# Performing Business Process Redesign with Best Practices: An Evolutionary Approach

Mariska Netjes[1], Selma Limam Mansar[2], Hajo A. Reijers[1], and Wil M.P. van der Aalst[1]

[1] Eindhoven University of Technology, PO Box 513, 5600 MB Eindhoven, The Netherlands, m.netjes@tue.nl,
[2] Carnegie Mellon University, Information Systems Program, PO Box 24866, Doha, Qatar

**Abstract.** Although extensive literature on BPR is available, there is still a lack of concrete guidance on actually changing processes for the better. In this paper we propose and detail out an evolutionary approach towards business process redesign. We describe the steps to derive a better performing redesign using the existing process model as input. The redesign steps are: 1) computing process measures, 2) evaluating condition statements to find applicable redesign "best practices", 3) creating alternative models according to these best practices, and 4) evaluating the performance of the created alternatives. The end result is a new process model. We show the applicability of the steps with a case study. Our approach has a formal basis to make it suitable for automation.

**Key words:** Business Process Redesign, process modelling, Business Process Management, workflows, best practices

## 1 Introduction

It seems hard to overestimate the value of good support for business process change projects. But as has been argued again and again (see e.g. [2, 20]), a general and broadly accepted understanding is lacking of what to change exactly in a business process to have it perform better (e.g. at lower throughput times) or conform better (e.g. to regulations). Although numerous papers and books on Business Process Redesign (BPR) were published during the past 15 years (e.g. [8, 9, 5]), guidance for concrete process redesign is scarce. Valiris and Glykas [20] identify that "there is a lack of a systematic approach that can lead a process redesigner through a series of steps for the achievement of process redesign". This paper fits within our aim to fill this void by providing a redesign approach which describes and supports the steps to derive from an existing process a better performing one.

A key element of the approach that we propose in this paper is the generation of diagnostic information on an existing business process. Such a diagnosis is performed using so-called *process measures*, which provide a global view on the characteristics of the process. Actual values for the process measures may reveal weaknesses in the process. Identified weaknesses are removed by the application of one or more *redesign best practices*. A redesign best practice describes a well-tried way to remove a particular problem from a process to improve its performance. In our earlier work we have described 29 redesign best practices and presented a qualitative description, the potential effects and possible drawbacks for each best practice [19]. Finally, in our approach, the application of redesign best practices leads to one or more alternative candidates for the existing process. The evaluation of the performance of each alternative shows the best candidate process which should replace the existing one.

Our approach builds on the formal representation of a business process using Petri nets, in particular WorkFlow nets [1]. Some of the features of our model are inspired by the process modelling tool Protos [18]. Current versions of Protos are in use by thousands of organizations in more than 25 countries. In The Netherlands alone, more than half of all municipalities and insurance companies use Protos for the specification of their business processes. The focus on this real-life tool illustrates the applicability of our approach, while it is still easy to see how our approach can be generalized to other modelling techniques and tools (e.g. ARIS).

We envision as the ultimate goal of our research the delivery of an automated redesign tool. This tool would support all steps of the approach in an “intelligent” way. By this, we mean that the tool will not only automate the various steps of the approach, but will also interact with the redesigner. The redesigner will be able to indicate which performance dimensions (time, costs, quality) should be improved, whether certain process characteristics should perhaps not be changed, and which promising alternatives should be combined in constructing the best alternative. Our approach is a solution that should primarily help redesign novices in finding process alternatives based on best practices. Secondly, more experienced redesigners are supported in the creation and evaluation of such alternatives in a structured and less time-consuming manner.

The structure of the paper is as follows, Section 2 describes the related work, Section 3 details out the steps of our approach using a running example and Section 4 presents our conclusions and future work.

## 2 Related Work

Various more structured approaches to process redesign were proposed earlier, most notably the ProcessWise methodology [7] and the MIT Process Handbook [12]. Also, a variety of tools is available, e.g. MIT’s process recombinator tool [6], a number of tools that apply case-based reasoning [10, 13], and the KOPeR tool by Nissen [16]. Many existing approaches and tools are limited in their application domain, while none of the approaches has succeeded to gain widespread adoption in industry. We have provided a more extensive literature review on this topic in [14].

