Quantitative assessment of workflow performance through PH reduction

Salvatore Distefano
University of Messina, Italy
sdistefano@unime.it
Kazan Federal University, Russia
sdistefano@kpfu.ru

Marco Scarpa
University of Messina, Italy
Department of Engineering
Contrada di Dio, S. Agata, 98166 Messina, Italy.
mscarpa@unime.it

ABSTRACT

Workflows are logical abstraction of processes widely adopted in several contexts. Design and operation of workflows are critical stages where issues not manifested by the single block arise from compositions. To deal with such issues, proper techniques and tools should be implemented as support for workflow designers and operators. This paper proposes a solution for the evaluation of the workflow performance starting from the components' ones. Based on the stochastic characterization of the workflow tasks, phase type distributions and stochastic workflow reduction rules, our approach allows to overcome the limits of existing solutions, considering general response time distributions while providing parametric analysis on customer usage profiles and design alternatives. To demonstrate the effectiveness of the proposed solution an example from literature is evaluated.

Keywords

Workflow, non-Markovian behaviors, usage profile, design alternatives, phase type, performance.

1. INTRODUCTION

A workflow is usually composed of tasks or activities orchestrated through connectors specifying the process logic. The process design is therefore usually split in two phases: i) workflow logic specification and ii) task selection. Once defined, the *workflow management system* (WfMS) is in charge of setting-up, executing and monitoring the workflow based and acting on its components [22].

For a workflow development and operation both functional and non-functional requirements and properties are of strategic importance. This is particularly challenging because of the properties of a system usually emerge from those of its components, often providing functionalities not implemented by any component itself and manifesting behaviors that are not just the summation of components' ones. From a functional perspective this is the *"added value"* but, from a non-functional perspective, it introduces uncertainty and complexity to deal with. To investigate on workflow non functional properties is therefore a quite challenging goal. This is highlighted by the scientific literature, where different solutions are available. Related work mainly focuses on mathematical models such as Queueing Networks [7], Petri Nets [11], simulation [24], UML-based [9, 18], and graph-based [5] ones.

Focusing on performance, a solution should properly take into account the behavior of the system in time, consequently characterizing tasks in statistical-stochastic terms through adequate service time distributions (in general non-Markovian). To support the enforcement of service level agreement policies, it should also be able to deal with different classes of end users-customers (multi-class). Furthermore, different design alternatives have to be taken into account to support the design phase.

In this paper we propose a methodology to support a designer in workflow design and operation. The proposed framework allows to take into account different design alternatives and customer classes, providing a proper model able to deal with general distributions expressed in form of phase types. To the best of our knowledge, this is the first attempt in providing a framework for parametric workflow design. Other works mainly address these issues separately. For example, in [8], different customer classes have been considered in a service oriented computing context. On the other hand, in [13] a technique for dealing with different design alternatives in software families and software product lines is proposed. None of them deal with general, non-Markovian, distributions, mainly providing solutions in the Markovian case. However, starting from these approaches, here we propose to relax this assumption while merging their goals. We base our solution on the lightweight stochastic workflow reduction approach [5], characterizing the service time through distributions dealt with by the phase type expansion, then reduced through random variable algebra and related properties. This way a parametric phase type is obtained for the overall process, which could provide insights on it by posing and solving specific optimization problems targeting optimal design, deployment and operation.

The paper is organized as follows. In Section 2 the rationale and the related work are discussed. Then the proposed technique is presented and detailed in Section 3. Its effectiveness is demonstrated by a case study reported in Section 4 while conclusions are in Section 5.

2. PROBLEM STATEMENT

VALUETOOLS 2016, October 25-28, Taormina, Italy

DOI 10.4108/eai.25-10-2016.2266615

2.1 Rationale

To characterize the problem in formal terms, we start from an abstract workflow considered as a set of activities, interconnected through directed arcs.

DEFINITION 1. *A generic workflow W is a bipartite, connected, directed, rooted graph represented by the tuple*

$$W = \{\mathbf{T}_W, \mathbf{C}_W, \mathbf{E}_W\}$$

where:

- $\mathbf{T}_W \subseteq \mathcal{T}$ *is a non-empty set of tasks, and* $\mathcal{T}$ *is the task space set.*
- $\mathbf{C}_W \subseteq \mathcal{C}$ *is a set of workflow connectors, and* $\mathcal{C}$ *is the connector space set. At least two distinguished elements belong to* $\mathbf{C}_W$*,* C_s *and* C_e*. They represent the* W *initial and final connectors, respectively, corresponding to the graph root and sink.*
- $\mathbf{E}_W \subseteq (\mathbf{C}_W \times \mathbf{C}_W) \cup (\mathbf{C}_W \times \mathbf{T}_W) \cup (\mathbf{T}_W \times \mathbf{T}_W) \cup (\mathbf{T}_W \times \mathbf{C}_W) \equiv \mathcal{E}$ *is a set of directed arcs connecting workflow elements, and* $\mathcal{E}$ *is the arc space set.*

Task nodes have *indegree* 1 and *outdegree* 1. On the other hand, workflow connector nodes may have either 0, 1 or greater indegree and outdegree. Indeed, the root C_s is the only node with indegree 0 and outdegree 1 while the sink C_e has indegree 1 and outdegree 0. After the above definition, we have to characterize workflow elements in terms of performance. To this purpose, considering a workflow W, we define a task T as follows:

DEFINITION 2. *A generic task* $T \in \mathbf{T}_W$ *is represented by its service time distribution*

$$T = \{F_T\}$$

where:

- $F_T : \mathbb{R}^+ \rightarrow [0,1] \subset \mathbb{R}$ *is the cumulative distribution function (CDF) of the* T *service time.*

Workflow connectors are logic elements that can be considered *instantaneous*. Thus, they can be formally represented as follows.

