



COVER SHEET

This is the author-version of article published as:

La Rosa, Marcello and Lux, Johannes and Seidel, Stefan and Dumas, Marlon and ter Hofstede, Arthur H. M. (2006) Questionnaire-driven Configuration of Reference Process Models.

Accessed from <http://eprints.qut.edu.au>

Questionnaire-driven Configuration of Reference Process Models

Marcello La Rosa, Johannes Lux, Stefan Seidel, Marlon Dumas
and Arthur H. M. ter Hofstede

BPM Group, Queensland University of Technology, Australia
{m.larosa, j.lux, s.seidel, m.dumas, a.terhofstede}@qut.edu.au

Abstract. Reference models are a widely accepted means to facilitate reusable information system and organizational design. At present, besides domain knowledge, the configuration of reference models requires a thorough understanding of both the reference model and the language it is captured in. This hinders the involvement of domain experts without specialized modeling background, in the configuration of reference models. In this paper, we propose a questionnaire-driven approach to reference model configuration which abstracts away from the modeling language. For illustration, we show how this approach can be applied to reference process models captured in the Configurable EPC notation. To demonstrate its applicability, the proposal has been implemented as a toolset that guides users through the configuration process by means of a form-based interface.

Key words: reference model, business process configuration, interactive questionnaire, Configurable EPC

1 Introduction

The benefits of reference models for Information Systems (IS) and organizational design are widely accepted in theory and practice [7]. The main objective is to streamline the development of individual models based on the reuse of complex and well-designed artifacts [8]. The use of reference models can facilitate the modeling process, thus lowering problems of time and cost [16]. This has led to the emergence of a number of reference models for specific domains such as IT service management (ITIL) [20] or supply chain management (SCOR) [19].

However, the process of configuring a reference model to the demands of an organization requires the user to have a thorough understanding of both the domain and the modeling language the reference model has been constructed in. They must be capable of estimating the impact of configuration decisions on the model. Consequently, the user who carries out the configuration must not only be a domain expert but also be skilled in reading and adapting reference models. This assumption is unrealistic in application domains where users are unfamiliar with modeling notations.

To facilitate the configuration of reference models we propose a questionnaire-driven approach based on the representation of choices and their dependencies.

Via so-called *facts* that represent answers, the *questions* are linked to variation points in reference models. The user, therefore, does not have to directly deal with the reference model anymore. Questions are expressed in natural language and can be answered by domain experts without extensive knowledge of the underlying reference model. The only major assumption made is that questions have a finite or discretized domain of possible answers. This assumption allows the models to be efficiently analyzed in order to prevent the user from entering conflicting responses to subsequent questions.

In this paper we show how this approach can be applied to configuring reference process models. In particular, we target real configuration scenarios, where complex and intricate interdependencies may render model-based configurations unacceptably arduous. To demonstrate this, we have formalized a mapping between our approach and Configurable Event-driven Process Chains (C-EPCs) [15]. C-EPC is a notation that allows modelers to explicitly define variation points and was therefore found suitable to show how questions can be linked to these variation points. C-EPCs have also been used to construct a reference process model that serves as a working example for this research. In order to illustrate how the configuration process can be simplified and automated, the presented concepts have been implemented as a toolset that guides users through the configuration process by means of a form-based interface.

The rest of the paper is organized as follows. In the next section we briefly describe C-EPCs. We then introduce a working example and demonstrate how the formalized approach to questionnaire-driven configuration can be applied. We further show how the automated configuration has been implemented including the mapping between C-EPC and interactive questionnaires. The paper concludes with related work and an outlook including our future research agenda.

2 Background: Configurable Event-driven Process Chains

Event Driven Process Chains (EPCs) [9, 1] are a widely used modeling language whose main components are *events*, *functions*, *connectors* and *arcs* linking these elements. Events represent triggers or conditions, functions correspond to tasks, and connectors denote splits and joins of type *AND*, *OR* or *XOR*.

C-EPCs extend EPCs by providing a means to explicitly represent variability in EPC reference process models. This is achieved by identifying a set of variation points (*configurable nodes*) in the model, to which possible values (*alternatives*) can be assigned, as well as constraints to restrict the combination of allowed values. By configuring each variation point to exactly one value among the ones allowed, it is possible to derive an EPC model from the starting C-EPC.