Nissen’s work [15, 16, 17] is most related to our approach which motivates its following deeper discussion. Its main contribution with the KOPeR tool is the construction of a set of measures that, applied to processes, would at first help to diagnose pathologies. The pathologies are then mapped to matching transformations that may be applied to the processes in order to improve their performances. Although the tool does not generate new designs itself, experiments suggest that the tool “performs redesign activities at an overall level of effectiveness exceeding that of the reengineering novice”.

Nissen's work has inspired us to come up with a similar approach that nonetheless overcomes some of the KOPeR tool's shortcomings:

– On the process modelling side, the process model in use by KOPeR is yet simple and the provided examples are rather simplistic. Our process model is defined as an enriched WorkFlow net allowing for the modelling of realistic, complex business processes.
– Nissen used graph-based definitions of the measures in order to operationalize them. We have noticed that the exact meaning of some of the measures is unclear. We use a formal notation to overcome this and define our measures unambiguously.
– Nissen adds an extra layer of indirection (process pathologies and process transformations). We only define a set of measures and a set of transformation rules to immediately find the applicable transformations.
– Nissen's set of presented transformation serves as an illustration and is far from a complete coverage of the spectrum of redesign options. We provide a more exhaustive list of rules based on our set of 29 best practices [19].

## 3 Evolutionary Approach

In our *evolutionary* approach towards workflow process redesign we take an existing process and improve it using redesign best practices. It is evolutionary, because an existing process is taken as starting point instead of a clean sheet. Our approach starts with a (formal) model of an existing process and consists of four steps:

(1) Compute process measures,
(2) Evaluate condition statements to find applicable best practices,
(3) Create alternative models based on the selected best practices,
(4) Evaluate the performance of the created alternatives.

The best alternative is the new process (model) that should replace the process we started with. Our approach is depicted in Figure 1.

The computed process measures of step (1) can be seen as global process characteristics. Values for the process measures are derived from the existing process model and point out weaknesses in the process. For each best practice it is known which process weaknesses it could solve and with step (2) the corresponding process measures are combined in one condition statement per best practice. When a statement evaluates to true it suggests the application of the associated best practice. All condition statements are evaluated to find the best practices which are eligible to be applied to the process. We strive to include as many redesign best practices as possible in our approach and we assume that our set of best practices is complete. In step (3) the selected best practices are used to create alternative models. A best practice has essentially the following parts: some kind of construction or pattern that can be distinguished in the existing process, an alternative to be incorporated for the redesign and a context-sensitive justification for this redesign. Finally, in step (4), the performance of the created alternatives is evaluated and the best alternative is selected. This alternative process model is implemented

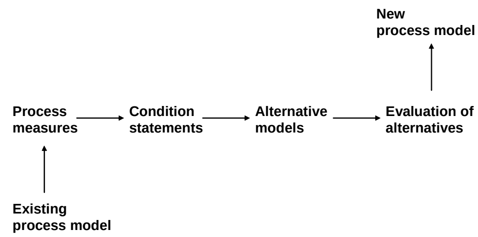


**Fig. 1.** Evolutionary approach towards redesign

as the new process.

In the remainder of this section we detail out the four steps from Figure 1. We use a case study to illustrate the concrete application of the steps in practice.

### 3.1 Process Model

The starting point of our evolutionary approach to workflow process redesign is the existing process model. In order to illustrate our findings we use a case study that describes the process of handling insurance claims.
Let us first describe the process: The process handles the insurance claims of both individual and business clients. The process starts when a claim is received. After receipt, the claim is classified as “individual” or “business”. Then the claim is checked for validity. Three checks, *Check policy*, *Check amount* (only for business clients, requires the receipt of a damage report) and *Check legal* are performed. A check either results in OK (proceed with next check) or not OK (reject claim). Claims that pass all checks are accepted and paid. Payments are authorized at the end of each day by the finance manager. For all claims (both rejected and accepted) a letter is written and the claim is archived.
To model this process, we use Worfklow nets. A workflow process is case-based, i.e. every piece of work is executed for a specific case, and make-to-order. A Petri net which models a workflow process definition (i.e. the life-cycle of one case in isolation) is called a WorkFlow net (WF-net). In a WF-net, the workflow management concept *task* is modelled as the Petri net concept *transition*, *conditions* are modelled by *places*, and *cases* are modelled by *tokens*. Definition 1 gives the WF-net definition. For more information on WF-nets the reader is referred to [1].