DEFINITION 3. *A generic workflow connector* $C \in \mathbf{C}_W$ *can be represented by the tuple:*

$$C = \{Id, \mathbf{In}, \mathbf{Out}, Class, Op, \mathbf{\Pi}\}$$

where:

- $Id \in \mathbb{N}^+$ *is the connector unique identifier;*
- $\mathbf{In} \subseteq \mathbf{T}_W \cup \mathbf{C}_W \cup \emptyset$ *is the set of input tasks and/or connectors such that* $\forall In_i \in \mathbf{In} \Rightarrow (In_i, C) \in \mathbf{E}_W$*;*
- $\mathbf{Out} \subseteq \mathbf{T}_W \cup \mathbf{C}_W \cup \emptyset$ *is the set of output tasks and/or connectors such that* $\forall Out_o \in \mathbf{Out} \Rightarrow (C, Out_o) \in \mathbf{E}_W$*;*
- $Class \in \{Start|End|Split|Join\}$ *identifies the specific connector type among the 4 here defined;*
- $Op \in \{AND|XOR|\emptyset\}$ *specifies the operation applied to connectors;*
- $\mathbf{\Pi}$ *is the* weight vector*, which generic element* $\mathbf{\Pi}[h]$ *characterizes the h-th input or output in case of multiple inputs or outputs.*

Given a connector, its **In** and **Out** sets are such that $\mathbf{In} \cap \mathbf{Out} = \emptyset$. A connector of type *Start* has no input arcs ($\mathbf{In} = \emptyset$) and only one output arc, while connectors in the class *End* have only one input arc and no output arcs ($\mathbf{Out} = \emptyset$). We denote such kind of connectors with C_s and C_e, respectively, and they are characterized by $OP = \emptyset$. *Split* and *Join* connectors are associated with logical operations: AND represents *concurrent* executions of all the multiple branches while XOR represents the *exclusive choice* of just a branch among the available ones. A connector in class *Split* is characterized by one incoming arc and multiple outgoing arcs, while a connector in class *Join* has multiple incoming arcs and just one outgoing arc. The weight vector $\mathbf{\Pi}$ is related to the set **In** for *Join* connectors and to the set **Out** for *Split* connectors. In the case of conditional patterns (XOR split and join), $\mathbf{\Pi}$ of the XOR split corresponds to a probabilistic choice and its elements must sum to 1. For parallel patterns (AND split), $\mathbf{\Pi}$ of the AND split is a vector of 1 $\mathbf{\Pi} = \{1, .., 1\}$. In the case of loops ($XOR$ split and join), the XOR split $\mathbf{\Pi}$ is the same of conditional choices, while the XOR join includes the information of the matching split in $\mathbf{\Pi}[0] = XORS.Id$. Furthermore, if the loop has fixed number of iterations $r \in \mathbb{N}^+$, this parameter is specified in the XOR join $\mathbf{\Pi}[1] = r$, while all the other elements of the XOR join $\mathbf{\Pi}$ are 0. In all the other cases the join $\mathbf{\Pi} = \{0, .., 0\}$.

2.2 Parametric workflow

To ensure a proper quality of service able to satisfy SLA constraints, it is often required to discriminate among customers. In terms of the overall process, this implies an in depth investigation on its workflow aiming at characterizing the *usage profile* of a class of customers knowing the corresponding workload. This means to identify the paths, the alternatives selected by the customer of a class, by mainly quantifying and stochastically characterizing conditional choices and branches. This way the usage profile of a customer class can be specified by a vector composed of the probabilities associated with all the conditional patterns in the workflow.

It could also be of interest to include possible *design options or alternatives*. Typical examples are software families and software product lines, where specific variability modeling solutions are used to perform parametric evaluations. Such alternatives can be included on a workflow as a special kind of conditional statements, which branches represent the design choices available thus turning the workflow into a parametric one. Also in this case the design parameters can be represented by a vector composed of the parameter associated with the design alternatives in the workflow.

An interesting feature of the proposed technique is the possibility of performing parametric analysis, taking into account both the usage profile of customers and different, alternative options in the workflow design. These options, identifying alternative workflow solutions for the composite process or parts of it, can be represented through extra exclusive choices and thus included in the main workflow containing the process choices characterizing the usage profile thus obtaining a single, parametric model. The multiple exclusive choice $XOR-Split/XOR-Join$ pattern is there-

fore the key pattern for expressing both usage and design alternative parameters into the workflow.

To this end, a parameter vector can be associated with a $XOR-Split/XOR-Join$ connector of the workflow, characterized by the $\mathbf{\Pi}_i$ weight vector of the ith $XOR-Split$. This way, a parameter set $\mathbf{V} = \{\mathbf{\Pi}_i\}$ is specified. Such parameters can be categorized into two groups, the usage profile and the design alternative ones, thus identifying a partition of $\mathbf{V}$ into two subsets $\mathbf{U}$ and $\mathbf{Z}$, respectively, such that $\mathbf{V} = \mathbf{U} \cup \mathbf{Z}$. In the case of a design alternative parameter the weight vector associated with the $XOR-Split$ of the corresponding exclusive choice is a binary stochastic vector where the element related to the selected branch (e.g. the jth one with $1 \leq j \leq m$) is 1 and the others are 0, i.e. $\mathbf{\Pi} = \{\underbrace{0,..,0}_{j-1}, 1, \underbrace{0,..,0}_{m-j}\}$.

2.3 Related work

Several methods have been proposed in the literature to evaluate workflow performance, including simulation [24], process calculi [18], queuing networks [7], Petri nets [11] and graphs (DAG, workflow graphs, control flow graph, process flow graphs) [5]. However, the most widely adopted formalism for workflow performance evaluation is queuing networks. In [7], workflow performance requirements and specifications are expressed by P-WSDL (Performance-enabled WSDL) and then QoS parameters such as average response time and utilization are evaluated by a layered queueing network (LQN) model generated by the proposed tool. Similarly, in [15], the performance of a Web service workflow are evaluated through bound analysis, while [14] performed bottleneck analysis of a personnel application modeled as a LQN with three different customer classes, thus allowing to take into account different usage profiles.

Even if less investigated than others, there are some interesting work on design alternatives available in literature. In [20], a delta-oriented approach is adopted to deal with variants in a software product line model. This technique is also applied to the performance evaluation of workflows expressed as performance annotated activity diagrams in [13]. However, most of the above referred techniques, and in particular those based on state space models, have the problem of state space explosion, thus limiting the applicability of related techniques to low complexity problem. To relax this restriction, workflow reduction techniques, sometimes combined with simulation, have been proposed in [5, 10] to estimate WS workflow QoS parameters.

The main problem to deal with in the above mentioned work is related to complexity and state space explosion. Furthermore, one of the goal of this work is to take into account both usage profiles and design alternatives altogether, which introduce further complexity on the overall model. To this purpose a solution should base on workflow reduction, which anyway does not allow to represent workflow performance with stochastic behaviors and related quantities such as cumulative distribution functions. The main goal of this paper is therefore to relax these assumptions, allowing, on the one hand, to deal with complex problems including usage profiles and design alternatives through workflow reduction while, on the other hand, taking into account the stochastic behavior of a system not limiting to Markovian/exponential distributions.

3. WORKFLOW REDUCTION AND EVALUATION

Since the model we are dealing with is in general non-Markovian, we approach the problem by using phase type distributions (PHs); we assume that the CDFs in $\mathbf{F}$ are approximated by the corresponding phase types through specific fitting algorithms. When PHs are obtained, we apply a reduction step in order to obtain a PH distribution representing the whole process in a parametric form.

