Smart Tools and Visual Analytics

Hartmann Genrich, Process Analytica, Germany
Robert Shapiro, Process Analytica, USA

INTRODUCTION

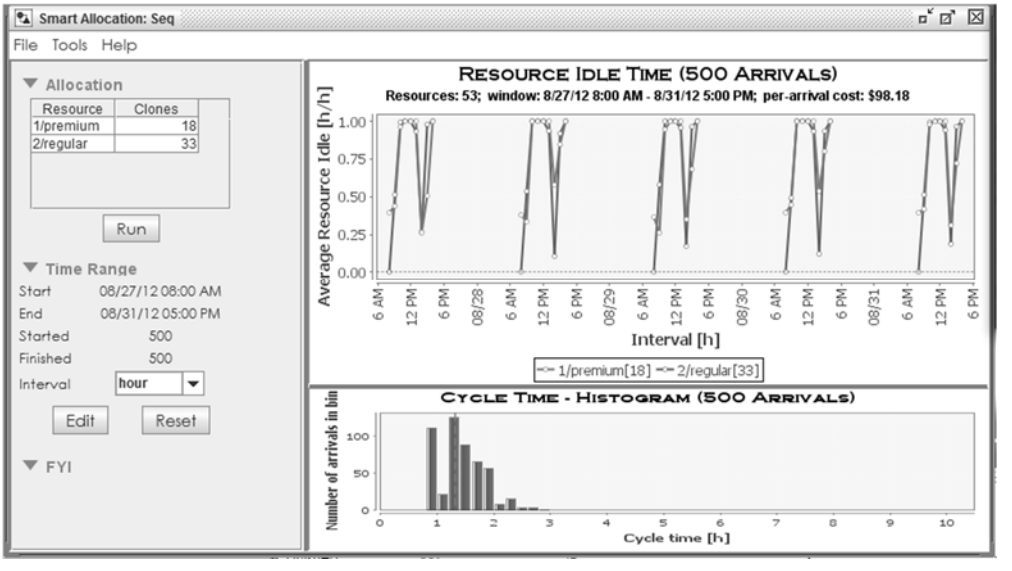
Smart tools of *PA Optima* support fast optimization by simulation and analysis using only that subset of the analytics required by the particular optimization technique. They are rather 'autistic' and focus on efficiency rather than interaction and generality.

So far two such tools are available: *Smart Resource Allocation* and *Smart Productivity Improvement*. We briefly describe Smart Allocation because the optimization algorithms in it are also used in Smart Productivity.

SMART ALLOCATION

Once started, the *Smart Resource Allocation* tool allows in a quick 'assess, change and try' iteration the changing of the number of copies ('clones') of each resource, simulation of the given set of work item arrivals and assessing the results in terms of resource idleness (upper chart), cycle times (lower chart).

Figure 1



Approach

The approach of *Smart Resource Allocation* is to support a quick iteration of 'assess, change and try' steps. In addition to changing the allocation manually there are two automatic allocation modes: *initial allocation* and *idleness reduction*.

An *idleness* chart, and a *cycle time* chart support assessment.

Assessment

The tool allows the assessment of a particular allocation in several respects.

1. The *per-arrival costs* appear in the title of the resource idle time chart (upper pane) and on top of the FYI text field. The costs are calculated on the basis of the times the resources are on shift (scheduled) during the given time window (default: simulation period) and their respective salaries, divided by the number work items that finish in that window.
2. The idle time chart (upper pane) plots idleness of every resource over the time axis.
3. The FYI text shows a simple characterization of *idleness distribution*. The plot is virtually divided into three bands (*upper*, *middle* or *lower*) of equal width and for each resource the percentage of idle time in each band is listed.
4. The *cycle times* are plotted in the lower pane of the window, optionally as histogram or scatter plot.[2] They allow a quick assessment of performance under the chosen allocation.

The initial example was created by initial resource allocation with maximum hourly workload (see *Initial Resource Allocation* below). It obviously gives high performance at a high cost. In the sequel we lower the performance to a tolerable level (work item finished within one workday) in order to lower the costs.

Resource Idleness

A *resource idleness* chart shows for every period of selected length (full hour | full day) and every resource an *average idleness* between 0 and 1. These values are calculated as follows by *resource utilization* analysis.