**Definition 1 (WF net)** *A Petri net $PN = (P, T, F)$ is a WF-net (Workflow net) if and only if:*

*(i) There is one source place $i \in P$ such that $\bullet i = \emptyset$.*
*(ii) There is one sink place $o \in P$ such that $o \bullet = \emptyset$.*
*(iii) Every node $x \in P \cup T$ is on a path from $i$ to $o$.*

The WF-net represents the process structure or control flow. The process structure is annotated with information related to transitions (such as external triggers, the type of activity to be executed, XOR-splits and -joins to model choices, responsible departments, required applications, and handled products and services). The organizational model uses roles at its foundation. A role is a collection of complementary skills. Allocating roles to transitions ensures that work is performed by the relevant person. Roles have a hierarchical relation, i.e. if two roles have the following relation $(r', r)$ this means that role r is one step higher in the hierarchy than role $r'$ and that role r is also able to perform the transition(s) allocated to role $r'$. The related information and organizational model are defined as an annotation of the WF-net in Definition 2.

**Definition 2 (Annotation)** *Let $PN = (P, T, F)$ be a WF-net.*
*$AN = (B, C, S, J, D, DT, E, ET, G, GT, R, H, U, A, AH)$ is an annotation of PN where*

- *$B : T \to \mathcal{P}(\{time, periodic, digital, mail, telephone\})$ relates each transition to zero, one or more trigger types;*
- *$C : T \to \mathcal{P}(\{basic, communication, check, authorize, batch\})$ relates each transition to zero, one or more activity types;*
- *$S : T \to \{AND, XOR\}$ relates each transition to a split type element;*
- *$J : T \to \{AND, XOR\}$ relates each transition to a join type element;*
- *D is a non empty, finite set of departments;*
- *$DT : T \to \mathcal{P}(D)$ relates each transition to zero, one or more departments;*
- *E is a finite set of applications (i.e. software tools);*
- *$ET : T \to \mathcal{P}(E)$ relates each transition to zero, one or more applications;*
- *G is a non empty, finite set of products and services;*
- *$GT : T \to \mathcal{P}(G)$ relates each transition to zero, one or more products and services;*
- *R is a non empty, finite set of roles;*
- *$H \subseteq (R \times R)$ is a (acyclic) set of hierarchical role relations;*
- *$U \in R \to \mathcal{N}$ is a non empty finite bag of users. $\mathcal{N}$ is the set of natural numbers $\{0, 1, 2, ...\}$. U is a bag, i.e., for some role $r \in R$, U(r) denotes the number of users having role r as the highest role;*
- *$A : T \not\to R$ relates each transition to zero or one roles (allocation);*
- *$AH : T \to \mathcal{P}(R)$ relates each transition to zero, one or more roles (hierarchical allocation) [Note that for $t \in dom(A)$: $AH(t) = \{r \in R | \ (A(t), r) \in H^*\}$ (with $H^*$ being the reflexive transitive closure of H) and for $t \notin dom(A)$: $AH(t) = \emptyset$].*

*$(PN, AN)$ is an annotated WF-net.*

A process can be modelled according to the annotated WF-net with the modelling tool Protos [18]. We made one assumption regarding the creation of a process model. We only take into account the structural properties of a process model and abstract from behavioral information.

Let us now use the Protos model for our insurance claim process. The model of the insurance claim process is shown in Figure 2 and is easy to understand.

The rectangular boxes in the process model are the transitions ($T$, Definition 1), each transition has an activity type indicated by the symbol in the box ($C$, Definition 2). The talking balloon indicates the activity type “communication”, the trapezium indicates “basic”, the check box “check”, and the V-symbol “authorize”. The three rectangles on