Variation points are nodes of type *function* or *connector*, highlighted in bold in the model. *Configurable functions* can be set as included (*ON*), excluded (*OFF*) or conditionally skipped (*OPT*). The first two alternatives allow one to decide a priori whether to keep or permanently discard the function; the last option permits the deferral of this choice to run-time, where the execution of the function can be skipped on an instance-by-instance basis. *Configurable*

connectors can only be mapped to equally or less expressive connector types. Consequently, a configurable *AND*-connector can only be mapped to a regular *AND*-connector. A configurable *XOR* can be set to a regular *XOR* or to an outgoing/incoming sequence SEQ_n of events and functions (where n is the node starting the sequence). A configurable *OR* can be mapped to a regular *OR*, *XOR*, *AND* or to a single sequence. Moreover, *configuration requirements* formalize constraints over the values of variation points, whilst *configuration guidelines* express advices and industry best practices to aid the configuration process. They are both expressed in the form of logical predicates and depicted as notes attached to the variation points involved. Only requirements are mandatory and must hold in order for a configuration to be valid. Finally, a partial order over variation points can be defined as a suggested order for configuring the nodes of the model.

The following definitions formalize the above concepts and closely follow [15]:

Definition 1 (Configurable EPC). *A configurable EPC is a ten-tuple C-EPC $= (E, F, C, l, A, F^C, C^C, O^C, R^C, G^C)$ where:*

- E, F, C, l and A refer to standard EPC sets of events, functions, connectors, a mapping to define a label *AND*, *XOR*, or *OR* for each connector, and arcs,
- $F^C \subseteq F$ is the set of configurable functions,
- $C^C \subseteq C$ is the set of configurable connectors,
- $O^C \subseteq (F^C \cup C^C) \times (F^C \cup C^C)$ is a partial order over the configurable nodes,
- R^C is the set of configuration requirements,
- G^C is the set of configuration guidelines.

Definition 2 (Partial Order for Connectors). *The partial order \leq^C is defined on $CT \cup CTS$ where $CT = \{AND, OR, XOR\}$ is the set of connector types and $CTS = \{SEQ_n \mid n \in E \cup F \cup C\}$ is the set of sequence operators. $\leq^C = \{(n, n) \mid n \in CT\} \cup \{(XOR, OR), (AND, OR)\} \cup CTS \times \{XOR, OR\}$.*

The partial order \leq^C is used to determine, by restriction, the set of values each configurable connector can be mapped to. For example $XOR \leq^C OR$ implies the second configurable connector type can be mapped to the first connector type.

A *configuration* is a mapping that links a configurable node to an allowed value, according to the node type. It also ensures that a sequence can be chosen as value, only if it is an incoming branch for a configurable join-connector, or an outgoing branch for a configurable split-connector.

Definition 3 (Configuration l^C). *Let C-EPC $= (E, F, C, l, A, F^C, C^C, O^C, R^C, G^C)$ be a configurable EPC. The mapping $l^C \in (F^C \rightarrow \{ON, OFF, OPT\}) \cup (C^C \rightarrow CT \cup CTS)$ is a configuration of C-EPC iff for each $c \in C^C$:*

- $l^C(c) \leq^C l(c)$,
- if $c \in C_J$ and $l^C(c) = SEQ_n$ for some $n \in E \cup F \cup C$, then $(n, c) \in A$, where C_J is the set of join connectors,
- if $c \in C_S$ and $l^C(c) = SEQ_n$ for some $n \in E \cup F \cup C$, then $(c, n) \in A$, where C_S is the set of split connectors.

When the above requirements are met, we write $valid_{C-EPC}(l^C)$, or simply $valid(l^C)$ if from the context the C-EPC involved is clear.

3 Working Example

As part of a research effort focusing on business process management for the Post-Production phase in the screen business, we developed a C-EPC reference process model [17]. An extract of this model is presented in Fig. 1 and will be used as working example throughout the paper.

