SIMULATING HUMAN RESOURCE CAPABILITY AND PRODUCTIVITY IN SOFTWARE PROCESS SIMULATIONS

Štěpán Kuchař[(a)], Ivo Vondrák[(b)]

VSB - Technical University of Ostrava, IT4Innovations
17. listopadu 15/2172, 708 33 Ostrava, Czech Republic

[(a)]stepan.kuchar@vsb.cz, [(b)]ivo.vondrak@vsb.cz

ABSTRACT
Software processes are considered to be one of the most complex processes because they are very dependent on human behaviour, creativity and productivity. Simulation models are trying to capture and model these properties to provide more precise information about the progress and parameters of the process. This paper describes a method to describe competencies of the workers in the process using competency models. These competencies are used to simulate human resource capability and productivity that influences the duration and allocation of resources to activities in the process. This method is then integrated into the BPM Method modelling and simulation environment that is used in an experiment to compare the allocation strategies in software process simulations for a middle-sized software development company.

Keywords: Software Process Simulation, Discrete Event Simulation, BPM Method, Human Resource Productivity, Competency Model

1. INTRODUCTION

Business processes represent the core of each organization's behaviour (Madison 2005). They define a set of activities that have to be performed to satisfy the customers' needs and requirements, roles and relationships of the employees that are needed for actually performing these activities and objects that are consumed or produced by these activities (Šmída 2007). Software processes are also a special type of business processes that are highly dependent on human creativity, competencies, experience and interaction (Dutoit et al. 2006). Human-based processes tend to be more uncertain than automated processes performed by machines, because human behaviour is very complex and depends on a lot of factors, including capabilities, emotions, social status, health, etc. (Urban and Schmidt 2001). All these aspects influence the productivity of workers in the process and subsequently change the cost and duration of the process and the quality of the final product.

Unfortunately, existing process simulation models are not very concerned with accurate human resources modelling and description (Rozinat et al. 2009) and this can lead to the loss of precision in the simulation results. This paper proposes a capability and productivity model for allocating competent workers to activities during the simulation and dynamically changing duration of such activities based on their skills and abilities.

2. RELATED WORK

Numerous simulation models for software processes were created to evaluate different types of processes to help the companies with determining the best process for their needs and to help them estimate the total cost and duration of their software development projects. Several of these simulation models contain a specification of workers and their capabilities to support the allocation or estimate their productivity.

(Abdel-Hamid and Madnick 1991) proposed a system dynamics model which divided workers in the process to two groups – experienced and newly hired. Part of the experienced workers workload was to train the newly hired workers that were gradually assimilated to the experienced workers stock. This model was used for modelling the delays for the introduction of newly hired personnel to the process.

Experience and capability on a more detailed level was specified in the Generalized Stochastic Petri Net simulation model designed in (Kusumoto et al. 1997). Individual workers were modelled with their experience level in mind with three possible values – novice, standard, expert. Each activity is modelled as a loop of possible communication effort, thinking and writing/creating the result. Each of these transition durations and workloads are influenced by the experience of the worker. The activities do not have any requirements and no allocation model is specified.

(Hanakawa et al. 1998) defines a simulation model with exactly specified activity, productivity and learning models that served as a base for our own model. Each activity is divided into primitive activities and each one of these specifies the required knowledge level. The primitive activities' knowledge levels are distributed normally for each activity. The productivity model then evaluates the amount of work produced in one time step based on the required knowledge level and worker's acquired knowledge. Finally,

the knowledge level uses a learning curve to calculate the increase in the worker's acquired knowledge. In (Hanakawa et al. 2002), this model is further enhanced by detailing the worker's knowledge by a cognitive map. This enhancement is similar to our competency approach, but the paper does not work with the process as the whole and only works with individual activities and individual workers, ignoring the allocation of the right resource for the right activity.

(Hanne and Neu 2004) uses a similar model as (Hanakawa et al. 1998) with each activity specified by one skill and learning curve used for changing the knowledge level of this skill dynamically during the process. Influences of several emotional factors like stress and boredom are also integrated to the model. Allocation of resources is not dealt with.

(Raunak 2009) specifies a very well thought out model for human resource modelling that includes multiple capabilities for workers and multiple requirements for activities, constraints for allocating resources of specified groups or in specified order, bidding mechanism for deciding allocated resource based on his availability and motivation, dynamic change of requirements based on the state of the process, etc. This model has a similar objective as ours, but it does not work with different competency levels and requirements, define importance of individual competencies for an activity and does not provide the means to evaluate the worker's productivity and its effect on the duration and cost of the process.