Let:

res be a resource,

R_{res} = (*res.clones* + *1*) be its capacity (the number of copies),

schd = *res.schedule* be its schedule,

prd be a period of the x-axis[1].

The algorithm calculates variables

$busy_{res,prd}$*:* the sum of times some copy of *res* spent processing a task during period *prd,*

$onshift_{res,prd}$*:* the time period prd intersects with the times that resource res is scheduled to work.

It takes the set of all occurrences *occ* of tasks in all histories (see [1]) where the resource field of *occ* denotes a copy of *res*; it adds to *busy* the sum of the time *occ* spent processing during period *prd*. This requires splitting the processing period of *occ* into pieces belonging to shift instances of schedule *schd* and add the intersections of those pieces with period *prd*.

The results are stored in maps

$OnShift : res \times prd \rightarrow onshift_{res,prd}$

$Busy : res \times prd \rightarrow busy_{res,prd}$

[1] The x-axis starts with the period that contains the simulation start time and ends with the period that contains the simulation end time.

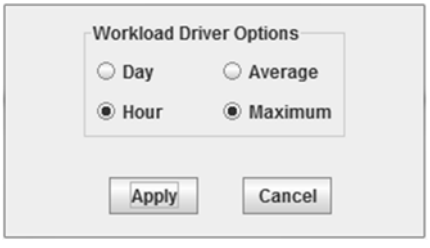
The variable $idle_{res,prd} = (onshift_{res,prd} * R_{res} - busy_{res,prd}) / onshift_{res,prd} * R_{res}$ is the *average idleness* of (the copies of) *res* relative to the time *res* is on shift during period *prd*. It is plotted in the idleness chart.

Initial Resource Allocation

Initial resource allocation uses a two pass approach.

First, the workflow is simulated with a *'universal'* resource, a special device of the simulator providing unlimited resource capacity for any performer. The *universal* resource can perform any role and work concurrently on multiple tasks. It allows the simulation to generate the 'ideal' workload drivers. The business schedule determines when the business is 'open'. The resulting simulation produces the fastest end-to-end cycle time as if we were willing to schedule enough resources to handle the sharpest peak loads.

In a variation of the resource utilization algorithm above, the *Busy* map becomes the set of workload drivers for each performer. Depending on the selected option the demand per hour for each performer is determined. Then a resource is defined for each performer with the default salary and performance defined for the performer and the number of clones determined by the workloads and the selected options.



Example: if option *hour&max* is selected and the peak hourly demand of performer *premium* is 20 hours, then a resource is created that plays role *premium* and has 20 copies (19 clones).

With the resulting resource allocation the workflow is simulated again. The results of the second pass are then loaded into a new copy of the allocation tool.

Idleness Reduction

The idleness reduction algorithm, too, uses the results of the *resource utilization* algorithm above. For each resource it removes clones one by one as long as its remaining average idleness does not fall below 20%; this lower limit 0.2 of idleness is a wired-in heuristic constant that prevents the resources from being maxed out.

We started with the initial allocation created above, see Once started, the *Smart Resource Allocation* tool allows in a quick 'assess, change and try' iteration the changing of the number of copies ('clones') of each resource, simulation of the given set of work item arrivals and assessing the results in terms of resource idleness (upper chart), cycle times (lower chart).

Figure 1. After two applications of idleness reduction – the first one leaves a little room for a second and final round – we get an allocation with clone counts considerably reduced; see Figure 2 and Table 1. The cost per arrival goes down from $98.18 to $24.44 and the work is still finished within one work day (the outliers are due to the weekend).