After introducing symbols and definitions used in the rest of the paper, we give the algebraic expressions describing all the possible patterns defined; they are used to perform the reduction step.

3.1 PH expansion

Let us consider a continuous time Markov chain $\mathcal{X}$ with $\nu+1$ states, of which states $1, \cdots, \nu$ are transient and state $\nu+1$ is *absorbing*. Let us suppose $\boldsymbol{\pi}(0) = [\alpha_1, \cdots, \alpha_\nu, \alpha_{\nu+1}]$ is the Markov chain initial probability distribution. Let ξ denote the random variable that gives the time until absorption of $\mathcal{X}$. Then we say, according to the definition given in [17], that ξ has a *phase-type distribution*. More formally:

Definition 1. *A continuous PH distribution (*CPH*) [17] is defined as the CDF of the time till absorption of a continuous time Markov chain (CTMC) with ν transient states and one absorbing state $(\nu+1)$, given an initial probability vector.*

More specifically, we say that ξ is distributed according to a phase type distribution with *representation* $(\boldsymbol{\alpha}, \boldsymbol{G})$ and order ν, where $\boldsymbol{\alpha}$ and $\boldsymbol{G}$ are defined as follows:

- let $\boldsymbol{\pi}(0) = [\,\boldsymbol{\alpha},\ \alpha_{\nu+1}\,]$ be the $(\nu+1)$ initial probability (row) vector, then the following holds: $\alpha_{\nu+1} = 1 - \sum_{i=1}^{\nu} \alpha_i$.
- let $\widehat{\boldsymbol{G}}$ (of dimension $(\nu+1)\times(\nu+1)$) be the infinitesimal generator matrix of the CTMC $\mathcal{X}$. $\widehat{\boldsymbol{G}}$ can be partitioned in the following way:

$$\widehat{\boldsymbol{G}} = \left[\begin{array}{c|c} \boldsymbol{G} & \boldsymbol{B} \\ \hline \mathbf{0} & 0 \end{array}\right],$$

  where, $\boldsymbol{G}$ is a $(\nu\times\nu)$ matrix that describes the transient behavior of the CTMC and $\boldsymbol{B}$ is a $(\nu \times 1)$ column vector grouping the transition rates to the absorbing state. Since the matrix $\widehat{\mathbf{G}}$ must be stochastic, we have $\boldsymbol{B} = -\boldsymbol{G} \cdot \mathbf{e}$, where $\boldsymbol{e}$ is a $(\nu \times 1)$ column vector with all the entries equal to 1.

Definition 1 implicitly assumes that the system is absorbed (i.e. reaches state $\nu+1$) with probability 1. This can be guaranteed irrespective of the value of $\boldsymbol{\alpha}$ if state $\nu+1$ is reachable from every other state. Using this notion and the basic properties of the matrix $\boldsymbol{G} = [g_{ij}]$ (i.e. $g_{ii} < 0$, $g_{ij} \geq 0$ for $i \neq j$, and $|g_{ii}| \geq \sum_{j\neq i} g_{ij}$), it can be showed that $\boldsymbol{G}$ must be non singular [12]. This means that all the eigenvalues of $\boldsymbol{G}$ must be non positive [16].

Given a CPH with representation $(\boldsymbol{\alpha}, \boldsymbol{G})$, its CDF is $F(t) = 1 - e^{\boldsymbol{G}t}\mathbf{1}$, and its k moment is written as

$$E(\xi^k) = k!\ \boldsymbol{\alpha}(-\boldsymbol{G})^{-k}\mathbf{1}.$$

Other relevant properties involves operation among CPHs; in particular, in the following of this paper, we exploit sum-

mation, mixture, maximum and minimum, thus we briefly recall the related expressions.

Let us consider two CPHs, X_1 and X_2, with representation $(\boldsymbol{\alpha}_1, \boldsymbol{G}_1)$ and $(\boldsymbol{\alpha}_2, \boldsymbol{G}_2)$ respectively and such that the initial probabilities to the absorbing state ν_{i+1} are null; thus the following properties hold:

1. the summation $X_S = X_1 + X_2$ is a CPH with representation $(\boldsymbol{\alpha}_S, \boldsymbol{G}_S)$, where $\boldsymbol{\alpha}_S = (\boldsymbol{\alpha}_1, \mathbf{0})$ and $\boldsymbol{G}_S = \left[\begin{array}{c|c} \boldsymbol{G}_1 & \boldsymbol{B}_1\boldsymbol{\alpha}_2 \\ \hline \mathbf{0} & \boldsymbol{G}_2 \end{array}\right]$;
2. the mixture $X_T = \left\{ \begin{array}{ll} X_1 & \textit{with probability } p \\ X_2 & \textit{with probability } (1-p) \end{array}\right.$ is a CPH with representation$(\boldsymbol{\alpha}_T, \boldsymbol{G}_T)$, where $\boldsymbol{\alpha}_T = (p\boldsymbol{\alpha}_1, (1-p)\boldsymbol{\alpha}_2)$ and $\boldsymbol{G}_T = \left[\begin{array}{c|c} \boldsymbol{G}_1 & \mathbf{0} \\ \hline \mathbf{0} & \boldsymbol{G}_2 \end{array}\right]$;
3. the minimum $X_m = \min(X_1, X_2)$ is a CPH with representation $(\boldsymbol{\alpha}_m, \boldsymbol{G}_m)$, where $\boldsymbol{\alpha}_m = \boldsymbol{\alpha}_1 \otimes \boldsymbol{\alpha}_2$ and $\boldsymbol{G}_m = \boldsymbol{G}_1 \oplus \boldsymbol{G}_2$;
4. the maximum $X_M = \max(X_1, X_2)$ is a CPH with representation $(\boldsymbol{\alpha}_M, \boldsymbol{G}_M)$, where $\boldsymbol{\alpha}_M = \left[\begin{array}{c|c|c} \boldsymbol{\alpha}_1 \otimes \boldsymbol{\alpha}_2 & \mathbf{0} & \mathbf{0} \end{array}\right]$ and $\boldsymbol{G}_M = \left[\begin{array}{c|c|c} \boldsymbol{G}_1 \oplus \boldsymbol{G}_1 & \boldsymbol{B}_1 \oplus \boldsymbol{I} & \boldsymbol{I} \oplus \boldsymbol{B}_2 \\ \hline \mathbf{0} & \boldsymbol{G}_2 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \boldsymbol{G}_1 \end{array}\right]$.

In the expression above, we used the symbols $\otimes$ and $\oplus$ to denote the Kronecker multiplication and Kronecker sum respectively.