3. THE BPM METHOD

A modelling and simulation method that is able to sufficiently model human-based processes was needed to provide the simulation environment for the capability and productivity models. For these purposes we used the BPM Method (Vondrák et al. 1999). We had to enhance this simulation environment with stochastic parameters (Kuchař and Kožusznik 2010) and also with the means to share generic resources between concurrent process instances and activities (Kuchař et al. 2012a). This method defines three basic models of the process – architecture of the process, objects and resources utilized in the process and the behaviour of the process. The most important one of these models for performing simulations is the behavioural model. This model is called the Coordination model and it specifies the behaviour of the process as a sequence of activities. It also specifies what resources the activities require and which artefacts they consume and produce. Alternative flow in the coordination model is enabled by multiple activity scenarios and concurrency of the activities can also be modelled using special modelling techniques. This model can also be converted to a Petri net to provide exact semantics for performing simulations (Kuchař and Kožusznik 2010).

The Coordination model is visualized by the Coordination diagram and a simple example of this diagram is shown in Figure 1.

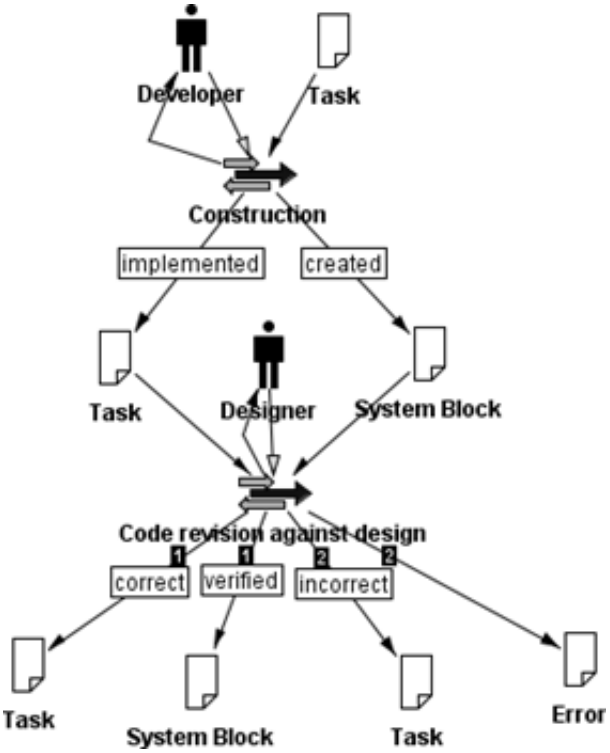

Figure 1: Part of the Coordination Diagram

This diagram describes a part of the Software Construction subprocess. The Designer and Developer active objects describe the roles of employees in the process and the Task passive object defines the task objects that serve as input for the Construction activity. System block can be constructed when the Task is created and the Developer resource is available. The yellow arrow shows that the Developer is responsible for executing the Construction activity. By completing this activity the state of the Task changes to implemented, new System block is created and the Developer is ready to implement another task.

The subsequent activity is Code verification that is performed by the Designer and consumes the implemented Task and created System block. This activity can end up in two ways. The first scenario signifies that the constructed code is correct and its outputs are marked by number 1. The second scenario shows that there were errors in the implementation and the process will continue by reporting and repairing the error. Outputs of the second scenario are marked by number 2.

4. HUMAN RESOURCE COMPETENCIES

Our proposed simulation method uses competency models to describe the capabilities of human resources to perform activities in the process. Competency models (see e.g. (Sinnott et al. 2002; Ennis 2008)) describe various competencies which are important for the process. Competencies are defined as sets of knowledge, abilities, skills and behaviour that contribute to successful job performance and the achievement of organizational results (Sinnott et al. 2002).

Competency models also describe how to measure and evaluate individual competencies. In most cases competencies are measured by a number of advancing stages where higher levels of competency include everything from their lower levels. There is no standard for how many levels a competency model should have and every model defines its own set of levels, so our method is able to work with an arbitrary competency level limit.

Competency levels of resource *r* in the process can be defined using the following vector:

$$\boldsymbol{r} = (l_{r,1}, l_{r,2}, \ldots, l_{r,n}) \quad (1)$$

where $l_{r,1}$, $l_{r,2}$, ..., $l_{r,n}$ are levels of competencies c_1, c_2, ..., c_n that have been mastered by the resource *r*.

Activities in the process also have some requirements on the resources and their competencies. The resources that fulfil these requirements are able to effectively perform the activity, but it does not always mean that resources without required competencies are not able to perform the activity at all. Such resources, which are lacking some of the required competencies, will of course have troubles with the activity, prolonging the duration and increasing the probability of faults in the activity's results. The effect of the lack or abundance of competencies on the activity is specified in section 5.

