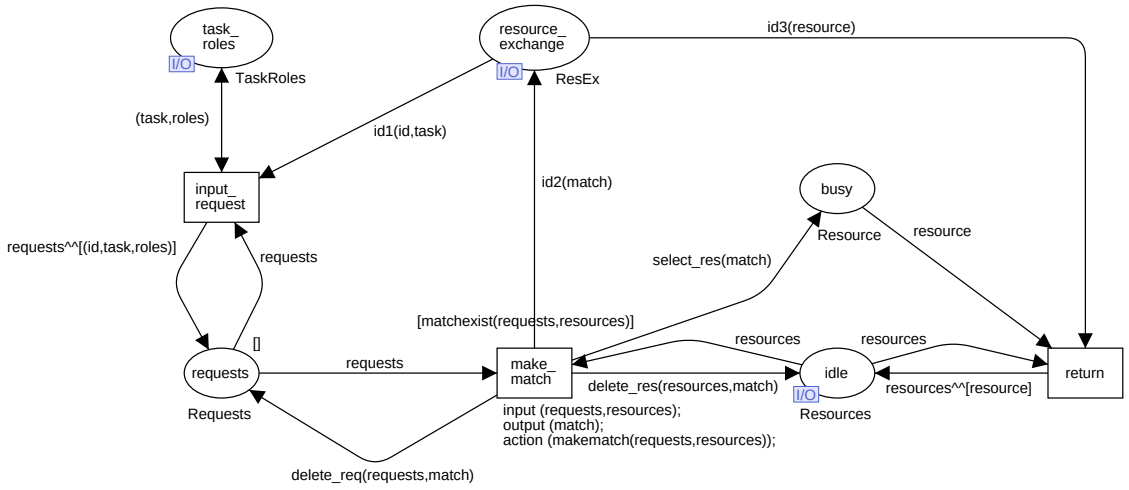

Fig. 3. Resource module

2.3 Resource module

The resource module has been developed to allocate resources to work items. In the module, each resource is assigned to exactly one role and the resource name is equal to the assigned role. For instance, a resource with role *postman* is called a postman. If there is more than one postman in the process, then all resources with role *postman* are called postman. A role is linked to one or more tasks, i.e., a resource with a certain role is capable of performing one or more tasks. Resources are allocated to work items by mapping the role of the resource with the task that needs to be executed for the work item.

The resource module is a generic solution, i.e., the use of the model is not limited to a specific number of tasks or resources and the functionality is the same for every resource-constrained process model. Figure 3 shows the resource module and Table 1 contains the color sets of the resource module. We will explain the resource module in more detail starting in place `resource_exchange` and following the arcs from there.

A resource request for a certain work item is received in the place `resource_exchange`. The resource request has the attributes case id and task name, because both attributes are necessary to return the filled request to the right work item. At the first transition, `input_request`, the roles that may perform the work item are added to the request. The request (with attributes case id, task name and roles) is put in a list with requests ordered in a First Come First Serve order. This is done by putting an arriving request at the end of the requests list. The requests will be matched with the available resources starting from the top of the requests list. The combination of the case id and the task name from the request and the allocated resource is called a match. When more role types are able to perform a certain task an allocation method is necessary to allocate role types in a specified order. Note that in the next section different allocation methods and the associated functions are described. The available resources are stored in the place `idle`. If a match exists, the transition `make_match` is enabled. When enabled, an attempt is made to match the first request from the requests list with an available resource. When a resource with the required role is available a match is found. A match is transferred to the place `resource_exchange` and from there to the corresponding task. The matching request and resource are deleted from the requests list and the resources list. Next to this, busy resources are stored in the place `busy_res`. This place can be perceived as redundant, but it is added so show which resources are actually busy. From head to tail the requests list is evaluated and matches are made. Requests for which no resource is available wait and stay in the requests list. When a resource is finished working on a task, it is put back in the resources list via place `resource_exchange` and transition `return`.

