Towards Process Design for Efficient Organisational Problem Solving

Dariusz W. Kaminski

Department of Computing and Communications
The Open University
Milton Keynes, United Kingdom
Email: dariusz.kaminski@ihs.com

Jon G. Hall

Department of Computing and Communications
The Open University
Milton Keynes, United Kingdom
Email: J.G.Hall@open.ac.uk

***Abstract*—This paper presents initial results on stochastic business process modelling. We interpret business processes as problem solving processes to show that certain trade-offs in stakeholder involvement can usefully be analysed. Our initial results illustrate the analytic potential of our stochastic model, which could be useful to business process analysts and designers. We show that our approach can be used to model and reason about business process resource usage. To support process improvement, and to help design process models enriched with relevant, measurable and comparable characteristics, we embed our business process modelling efforts in Problem Oriented Engineering, where we benefit from a systematic approach to problem solving.**

***Keywords*–Process modelling; Business process design; Problem Oriented Engineering.**

I. INTRODUCTION

Organisational problem solving has become an incredibly complex topic [12]. The development of business processes as an organisation's response to problems they face is hampered by this complexity, and the tools currently available for solving organisational problems are relatively unsystematic.

Research by the second author in Problem Oriented Engineering (POE) offers an approach to the structuring of organisational problems that has been successfully applied in many organisational contexts, from the design of seating arrangements to business process reengineering in a financial engineering setting [7].

Business problems and business processes do not exist in vacuum – they are embedded in organisational contexts. Each business process is enacted by a single organisation, although it may also interact with business processes performed by other organisations [2]. When identified, a structure of a business process comprises of a set of work activities across time and place, with a beginning and an end, as well as clearly specified inputs and outputs [4]. Process identification is only one part of a lifecycle, which falls under the umbrella of Business Process Management (BPM). BPM is the art and science that is focused on work designed and executed in an organisation so that it can be done consistently and efficiently [6]. Business processes go through a lifecycle of four iterative phases: design, implementation, enactment and diagnosis [5]. During the design phase, the process creation follows a detailed requirements analysis of business operations, and in this phase process models can be enriched with functionality such as simulation, which enables *what-if* analysis. Quantifying business process characteristics is an important issue, and a research agenda introduced by Wynn *et al.* [20] focuses on the significance of cost-awareness in the area of BPM.

In this paper, we interpret part of POE as a stochastic process and consider the analytic potential this provides for the modelling of business processes with the focus on the cost/resource use metric.

The paper is organised as follows. Section II provides some background on POE. Section III presents the link between POE and business processes. In Section IV, we map the POE Process Pattern to its stochastic representation. Section V gives an overview of the study from the industrial context, which we then model with our stochastic model. An evaluation of the study and its results is given in Section VII. Related work is discussed in Section VI, while Section VII concludes the paper.

II. BACKGROUND

Rogers' definition of engineering [18] states that (abbreviations *E*, *S*, and *N* added by the authors):

> "Engineering refers to the practice of organising the design and construction of any artifice (*S*) which transforms the physical world (*E*) around us to meet some recognised need (*N*)."

Problem Oriented Engineering encodes this definition as a problem solving exercise – an engineer's task is to find S that satisfies N in environment E: represented by the proposition $E, S \vdash N$ meaning that that, when S is installed in the environment E, their combination meets the need N. For reasons of space, we do not expand on the usage of POE problem notation; the interested reader is referred to [9].

For this paper it is sufficient to consider the structure that POE suggests for problems solving steps that is illustrated in Figure 1, in which rectangles are resource consuming activities; diamonds indicate requests to stake-holders for validation either – on the left – of problem understanding, or – on the right – of a candidate solution.