The requirements of the activity *a* can be described by the following vector:

$$\boldsymbol{a} = (\boldsymbol{rr}_{a,1}, \boldsymbol{rr}_{a,2}, \ldots, \boldsymbol{rr}_{a,m}) \quad (2)$$

where $\boldsymbol{rr}_{a,i}$ ($i \in \{1,\ldots,m\}$) is a vector of requirements for the *i*-th resource required by activity *a* and *m* is a number of resources the activity requires. The required resources are specified by the BPM method as the active objects that are entering the activity as inputs. Each requirement vector is then specified as:

$$\boldsymbol{rr}_{a,i} = (\boldsymbol{rc}_{a,i,1}, \boldsymbol{rc}_{a,i,2}, \ldots, \boldsymbol{rc}_{a,i,n}) \quad (3)$$

where $\boldsymbol{rc}_{a,i,j}$ is a vector that specifies the requirements for competency c_j concerning the *i*-th resource of activity *a*. Structure of the $\boldsymbol{rc}_{a,i,j}$ vector follows:

$$\boldsymbol{rc}_{a,i,j} = (ri_{a,i,j}, rll_{a,i,j}, rlt_{a,i,j}, rhl_{a,i,j}, rht_{a,i,j}) \quad (4)$$

where $ri_{a,i,j}$ specifies the importance of competency c_j for the *i*-th resource required by activity *a*.

$rll_{a,i,j}$ and $rhl_{a,i,j}$ represent the low and high limits for required levels of competency c_j for the *i*-th resource required by activity *a*.

$rlt_{a,i,j} \in \{strictly\ required, requested\}$ describe the type of the appropriate low requirement. The *strictly required* type defines a strict constraint on the competency level. This means that only resources that meet the condition $l_{r,j} \geq rll_{a,i,j}$ can be allocated to the activity. On the other hand, the *requested* type means that even resources with lower competency levels can be allocated if they are chosen by the allocation strategy. $rht_{a,i,j}$ follows a similar pattern only for the high requirement level.

The activity requirements have to be compatible with the resource competencies meaning they have to use the same vector of competencies, the competencies have to be defined on the same scale and have the same meaning. Exact specification of these conditions is stated in our previous work (Kuchař and Martinovič 2013).

4.1. Worker's Capability to Perform Process Activities

Whenever any activity with requirements in any process case needs to start its execution, the simulation needs to allocate one or more resources to perform this activity. These chosen resources have to be suitable for performing this activity by having all *strictly required* competency levels to fulfil the activity's requirements. On top of these hard constraints, both the *strictly required* and *requested* requirements are used to evaluate the resource's capability to perform the activity. This capability can be calculated by encoding the resource competencies and activity requirements to their comparable vector representation and evaluated in the vector space model. The capability of resource *r* for activity *a* is defined as similarity of their vector representations:

$$cap(r, a) = sim(rvect(\boldsymbol{r}), avect(\boldsymbol{a})) \quad (5)$$

where *rvect* is a function that converts a resource vector to the comparable vector representation and *avect* converts an activity vector to this representation (specific conversion and similarity evaluation is elaborated in (Kuchař and Martinovič 2013)).

5. HUMAN RESOURCE PRODUCTIVITY

The resource capability introduced in previous section specifies how skilled and knowledgeable the worker is to perform specific activity. This skill and knowledge level influences the productivity of the resource. (Hanakawa et al. 1998) also agrees with this statement and specifies a productivity model that simulates this influence. We will also use this model in our method.

The productivity model derives the worker's productivity by processing the worker's capability and the required capability to perform the activity. It uses a cumulative distribution function of the standard normal distribution to specify the productivity *P(r,a)* as:

$$P(r,a) = C_{r,a} \int_{-\infty}^{a_a(cap(r,a)-cap(r_a,a))} \frac{1}{\sqrt{2\pi}} e^{-t^2/2} dt = \\ = C_{r,a} \Phi\Big(a_a\big(cap(r,a) - cap(r_a,a)\big)\Big) \quad (6)$$

where $C_{r,a} > 0$ is a maximum of the productivity of resource *r* for activity *a*. $a_a \geq 0$ is a level of accuracy needed to perform the activity *a* and determines the sharpness of the decline in the productivity. *cap(r,a)* is a capability of resource *r* for activity *a* (see section 4.1).

cap(r_a,a), on the other hand, is a required capability for activity a. This capability is derived from the capability of referential resource r_a for activity a. The referential resource of the activity has exactly the same competency levels as those required by the activity and therefore has exactly an average productivity while performing the activity. This average productivity corresponds to the default activity durations specified in the process model.

To simplify further calculations with productivity, it is useful to universally set the $C_{r,a}$ parameter to 2. This leads to the fact that the referential resource r_a always has a productivity of 1 regardless of the value of parameter a_a because

$$cap(r_a,a) - cap(r_a,a) = 0 \quad (7)$$

$$P(r_a,a) = 2\Phi(a_a * 0) = 2 * 0.5 = 1 \quad (8)$$

This can then be perceived as doing 1 unit of work in 1 unit of time. Resources with higher capabilities than the referential resource have higher productivity and can therefore do more units of work in 1 unit of time (e.g. if the worker's productivity is 1.25, she can do 1.25 units of work in 1 unit of time). On the other hand, resources with lower capabilities have lower productivity and therefore less units of work in 1 unit of time.

Figure 2 shows the productivity curves for different values of parameter a_a with $C_{r,a}$ set to 2.