One could think that the model has some restrictions. One possible restriction is that it is not possible to withdraw an assignment of some work to a resource. This functionality is not added, because human resources will become demotivated when work is taken from them just before they will start working on it. More on assignment policies, the possible choices and the subsequent consequences can be found in [23]. Another point that can be seen as a limitation is that a resource can not work on several activities at the same time. This limitation follows from the definition of a task as an "atomic" piece of work. Once started, the task needs to be finished before the resource will be able to start another task [3].

2.4 Resource allocation methods

The resource module allocates resources to work items based on a certain allocation method. An allocation method specifies the order in which the allocation of different resource types should take place. Two allocation methods, priority based allocation and random allocation, are modeled for incorporation in the resource module. We have chosen to define allocation methods with ML functions. The functions are made on two levels, the top level functions are the same for each allocation method and these functions call lower level functions which define the specific allocation method. By doing so, the CPN sub model representing the resource module is the same regardless of the chosen allocation method, because the resource module only refers to the top level functions. We will briefly discuss the top level functions, they are all mentioned in Figure 3. All functions are

connected to the transition `make_match` in the resource module. This transition is enabled when the precondition in the guard is satisfied. The precondition function `matchexist` evaluates if at least one possible match exists between the list with requests and the resources. When the transition `make_match` is enabled it performs an action. The action part of the transition `make_match` is used to model the action and the action part is placed under transition `make_match` (see Figure 3). In the action part the input, the output and the actual action are specified. The inputs for the action part are the list with requests and the resource list. The actual action is the evaluation of the function `makematch`. When evaluated this function calls the lower level functions for a specific allocation method. The output of the action part is a match between a request and a suitable resource allocated with the underlying allocation method. This output, `match`, is used to put the right value on each arc going from transition `make_match` to one of the surrounding places. The `match` itself, with attributes case id, task name and resource, is returned to the requesting task via the place `resource_exchange`. The function `delete_req` evaluates the requests list and deletes the request which is in the `match`. The function `delete_res` deletes the matching resource from the resources list. The function `select_res` places the resource assigned to the `match` in the `busy` place.

We have predefined an allocation method to assign resources on priority and a method to assign resources randomly. The general idea of both methods will be described. We will first explain the priority based allocation and then the random allocation.

With priority based allocation it is already known before the actual allocation in which order resources will be assigned to work items. The roles able to perform a task are prioritized and the order in which resources are assigned is based on the priority of their role. The priority of the roles is defined in the place `task_roles`. For each task the roles able to perform the task are specified. The roles are ordered from high priority to low priority. From the available resources the resource with the highest priority role is selected and assigned.

With random allocation the available resources with a suitable role all have an equal change of being selected. For each resource request a preselection of all resources with a suitable role is made. From this preselection a resource is randomly chosen and allocated.

For validation purposes a tandem M/M/1 queueing system has been modeled with the task building blocks and the resource module. The simulation results from this model are equal to the results calculated with queueing theory [10]. This validation suggests the model is working as we expected. To collect the statistics we have used the monitoring facility of CPN Tools. This facility has not yet been released. However, we were able to use a prerelease. The facility allowed us to collect measurement data without changing the functional model and thus very helpful in maintaining the readability of the models.

3 Application of the CPN model

In this section we will use the developed CPN model to investigate how process performance is influenced by the ratio of specialized and generic resources in the process and the allocation method applied to these resources. In this section we will first give a short introduction on the best practices and the example process on which they are applied. Secondly, we will apply the specialist-generalist best practice to find the optimal ratio of specialists and generalists. Next to that, we will investigate when flexible assignment of resources should be applied.

3.1 Introduction

We would like to have more (quantitative) insight in the best practices for business process improvement. In this paper we consider *two best practices related to the assignment of specialized or generic resources*. We will investigate under which conditions process improvements are achieved with the application of the best practices. We look for improvements of the average throughput time of the process and this measure is used as performance indicator. With the developed CPN

model we model an example process on which we apply the best practices. This example process deals with the opening of a bank account. A short description of this bank process is as follows. When a registration for a bank account is received the data is entered into the system. Secondly, the bank account is initiated and finally a letter is sent to the customer. The process model is built with three task building blocks and the resource module and is depicted in Figure 4.