POE characterises the problem solving process as illustrated in Figure 2. Briefly, in the POE Process Pattern (PPP) there are four groups of agents that interact with each other during the problem solving activity: problem explorers, problem validators, solution explorers and solution validators. During problem (resp., solution) exploration a problem's context and requirements (resp., a solution) are understood and described. Problem (resp., solution) validators are available to

 ISBN: 978-1-61208-395-7

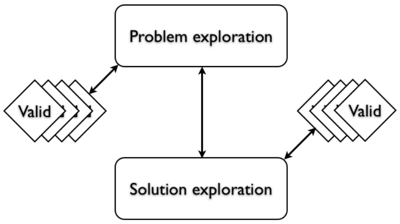

Figure 1. The POE Process Pattern: iteration between problem and solution exploration with interleaved validation (adapted from [8]).

problem (resp., solution) explorers and will perform validation (if requested) of the current problem (solution) description [7].

Each state in the PPP can be linked to the state of a problem's elements during problem solving, states represented by $\times$ (meaning element unknown or invalid), c (meaning element has a candidate description) and ✓ (meaning element has been validated). Thus, for example, $E\checkmark, S\times \vdash Nc$ represents a POE problem whose environment has been validated (by a problem validation step), the need has been proposed but not yet validated, and the solution has not yet been explored.

As shown, unsuccessful problem validation will typically result in problem exploration being extended. Successful problem validation will typically result in solution exploration being begun. Problem validation allows developmental risk to be transferred from explorer to validator. Unsuccessful solution validation is more complex as not only can it reveal that the solution does not satisfy a problem (the feedback loop between solution validation and problem exploration in Figure 2), but also that the problem was misunderstood (whether validated or not; the arc to 8 in Figure 2).

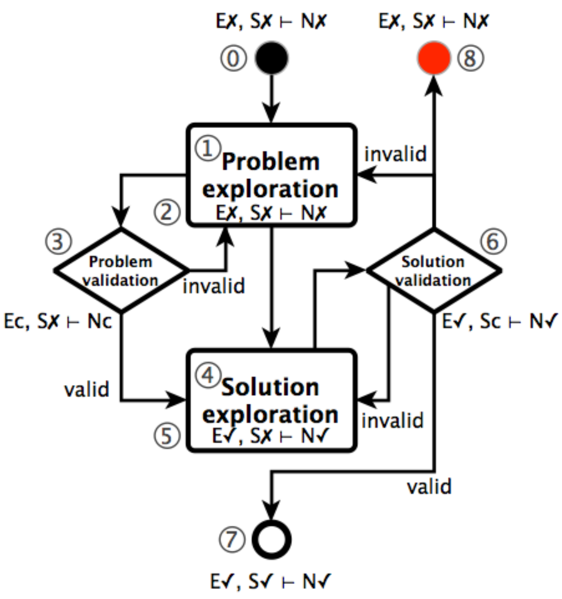

Figure 2. POE problem $E, S \vdash N$ at different points of problem solving in PPP. This is a simplified version of the pattern, where processes are sequentialised for the purposes of modelling (based on [8]). The numbers representing states will be explained in Section IV.

In PPP, it is possible for the problem exploration to be followed immediately by the solution exploration without the problem validation step, i.e., when the problem validator is not asked for validation. This possibility is shown in Figure 2 as the transition from the problem exploration to the solution exploration.

III. Business Processes as Problem Solving

Jonassen defines a problem [14] in terms of two attributes: 1) a problem is an unknown entity in some situation (the difference between a goal state and a current state); 2) finding or solving for the unknown must have some social, cultural, or intellectual value. The finding takes place during the *process of problem-solving*, so that a problem involves bridging from the current state to a goal state.

POE has been shown capable of modelling business processes (for instance, [17]). As a result of that modelling, effective business process changes have been suggested based on the properties that POE processes are expected to have. One could argue, however, that this success is to some extent fortuitous, requiring business processes that are susceptible to modelling in POE and it is not obvious that all business processes can be seen in this way. So, to be able to speak about general business processes in a POE setting, we must be able to encode them as problem solving processes.

Business processes are defined by van der Aalst and Stahl [2] as:

> "A business process consists of a set of activities that is performed in an organisational and technical environment. These activities are coordinated to jointly realise a business goal."

and so can be associated with a) an organisational and technical environment, and b) a business goal. To show that a business process is a problem-solving process in the POE sense, we will construct a problem that the business process solves from this environment and goal (the need that the solution satisfies). For the business process BP, call the environment ENV_{BP} and the goal $GOAL_{BP}$. Then form the problem:

$$Problem_{BP} : ENV_{BP}, S \vdash GOAL_{BP}$$

We see that:

1. $Problem_{BP}$ is a POE problem;
2. BP solves $Problem_{BP}$ in that when, as S, it is installed in ENV_{BP} it establishes the goal $GOAL_{BP}$.