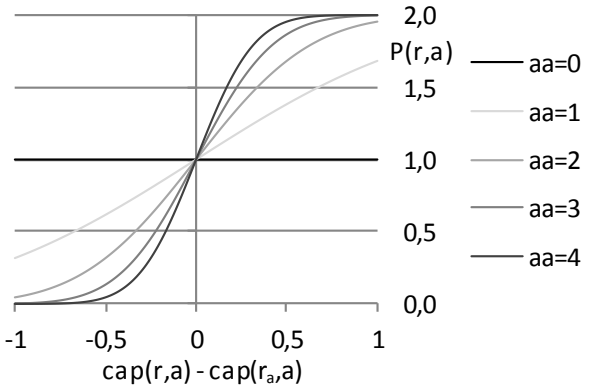

Figure 2: Productivity Curves

The plot in Figure 2 shows that the values of parameter a_a greatly influence the productivity values. Bigger values of the a_a parameter cause sharper rise of the curve, meaning that small changes in capability cause a major change in productivity. This property can be used to differentiate between specialized and universal activities in business processes, specialized activities having a steeper curve and universal activities having a flat curve. It is also useful to allow setting the the a_a parameter to 0 for fully automated activities or for disabling the influence of the productivity for other types of simulations where productivity is not required.

5.1. Productivity and Duration of Activities

The last section described the notion of workers' productivity but how does this productivity translate into the duration of the process instances? Each activity in the process is performed by some resource (either human or non-human) and can be influenced by the resource's competencies and productivity. Each activity can therefore take a different amount of time for different workers even though the amount of work for the activity does not change.

The worker's productivity specifies how much faster the worker can perform an activity. The duration of activity a being performed by resource r can therefore be defined as:

$$d_a(r) = d_a * m(r,a) \quad (9)$$

where d_a is the standard duration of activity a for an average worker and *m(r,a)* is a duration multiplier for resource r and activity a that is specified as:

$$m(r,a) = \frac{P(r_a,a)}{P(r,a)} \quad (10)$$

where *P(r_a,a)* is a productivity of the referential resource r_a for activity a and *P(r,a)* is a productivity of resource r for activity a.

The productivity of the referential resource in equation (10) has to be taken into account, because productivities of all resources are related to the referential worker (see equation (6)). And here, the choice of the $C_{r,a}$ parameter having a value of 2 pays off, because then *P(r_a,a)* is always 1 (see equations (7) and (8)) and it can be removed from equation (10) leading to:

$$m(r,a) = \frac{1}{P(r,a)} \quad (11)$$

Figure 3 shows the duration multipliers *m(r,a)* for different values of parameter a_a.

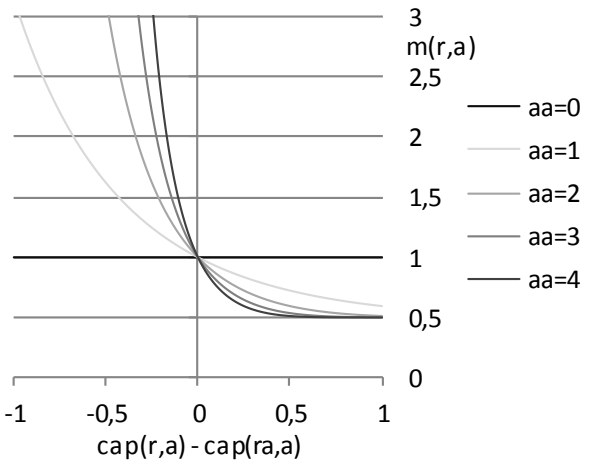

Figure 3: Duration Multipliers

5.2. Introducing Productivity into the BPM Method

The theory described in previous sections can be easily introduced into the simulation engine of the BPM Method because the simulation engine is already prepared to take worker's competencies and capabilities into account (see Kuchař et al. 2012b). The only problem is to implement the productivity enhancements into the simulation engine and link them with the probability distributions that determine the duration of activities.

The duration of activities in the BPM Method is modelled by using a normal distribution specified by two percentiles – low and high boundaries (for more specific details see Kuchař and Kožusznik 2010). These two percentiles are then used to calculate the mean value and variation of the normal distribution. Every time the activity is started, its duration is determined as a random variable from the distribution.

There are three possible ways how to influence the distribution with productivity:

1. Multiply the final duration value acquired from the distribution just at the time when the activity starts.
2. Change the parameters of the distribution just before the final duration value is acquired from the distribution.
3. Change the values of the percentiles before the distribution parameters are determined.

Concerning the normal distribution, all these three possibilities are equal in their accuracy. The first option would be the most efficient, because the basic duration without productivity effect could be evaluated before allocating any resource and then changed right at the start of the activity. But in the future, we would like to add additional distributions for durations, mainly the lognormal distribution that has better properties for modelling the duration of activities (Hanne and Neu 2004). Using the first and second option directly for other distributions could potentially skew the probability of resulting values and each distribution would need to specify the conversion method for them to preserve the distribution of the results. Therefore we chose the safest third option that can be used directly regardless of the chosen stochastic distribution because the distribution parameters are determined after the productivity change itself.