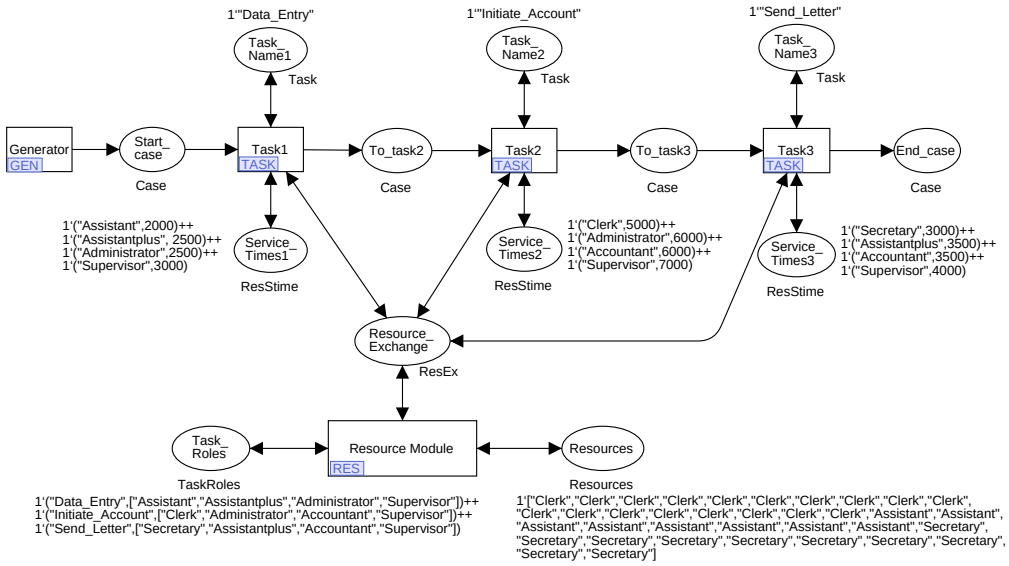

Fig. 4. Process model: Opening a bank account

In the model each task has two places specifying the associated task variables. The task names are stated above place `Task_Name`. The service times are stated next to place `Service_Times`. For instance, the first task named 'Data_Entry' can be executed by one resource with any of the four stated roles. Each role needs a different average service time to execute the task (e.g. the first role 'assistant' needs on average 2 minutes to perform a data entry). The resource module also has two specification places. The place `Task_Roles` describes for each task which roles are able to execute it. The task 'Data_Entry' is linked to the roles 'assistant', 'assistantplus', 'administrator', and 'supervisor'. The roles are ordered from specialized resources to generalists. The assistant is specialized in entering data, the assistantplus can enter data and send letters, the administrator could do data entries and account initiations and the supervisor is a generalist and is able to do all three tasks. The place `Resources` contains the resources available for the process. In Figure 4 there are 36 specialists available, namely 18 clerks, 8 assistants, and 10 secretaries.

3.2 Specialist-generalist trade-off

In this section we will investigate one of the two best practices, the specialist-generalist trade-off, and its underlying mechanisms. This best practice aims at *finding the optimal ratio of specialized and generic resources*. This implies that the total number of resources is kept constant, while it is decided how many of the resources should be specialized and how many should be generic to have an optimal process performance. The performance, indicated by the average throughput time, of a mixture of specialists and generalists is determined by two mechanisms.

– The specialist performs the work faster and using specialists will lower the average service time spent on a case. A generalist needs more time for service and this first mechanism is in favor of the specialized resources.

– According to our definition a specialist can only perform one task. When the utilization of the resources increases, work items will have to wait longer for 'their' specialist. The use of generalists will add flexibility and a better utilization of resources to the process. The second mechanism favors generic resources, because they are suited for more tasks and using generalists will lower the average waiting time of a case.

We would like to have more insight in the trade-off between these two mechanisms and find the optimal ratio of specialists and generalists for the bank process.