Hence, to each business problem we can associate a POE problem that it solves. In essence, from an engineering perspective, business processes are designed and implemented in response to organisational problems. The above construction shows that business processes are problem solving processes, and we have given a representation of them in POE.

IV. Stochastic Semantics

In this section, we model a single instance of the PPP as a Markov chain, where we map PPP elements to states (numbers in Figure 2). The PPP process can be seen as a transition system. While the choices made by the validators will not be known until the execution, we can model the behaviour with Markov chains. They are a useful means of modelling processes, and can be seen as transition systems, in which state transitions are decided probabilistically [3]. This extends the PPP model by a set of probabilities, which could

 ISBN: 978-1-61208-395-7

be interpreted as characteristics of the agents taking part in the problem solving.

By mapping the POE Process Pattern to a Markov chain, we identified nine states (state p in Figure 2 and Figure 3 – numbered from (0) to (8)). States (0) and (7) represent the process in the initial and successfully solved state (they correspond to the black and white circles in Figure 2, respectively). Problem exploration (2) and solution exploration (5) are followed by their respective validation check points: problem validation (3) and solution validation (6). Additionally, we introduced states (1) and (4) to help model resource expenditure during explorations, i.e., transitions *Pexp* and *Sexp* increment the cost. When the process runs out of resources, or when the solution validator declares it unsolvable, then it remains unsolved – state (8). All these states, and their transition probabilities are shown in Figure 3.

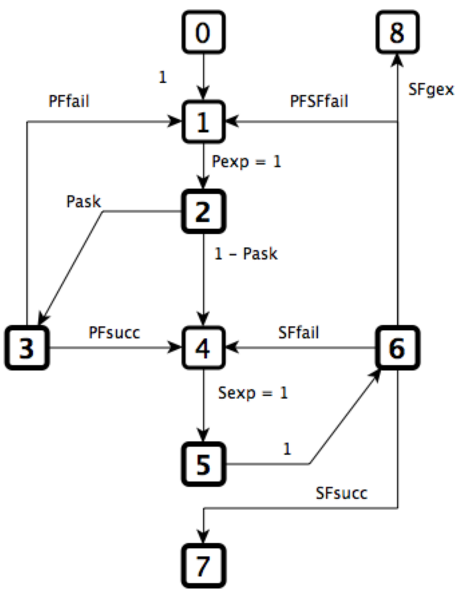

Figure 3. PPP as Markov chain.

With our model of PPP encoded as a Markov chain, we can use the PRISM model checking tool [16], which allows for process simulation and analysis. It is useful, because with this we can model the overall probability of successful problem solving under budgetary constraints, i.e., we will be able to answer questions such as, for example, this: *with a given budget of X, what is the probability that the problem solving will be successful?*