The resulting values of the low percentile $ldp_a(r)$ and high percentile $hdp_a(r)$ of activity a influenced by the productivity of resource r can be evaluated as:

$$ldp_a(r) = ldp_a * m(r,a) \quad (12)$$

$$hdp_a(r) = hdp_a * m(r,a) \quad (13)$$

where ldp_a is the low percentile for activity a, hdp_a is the high percentile for activity a and $m(r,a)$ is the duration multiplier for resource r and activity a.

6. HUMAN RESOURCE ALLOCATION

Knowing how the capability of resources influences their productivity and duration of the process is only half of the problem of simulating processes with specific resources. The second half is a proper allocation of these resources to activities in the process. In manual simulations the best approach is to allow manual allocation of workers by providing information about their capability, productivity and availability, letting the user decide which worker should be allocated to the current activity. This is unfortunately not possible for automatic simulations that have to run without user input during the simulation and have to work only with predefined settings. Because of the stochastic nature, changing conditions, activity and process instance concurrency, every instance of the process is unique and cannot be easily predicted or pre-set. On the other hand it is possible to define several allocation strategies based on the resource capabilities and availability that would find the most appropriate worker for different process conditions.

6.1. Resource Availability, Utilization and Process Waiting Time

Before defining allocation strategies, it is important to define resource availability. One resource cannot perform two activities at the same time and when he is executing some activity in the process he is unavailable to other concurrent activities. If another activity needs the same worker (e.g. one developer is needed to implement a new feature in one system and at the same time needs to repair a fault in another system), she has to perform these tasks sequentially by:

- finishing the first task and then starting the second one, or
- pausing the first task and returning to it after finishing the second one, or
- switching back and forth between these tasks.

The BPM Method is only able to model the first sequencing option and it opens a question if another worker is able to perform the task in shorter timeframe instead of waiting for the unavailable worker. Here, the workers' capabilities and productivity can be used to evaluate if it would be better to wait or to allocate another worker.

The unavailability and demand also present two interesting statistical indicators of the process that will be used in the case study – utilization and process waiting time.

Utilization is measured by simply counting up the time when the resource is performing any activity. It is an interesting result of the simulation but it is not very useful for looking at the performance of the process. Performance is not about how long one resource was doing something in the process, but rather how long did the process have to wait for unavailable resources when they were needed to perform another activity.

It is therefore important to be able to simulate and measure these process waiting times. Whenever an activity is enabled but the resource is unavailable, the BPM Method counts and notes the time needed for the resource to become available to perform the activity. Total waiting time for each resource is then computed by adding up these noted times for the appropriate resource.

6.2. Automatic Allocation Strategies

The notion of unavailability in connection with capability and productivity is very useful for determining allocation strategies. Each activity in the process should have some allocation strategy associated with it to enable running automatic simulations and performing automatic allocations of resources to these activities. Our possible strategies consider following problems and their possible solutions:

1. How high has to be the capability of the worker that can be allocated to this activity?
   (a) based on the percentage of referential resource's capability (e.g. all workers that have higher capability then 80% of referential resource's capability)
   (b) specific number of workers with highest/closest to referential/closest but lower than referential/closest but higher than referential capability
2. In what order should the capable workers be considered for allocating to this activity?
   (a) from highest to lowest
   (b) from lowest to highest
   (c) by the distance from the referential resource
   (d) by the distance from the resource in specific position from the highest/lowest capability
3. Should the activity wait for the first unavailable resource or should it allocate the first resource that is currently available?
   (a) always use the first available resource
   (b) compare unavailability and performance durations and use the resource that can finish the activity first
   (c) always wait for the first resource
4. Which worker should be requested if all capable workers are unavailable?
   (a) use the first worker that will become available
   (b) compare unavailability and performance durations and use the resource that can finish the activity first

By combining possible mentioned solutions different strategies can be created. One possible strategy would for example be to consider all workers that have higher capability then 100% capability of the referential resource, sort them from the highest to lowest capability, always use the most capable but available worker and if all considered workers are unavailable, then use the first worker that will become available.

To abbreviate the description of such strategies for further use in the experiment, we will be using a code based on the numbering used in the solution list. A code for the previous example would then be 1a(100%),2a,3a,4a.

7. EXPERIMENT

We have conceived an experiment to compare the impact of different allocation strategies after implementing the resource productivity extension to the BPM Method. This experiment is based on a simplified software process based on a real process described and modelled in a middle-sized software development company from Czech Republic. This simplified process contains 37 basic activities grouped into 7 standard subprocesses – Requirement Specification, Analysis, Design, Implementation, Testing, Deployment and Post-Deployment Support.