According to the description of the specialist-generalist trade-off we should try to *find the optimal ratio of specialists and generalists*. In the bank process there are 36 resources available and each resource could perform one out of seven roles. This means we can form approximately 6 billion different ratios of specialists and generalists, which all should be evaluated. And each result should be compared with the results for the other ratios to see which alternative has the optimal ratio of specialists and generalists. We think a more pragmatic and more efficient approach can distinguish at least one ratio that is favorable compared to other ratios. For the bank process we would like to find an alternative process that has *a throughput time significantly lower than the initial situation with only specialists*. To find this alternative with a *distinctive ratio of specialists and generalists* we first perform a global search for alternatives that seem fruitful for further investigation. From these alternatives we will derive promising alternatives to evaluate and enclose the alternatives with a *distinctive* ratio.

For the global search we make several process alternatives with specialist-generalist ratios ranging from mainly specialists to mainly generalists. In our initial process model (see Figure 4) only specialized resources are available to execute the process. The roles of resources stored in place `Resources` are easily changed to create an alternative process model. Note we have two types of generalists in the bank process. There are generalists like the administrator which are able to perform two tasks and this type is called `2task-generalist`. The other type of generalists is able to perform three tasks. Since there is only one such generalist type in the bank process we name this type `supervisor`. We form five alternative process models with different ratios of specialists and generalists. The initial process is called process alternative 0. The six process alternatives are stated in Table 2 with the ratio of specialists, 2task-generalists, and supervisors per alternative and the available resources.

Table 2. Definition of process alternatives

Alternatives	**Ratio of spec-gen**	**Resources**
0	36 specialists	8 assistants, 18 clerks, 10 secretaries
1	27 specialists 9 2task-generalists	6 assistants, 14 clerks, 7 secretaries 3 administrators, 3 accountants, 3 assistantsplus
2	18 specialists 18 2task-generalists	4 assistants, 9 clerks, 5 secretaries 6 administrators, 6 accountants, 6 assistantsplus
3	9 specialists 18 2task-generalists 9 supervisors	2 assistants, 4 clerks, 3 secretaries 6 administrators, 6 accountants, 6 assistantsplus 9 supervisors
4	9 specialists 9 2task-generalists 18 supervisors	2 assistants, 4 clerks, 3 secretaries 3 administrators, 3 accountants, 3 assistantsplus 18 supervisors
5	18 2task-generalists 18 supervisors	6 administrators, 6 accountants, 6 assistantsplus 18 supervisors

Each alternative allocates resources randomly, i.e., flexible assignment is not used. For each of the alternatives we perform a simulation. The arrival intensity is 200 arrivals per hour (Poisson arrival process) and for each alternative we run a simulation run divided in 30 sub runs handling 1000 cases each. We measure the average throughput time and the average utilization and for the

average throughput time we calculate a 95%-confidence interval. The results are presented in Table 3. For each alternative the 95%-confidence interval for the average throughput time and the average utilization rate are given. The average throughput time for two alternatives differ significantly if the confidence intervals do not overlap. The results show us that alternative 1 with nine 2task-generalists has the lowest average throughput time, although the difference is not significant when compared to alternative 0 (the initial process). Next to this, we see that alternative 5 (with only generic resources) could be significantly improved by specializing some resources, leading to alternative 4. The same holds for alternative 4 for which making more resources specialist would lead to the improved performance of alternative 3. The same argumentation could be applied to alternative 2 and 3.

Table 3. Results for random allocation

Alternatives	Throughput time	Utilization rate
0	[16.134 ; 18.604]	0.84
1	[14.541 ; 16.451]	0.89
2	[16.438 ; 19.086]	0.89
3	[29.694 ; 34.838]	0.92
4	[36.390 ; 43.550]	0.93
5	[51.831 ; 58.895]	0.94

It seems that a process with mainly specialists and some generalists could perform significantly better than alternatives with other ratios of specialists and generalists. Our next step is to focus our search on alternatives with approximately the same ratio as alternative 1 to find a *distinctive ratio* of specialists and generalists. Two configurations of a process with 6 2task-generalists and 30 specialists have an average throughput time which is significantly lower than the average throughput time of the initial process. The configuration with 2 administrators, 2 accountants, 2 assistantsplus, 8 secretaries, 16 clerks, and 6 assistants is called alternative 1a. The configuration with 2 administrators, 2 accountants, 2 assistantsplus, 9 secretaries, 16 clerks, and 5 assistants is called alternative 1b. The results for the alternatives 1a and 1b are stated in Table 4 and for comparison the result for the initial process is included. We can see that alternative 1a and alternative 1b outperform the initial process (and also alternatives 2-5).