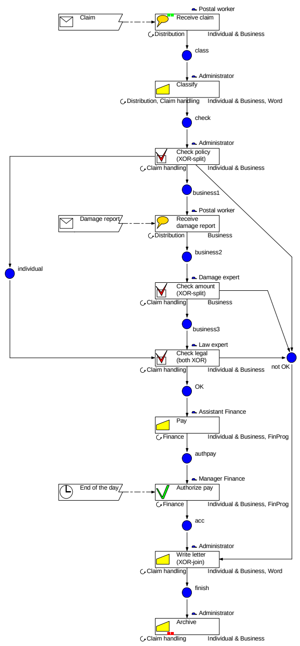


**Fig. 2.** The existing insurance claim process.

the left side of the process are triggers (*B*). A trigger indicates that a certain (external) condition has to be fulfilled before the transition it links to can be executed. In this example, a "Claim", a "Damage report" and the "End of the day" are used. The split and join types are set to their values in the transition properties, but to show them in the model XOR-splits and -joins are also explicitly stated in the name of a transition. On the top right side of the transition the role allocated to the transition is given (For example, the first transition "Receive claim" is executed by the role "Postal worker"), on the bottom left side the related department(s) (Department "Distribution" for the transition "Receive claim"), and on the bottom right side the products / services ("Individual and business" for the transition "Receive claim") and the required applications ("Word" for the transition "Classify"). The organizational model related to the process model is depicted in Figure 3 and is also created with Protos. It shows the roles and the number of resources per department.

Looking at Figure 2 and 3, it is not easy to spot inefficiencies in the process. Hence the next step of our approach, the use of process measures.

### 3.2 Process Measures

The first step of our evolutionary approach is the computation of the process measures for the existing process model. In this section we define our set of process measures. The starting point for the creation of the process measures has been the work of Nissen [16]. Nissen identified 19 (static) process measures [15] of which ten appeared to be

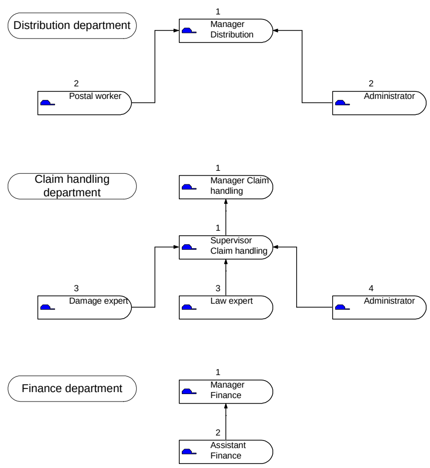


**Fig. 3.** Organizational model of insurance claim process.

relevant in relation to the redesign best practices. The graph-based definition Nissen presented for these measures is replaced by our formal definition because the formal definition provides a more precise and unambiguous meaning for the measures.
Besides the measures included from Nissen we developed eight new measures. These measures are all related to the additional information incorporated in the annotated WF-net. In Table 1 the process measures are defined, their range is given and it is indicated which measures are taken from Nissen and which are new. Table 1should be read with Definition 1 and 2 as context.

To illustrate our results on the process measures, let us apply them to our insurance claim process. Using the set of measures defined in Table 1 we compute the process measures. The computed values are presented in Table 2. For instance, the measure *level of control* is defined as the percentage of control tasks. In the insurance claim process, there are 3 (control transitions) divided by 10 (transitions) resulting in a *level of control* of 0.3.

In the next step, we use and combine the set of process measures to determine condition statements per best practice. We know which weaknesses in a process a best practice would help to alleviate and we derive in the next section condition statements accordingly. Let us note that the complete set of measures has been created and defined iteratively with the development of the condition statements. Each condition statement is connected to one best practice. We assume our set of best practices is complete and in this work we include all best practices which require process, data or resource information for their selection. We hypothesize that all relevant measures which could be derived from this information and which are necessary for the condition statements are included in our set of measures.