7.1. Configuration of the Experiment

Requirements for activities in the process were evaluated for 16 basic competencies on a 10-level scale with 8 of these competencies being further specified by 16 process instance parameters, thus creating a total of 22 specific competencies. Each activity was assigned to one of 7 roles in the process, each role containing a different number of workers based on a standard project team structure. Analysis was created and managed by 1 Analyst and the following design was elaborated by 2 Designers that were also overseeing code revisions in the Implementation phase. Implementation was performed by 6 Developers and 2 Testers did the testing of the software. Everyone was supported by 1 Administrator and supervised by 1 Project Manager. The post-deployment support of the project results was backed by web-based helpdesk software that was managed by 1 Incident Manager that ensured proper categorization and reporting of incidents. Each of these human resources in the project was evaluated for the same set of 22 competencies on a 10-level scale as activity requirements to ensure their compatibility.

We simulated this process with 12 different allocation strategies to see how the process behaves with the resource productivity extension specified in this paper. For easier comparison of the results and their impact, all activities in the process were using the specified strategy for each simulation run. In real-life simulations, each activity could specify its own allocation strategy that best suits this type of activity. Activity accuracy a_a was also specified globally for all activities and was set to 3.

The first 6 allocation strategies used in the experiment were:

1. 1a(100%),2a,3a,4a
2. 1a(100%),2b,3a,4a
3. 1a(100%),2c,3a,4a

4. 1a(100%),2a,3b,4b
5. 1a(100%),2b,3b,4b
6. 1a(100%),2c,3b,4b

The second 6 strategies were the same with the exception that 1a was set to 80% to identify how the enabling of lower skilled resources influences the process.

7.2. Results of the Experiment

To compare the impact of various allocation strategies and productivity extension on the process, we measured the total duration of the process instance. Each simulation run was done by executing 200 iterations of the simulation to weaken the impact of stochastic properties and risks in the process to enable proper statistical analysis of the results. Figure 4 shows a box plot of total process instance durations for different allocation strategies.

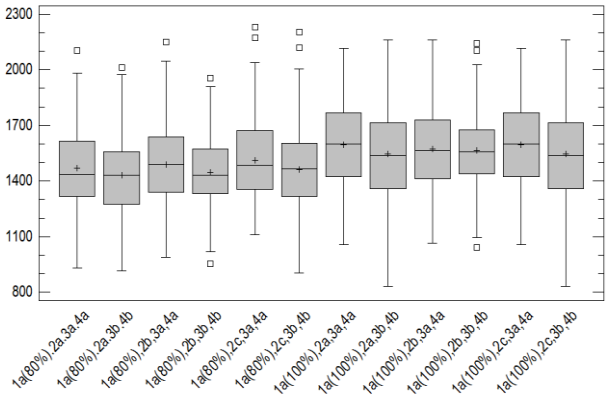

Figure 4: Process Instance Durations for Different Allocation Strategies

By using the test for normal distribution using standard skewness and kurtosis, all runs but one were deemed to follow the normal distribution. The only non-normal run was the 1a(80%),2c,3a,4a with skewness evaluated to 3.13. Looking at the box plot, it is clear that this result is skewed towards lower values. This was probably caused by frequent assignment of lower values for stochastic properties in the process (activity durations and error functions). This proved to be the case because repeated executions of this simulation run had normal skewness and kurtosis.

The first interesting result is directly visible from the plot on top of it being statistically significantly different (at the 95% confidence level). This difference is that the second half of the strategies results in a longer duration. This result was expected because these strategies work only with resources more capable than the referential resource unlike the other strategies that allocate even resources with lower capability (down to 80% of referential resource capability). This difference is also shown in Figure 5 that compares utilization of workers in the process for two representative strategies from different groups. Utilization in the 1a(100%) strategy is very high for few highly skilled workers (Designer1, Developers 0-3, Tester1) and very low for workers with lower capabilities (Designer0, Developers 4-5, Tester0) that could not be allocated to some activities in the process. In 1a(80%) the allocation is more evenly distributed and this leads to lower total duration times even though it takes longer to finish the activities for slightly lower skilled workers.

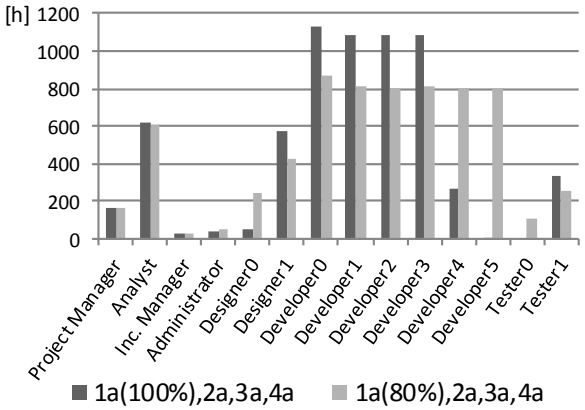

Figure 5: Comparison of Resource Utilization for Strategies 1a(100%),2a,3a,4a and 1a(80%),2a,3a,4a