A. Team characteristics

It is reasonable to characterise the problem and solution exploration expertise of agents that perform them. A good problem exploration team is likely to reach a validatable problem understanding more quickly and within budget. Conversely, a poor team can spend all allocated resources without producing a validatable problem understanding – leaving a problem unsolved. In the Markov model, poor problem exploration team expertise translates to high *PFfail* ($= 1 - PFsucc$) and *vice versa*. Similarly, poor solution exploration team expertise translates to low *SFsucc*. Other probabilities in the model relate to: the teams' propensity to ask for validation (*Pask*),

```
1  dtmc
2          const double PFfail = 1 - PFsucc;
3          const double PFsucc;
4          const double Pask = 1.0;
5          const double PFSFfail = 0.1;
6          const double SFfail = 0.1;
7          const double SFsucc = 0.7;
8          const double SFgex = 0.1;
9          const int BUDGET;
10         const int Unsolved = 8;
11         formula Res = (cost > BUDGET);
12 module PPP
13         p: [0..8];
14         [START] p=0 & !Res -> 1: (p'=1);
15         [PEXP] p=1 & !Res -> 1: (p'=2);
16         [PASK] p=2 & !Res -> Pask: (p'=3) + (1-Pask): (p'=4);
17         [PFVAL] p=3 & !Res -> PFfail: (p'=1) + PFsucc: (p'=4);
18         [SEXP] p=4 & !Res -> 1: (p'=5);
19         [SASK] p=5 & !Res -> 1: (p'=6);
20         [SFVAL] p=6 & !Res -> PFSFfail:(p'=1)
21                 + SFfail:(p'=4)
22                 + SFsucc:(p'=7) + SFgex:(p'=8);
23         [UNSOLVED] Res -> 1: (p' = Unsolved);
24         [SOLVED] (p=7) -> 1: (p'=7);
25         [GEX] (p=8) -> 1: (p'=8);
26 endmodule
27 module Cost
28         cost : [0..BUDGET];
29         [PEXP] !(cost >= BUDGET) -> 1: (cost'=cost + 1);
30         [SEXP] !(cost >= BUDGET) -> 1: (cost'=cost + 1);
31 endmodule
```

Figure 4. PPP as a discrete-time Markov chain `dtmc` model in PRISM, which comprises two modules: `module PPP` and `module Cost`. For illustration of this experiment, we assigned arbitrary probability values for *PFSFfail*, *SFfail*, *SFsucc* and *SFgex*, but these would be expected to be variable.

probability of failed problem validation (*PFfail*), probability of failed solution validation due to an invalid problem (*PFSFfail*), probability of failed solution validation (*SFfail*), and finally, probability of a catastrophic solution validation leading to a 'global' exception (*SFgex*). We should note that, as for any Markov chain, the sum of transition probabilities from any given state has to be equal to 1, e.g., for state (6), we have $SFsucc + SFfail + PSSFfail + SFgex = 1$.

In Figure 4, the Markov chain is based on a set of probabilities guarding the transitions (lines from 2 to 8 in Figure 3), and it is composed of two modules: `PPP` and `Cost`. The former represents the states and transitions between them. For example, in line 16 we have an action `[PASK]` from state `p=2` to states `p=3` or `p=4`, guarded by their respective probabilities *Pask* and $1 - Pask$. Module `Cost` declares two actions `[PEXP]` and `[SEXP]`, each of which increments variable `cost` by an arbitrary value of 1. Formula `Res` (line 11) is a guarding condition for many transitions of the model, which monitors if the overall cost has reached the budgetary constraints (integer variable `BUDGET`).

B. Problem solving resource usage

Assigning a constant cost to each exploration action in discrete-time Markov chain (DTMC) allows us to consider the overall cost of problem solving as probabilities on the arcs change. Each team expends resources, and when a process runs out of resources, it declares the problem as unsolved.

Figure 5 shows the results from running experiments on our PPP model in PRISM, with discrete `BUDGET` (in this

 ISBN: 978-1-61208-395-7

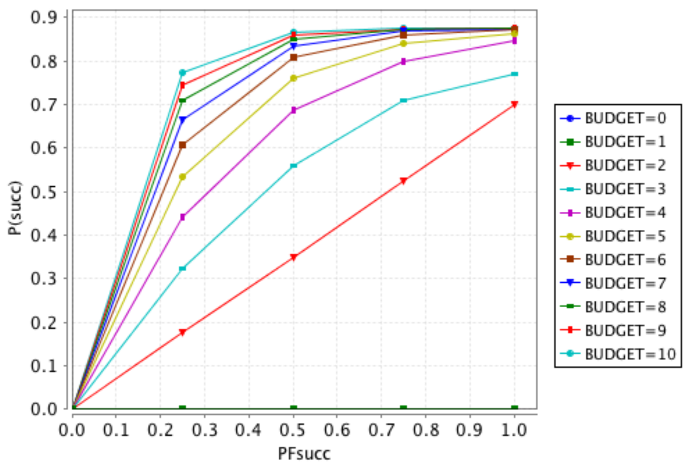

Figure 5. Probability of successful problem solving depending on *PFsucc* (curves plotted for different values of the available `BUDGET`).