The use of CPHs is justified because most of generally distributed continuous r.v. can be approximated by a CPH distribution by using appropriate fitting algorithms [2,3,21].

3.2 Reduction rules

The evaluation performed at this step has to deal with the high complexity of the process, especially when its workflow has branches and nesting that can also represents alternative design choices as discussed above. The evaluation technique is based on well known aggregation-reduction techniques [5, 6]. This section deals with the analytic formulas expressing the service time of each workflow basic pattern in terms of those of its tasks. Both one branch and multiple branch patterns are investigated. In the latter case, for the sake of readability and clarity, we just consider two branches, $T_1 = \{F_{T_1}, C_{T_1}\}$ and $T_2 = \{F_{T_2}, C_{T_2}\}$, characterized by the corresponding service time random variable (r.v.) X_1 and X_2 with their CPH distributions $F_{T_1} = (\boldsymbol{\alpha}_1, \mathbf{G}_1)$ and $F_{T_2} = (\boldsymbol{\alpha}_2, \mathbf{G}_2)$ and costs, respectively. Anyway, complex multi-branch patterns with more than two branches can be easily decomposed into 2-branch nested patterns and thus reduced by iteratively applying the following rules.

In the following, we evaluate the aggregation reduction formulas for the main process workflow single and multi-branch patterns $W = \{\mathbf{T}_W, \mathbf{C}_W, \mathbf{E}_W\}$ in terms of the involved T_1 and T_2 task service time CDFs and related kth moments $E(X_1^k)$ and $E(X_2^k)$.

3.2.1 Sequence

Considering a workflow composed of the *sequence* $S = \{\mathbf{T}_S, \mathbf{C}_S, \mathbf{E}_S\}$ of two tasks T_1 and T_2, such that $\mathbf{T}_S = \{T_1, T_2\}$ (where T_1 and T_2 are the same above specified), $\mathbf{C}_S = \{C_s, C_e\}$ and $\mathbf{E}_S = \{(C_s, T_1), (T_1, T_2), (T_2, C_e)\}$. Since S returns control when the tasks complete following the sequence order, we have that

$$X_S = X_1 + X_2 \ . \tag{1}$$

The sequence pattern S CPH representation $(\boldsymbol{\alpha}_S, \mathbf{G}_S)$ can be thus expressed as:

$$\boldsymbol{\alpha}_S = [\boldsymbol{\alpha}_1, \mathbf{0}], \quad \mathbf{G}_S = \left[\begin{array}{cc} \mathbf{G}_1 & \boldsymbol{B}_1\boldsymbol{\alpha}_2 \\ \mathbf{0} & \mathbf{G}_2 \end{array}\right] \tag{2}$$

3.2.2 Multi branch patterns

According to the definitions of Section 2.1, three possible combinations are meaningful for split/join multibranch patterns: XOR/XOR, AND/AND, AND/XOR, while the XOR/AND configuration is semantically wrong.

XOR/XOR.

XOR/XOR patterns implement conditional exclusive choices among alternative paths each associated with the corresponding task or subworkflow. The 2-way XOR/XOR exclusive choice $XOR/XOR = \{\mathbf{T}_{XOR/XOR}, \mathbf{C}_{XOR/XOR}, \mathbf{E}_{XOR/XOR}\}$ is expressed as

$$\mathbf{T}_{XOR/XOR} = \{T_1, T_2\}, \mathbf{C}_{XOR/XOR} = \{C_s, C_{XORS}, C_{XORJ}, C_e\}$$

and

$$\begin{aligned} \mathbf{E}_{XOR/XOR} = \{&(C_s, C_{XORS}), (C_{XORS}, T_1), (C_{XORS}, T_2), \\ &(T_1, C_{XORJ}), (T_2, C_{XORJ}), (C_{XORJ}, C_e)\}. \end{aligned}$$

If the X_c r.v. identifies the selected branch and thus $X_c \in \{1, 2\}$, the service time r.v. of the XOR/XOR pattern $X_{XOR/XOR}$ can be expressed as follows:

$$X_{XOR/XOR} = \left\{ \begin{array}{l} X_1 \text{ if } X_c = 1 \\ X_2 \text{ if } X_c = 2 \end{array}\right. . \tag{3}$$

Assuming that all such random variables are statistically independent and also that the probability to choose T_1 is given by $\boldsymbol{\Pi}_{XORS}[1] = Pr\{X_c = 1\} = p_c$ while the T_2 one is $\boldsymbol{\Pi}_{XORS}[2] = Pr\{X_c = 2\} = 1 - p_c$ the CPH describing the XOR/XOR multiple branch $(\boldsymbol{\alpha}_{XOR/XOR}, \mathbf{G}_{XOR/XOR})$ is

$$\boldsymbol{\alpha}_{XOR/XOR} = [p_c\boldsymbol{\alpha}_1, (1-p_c)\boldsymbol{\alpha}_2], \boldsymbol{G}_{XOR/XOR} = \left[\begin{array}{cc} \boldsymbol{G}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{G}_2 \end{array}\right] \tag{4}$$

The XOR/XOR pattern could also have a *null* task of one its branches. In this case the CPH associate is smaller since it is defined by the transient behavior of the non null task that is executed with a given probability. Let us suppose X_1 is the r.v. associated to the non null task, thus the representation is:

$$\boldsymbol{\alpha}_{XOR/XOR} = p_c\boldsymbol{\alpha}_1, \quad \boldsymbol{G}_{XOR/XOR} = \boldsymbol{G}_1 \tag{5}$$

We point out that the non null activity modifies the structure of the resulting CPH that does not start from a transient phase with probability 1.0 because the $\nu+1$-th element of its initial probability vector is $(\alpha_{XOR/XOR})_{\nu+1} = 1-p_c \neq 0$.

A *null* task is instead semantically without sense for the other multi-branch patterns.

AND/AND.