**Table 1.** Process measures.

| Process measures with range [0, ..., 1] | | |
|---|---|---|
| Parallelism* | $= \frac{\vert T_{par}\vert}{\vert T\vert}$, perc. of parallel tasks[1] | |
| Process contacts | $= \frac{\vert\{t\in T\mid communication\in C(t)\}\vert}{\vert T\vert}$, perc. of communication tasks | |
| Batch | $= \frac{\vert\{t\in T\mid batch\in C(t)\}\vert}{\vert T\vert}$, perc. of batch tasks | |
| Periodic | $= \frac{\vert\{t\in T\mid periodic\in B(t)\}\vert}{\vert T\vert}$, perc. of periodic tasks | |
| Level of control | $= \frac{\vert\{t\in T\mid check\in C(t)\}\vert}{\vert T\vert}$, perc. of control tasks | |
| Level of authorization | $= \frac{\vert\{t\in T\mid authorize\in C(t)\}\vert}{\vert T\vert}$, perc. of authorization tasks | |
| IT automation* | $= \frac{\alpha\cdot\vert\{t\in T\mid ET(t)\neq\emptyset\wedge t\notin dom(A)\}\vert + \beta\cdot\vert\{t\in T\mid ET(t)\neq\emptyset\wedge t\in dom(A)\}\vert}{(\alpha+\beta)\cdot\vert T\vert}$, perc. of (semi-)automated tasks | |
| IT comm.* | $= \begin{cases} \frac{\vert\{t\in T\mid digital\in B(t)\wedge communication\in C(t)\}\vert}{\vert\{t\in T\mid communication\in C(t)\}\vert}, & for\ \{t\in T\mid communication\in C(t)\}\neq\emptyset \\ 1, & for\ \{t\in T\mid communication\in C(t)\}=\emptyset \end{cases}$ | |
| Department involvement* | $= \frac{\vert D\vert}{\vert T\vert}$, perc. of departments | |
| Department share | $= \frac{\vert\{t\in T\mid \vert DT(t)\vert\geq 2\}\vert}{\vert T\vert}$, perc. of tasks shared by departments | |
| Process hand offs* | $= \frac{\vert\{t_1,t_2\in T\mid t_1\bullet\cap\bullet t_2\neq\emptyset\wedge AH(t_1)\cap AH(t_2)=\emptyset\}\vert}{\vert\{t_1,t_2\in T\mid t_1\bullet\cap\bullet t_2\neq\emptyset\}\vert}$, perc. of work that is handed over to another role | |
| Specialization* | $= \frac{\vert\{A(t)\mid t\in dom(A)\}\vert}{\vert T\vert}$, specialization of roles (with a higher perc. meaning more specialists) | |
| Role usage* | $= \frac{\vert\{A(t)\mid t\in dom(A)\}\vert}{\vert R\vert}$, perc. of actively involved roles | |
| Managerial layers* | $= \frac{\vert lrp\vert}{\vert R\vert}$, perc. of hierarchical layers[2] | |
| Knock outs | $= \frac{\vert\{p\in P\mid \vert\bullet p\vert>1\wedge(\forall_{t\in\bullet p}\ check\in C(t)\wedge S(t)=XOR\wedge\vert t\bullet\vert>1)\wedge(\exists ep\in EP_{PN}\mid \bullet p\subseteq\vert\vert ep\vert\vert)\}\vert}{\vert P\vert}$, perc. of k.outs[3] | |
| | | Range |
| Process size* | $= \vert T\vert$, the number of transitions | [1, 2, ..., number of transitions] |
| Versions | $= \vert G\vert$, the number of products and services | [1, 2, ..., number of products and services] |
| User involvement* | $= \frac{\vert U\vert}{\vert T\vert}$, the average number of users per task | [0, ..., 1, ..., number of users] |

* = measure taken from Nissen [15]

1. A parallel transition, $T_{par}$, is defined as $T_{par} \subseteq T$ such that $t \in T_{par}$ if and only if there exist two elementary paths that both start in an AND-split and end in an AND-join, $t$ is on only one of these two paths, and the AND-split and AND-join are the only two nodes these paths have in common.
2. A role path, rp, in PN is defined as a nonempty sequence $r_k...r_1$ of roles which satisfies $(r_k, r_{k-1}), ..., (r_2, r_1) \in H$. Let $RP_{PN}$ be the set of all role paths in PN. Then a longest role path, $lrp \in RP_{PN}$, is defined as a role path which satisfies $\forall_{rp\in RP_{PN}}\ |lrp| \geq |rp|$.
3. An elementary path, ep, in PN is defined as a nonempty sequence $a_1...a_k$ of nodes which satisfies $(a_1, a_2), ..., (a_{k-1}, a_k) \in F \wedge \forall_{1\leq i<j\leq k}\ a_i \neq a_j$. Further, let $EP_{PN}$ be the set of elementary paths in PN and let $||ep||$ be the set of all nodes in the elementary path ep.