case, values from 0 to 10 are representative enough, curves for values closer to 10 will be similar, i.e., the more budget available, the higher the probability of success), where we plot the curves representing the probability of successful solving for different values of *PFsucc*. The process never completes when *BUDGET* < 2 (the lines for *BUDGET* equal to 0 and 1 lie on the X-axis), because the PPP model needs at least 2 units of resource to complete both explorations. Depending on the available budget, we can state that with more available budget the probability of success increases. We can see that the better the team, the higher the overall probability of success.

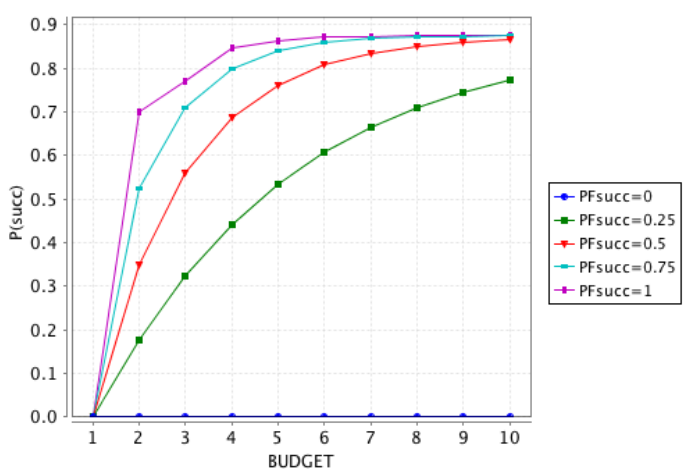

Figure 6. Probability of successful problem solving depending on the available `BUDGET` (curves plotted for different values of probability *PFsucc*).

In Figure 6, we plotted curves for probability of successful solving, and we observe, that this probability is increasing for increasing values of *BUDGET*. Since we only consider discrete values of *BUDGET*, from the plotted points we see, that probability of solving is 0, when *BUDGET* < 2. These results confirm that the probability of success depends on *PFsucc* – quantitative characteristic of our problem solving team, and in this case, the better the team (and the more *BUDGET* available), the higher the likelihood of solving the overall process.

V. CASE STUDY

Nkwocha, Hall and Rapanotti model part of the defect tracking process of mortgage calculation software [17]. Essentially, software defects lead to incorrect mortgage calculations. There are two remedies: the first is tactical – incorrect data values are corrected and the calculation retried; the second is strategic – the software is debugged and the session rerun.

Tactical solutions are less expensive than strategic solutions and so are preferred by the solution provider. However, a tactical solution does not always solve the problem and, when it fails, the customer has to request the strategic solution leading to customer dissatisfaction.

In this section, we model the trade-off between the solution provider's and the customer's positions.

The process is complex (see [17]) so, for reasons of brevity, we have modelled before and after adding the problem validation step as shown in Figure 7, in which a) shows the process without and b) with problem validation. Analysis of the problem leads to the observation that problem validation was not being used to check whether the defect required a tactical or a strategic solution: the problem team would assume they understood the problem, setting the solution team to solve it (Figure 7a). Solution validation would reveal, late, that the problem understanding was deficient, and cause the problem to be re-explored. The problem that POE was used to solve was that the trade-off was not successful, too often, a tactical remedy was not effective and the defect was logged again, causing a strategic remedy. This led to an expensive process including some dissatisfaction on the part of the client. The remedy, as suggested by POE, was to insert a problem validation step into the process so that the client could confirm the problem understanding before moving onto solution exploration. This situation is presented in Figure 7b. For simplicity, and rather than build two DTMC models, we can use the probabilities associated with state transitions outgoing from state (2) in Figure 3 to omit problem validation ($Pask = 0$) and include it ($Pask = 1$).