Table 4. Results for distinctive ratios

Alternatives	Throughput time	Utilization rate
0	[16.134 ; 18.604]	0.84
1a	[13.529 ; 14.867]	0.86
1b	[13.852 ; 15.538]	0.86

The specialist-generalist trade-off suggests *the improvement of a process by using a distinctive ratio of specialists and generalists*. Our conclusion is that a *distinctive ratio* of specialists and generalists is a ratio with mainly specialists and one or a few generalists. With this ratio the best trade-off is made between the shorter service times of the specialists and the flexibility offered by a generalist. When applying the best practice to the problem the total number of resources has been kept constant. A similar problem would have been encountered if the total salary or another variable had been kept constant.

A precondition for finding a *distinctive ratio* is a high resource utilization, because then the advantage of shorter service times for specialists does not compensated for the flexibility offered

by one or a few generalists. When the utilization rate is lower suitable specialists will always be available and the flexibility is not necessary. Next to this, the distribution of the resources over the different specialized resource types should be proportional, so each resource type has an equal utilization rate. We also made the explicit assumption that generalists need more time to perform a task than specialists. If specialists and generalists have an equal service time, a process with only generic resources will have the best performance.

3.3 Flexible assignment policy

With the flexible assignment of resources *the most specialized (available) resource will be assigned to a work item*. In this way, the more generic resources are preserved and the possibilities for having a suitable resource for the next work item are maximal. There is a smaller chance that a case has to wait for a specific resource and it is expected that the overall queueing time in the process will be reduced. Next to the flexibility, a specialist performs a task faster than a generalist leading to lower service times. When we compare flexible assignment with random allocation we expect the following for the initial process and the five process alternatives (see Table 2 for the definition of the process alternatives). For the initial process both allocations will perform the same, because there are only specialists to allocate. Also, in the first alternative there are mostly specialists available and no difference is expected. In the second alternative the ratio specialists-generalists is half-half and it could be that flexible assignment will select a specialist more frequently than would happen randomly. For the alternatives 3 and 4 there are more generalists than specialists available and it is likely that flexible assignment will allocate significantly more specialists than random allocation would. We expect that allocating the specialist will lead to lower queuing and service times for the process. So, we expect a reduced average throughput time for process alternatives 3 and 4. Alternative 5 has two types of generic resources and maybe allocating the least generic resource will lead to an improvement.

We apply the flexible assignment policy to the six process alternatives specified in Table 2. We also apply it to the alternatives 1a and 1b (with a *distinctive ratio* of specialists and generalists). We use priority based allocation to model flexible assignment. In place *Task_Roles* (see Figure 4) the roles for each task are already prioritized with the specialized resource having the highest priority, the 2task-generalist having a middle priority and the supervisor having a lower priority. Through this priority setting an available specialist will be assigned and generic resources are preserved for the near future. The results of the application of flexible assignment to the 8 alternatives are presented in Table 5.

Table 5. Results for flexible assignment

Alternatives	Throughput time	Utilization rate
0	[16.703 ; 19.731]	0.84
1	[14.569 ; 16.613]	0.88
1a	[14.066 ; 16.158]	0.87
1b	[14.149 ; 15.687]	0.87
2	[15.817 ; 17.410]	0.90
3	[29.131 ; 34.279]	0.93
4	[38.070 ; 44.872]	0.92
5	[47.594 ; 54.428]	0.94

We see that alternatives 1a and 1b again have a significantly lower throughput time than the initial process and also alternative 1 performs significantly better. Next to this, we compare the results of flexible assignment (Table 5) with the results of random allocation (Table 3 and Table 4). Flexible assignment does not seem to improve process performance, because for all alternatives both allocation methods lead to similar results.