**Table 2.** Values for process measures.

| | |
|---|---|
| Parallelism = 0 | Department share = 0.1 |
| Process contacts = 0.2 | Process hand offs = 0.4 |
| Batch = 0 | Specialization = 0.7 |
| Periodic = 0.1 | Role usage = 0.7 |
| Level of control = 0.3 | Manag. layers = 0.3 |
| L. of authorization = 0.1 | Knock outs = 0.1 |
| IT automation = 0.2 | Process size = 10 |
| IT communication = 0 | Process versions = 2 |
| Department inv. = 0.3 | User involvement = 2 |

### 3.3 Condition Statements

The second step of our evolutionary approach is the evaluation of condition statements to select applicable redesign best practices. For each best practice we derive one condition statement which includes one or more process measures. The application of a certain best practice should be suggested when the condition statement is fulfilled. The values used in the condition statements are based on our own redesign experience and expectation of when a certain best practice is applicable. The validation of these values will be future work.

We have created condition statements for 17 out of the 29 best practices.

Regarding the remaining 12 best practices: 1) four best practices appeared to have conditions similar to other best practices and are combined, 2) four best practices are not included because measures beyond the process level are necessary to come to a proper condition statement, 3) four best practices can not be covered due to behavioral dependencies which are not incorporated in the process definition.

We have created the following condition statements:

**Task Elimination**: When applying task elimination unnecessary tasks are removed (remember: task is a synonym for transition).
Condition statement: Apply task elimination IF *Level of control* $> 0.2$.

**Task Automation**: When applying task automation tasks are automated.
Condition statement: Apply task automation IF *IT automation* $< 0.5$ OR (*IT communication* $< 0.5$ AND *Level of control* $> 0.2$).

**Knock-Out**: When applying knock out tasks resulting in a knock out are re-ordered.
Condition statement: Apply knock-out IF *Knock outs* $> 0$.

**Parallelism**: When applying parallelism tasks are placed in parallel.
Condition statement: Apply parallelism IF *Parallelism* $< 0.1$.

**Split Responsibilities**: When applying split responsibilities the responsibility for a task will be given to one department.
Condition statement: Apply split responsibilities IF *Department share* $> 0$.

**Numerical Involvement**: When applying numerical involvement the number of departments / roles / resources in the process is reduced.
Condition statement: Apply numerical involvement IF *Department involvement* $> 0.25$ OR *User involvement* $> 1$ OR *Role usage* $< 0.5$.

**Specialist-Generalist**: When applying specialist-generalist resources may be turned from specialists into generalists or visa versa.
Condition statement: Apply specialist-generalist IF *Specialization* $< 0.4$ OR *Specialization* $> 0.6$.

**Contact Reduction**: When applying contact reduction contacts are eliminated or combined.
Condition statement: Apply contact reduction IF *Process contacts* $> 0.1$.

**Case Types**: When applying case types new workflow processes and product types are distinguished.
Condition statement: Apply case types IF *Process versions* $> 1$.

**Technology**: When applying technology, systems like WorkFlow Management Systems and Database Management Systems are introduced.
Condition statement: Apply technology IF *IT automation* $< 0.5$ OR (*Parallelism* $< 0.25$ AND *Process hand offs* $> 0.5$).

**Case-Based Work**: When applying case-based work each case is handled individually.
Condition statement: Apply case-based work IF *Batch* $> 0$ OR *Periodic work* $> 0$.

**Task addition**: When applying task addition controls are added (at the beginning and end of the process).
Condition statement: Apply task addition IF *Level of control* $< 0.05$.

**Task composition**: When applying task composition tasks with the same role are combined.
Apply task composition IF *Parallelism* $< 0.25$ AND *Process hand off* $< 0.3$ AND *Process versions* $< 2$.

**Control relocation**: When applying control relocation controls are moved to the client.
Condition statement: Apply control relocation IF *Level of control* $> 0.2$ AND *IT communication* $> 0.5$.

**Triage**: When applying triage tasks are divided in alternative tasks for different case types.
Condition statement: Apply triage IF *Process versions* $> 1$ AND *User involvement* $>$ *Process versions*.