The AND/AND pattern is the simplest concurrent pattern splitting the main flow into parallel threads and finally

synchronizing them through the join. It thus terminates only when *all* the parallel threads return control regardless the order. The 2-way AND/AND concurrent pattern $AND/AND = \{\mathbf{T}_{AND/AND}, \mathbf{C}_{AND/AND}, \mathbf{E}_{AND/AND}\}$ is expressed as $\mathbf{T}_{AND/AND} = \{T_1, T_2\}$, $\mathbf{C}_{AND/AND} = \{C_s, C_{ANDS}, C_{ANDJ}, C_e\}$ and $\mathbf{E}_{AND/AND} = \{ (C_s, C_{ANDS}), (C_{ANDS}, T_1), (C_{ANDS}, T_2), (T_1, C_{ANDJ}), (T_2, C_{ANDJ}), (C_{ANDJ}, C_e)\}$. The service time r.v. of an AND/AND concurrent pattern $X_{AND/AND}$ with 2 parallel branches T_1 and T_2 characterized by the X_1 and X_2 service time r.v. is therefore identified by the maximum among such random variables. Let $\mathbf{X}^{AND/AND}$ be the set or the *largest order statistic*

$$X_{AND/AND} = \mathbf{X}^{AND/AND}_{(2)} = \max\{X_1, X_2\} \ . \tag{6}$$

Then, the AND/AND CPH with representation

$$(\boldsymbol{\alpha}_{AND/AND}, \mathbf{G}_{AND/AND})$$

is expressed as:

$$\boldsymbol{\alpha}_{AND/AND} = \left[\ \boldsymbol{\alpha}_1 \otimes \boldsymbol{\alpha}_2, \quad \mathbf{0}, \quad \mathbf{0} \ \right] \tag{7}$$

and

$$G_{AND/AND} = \begin{bmatrix} \mathbf{G}_1 \oplus \mathbf{G}_2 & \boldsymbol{B}_1 \otimes \mathbf{I}_2 & \mathbf{I}_1 \otimes \boldsymbol{B}_2 \\ \mathbf{0} & \mathbf{G}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{G}_1 \end{bmatrix} \tag{8}$$

AND/XOR.

Another concurrent pattern is the AND/OR one, where all the threads concurrently start and the first completing closes the pattern. The 2-way AND/XOR concurrent pattern $AND/XOR = \{\mathbf{T}_{AND/XOR}, \mathbf{C}_{AND/XOR}, \mathbf{E}_{AND/XOR}\}$ is expressed as $\mathbf{T}_{AND/XOR} = \{T_1, T_2\}$, $\mathbf{C}_{AND/XOR} = \{C_s, C_{ANDS}, C_{XORJ}, C_e\}$ and $\mathbf{E}_{AND/XOR} = \{(C_s, C_{ANDS}), (C_{ANDS}, T_1), (C_{ANDS}, T_2), (T_1, C_{XORJ}), (T_2, C_{XORJ}), (C_{XORJ}, C_e)\}$. This way, if $\mathbf{X}_{AND/XOR}$ r.v. is the *first order statistic* (or smallest order statistic), i.e. the minimum between X_1 and X_2, we can describe the AND/XOR pattern service time r.v. by the following formula:

$$X_{AND/XOR} = \mathbf{X}^{AND/XOR}_{(2)} = \min\{X_1, X_2\}. \tag{9}$$

Then, the AND/XOR CPH $(\boldsymbol{\alpha}_{AND/XOR}, \mathbf{G}_{AND/XOR})$ is expressed by:

$$\alpha_{AND/XOR} = \boldsymbol{\alpha}_1 \otimes \boldsymbol{\alpha}_2, \quad G_{AND/XOR} = \mathbf{G}_1 \oplus \mathbf{G}_2 \tag{10}$$

3.2.3 Loop

According to the usual notion, we define the *Loop* pattern as a structure able to execute a number of iterations. The number of performed iterations could be either fixed or undefined and probabilistically controlled. In the following, the rules for both cases will be introduced.

Fixed number of iterations.

The loop pattern is composed of the body task T that has to be executed a r times. It is therefore represented as a sequence of the task T and the following loop condition, and can be unrolled into a sequence of task T repeated r times. Formally, the loop pattern $L = \{\mathbf{T}_L, \mathbf{C}_L, \mathbf{E}_L\}$ is composed of just one task to be iterated, and thus expressed as $\mathbf{T}_L = \{T\}$, $\mathbf{C}_L = \{C_s, C_{XORS}, C_{XORJ}, C_e\}$ and $\mathbf{E}_L = \{(C_s, C_{XORS}), (C_{XORS}, T), (C_{XORS}, C_{XORJ}), (T, C_{XORJ}), (C_{XORJ}, C_e)\}$. Thus, assuming the iteration service times X_T r.v. are independent and identical distributed, the loop CPH $(\boldsymbol{\alpha}_L, \mathbf{G}_L)$ can be iteratively described by

$$\boldsymbol{\alpha}_L = (\boldsymbol{\alpha}, \underbrace{\mathbf{0}, .., \mathbf{0}}_{r}) \tag{11}$$

and

$$G_L = \begin{bmatrix} \mathbf{G}_L^{r-1} & \boldsymbol{B}_L^{r-2}\boldsymbol{\alpha} \\ \mathbf{0} & \mathbf{G}_L^{r-1} \end{bmatrix} \tag{12}$$

where:

$$G_L^{r-1} = \begin{bmatrix} \mathbf{G}_L^{r-2} & \boldsymbol{B}^{r-3}\boldsymbol{\alpha} \\ \mathbf{0} & \mathbf{G}_L^{r-2} \end{bmatrix}$$

and

$$G_L^2 = \begin{bmatrix} \mathbf{G} & \boldsymbol{B}\boldsymbol{\alpha} \\ \mathbf{0} & \mathbf{G}_L \end{bmatrix}$$

for $r > 1$, while if $r = 1$ then $\boldsymbol{\alpha}_L = \boldsymbol{\alpha}$ and $\mathbf{G}_L = \mathbf{G}$, corresponding to the T CPH distribution attributes.

Probabilistic iterations.

In the general case the loop pattern is composed of the body task T_b followed by the loop condition characterized by the iteration probability p_l[1]. When the random choice selects a new iteration, another task T_e could be eventually executed. It is therefore represented as a sequence of the task T_b, the following loop condition and the task T_e. It can be considered as a sequence of tasks repeated a random number of times. Formally, the loop pattern $L = \{\mathbf{T}_L, \mathbf{C}_L, \mathbf{E}_L\}$ is composed of two tasks to be iterated, and thus expressed as $\mathbf{T}_L = \{T_b, T_e\}$, $\mathbf{C}_L = \{C_s, C_{XORS}, C_{XORJ}, C_e\}$ and $\mathbf{E}_L = \{(C_s, C_{XORS}), (C_{XORS}, T_b), (T_b, C_{XORJ}), (C_{XORJ}, T_e), (T_e, C_{XORS}), (C_{XORJ}, C_e)\}$ If no task has to be executed when a new iteration starts the task T_e could be considered *null*.