We expected that flexible assignment would have an impact on the performance of processes with many generalists and just a few specialists, because the outcome of the allocation methods would be different. With random allocation there is a high chance that a generalist would be selected, while flexible assignment would favor the selection of a specialist. Alternative processes 3 and 4 have 9 specialists and 27 generalists, but the results with flexible assignment do not differ from random allocation. When we look at the results for these alternatives we see that the utilization rate is rather high. When 36 resources have an utilization rate of 0.92 there are on average 33 resources busy and three resources available. Only a quarter of the total amount of resources is specialist and specialists can only perform one of the three tasks. It follows that most of the time only suitable generalists will be available when an allocation has to be made. Because of this, most of the time flexible assignment will also select a generic resource. This leads to the conclusion that flexible assignment will probably act the same as random allocation for processes with a high utilization rate, because in most cases no choice is offered between specialists and generic resources.

We perform new simulations for the alternatives 3 and 4 with less cases arriving to lower the utilization rate of the resources. Table 6 shows the results for an arrival intensity of 120 and an arrival intensity of 150 cases per hour (Poisson arrival process).

Table 6. Significant results for flexible assignment policy

Arrival rate	Alt.	Allocation method	Throughput time	Util. rate
120 cases per hour	3	Random allocation	[12.075 ; 12.249]	0.64
		Flexible assignment	[11.312 ; 11.478]	0.60
	4	Random allocation	[12.489 ; 12.647]	0.67
		Flexible assignment	[11.707 ; 11.993]	0.62
150 cases per hour	3	Random allocation	[12.276 ; 12.614]	0.78
		Flexible assignment	[11.814 ; 12.186]	0.76
	4	Random allocation	[13.076 ; 13.664]	0.81
		Flexible assignment	[12.563 ; 13.019]	0.79

We see that flexible assignment outperforms random allocation for both alternatives 3 and 4 and with both arrival rates. Another interesting point is that although the average throughput time is lower with an arrival intensity of 120 cases per hour the difference between flexible assignment and random allocation is larger. It seems that the proposed relationship between the utilization rate and having a choice between a specialist and a generalist is correct. Flexible assignment does not perform better than random allocation for processes with a high utilization rate, because then there is no choice between suitable specialists and generalists when an allocation has to be made.

With the application of the flexible assignment policy we have gained insight in when to use a flexible allocation method. The flexible assignment policy suggests *the improvement of a process by allocating a specialist* when a choice has to be made between specialized and generic resources. The application of flexible assignment to the bank process led to an improvement of process alternatives with more generic than specialized resources. Flexible assignment will select a specialist more frequent than would happen randomly if there is at least one generic resource. So, in general flexible assignment could improve each process with both specialized and generic resources.

Note that the benefits of flexible assignment will only be observed if the process is not loaded too heavily. There should be enough free specialists in the process to have a choice between suitable specialists and generalists. Next to this, the actual improvement gained with flexible assignment is dependent on the difference in service time between a specialist and a generalist. When the

specialist works faster there is not only the benefit of the preserved flexibility, but also of lower service times.

4 Related Work

In this section we briefly discuss related work on the two main topics of this paper. The first topic is the analysis of business processes with simulation. Regarding the analysis of business processes we describe its position in Business Process Management, the broad use of simulation, and the application of the simulation facility of CPN Tools to business processes. The second topic is the application of best practices. We present their general background, the history of the two evaluated best practices, and an empirical study on the use of specialists and generalists in a call center.

The redesign of business processes based on best practices is part of Business Process Management (BPM). BPM is defined as "Supporting business processes using methods, techniques, and software to design, enact, control, and analyze operational processes..." [4]. Within the four defined phases the focus has traditionally been on enactment and control [4]. In this paper we give attention to (re)design and analysis.

The analysis of business processes is performed with simulations. From practice, the use of simulation is advocated to compare the "to be" alternatives and to understand the "what" and the "why" of the current process and possible alternatives [5]. From a scientific point of view, simulation is used to evaluate quantitative criteria such as throughput time or costs as input for design decisions [6].

For the analysis of the best practices the simulation facility of CPN Tools was used. CPN Tools and Design/CPN, the predecessor of CPN Tools, have been applied in various industrial projects [24], but we have found only two projects for which a business process has been analyzed with simulation. In one project a planning process was modeled with Design/CPN and this process was simulated with the Design/CPN simulator [14]. In another project CPN Tools was used to study the bullwhip effect in supply chains. For this project the supply chain was modeled with CPN Tools and with simulation the bullwhip effect, inventory increase in the chain, was demonstrated [13].