There are two exceptions for the rule of 1a(100%) strategies taking longer than 1a(80%) ones. The comparison of 1a(80%),2c,3a,4a (lower skilled resources are used first and the first available resource is allocated), 1a(100%),2a,3b,4b (highly skilled resources are used first and the resource with fastest finish time for the activity is allocated) and 1a(100%),2c,3b,4b (resources close to the referential resource should be used first but in the end the resource with fastest finish time for the activity is allocated) provided statistically insignificant difference. This means that primarily allocating workers with low capabilities (and therefore lower productivity) yields similar results in this process as when using the currently fastest resource possible. Strategy 1a(100%),2b,3b,4b should also be included in this exception because it leads to the same strategy as the previous two, but it is slightly off the insignificant difference interval. This can be caused by frequent deviations of stochastic properties in the process or simply by falling out of the 95% confidence interval when deciding significance of their difference.

The second important result was already mentioned in the previous paragraph and concerns the insignificant difference inside each group of 3b,4b strategies. These choices of solutions to the third and fourth allocation questions effectively overshadow the choice for the second question. This is because the order of resources is ignored in favour of finding a worker that will manage to perform the activity first based on the availability of all resources. This hypothesis is proven by the data because all 1a(100%),2*,3b,4b strategies and all 1a(80%),2*,3b,4b strategies are not significantly different.

The same situation is with 1a(100%),2b,3a,4a and 1a(100%),2c,3a,4a that are not significantly different because sorting by the distance from the referential resource is the same as sorting from the lowest capability in this situation. This is because the referential resource is always the lowest capable resource in 1a(100%) strategies.

With expected results put aside, it is time to focus on comparing the remaining strategies with the solution to the second question in mind. Does it matter if the resources are allocated from highest to lowest or from lowest to highest or based by the distance from the referential resource? In this process, the results show that it does not matter, because all strategies that differed only in the second question solution had only insignificant differences. This is probably caused by high process waiting times of highly utilized resources (shown later in Figure 6). This means that highly utilized workers are still supplied by more work and all capable resources are unavailable when allocating resources for an activity. This distributes the work evenly because the process waits for the worker to finish his job. Every worker that finishes his work on an activity is immediately assigned to another activity without much emphasis on his capability. The second choice would be very important in processes that do not have such a high density of activities to be performed, but this is not the case with this experimental process.

This leaves us with the comparison of choosing the first available resource (3a,4a strategies) against choosing the resource that will finish the job first regardless of his availability (3b,4b strategies). It is interesting to look at the process waiting times for workers in the process to see the basic difference in these strategies (see Figure 6).

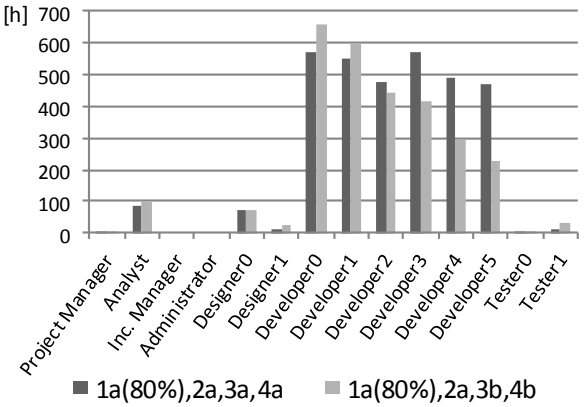

Figure 6: Comparison of Process Waiting Times for Strategies 1a(80%),2a,3a,4a and 1a(80%),2a,3b,4b

The process waiting times are fairly evenly spread in 3a,4a strategies because they always take the first available worker. At the start of the process, workers are allocated with regards to their capability but when all workers are unavailable, the first worker that finishes his activity is assigned to the waiting activity regardless of his capability (he only has to be capable enough to pass the limiting factor set by the first allocation question). On the other hand, 3b,4b strategies have a different pattern of the process waiting times. The most capable (and therefore most productive) workers have higher process waiting time than the less capable workers. This is the product of the "fastest one wins" solution which leads to prioritizing resources with high productivity. Unfortunately, this trend is not followed in the utilization of these workers that is still evenly distributed like in the 3a,4a strategies. This means that even though the more productive resources are chosen for the activity when they are still unavailable, they are allocated to other activity when they become available. This is caused by the fact that resources in the BPM Method can be chosen to several activities when they are unavailable but they can be allocated only to one activity afterwards, leaving other activities to other workers. This only delays the allocation for these activities and it leads to higher process durations. This is also mirrored in the process duration results that were expected to be better for 3b,4b strategies, but experiments showed that they are not significantly different from the 3a,4a strategies. There is only one significant difference between the 1a(*),2a,3b,4b and 1a(*),2c,3a,4a strategies meaning that allocating from the most capable resources on the "fastest one wins" bases leads to shorter durations than allocating from the less capable and taking the first available resource.