**Case manager**: When applying case manager for each case one person is appointed as responsible.
Condition statement: Apply case manager IF *IT automation* $> 0.75$ AND *Process contacts* $> 0.2$.

**Empower**: When applying empower decision-making authority is given to employees and middle management is reduced.
Condition statement: Apply empower IF *Managerial layers* $> 0.2$ AND *Level of authorization* $> 0.1$.

For the insurance claim process 11 of the 17 condition statements evaluate to true. It is straightforward, using Table 2, to check that the first 11 condition statements (from Task Elimination to Case-Based Work) are true for the insurance claim process. The related best practices are selected for the redesign of this process.

### 3.4 Alternative models

The third step of our approach derives new process models based on the selected best practices. At this stage our approach does not support an automatic identification of where in the process a best practice should be applied. This is included in our future work. However, for the sake of completeness, we here illustrate the remainder of the approach for the insurance claim process. For the best practices for which the condition statements evaluated to true we include a possible application:

- Task elimination: eliminate the control task *Check legal* and ask the client to indicate whether (s)he or some one else was responsible for causing the damage.
- Task automation: automate the receipt of client information with a web interface.
- Knock out: perform *Check amount* and *Check legal* in a different order if *Check amount* requires longer service times and / or has a lower rejection probability than *Check legal*.
- Parallelism: place the three checking tasks *Check Policy*, *Check amount* and *Check legal* in parallel which should reduce the throughput time of the process.
- Split responsibilities: give the responsibility for the task *Classify* solely to the Distribution department.
- Numerical involvement: reduce the number of departments and / or the number of users involved in the process.
- Specialist-generalist: make some of the roles in the process more general to have more flexibility in the process.
- Contact reduction: reduce the number of contacts with the client and ask the business client to send the damage report together with the claim.
- Case types: distinguish one workflow process for the individual claims and one for the business claims.
- Technology: introduce a workflow management system.
- Case-based work: remove or change the periodic activity *Authorize pay* which should reduce waiting times.

### 3.5 Evaluation of Alternatives

In the final step of our approach, the performance of the various alternatives is evaluated and one redesign alternative is selected for implementation. For the evaluation, performance data (time, cost and quality indicators) are necessary. Evaluation can be done by simulating the model or (in simple processes) by using more analytic approaches (e.g. queueing networks). The alternative that provides the best performance is selected. We have earlier found that 11 condition statements (thus best practices) evaluated to true for the insurance claim process. Implementing the best practices separately would lead to 11 redesign alternatives for the insurance claim process. Each redesign project has goals (for instance improvement on throughput time or operational costs) and project risks [11] which makes some alternatives more promising than others. For the insurance claim process an improvement in throughput time will be achieved with the application of *Task elimination*. An improvement on costs could result from the use of the *Knock out* best practice. A careful evaluation with performance data is necessary to see which alternative will indeed be the best and should replace the existing insurance claim process.

## 4 Conclusion and Outlook

In this paper we describe and illustrate an evolutionary approach towards workflow process redesign. Our contribution in this paper focuses on the first two steps of the approach leading to the proposal of the applicable best practices. This proposal will already help redesign novices with the creation of redesign alternatives. We introduce a formal process definition suitable for modelling realistic, complex business processes. Our process measures have a clear and unambiguous meaning because of their formal notation. Furthermore, our process measures are directly related to the redesign best practices with condition statements.

Our current work holds limitations that we will be addressing in the future. One direction for future research is the extension of the current process definition, for instance with performance data about historic process instantiations, to be able to set up condition statements for all redesign best practices (steps 1 and 2). In [3] we argue that performance information from a real process (collected in event logs which are derived from the execution of, for instance, a BPM system) may be used for this. However, at this point in time, existing BPM systems provide limited support for this log-based analysis.

Another important direction will be the exact place in the process model where a suitable best practice should be applied and the derivation of the alternative model (step 3). The simulation of an alternative model to obtain its performance may be based on data (event logs) derived from the actual process. Log-based extension of a process model with a new aspect or perspective (e.g., enriching the model with performance data) is part of the process mining research [4] (step 4).