Figure 2

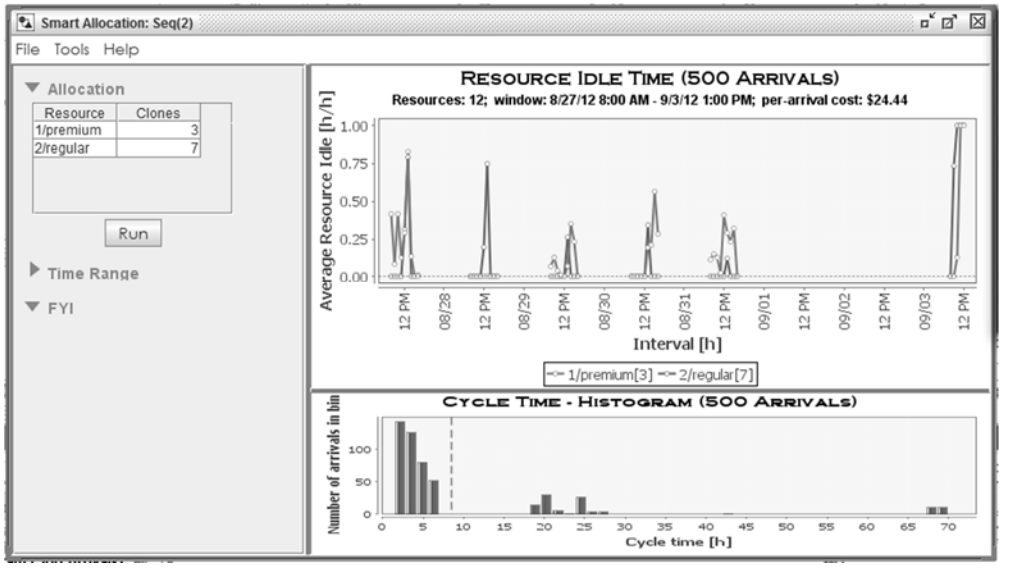
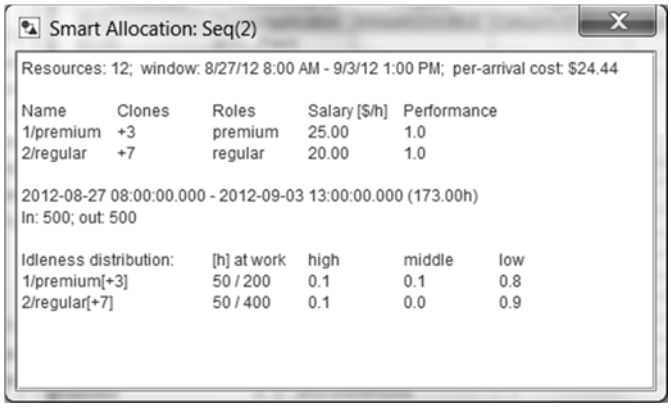


Table 1



The idleness distribution given in the text window confirms the visual impression of the idleness chart that for both resources their idleness is in the lower sector most of the time. Further manual reduction may push the per-arrival costs further down but the cycle times will go up drastically.

SMART PRODUCTIVITY

In some way all components of *Optima* are concerned with productivity improvement. This productivity tool, however, is different and novel. It goes one step beyond identifying potential improvement; it facilitates *Cost/Benefit* and *Return-on-Investment* analysis of *proposed improvement* measures.

Our running example is once more the somewhat superficial view (expressed in BPMN [3]) of a workflow handling the applications for a bank account of type *regular* or *premium*; see Figure 3 and Figure 4. We will see how different proposals can be assessed with respect to their costs and benefits and to the return on the necessary investments.

Figure 3
A workflow (top-level process, sub-processes collapsed)