8. CONCLUSION AND FUTURE WORK

This paper presented a method for simulating software processes and its extensions for enhancing automatic simulations of human-based processes to provide additional and more precise information about the bottlenecks in the process. Such bottlenecks can be caused by insufficient number of resources in the process or even by wrong allocation of these resources. The proposed simulation solution can be used to try different allocation strategies and find out about their advantages and disadvantages in the simulated process. The productivity extension enables a few of these more complex strategies and at the same time helps to individualize the resources in the process for more transparent simulation results. These extensions are integrated to the BPM Method to provide a robust simulation environment based on the Petri nets formalism.

Integration of the presented extensions to the BPM Method still has one problem that was identified during the experiment. One highly productive unavailable worker can be chosen for several waiting activities at one time but can only be allocated to one of these activities when he becomes available. This will be solved in our future work by creating a prioritized queue for each resource that will enhance the allocation by preferring prioritized activities and streamlining the potential allocation of unavailable resources.

This paper also presented only the time aspect of the productivity and capability of resources in the project but there are additional properties that can

change on the basis of resource capability. For example more capable workers make fewer errors and their capability should influence the error rate of the activity. Our future research will focus on analysing these properties and integrating them to the BPM Method.

Finally, all workers in the process simulations have their competencies pre-set and they do not change during the simulation. In some processes, it is possible to have some training activities that could provide additional competencies for the workers in the process. Even if it is not the case, people are honing their competencies by performing each activity in the process. This learning-by-doing aspect could also be introduced to the BPM Method in our future work.

ACKNOWLEDGMENT

This work was supported by the European Regional Development Fund in the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070).

REFERENCES

Abdel-Hamid T., Madnick S. E., 1991. *Software Project Dynamics - An Integrated Approach*. Prentice-Hall, Englewood Cliffs.

Dutoit A.H., McCall R., Mistrik I., 2006. *Rationale Management in Software Engineering*. Springer.

Ennis M.R., 2008. *Competency Models: A Review of the Literature and The Role of the Employment and Training Administration (ETA)*. US Department of Labor.

Hanakawa N., Morisaki S., Matsumoto K., 1998. A learning curve based simulation model for software development. *International Conference on Software Engineering* (ICSE) 1998, pp. 350-359.

Hanakawa N., Matsumoto K.-i., Torii K., 2002. A Knowledge-Based Software Process Simulation Model. *Annals of Software Engineering* 14, No. 1-4, pp. 383-406.

Hanne, T., Neu H., 2004. Simulating human resources in software development processes. Berichte des Fraunhofer ITWM 64.

Kuchař Š., Kožusznik J., 2010. BPM Method Extension for Automatic Process Simulation. *8th Industrial Simulation Conference 2010*, pp. 80-85. 7-9 June, Budapest, Hungary.

Kuchař Š., Ježek D., Kožusznik J., Štolfa S., 2012. Sharing Limited Resources in Software Process Simulations. *10th Industrial Simulation Conference 2012*, pp. 33-37. 4-6 June, Brno, Czech Republic.

Kuchař Š., Podhorányi M., Martinovič J., Vondrák I. 2012. Simulation of the Flood Warning Process with Competency-based Description of Human Resources. *The 11th International Conference on Modeling and Applied Simulation 2012*, pp. 100-105. 19-21 September, Vienna, Austria.

Kuchař Š., Martinovič J., 2013. Human Resource Allocation in Process Simulations Based on Competency Vectors. *Advances in Intelligent Systems and Computing 188*, pp. 231–240. Springer Berlin Heidelberg.

Kusumoto S., Mizuno O., Kikuno T., Hirayama Y., Takagi Y., Sakamoto K., 1997. A new software project simulator based on generalized stochastic. *Proceedings of 19th International Conference on Software Engineering*, pp. 293-302.

Madison D. 2005. *Process Mapping, Process Improvement and Process Management*. Paton Press.

Raunak M.S., 2009. Resource Management in Complex and Dynamic Environments. *Open Access Dissertations*. Paper 141.

Rozinat A., Wynn M.T., Aalst W.M.P. van der, Hofstede A.H.M. ter, Fidge C. J., 2009. Workflow simulation for operational decision support. *Data & Knowledge Engineering* 68 (9), pp. 834–850.

Sinnott G.C., Madison G.H., Pataki G.E., 2002. *Competencies: Report of the competencies workgroup, workforce and succession planning work groups*. New York State Governor's Office of Employee Relations and the Department of Civil Service.

Šmída F. 2007. *Zavádění a rozvoj procesního řízení ve firmě*. Grada Publishing, a.s.

Urban C., Schmidt B., 2001. PECS - Agent-Based Modelling of Human Behaviour. *Emotional and Intelligent II - The Tangled Knot of Social Cognition*, AAAI Fall Symposium.

Vondrák I., Szturc R., Kružel M., 1999. BPM – OO Method for Business Process Modeling. *ISM '99 Proceedings*, pp.155-163. Rožnov pod Radhoštěm, Czech Republic.