We also aim at automating our approach with a highly interactive redesign tool. In addition to merely generating process alternatives on the basis of an existing model, such a tool will be able to process the preferences of the redesigner for a subset of the alternatives to continue its search for a satisfactory design. The interaction with the redesigner and the advanced support will hopefully make our tool a truly “intelligent” system for BPR.

## Acknowledgement

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. Aalst, W.M.P. van der. The Application of Petri Nets to Workflow Management. *The Journal of Circuits, Systems and Computers*, 8(1):21–66, 1998.
2. Aalst, W.M.P. van der. Reengineering knock-out processes. *Decision Support Systems*, 30(4):451–468, 2000.
3. Aalst, W.M.P. van der, Netjes, M., Reijers, H.A. Chapter 4: Supporting the Full BPM Life-Cycle Using Process Mining and Intelligent Redesign. In Siau, K., editor, *Contemporary Issues in Database Design and Information Systems Development*, pages 100–132. IGI Global, Hershey, USA, 2007.
4. Aalst, W.M.P. van der, Weijters, A.J.M.M. Process Mining: A Research Agenda. *Computers in Industry*, 53(3):231–244, 2004.
5. Al-Mashari, M., Zairi, M. BPR implementation process: An analysis of key success and failure factors. *Business Process Management Journal*, 5(1):87–112, 1999.
6. Bernstein, A., Klein, M., Malone, T.W. The process recombinator: a tool for generating new business process ideas. *International Conference on Information Systems*, pages 178–192, 1999.
7. Calvert, P. An Advanced BPR Methodology with Integrated, Computer-based Tools. In Glasson, B.C., Hawryszkiewycz, I.T., Underwood, B.A., Weber, R.A., editor, *Business Process Re-engineering: Information Systems Opportunities and Challenges*, pages 161–170. Elsevier, 1994.
8. Grover, V., Jeong, S., Kettinger, W., Teng, T. The Implementation of Business Process Reengineering. *Journal of Management Information Systems*, 12(1):109–144, 1995.
9. Kettinger, W., Teng, J., Guha, J. Business process change: A study of methodologies, techniques, and tools. *MIS Quarterly*, 21(1):55–80, 1997.
10. Ku, S., Suh, Y.H. An Investigation of the K-tree Search Algorithm for Efficient Case Representation and Retrieval. *Expert Systems with Applications*, 11(4):571–581, 1996.
11. Limam Mansar, S., Reijers, H.A., Ounnar, F. BPR Implementation: A Decision-Making Strategy. In Bussler, C., Haller, A., editor, *Business Process Management Workshops: BPM 2005*.
12. Malone, T.W., Crowston, K., Herman, G.A. *Organizing Business Knowledge: The MIT Process Handbook*. MIT Press, 2003.
13. Min, D.M., Kim, J.R., Kim, W.C., Min, D., Ku, S. IBRC: Intelligent Bank Reengineering System. *Decision Support Systems*, 18(1):97–105, 1996.
14. Netjes, M., Vanderfeesten, I., Reijers, H.A. “Intelligent” Tools for Workflow Process Redesign: A Research Agenda. In Bussler, C., Haller, A., editor, *Business Process Management Workshops: BPM 2005*, volume 3812 of *Lecture Notes in Computer Science*, pages 444–453. Springer Verlag, Berlin, 2006.

15. Nissen, M. Knowledge-Based Organizational Process Redesign: Using Process Flow Measures to Transform Procurement, PhD. Dissertation. University of Southern California, downloadable at http://web.nps.navy.mil/ menissen/, 1996.
16. Nissen, M. Redesigning Reengineering through Measurement-Driven Inference. *MIS Quarterly*, 22(4):509–534, 1998.
17. Nissen, M. An Experiment to Assess the Performance of a Redesign Knowledge System. *Journal of Management Information Systems*, 17(3):25–44, 2000.
18. Pallas-Athena. *Protos User Manual*. Pallas Athena BV, Apeldoorn, The Netherlands, 2004.
19. Reijers, H.A. and Limam Mansar, S. Best Practices in Business Process Redesign: An Overview and Qualitative Evaluation of Successful Redesign Heuristics. *Omega: The International Journal of Management Science*, 33(4):283–306, 2005.
20. Valiris, G. and Glykas, M. Critical review of existing BPR methodologies. *Business Process Management Journal*, 5(1):65–86, 1999.