Best practices should be seen as independent rules of thumb helping practitioners in their redesign effort. Best practices are often derived from practical experience within companies and 29 of these best practices have been collected in [18].

In this paper we have evaluated the application of two of these best practices, the specialist-generalist trade-off [17,19] and the flexible assignment policy [3]. [17] identifies cross-training as a process improvement point and refers to cross-training as *training staff to make them capable of performing each others' jobs*. When a specialist should become more generic cross-training is required. A large survey on techniques used for Business Process Re-engineering (BPR) showed that companies consider the training of employees as the most important technique [21]. Toyota, for instance, used a system of cross-training and job rotation to enrich the jobs of their employees [15].

So far, both best practices have been described qualitatively, but little research has been conducted on the underlying mechanisms and the actual improvement gained when applied. An experimental study on the use of specialists and generalists in a call center has been performed by [22]. With simulation it is evaluated if 1) a *one-level* design should be used with only specialists or 2) a *two-level* design with generalists on the first level receiving the call which could be forwarded to specialists on the second level. The results show that a *one-level* design leads to better quality measures and a more regular load sharing between employees, but to higher labor costs.

5 Conclusions and Future Work

In this paper we developed a model for the analysis of resource-constrained processes and the evaluation of process alternatives. The model represents an abstraction of a business process

consisting of a process structure formed by task building blocks placed in sequence or parallel, a generic resource module and a method to allocate resources. While applying the specialist-generalist trade-off and the flexible assignment policy, many process alternatives were modeled and evaluated. The developed CPN model turned out to be very useful for comparing the performance of many process models in a short time. The addition of more tasks or a change in the ordering of the tasks may also be done in a short period of time.

With the application of the specialist-generalist trade-off to a process, a distinctive ratio of specialists and generalists can be found. The distinctive ratio is a combination of mainly specialists and one or a few generalists. Using this ratio instead of the current ratio could improve the process significantly. With a flexible assignment policy a specialist is assigned to a work item when both suitable specialized and suitable generic resources are available. The application of this policy to a process with both specialized and generic resources could lead to an significant improvement of the process. It is remarkable that the use of the specialist-generalist trade-off and the flexible assignment policy is influenced by the same mechanisms, but that the working of these mechanisms is opposite to each other. A useful application of the specialist-generalist trade-off requires high utilization rates and approximate equal service times for specialists and generalists to be effective, while the flexible assignment policy pays off if the utilization rates are lower and the service times of specialists and generalists differ.

We see opportunities for the extension of the developed CPN model. Other process characteristics like different case types and different customer classes influence the allocation of resources and could be added to the model. Next to this more performance indicators could be taken into account. The current model focuses on the timing aspect, but also the performance on cost, quality or flexibility could be interesting. In our model the requests for resources are handled in a FIFO order. The use of other priority rules could also change the performance of the process. With an extended model more insight will be gained on how to allocate resources in an optimal way.

Acknowledgement

We would like to thank Kurt Jensen and Lisa Wells from the University of Aarhus. We appreciate the help Kurt gave in the early stages of the research presented in this paper and we are grateful that Lisa let us use the monitors.

This research is supported by the Technology Foundation STW, applied science division of NWO and the technology programme of the Dutch Ministry of Economic Affairs.

References

1. W.M.P. van der Aalst. Three Good Reasons for Using a Petri-net-based Workflow Management System. In: T. Wakayama et al., editors. Information and Process Integration in Enterprises: Rethinking Documents. Volume 428 of The Kluwer International Series in Engineering and Computer Science. Boston, Kluwer Academic Publicers, pp.161-182, 1998.
2. W.M.P. van der Aalst. Workflow verification: finding control-flow errors using Petri-net-based techniques. In: W.M.P. van der Aalst et al., editors. Business process management: models, techniques, and empirical studies. Lecture Notes in Computer Science 1806. Berlin, Springer-Verlag, pp.161-183, 2000.
3. W.M.P. van der Aalst, K.M. van Hee. Workflow management: models, methods and systems. London, MIT Press, 2002.
4. W.M.P. van der Aalst, A.H.M. ter Hofstede, M. Weske. Business process management: a survey. In: Proceedings of the 2003 International Conference on Business Process Management (BPM2003). Lecture Notes of Computer Science 2678. Berlin, Springer-Verlag, pp.1-12, 2003.
5. F. Ardhaldjian, M. Fahner. Using simulation in the business process reengineering effort. Industrial Engineering, 26(7), pp.60-61, 1994.
6. J. Desel, T. Erwin. Modeling, Simulation and Analysis of Business Processes. In: W.M.P. van der Aalst et al., editors. Business process management: models, techniques, and empirical studies. Lecture Notes in Computer Science 1806. Berlin, Springer-Verlag, pp.129-141, 2000.