Let X_b and X_e the service time r.v. of T_b and T_e tasks; the loop L service time r.v. is distributed according to a CPH computed as a sequence of X_b followed by X_e with probability p_L after which it start again; the task L completes with probability $(1 - p_L)$. Thus the loop service time is expressed by a CPH with representation $(\boldsymbol{\alpha}_L, \boldsymbol{G}_L)$ where

$$\boldsymbol{\alpha}_L = [\boldsymbol{\alpha}_b, \mathbf{0}], \quad \mathbf{G}_L = \begin{bmatrix} \boldsymbol{G}_b & p_L\boldsymbol{B}_b\boldsymbol{\alpha}_e \\ \boldsymbol{B}_e\boldsymbol{\alpha}_b & \boldsymbol{G}_e \end{bmatrix} \tag{13}$$

The representation of X_L become simpler when the task T_e is null. In fact, in that case, after the probabilistic selection the task T_b starts again with probability p_L, thus $\boldsymbol{\alpha}_L$ and $\mathbf{G}_L$ are:

$$\boldsymbol{\alpha}_L = \boldsymbol{\alpha}_b, \quad \boldsymbol{G}_b + p_L\boldsymbol{B}_b\boldsymbol{\alpha}_b \tag{14}$$

3.3 Reduction algorithm and evaluation

The formulas thus obtained in eq. (1)-(14) can be applied to a complex structured workflow hierarchically combining and nesting the above patterns into an ordered rooted tree as detailed in the Algorithm 1. It assumes a workflow W is specified as an ordered *workflow tree*, WFT, where tasks appear only on leaves and connectors are internal nodes [4,23], removing *Start* and *End* connectors. This way the reduction

[1]In this work, we assume the iteration probability is not dependent on the number of the already performed iterations.

algorithm mainly consists in a left to right in-order traversal of WFT to preserve the workflow connector order, resulting in the reduced workflow phase type distribution $(\mathbf{G}_W, \boldsymbol{\alpha}_W)$ representing the W service time. Function $ReduceWF$ is invoked to apply the formulae above specified to the corresponding patterns, thus returning the corresponding CPH distribution (step 6). In the leaf case, $CWFT[]$ is an empty vector and $ReduceWF$ returns the task CPH.

Algorithm 1: $WFTSWR$

input : a tree WFT corresponding to a non-null workflow
output: reduced workflow CPH $(\mathbf{G}_W, \boldsymbol{\alpha}_W)$
local : the $CWFT[]$ vector storing partial reduction, index i initialized to 0

1 **begin**
2 **if** *WFT is not a leaf* **then**
3 **foreach** $C \in$ {the ordered set of children of WFT} **do**
4 $CWFT[i] \longleftarrow WFTSWR(C)$;
5 $i \longleftarrow i + 1$;
6 **return** $ReduceWF(WFT, CWFT[])$;

This way, the workflow service time CPH $(\mathbf{G}_W, \boldsymbol{\alpha}_W)$ is obtained as a function of its task parameters $(\mathbf{G}_i, \boldsymbol{\alpha}_i)$ and C_i $\forall i = 1, .., n$, i.e.

$$\mathbf{G}_W = f_{\mathbf{G}_W}(\mathbf{G}_1, .., \mathbf{G}_n) = f_{\mathbf{G}_W}(\mathbf{Z}, \mathbf{U})$$

$$\boldsymbol{\alpha}_W = f_{\boldsymbol{\alpha}_W}(\boldsymbol{\alpha}_1, .., \boldsymbol{\alpha}_n) = f_{\boldsymbol{\alpha}_W}(\mathbf{Z}, \mathbf{U})$$

which can be expressed in parametric form in terms of $\mathbf{U}$ and $\mathbf{Z}$ parameters.

The reduction algorithm together with the reduction rules (2)-(14) generate a block matrix representing an irreducible Markov chain with one absorbing state whose evaluation gives the CDF of the workflow response time. Due to the structure of the stochastic process a semi-symbolic analysis algorithms [19] can be used and an expression giving the completion time probability could be obtained as a function of usage and design parameters. Generally speaking the result of the evaluation is a function $F(t; \boldsymbol{u}, \boldsymbol{z})$, with $\boldsymbol{u} \in \mathbf{U}$ and $\boldsymbol{z} \in \mathbf{Z}$, and the $F(t; \boldsymbol{u}, \boldsymbol{z})$ has the shape of an expolynomial expression.

4. AN EXAMPLE

In order to explain the proposed technique and demonstrate its effectiveness, we applied it to an example taken from literature [13]. A high-level representation of the example workflow through activity diagrams is shown in Figure 1. It represents a logical workflow, composed of a *do while* loop iterating a sequence of three tasks where a conditional choice identifies the second one (Figure 1a). Therefore, in the main workflow there are two conditional choices C1 and C2, the former associated with the loop while the latter with the inner condition, which characterizes its usage profile. After performing the process design the designer has to select tasks and orchestrate the overall workflow. In this stages, starting from functional properties, but also based on non functional ones, tasks are selected and eventual possible project alternatives identified. The resulting workflow is thus reported in Figure 1b, including 2 design alternatives D1 and D2. As highlighted in this figure they are correlated, since branches associated with the same letter (**a** or **b**) correspond, thus identifying in total two possible combinations among the four available.

Par. i	1	2	3	4	5
γ_i	0.5775	0.4331	0.2887	0.2475	0.8663
β_i	2.0	2.0	2.0	2.0	2.0

Table 1: Distribution associated with the ith task ($i = 1, .., 5$) of the example workflow.

The completion time distributions associated with the five tasks composing this workflow are reported in Table 1. These are all Weibull distributions in the form $F_i(t, \gamma_i, \beta_i) = 1 - e^{-(x/\gamma_i)^{\beta_i}}$, where γ_i is the scale parameter and β_i the shape one, with $i = 1, .., 5$. These parameters have been specified according to the [13] ones, preserving the mean completion time but considering here non exponential distributions.

The distributions thus identified in Table 1 have been then represented by a 2-stage CPHs in canonical form CF-A [1] as the one showed in Figure 2a, whose representation is $(\boldsymbol{\alpha}_i, \boldsymbol{G}_i)$. This means that each of such distributions are characterized by the three parameters $\lambda^i_{12}, \lambda^i_{13}, \lambda^i_{23}$, whose numerical values are reported in Table 2.

Par. i	1	2	3	4	5
λ_{12}	3.986494	5.315632	7.974369	9.301820	2.657509
λ_{13}	0.000923	0.001230	0.001845	0.002153	0.000615
λ_{23}	3.964869	5.286798	7.931112	9.251362	2.643094

Table 2: CPH rates associated with the ith task of the workflow example.

After that, we have to specify the workflow tree corresponding to the example workflow. By applying the rules discussed in Section 3.3 the workflow tree shown in Figure 2b is obtained. In particular, since conditional constructs needs at least two branches, a *null* fictitious task is inserted in the conditional choice D2. The loop is composed by two branches: the direct branch made by a sequence of composed tasks and the reverse branch containing two tasks depending on the design alternative. We used a *null* task to model the design alternative **a** in the reverse task.