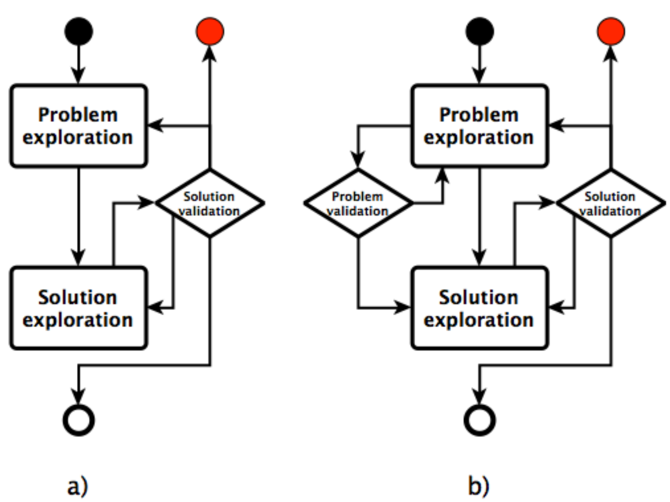

Figure 7. Process a) without asking for problem validation ($Pask = 0$); b) with problem validation ($Pask = 1$).

 ISBN: 978-1-61208-395-7

We suppose that the tendency of the team is to default to the tactical solution as it is least resource intensive, only changing to the strategic solution when indicated, perhaps because there are pressures on the problem exploration team to get an answer. They can ignore the problem validation, and go for the tactical solution every time ($Pask = 0$). Indications can be made by the problem validation (PV) requiring the strategic solution, or failure to gain solution validation (SV), for example, due to a missed PV required strategic.

We set up a simple PRISM-based experiment based on the DTMC shown in Figure 4. We varied $Pask$ between 0 and 1, and measured the probability of a successful solution being found ($P(succ)$) for a given budget for each datum. The results are shown in Figure 8 and described below.

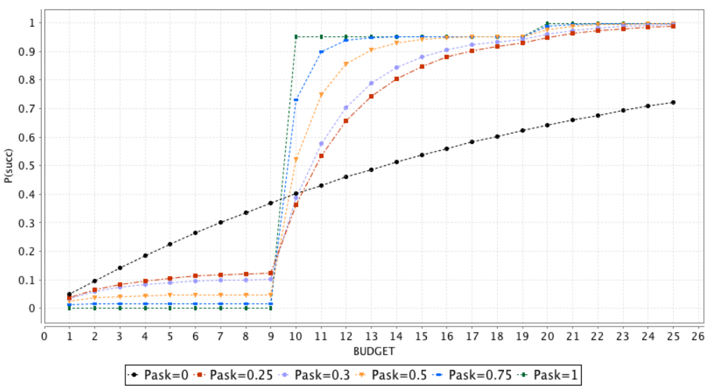

Figure 8. Results of PRISM-based experiment: the relationship between the available $BUDGET$ and the probability of successful solving $P(succ)$ for various values of $Pask$.

For *the before* case of wholly tactical solutions (modelled by $Pask = 0$, the black line in Figure 8), the reader will note that $P(succ)$ increases slowly with available $BUDGET$. For example, when the process has 10 units of $BUDGET$ to spend, $P(succ) = 0.4$. For *the after* case of problem analysis with customer validation, (modelled by $Pask = 1$, the green line in Figure 8) there is a sharp rise from $P(succ) = 0$, i.e., a solution is never found, to $P(succ) = 0.95$, a solution is almost always found. If the available budget is less than the cost of strategic solving (here, we supposed an arbitrary 10 units for strategic and 1 unit for tactical), it is better not to ask for PV, i.e., if $Pask > 0$, then PV could suggest expensive (over budget) solution, which would reduce the probability of successful solving.

VI. Related Work

With the abundance of business process modelling techniques, many of them are centred around capturing and visualisation of process structure, and only a limited number of techniques allow for quantitative analysis and structured process improvement [19]. While the business process modelling domain has become a ubiquitous part of the modern business enterprise, and many organisations view their operations in terms of processes [13], many approaches suffer from the common issues related to choice of languages, standardisation, and interoperability. Our approach of mapping POE processes to stochastic models goes beyond a visual process representation, because it makes it possible to annotate states and arcs with quantitative characteristics, which could then be modelled, analysed and simulated.