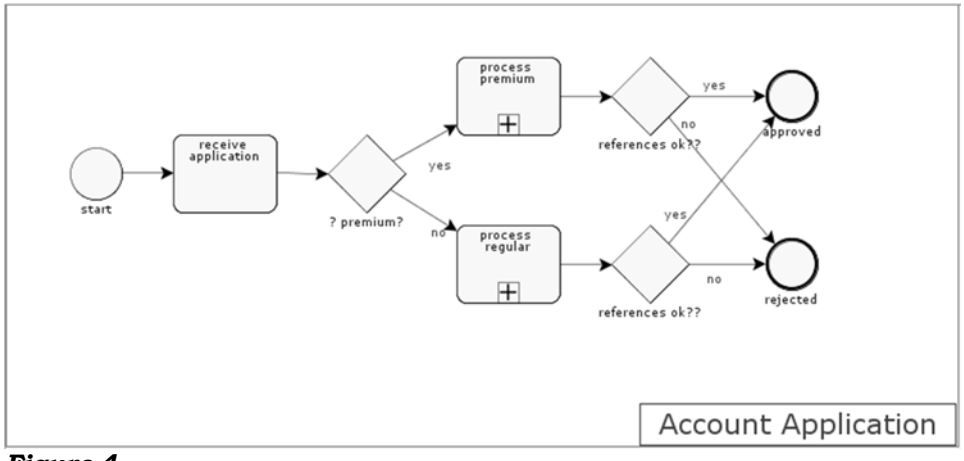
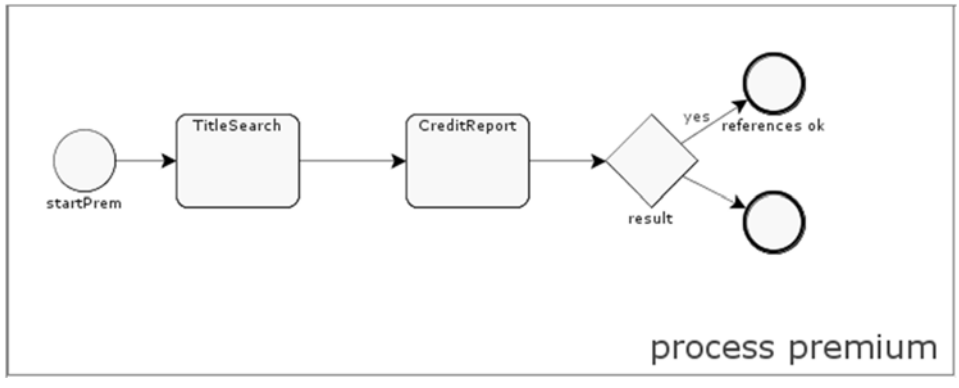


Figure 4
Sub-process details

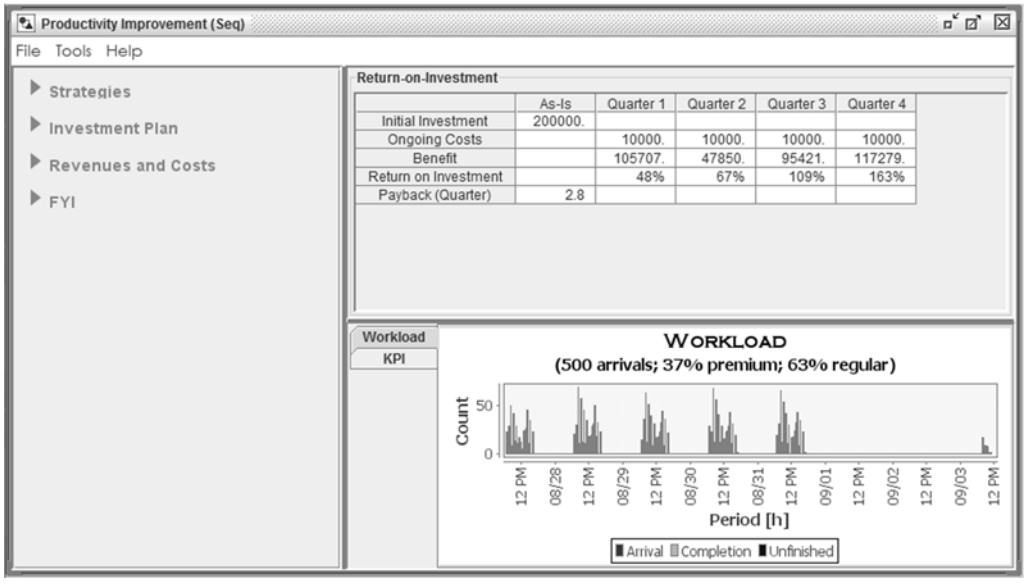


There are two types of windows in the *Productivity Improvement* tool. The primary window, the *project* window (see Figure 5), is opened for a loaded scenario via the *Tools* menu of *OPTIMIZE* or the respective scenario button in *ANALYZE* (chart, table and tool browser). It has three major panes, the *Parameter Setting* pane, the *Return-on-Investment* prediction pane and a tabbed pane with two auxiliary charts, *Workload* and *Cycle Time*, for a quick assessment of performance. An optional text window, *FYI*, presents additional information in textual or tabular form.

The secondary, auxiliary windows (see Figure 6) present the details of the *Cost/Benefit* analysis that populates the ROI prediction. It is started by a right-click on the *As-Is* or some *To-Be* column.