7. T. Forsberg, L. Nilsson, M. Antoni. Process orientation: the Swedish experience. Total Quality Management, Volume 10(4-5/July), pp.540 - 547, 1999.
8. K. Jensen. Colored Petri Nets: Basic Concepts, Analysis Methods and Practical Use Volume 1. Berlin, Springer-Verlag, 1992.
9. K. Jensen. Colored Petri Nets: Basic Concepts, Analysis Methods and Practical Use Volume 2. Berlin, Springer-Verlag, 1995.
10. L. Kleinrock. Queueing Systems, Volume 1: Theory. New York, John Wiley and Sons, 1975.
11. L.M. Kristensen, S. Christensen, K. Jensen. The Practitioner's Guide to Colored Petri nets. International Journal on Software Tools for Technology Transfer, 2(1998), pp.98-132, 1998.
12. A. Kumar, W.M.P. van der Aalst, H.M.W. Verbeek. Dynamic Work Distribution in Workflow Management Systems: How to balance quality and performance? Journal of Management Information Systems, 18(3), pp.157-193, 2002.
13. D. Makajic-Nikolic, B. Panic, M. Vujosevic. Bullwhip effect and supply chain modelling and analysis using CPN Tools. CPN Workshop 2004.
14. B. Mitchell, L.M. Kristensen, L. Zhang. Formal Specification and State Space Analysis of an Operational Planning Process. CPN Workshop 2004.
15. R. Muramatsu, H. Miyazaki, K. Ishii. A Successful Application Of Job Enlargement/Enrichment At Toyota. IIE Transactions, 19(4), pp.451-459, 1987.
16. K. McCormack. Business process orientation: Do you have it? Quality Progress, Volume 34(1), pp.51-58, 2001.
17. G. Poyssick, S. Hannaford. Workflow reengineering. Mountain View, Adobe Press Editions, 1996.
18. H.A. Reijers, S. Limam Mansar. Best practices in business process redesign: an overview and qualitative evaluation of successful redesign heuristics. Omega, Volume 33(4), pp.283-306, 2005.
19. A. Seidmann, A. Sundararajan. Competing in information-intensive services: analyzing the impact of task consolidation and employee empowerment. Journal of Management Information Systems, 14(2), pp.33-56, 1997.
20. A. Vinter Ratzer, L. Wells, H.M. Lassen et al. CPN Tools for Editing, Simulating, and Analysing Coloured Petri Nets. In: W.M.P. van der Aalst, E. Best (Eds.). Applications and Theory of Petri Nets 2003. Lecture Notes in Computer Science 2679. Berlin, Springer-Verlag, pp. 450-462, 2003.
21. M. Zairi, D. Sinclair. Business process re-engineering and process management: a survey of current practice and future trends in integrated management. Business Process Re-engineering & Management Journal, 1(1), pp.8-30, 1995.
22. M. Zapf, A. Heinzl. Evaluation of Generic Process Design Patterns: An Experimental Study. In: W.M.P. van der Aalst et al., editors. Business process management: models, techniques, and empirical studies. Lecture Notes in Computer Science 1806. Berlin, Springer-Verlag, pp.83-98, 2000.
23. M. zur Muehlen. Organizational Management in Workflow Applications - Issues and Perspectives. Information Technology and Management, 5(2004), pp.271-291, 2004.
24. http://www.daimi.au.dk/CPnets/intro/example_indu.html