A good example of an approach to business process modelling that proved to be stable and relevant are Petri Nets [2]. They combine both visual and formal aspects, and can be used for both qualitative and quantitative modelling of processes. Petri Nets can also be annotated with resource use. Our approach to modelling of POE processes is based on the POE Process Patterns expressed as Markov chains, which, unlike classic Petri Nets, allow for analysis of probability of success or failure.

Wynn *et al.* present a holistic approach to managing the cost of business operations by making an explicit link between cost and processes in all phases of BPM lifecycle. Their approach is bottom-up and it is focused on real-time process-based cost information using process mining techniques, and it requires organisations to maintain accurate cost data and also to keep track of process behaviour in the form of event logs [20]. This can be useful for organisations to make cost informed and operational decisions after processed have been deployed, enacted and monitored. Our approach is top-down and focuses on a cost metric in the design phase of BPM lifecycle, where we also establish a link between team characteristics and the likelihood of successful problem solving. By annotating transitions between states in Markov chains with cost metrics, we benefit from the techniques and analytical power offered by stochastic modelling. This can be useful to process analysts and designers who wish to model optimal processes before they decide to operationalise them in their organisations.

As shown by Hillston, Markov models can be expressed as higher level constructs, and this was achieved with Performance Evaluation Process Algebra (PEPA), which is an example of a high-level description language for Markov processes, whose purpose is to model systems behaviour and evaluate response times during execution [10], [11]. PEPA models describe components, and interactions between processes built out of such components, by using a relatively small number of operators/combinators. Each action executed in the model is annotated with a parameter that specifies task duration. In our case, we do not consider task duration, but we annotate some of the actions (transitions between states) with a cost parameter, which in our view is more flexible, because it makes our models more generic, i.e., depending on types of properties, we could further enhance our models (by annotating) with other meta-data about processes, such as the risk or cycle time.

VII. Discussion

We have characterised business processes as problem solving activities, and used this result to model them as stochastic processes using the POE Process Pattern. This has allowed us to develop a model, in which we measure the comparative probabilities of success under simple process transformations: *viz.*, with and without problem validation. We have applied our techniques to a case study from [17], which provides supporting evidence that the transformation described there led to more cost effective processes. However, we have also identified that there is a point before which the trade-offs are not economic. This is a new result and one that we will further investigate.

We recognise that Markov models require statistical observations to establish the probabilities between the transition states, and one way possible way of finding out these values could be done perhaps with process mining [1]. In this research, however, we consider such probabilities as parameters only, and we do not yet investigate their relation to processes in a real world case.

Based on the stochastic semantics presented here, future research will explore characteristics of more complex models, i.e., we will investigate arbitrarily complex processes created by composing POE processes in sequence, parallel and fractally [7]. With an increasing complexity of real-world processes, we are aware that problems often need to be decomposed into sub-tasks/problems. In this paper, we only focused on a single POE Process Pattern. While this may seem simplistic, we acknowledge that further work is in progress, based on the work by Hall and Rapanotti [7], which allows building more complex processes by composing processes together.

While we initially focus on a single POE Process Pattern, our research will further contribute to the POE Process Algebra (PPA) [15], where we introduced operators that allow for building of arbitrarily complex POE processes. PEPA is an example of a stochastic process algebra that offers a framework, which is suitable for capturing both qualitative and quantitative aspects of a system [11]. It is also a high-level language that hides the details of underlying processes, and makes it possible to operate on a more convenient level of abstraction. According to Hillston, process algebras have a number of characteristics that make them compelling: 1) they allow for building large systems (compositionality); 2) they are adequate for qualitative analysis; 3) they have a wider acceptance outside academia. For these reasons, we believe that combining a process algebraic approach with stochastic process modelling will enhance qualitative and quantitative analysis of business problem solving with POE, where the focus is inherently on activities performed by agents, and the exchange of information between them in order to transform a POE problem $E\times, S\times \vdash N\times$ into $E\checkmark, S\checkmark \vdash N\checkmark$.

ACKNOWLEDGMENT

We would like to thank Lucia Rapanotti and Michel Wermelinger from The Open University for their help with earlier versions of this paper.