Project Window

Figure 5
The Productivity Improvement Window

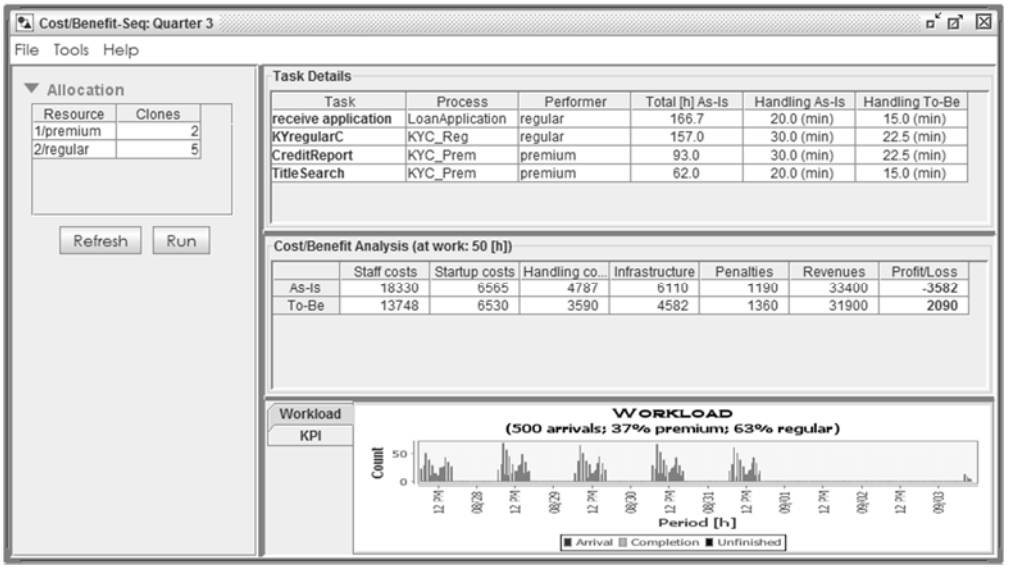


Cost/Benefit Window

A window with detailed *Cost/Benefit* analysis, of the *As-Is* scenario or of any of the *To-Be* intervals, is started by a right-click on the respective column of the ROI table. The tool

- depicts various details of the run for the respective interval,
- allows a quick series of 'fine tuning' experiments with changed resource allocation and/or task simulations details.

Figure 6
Cost/Benefit Details



Functionality, Modes of Operation

Auto-Allocation

In order to make resource costs comparable the resource capacities (numbers of copies/clones for each resource) are re-calculated with the same rules for each step. The algorithm is a simple version of initial resource allocation (see *Smart Resource Allocation* [1]).

For every performer the total workload entailed by the set of arrivals is calculated.

Then a resource is constructed with a capacity (number of copies/clones) sufficient to accomplish this work load during the same number of workdays as work items arrive (principle *of balanced load and capacity*). The assumption here is that

- distribution and period of arrivals is representative,
- task performers are chosen such that resources don't have to wait for work.

Example:

arrival period:	*5 workdays*
workday (business schedule):	*8 hours*
total workload for performer *regular*:	*300 hours*
resource capacity required for *regular*:	*300 / (5 * 8) = 7.5*
resource[2] :	*'2/regular'* with *7* clones

WorkLoad Growth

If we set the growth parameter of the strategies group to a percentage ≠ *0* we expect the workload, the collection of work item arrivals, to grow (or to shrink, for

[2] The resource is created and named by the auto-allocation algorithm. Its name consists of a sequence number (id) and the first characters of the performer.

that matter). Arrivals are specified as *batches* with a *start* date/time, a *size*, a *recurrence pattern* and value assignments for *data fields* (process properties). These batches are expanded by the simulator into collections of *data field inputs* (often called arrivals, too) at the start of a simulation.

Applying a growth ≠ *0%* must only change the *size* of batches which is an *integer* number > *0*. If the sizes are big enough we can just add or subtract the specified amount without much loss of precisions. If some sizes are too small, however, this might change the character of the workload too much.

An alternative approach might be to pick at random the required number of data field inputs (batch instances) and remove them or add a copy. However, it is the arrival batches that belong to the scenarios and they cannot be abstracted from the list of individual batch instances.

Hence the growth is applied by a combination of the two ideas.