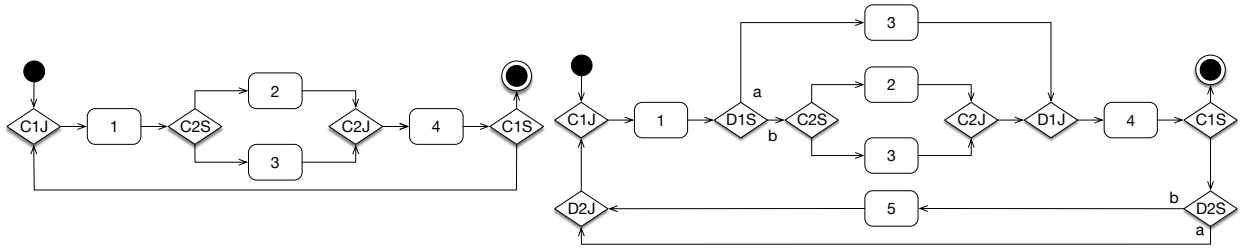
Given that we can obtain the full matrix representing the whole process by just apply the workflow reduction rules identified in Section 3.2. Starting from the leaves of the tree in Figure 2b, the matrices characterizing the CPHs of the XOR/XOR pattern C_2 and D_2 are built:

$$\boldsymbol{\alpha}_{C_2} = [p_{C_2}\boldsymbol{\alpha}_2, (1 - p_{C_2})\boldsymbol{\alpha}_3], \quad \boldsymbol{G}_{C_2} = \begin{bmatrix} \boldsymbol{G}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{G}_3 \end{bmatrix}$$

and

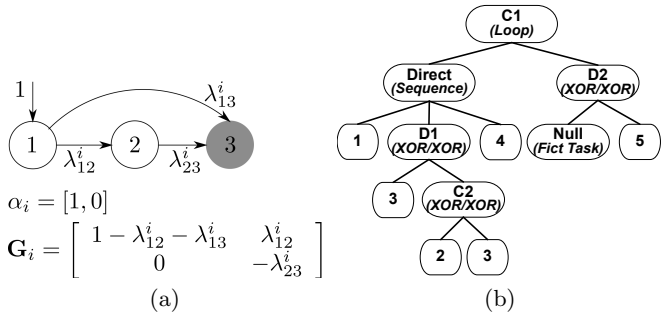
$$\boldsymbol{\alpha}_{D_2} = p_b\boldsymbol{\alpha}_5, \quad \boldsymbol{G}_{D_2} = \boldsymbol{G}_5$$

where p_{C_2} is the probability that the task 2 is selected in the branch C_2 ($p_{C_2} \in \mathbf{U}$), and $p_b \in \mathbf{Z}$ is the binary quantity used to select the design alternative **b**. We point out that when the alternative **b** is under evaluation $p_b = 1$ and correspondingly $p_a = 0$; at the opposite when the design alternative **a** is evaluated $p_a = 1$ and $p_b = 0$, thus the vector $\boldsymbol{\alpha}_{D_2}$ will be null deselecting the corresponding branch.



(a) Original (b) With design aternatives

Figure 1: The example workflow.



$\alpha_i = [1, 0]$

$\mathbf{G}_i = \begin{bmatrix} 1-\lambda_{12}^i-\lambda_{13}^i & \lambda_{12}^i \\ 0 & -\lambda_{23}^i \end{bmatrix}$

(a) (b)

Figure 2: The example CPH (a) and workflow tree (b).

Iteratively visiting the tree toward the root node, the block matrices of the global CPH are similarly obtained:

$$\boldsymbol{\alpha}_{Direct} = [\boldsymbol{\alpha}_1, \mathbf{0}, \mathbf{0}], \quad \boldsymbol{G}_{Direct} = \begin{bmatrix} \boldsymbol{G}_1 & \boldsymbol{B}_1\boldsymbol{\alpha}_{D_1} & \mathbf{0} \\ 0 & \boldsymbol{G}_{D_1} & \boldsymbol{B}_{D_1}\boldsymbol{\alpha}_4 \\ \mathbf{0} & \mathbf{0} & \boldsymbol{G}_4 \end{bmatrix}$$

$$\boldsymbol{\alpha}_{D_1} = [p_a\boldsymbol{\alpha}_3, p_b\boldsymbol{\alpha}_{C_2}], \quad \boldsymbol{G}_{D_1} = \begin{bmatrix} \boldsymbol{G}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{G}_{C_2} \end{bmatrix}$$

$$\boldsymbol{\alpha} = [\boldsymbol{\alpha}_{Direct}, \mathbf{0}],$$

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{G}_{Direct} + p_a p_{C_1}\boldsymbol{B}_{Direct}\boldsymbol{\alpha}_{Direct} & p_b p_{C_1}\boldsymbol{B}_{Direct}\boldsymbol{\alpha}_5 \\ \boldsymbol{B}_5\boldsymbol{\alpha}_{Direct} & \boldsymbol{G}_5 \end{bmatrix}$$

The parameter sets of the workflow are $\mathbf{U} = \{p_{C_1}, p_{C_2}\}$ and $\mathbf{Z} = \{p_a, p_b\}$.

User class	p_{C_1}	p_{C_2}
$U1$	0.5	0.3
$U2$	0.5	0.7
$U3$	0.2	0.3
$U4$	0.2	0.7

Table 3: User classes definition.

Given the matrices $\boldsymbol{G}$ and $\boldsymbol{\alpha}$, we evaluated the response time distribution of the considered workflow, $Fw(t)$, assuming four different classes of users characterized by the values as in Table 3. The user class U1 is the same evaluated in [13]. We evaluated the different classes with respect to the two design choices also. Other interesting measure could be derived, but we omit them for lack of space.

The resulting CDFs are depicted in Figure 3. The graph in Figure 3a shows the CDFs for the users classes considered in the case of design choice **a**. As clearly shown the parameter p_{C_2} does not affect the behavior of the system because the C_2 pattern is not used in this design choice while p_{C_1} is the same for $U1$ and $U2$, as well as for $U3$ and $U4$, respectively, thus identifying just two curves. When the design choice **b** is considered the parameter p_{C_2} has some effect slightly worsening the performances by increase as shown in Figure 3b. At the opposite, the analysis reveals a great impact of the parameter p_{C_1} on both the design choices: response time increases when the value of p_{C_1} increases. This is due to increased number of iterations in the loop.

5. CONCLUSIONS

Workflow design and orchestration are critical phases where issues may arise from task composition compromising the overall process. Modeling and evaluation technique can be a valid support to reduce the impact of such issues in early stage, predicting design flows or defects.