REFERENCES

[1] W. van der Aalst, Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer, 2011.

[2] W. van der Aalst and C. Stahl, Modeling Business Processes: A Petri Net-Oriented Approach. The MIT Press, 2011.

[3] C. Baier and J.-P. Katoen, Principles of Model Checking. MIT Press, 2008.

[4] T. Davenport, Process Innovation: Reengineering Work Through Information Technology. Ernst & Young, 1993.

[5] M. Dumas, W. M. van der Aalst, and A. H. ter Hofstede, Process-aware information systems: bridging people and software through process technology. John Wiley & Sons, 2005.

[6] M. Dumas, M. La Rosa, J. Mendling, and H. A. Reijers, Fundamentals of Business Process Management. Springer, 2013.

[7] J. G. Hall and L. Rapanotti, "Assurance-driven design in Problem Oriented Engineering," International Journal On Advances in Systems and Measurements, vol. 2, no. 1, 2009, pp. 119–130.

[8] J. G. Hall and L. Rapanotti, "A General Theory of Engineering: Thinking Bigger than Software," Department of Computing and Communications, The Open University, Technical Report 2015/01, January 2015. [Online]. Available: http://computing-reports.open.ac.uk/2015/TR2015-01.pdf

[9] J. G. Hall, L. Rapanotti, and M. Jackson, "Problem oriented software engineering: A design-theoretic framework for software engineering," in Proceedings of 5th IEEE International Conference on Software Engineering and Formal Methods. IEEE Computer Society Press, 2007, pp. 15–24, doi:10.1109/SEFM.2007.29.

[10] J. Hillston, A Compositional Approach to Performance Modelling. Cambridge University Press, 1996.

[11] J. Hillston, "Process Algebras for Quantitative Analysis," in Proceedings of the 20th Annual IEEE Symposium on Logic in Computer Science (LICS' 05). Chicago: IEEE Computer Society Press, Jun. 2005, pp. 239–248.

[12] M. Hobday, G. Dosi, and L. Marengo, "Problem solving behaviours, organizational forms and the complexity of tasks," in The SMS Blackwell Handbook of Organizational Capabilities, C. Helfat, Ed. Blackwell, 2003, pp. 167–192.

[13] A. ter Hofstede and N. Russell, Modern Business Process Automation. Springer, 2010, ch. The Language: Rationale and Fundamentals.

[14] D. H. Jonassen, "Towards a design theory of problem solving," Educational technology research and development, vol. 48, no. 4, 2000, pp. 63–85.

[15] D. W. Kaminski, J. G. Hall, and L. Rapanotti, "Towards Executable Business Processes with the Problem Oriented Engineering Process Algebra," Software Engineering Advances, 2011. ICSEA 2011. The Sixth International Conference on, 2011, pp. 239–244.

[16] M. Kwiatkowska, G. Norman, and D. Parker, "PRISM 4.0: Verification of probabilistic real-time systems," in Proc. 23rd International Conference on Computer Aided Verification (CAV'11), ser. LNCS, G. Gopalakrishnan and S. Qadeer, Eds., vol. 6806. Springer, 2011, pp. 585–591.

[17] A. Nkwocha, J. G. Hall, and L. Rapanotti, "Design rationale capture in the globalised enterprise: An industrial study," in Proceedings of Fifth International Conference on Software Engineering Advances (ICSEA 2010). IEEE, 2010, electronic proceedings.

[18] G. Rogers, The Nature of Engineering. The Macmillan Press Ltd., 1983.

[19] K. Vergidis, A. Tiwari, and B. Majeed, "Business process analysis and optimization: beyond reengineering," Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, vol. 38, no. 1, 2008, pp. 69–82.

[20] M. T. Wynn, J. De Weerdt, A. H. ter Hofstede, W. M. van der Aalst, H. A. Reijers, M. J. Adams, C. Ouyang, M. Rosemann, and W. Z. Low, "Cost-Aware Business Process Management: A Research Agenda," 24th Australasian Conference on Information Systems, Dec 2013, Melbourne, 2013.