Let *G* be the growth *factor*, *G = 1 + growth/ 100%*.

Let *delta* be a variable that holds the accrued rounding errors; it is *0* initially;

In a first pass, for each batch with size S and recurrence count R,

the size is set to *S' = round(S * G)*,

the error *(S' – S * G) * R* is added to *delta*.

In a second pass, repeatedly,

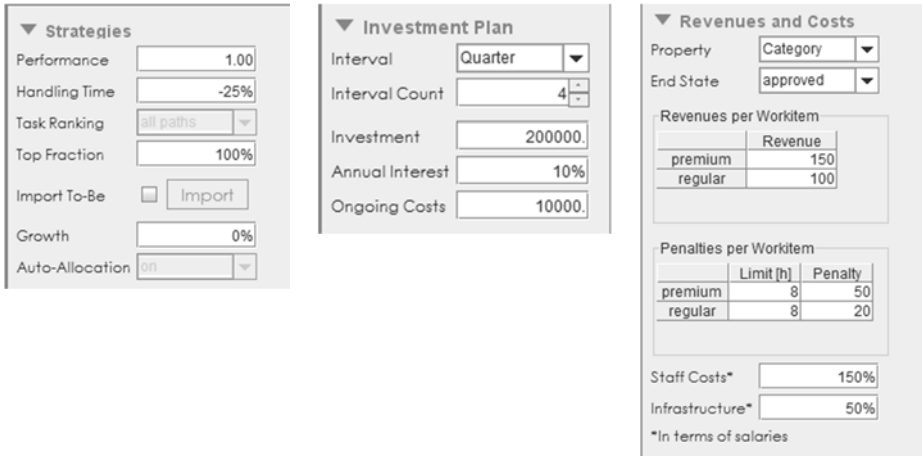
a batch is selected at random,

its *size* is increased respectively decreased by *1*,

delta is reduced by the batch's recurrence count, *R*,

until *delta* is exhausted.

Return-on-Investment Prediction



STRATEGIES

The tool offers three options for improving the productivity.

1. Invest in training to improve staff performance.
2. Invest in reducing the handling time at tasks.
   a. Tasks are ranked according to their total handling time. You can choose to invest in the top n %.
3. You can import a separate To-Be model with its own simulation details.

   a. If you check 'import' the import button becomes enabled.
   b. If you start a run with import checked and no To-Be imported yet you will be asked to do so.
4. If you plan the investment to prepare for more business you can set the growth per interval to the expected percentage – to be applied from the second interval on.

REVENUES AND COSTS

The assumption is that work items that *end* 'successfully' yield *revenues* and that the amount depends on a data field denoting a certain *kind* of work item. Similarly, work items that take too long to finish may result in some loss that can be quantified by a *penalty.*

In our example,

- revenues are specified for work items that finish with an *approved* state and the amount depending on the values *premium* or *regular* of process property *Category*.
- similarly, penalties have to be experienced if the cycle times exceed 8 hours.

For the operating costs it is assumed that they consist of staff costs (salaries plus benefits etc.) infrastructure costs (e.g. office space, equipment) and can be expressed in proportion to the costs for staff salaries. In our example, the total costs of operation are assumed to be twice the costs for salaries

CONCLUSION

The *Smart Productivity Improvement* tool is novel and different from all other components of *Optima*. It goes one step beyond identifying potential improvement. It combines functionality of existing applications for cost/benefit and return-on-investment analysis with the *Visual Analytics* power of *PA Optima*; it facilitates the quick assessment of *proposed* improvement measures.

The tool is still in the state of a prototype rather than a full-fledged return-on-investment application. However, it demonstrates the feasibility of the approach as well as the power of *PA Optima* in general.

REFERENCES

[1] *Workflow Histories in Optima*. Technical Report. Process Analytica (2013)

[2] *The JFreeChart Class Library*, Version 1.0.13. Developer Guide. Object Refinery Ltd. (2009)

[3] Bruce Silver: *BPMN Method and Style*, 2nd Edition. Cody-Cassidy Press (2011)

[4] *Smart Resource Allocation*. Technical Report. Process Analytica (2013)

[5] Robert M. Shapiro and Hartmann J. Genrich: *Optima Visual Analytics.* Process Analytica (In Preparation)