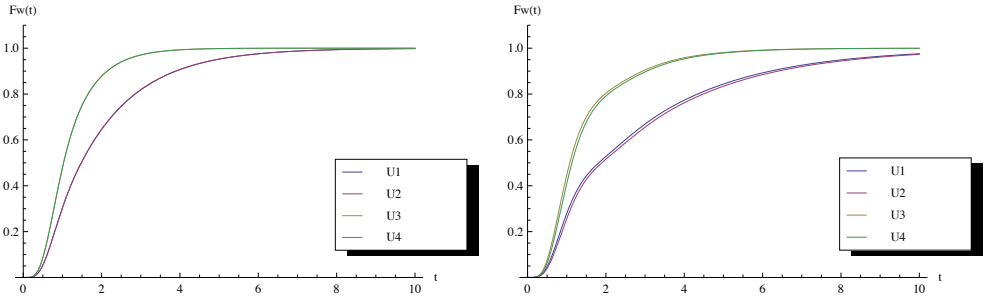
In this paper a technique for dealing with performance evaluation of workflows is proposed. It allows to take into account different customer classes, characterized by their usage profiles, and design alternatives as well. These are probabilistically characterized in a parametric workflow by conditional choices, which analysis parameters are the split probabilities. The proposed technique also allows to take into account the full stochastic characterization for the time behavior of components considering generic distributions. Based on stochastic workflow reduction rules, phase type distributions and related properties, it is able to provide parametric analyses of the worklow on different usage profiles and design choices.

6. REFERENCES

[1] A. Bobbio and A. Cumani. *ML estimation of the parameters of a PH distribution in triangular canonical form*, pages 33–46. Elsevier Science Publishers, 1992.

[2] A. Bobbio, A. Horvath, M. Scarpa, and M. Telek. Acyclic discrete phase type distributions: properties and a parameter estimation algorithm. *Performance Evaluation*, 54(1):1 – 32, 2003.

[3] A. Bobbio and M. Telek. A benchmark for PH estimation algorithms: results for acyclic-PH. *Communications in Statistics. Stochastic Models*, 10(3):661–677, 1994.



(a) Design alternative **a** (b) Design alternative **b**

Figure 3: Workflow completion time distributions.

[4] D. Bruneo, S. Distefano, F. Longo, and M. Scarpa. Stochastic Evaluation of QoS in Service-Based Systems. *IEEE Trans. Parallel Distrib. Syst.*, 24(10):2090–2099, 2013.

[5] J. Cardoso, A. Sheth, J. Miller, J. Arnold, and K. Kochut. Web semantics: Science, services and agents on the world wide web; quality of service for workflows and web service processes. *Journal of Web Semantics, Elsevier*, 1(3):281–308, 2004.

[6] J. A. Cardoso. *Quality of Service and Semantic Composition of Workflows*. PhD thesis, Graduate School of the University of Georgia, Athens, Georgia, Aug 2002.

[7] A. D'Ambrogio and P. Bocciarelli. A model-driven approach to describe and predict the performance of composite services. In *WOSP '07: Proceedings of the 6th international workshop on Software and performance*, pages 78–89, New York, NY, USA, 2007. ACM.

[8] S. Distefano and G. Serazzi. Performance driven WS orchestration and deployment in service oriented infrastructure. *J. Grid Comput.*, 12(2):347–369, 2014.

[9] K. H. HAN, S. K. YOO, and B. KIM. Qualitative and quantitative analysis of workflows based on the uml activity diagram and petri net. *WSEAS Transactions on Information Science and Applications*, 6(7):1249–1258, July 2009.

[10] S.-Y. Hwang, H. Wang, J. Tang, and J. Srivastava. A probabilistic approach to modeling and estimating the qos of web-services-based workflows. *Inf. Sci.*, 177(23):5484–5503, 2007.

[11] G. Janssens, J. Verelst, B., and Weyn. Techniques for modeling workflows and their support of reuse. In W. Van der Aalst, J. Desel, and A. Oberweis, editors, *Business Process Management: Models Techniques and Empirical Studies-LNCS1806*, pages 2–15, London, UK, 2000. Springer-Verlag.

[12] K. Kant. *Introduction to Computer System Performance Evaluation*. Mc Graw-Hill, 1992.

[13] M. Kowal, I. Schaefer, and M. Tribastone. Family-based performance analysis of variant-rich software systems. In *Proceedings of the 17th International Conference on Fundamental Approaches to Software Engineering - Volume 8411*, pages 94–108, New York, NY, USA, 2014. Springer-Verlag New York, Inc.

[14] M. Litoiu, J. Rolia, and G. Serazzi. Designing process replication and activation: a quantitative approach. *Software Engineering, IEEE Transactions on*, 26(12):1168 –1178, dec 2000.

[15] M. Marzolla and R. Mirandola. Performance prediction of web service workflows. In *Proceedings of the Quality of software architectures 3rd international conference on Software architectures, components, and applications*, QoSA'07, pages 127–144, Berlin, Heidelberg, 2007. Springer-Verlag.

[16] M. Neuts. Probability distributions of phase type. In *Liber Amicorum Prof. Emeritus H. Florin*, pages 173–206. University of Louvain, 1975.

[17] M. Neuts. *Matrix Geometric Solutions in Stochastic Models*. Johns Hopkins University Press, Baltimore, 1981.

[18] S.-C. Oh, D. Lee, and R. T. Kumara. Misq:a framework to analyze and optimize web service composition in business service networks. *INTERNATIONAL JOURNAL OF CASES ON ELECTRONIC COMMERCE*, 1(4):35–55, 2005.

[19] R. Sahner, K. Trivedi, and A. Puliafito. *Performance and reliability analysis of computer systems*. MKluwer Academic Publisher, Norwell, Massachusettes, 02061, USA, 1st edition, 1996.

[20] I. Schaefer. Variability modelling for model-driven development of software product lines. In *VaMoS*, pages 85–92, 2010.

[21] A. Thummler, P. Buchholz, and M. Telek. A novel approach for phase-type fitting with the em algorithm. *IEEE Transactions on Dependable and Secure Computing*, 3(3):245–258, July 2006.

[22] W. Van Der Aalst and K. M. Van Hee. *Workflow management: models, methods, and systems*. MIT press, 2004.

[23] M. Wimmer, M.-C. Albutiu, and A. Kemper. Optimized workflow authorization in service oriented architectures. In *Emerging Trends in Information and Communication Security*, pages 30–44. Springer, 2006.

[24] H. P. Zhu, S. W. Xiao, and Z. Che. Research on workflow simulation supported business process analysis. *Mini-Micro Systems*, 28(8):1531–1535, 2006.