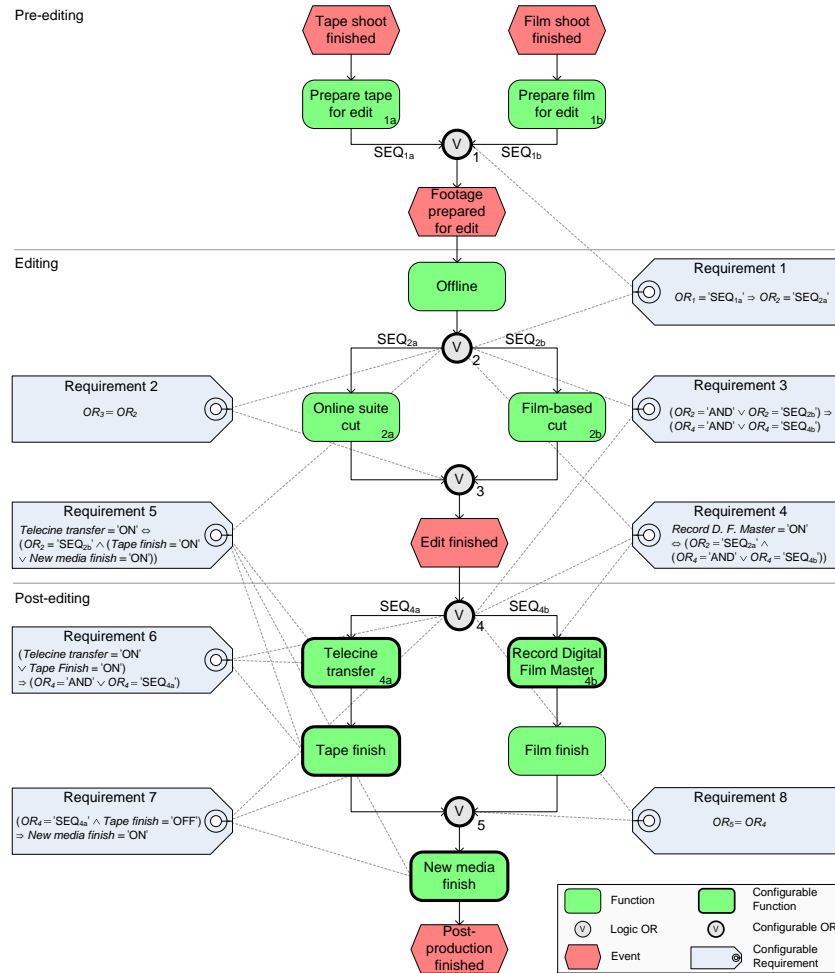


Fig. 1. The Post-Processing reference process model example.

Post-Production aims at the edit and technical completion of a screen business project and encompasses three main steps: pre-edit, edit and post-edit. In the first phase the footage arriving from the shooting is prepared for editing by synchronizing audio and video. The shooting format can be tape, film, or both media. Of the two, film results in a more costly operation as special treatments are required, for making it visible and permanent. The choice of the medium is

modeled in the C-EPC example via configurable connector OR_1 , which can be set to the value SEQ_{1a} for tape, SEQ_{1b} for film, or AND for both.

The first activity of the edit phase is the Offline (function *Offline* in the model), where the main creative editing part is carried out on a low resolution medium. This is followed by the cut stage, where the editing decisions previously taken are committed in a high quality format. The cut can be *Film-based* or carried out in an *Online Suite*, according to the format of the shooting media. This variability is achieved in the model by configuring OR_2 and OR_3 to just one of the two branches (SEQ_{2a}, SEQ_{2b}) or to their parallel execution (AND). However, the film-based variant can be selected only if at least parts of the project were shot on film. This is enforced by *Req. 1*, while *Req. 2* ensures OR_3 to be configured the same way as OR_2 , since they share the same incoming/outgoing branches. As a result, we can impose requirements only on the split, as the join connector will be configured accordingly. Note that we could have defined $(OR_2, OR_3) \in OC$ to specify that OR_2 should be configured before OR_3 . However the partial order is just a suggestion, and does not compel OR_3 to behave as OR_2 . Therefore *Req. 2* is needed.

In post-edit, the project is finished for delivery on tape, film, new media or any combination of these formats. The overall finishing process varies on the basis of the delivery media and may involve further tasks, according to the configuration choices made before. A *Film-based cut* is an expensive activity. Therefore, if performed, it must lead at least to a *Film finish*. This is guaranteed by *Req. 3* attached to connector OR_4 . In fact, if the Film-based branch is enabled by OR_2 , then function *Film finish* will be executed either if OR_4 is equal to AND or SEQ_{4b} . However, if the cut has been done only in the online suite, then a further task, modeled by configurable function *Record Digital Film Master*, is needed to transfer the editing results to the so-called ‘film master’. This is stated in *Req. 4*. Analogously, *Telecine transfer* is used only if the cut is film-based and if at least a finish on tape or new media is expected. This behavior is enforced by *Req. 5*, attached to configurable functions *Tape finish* and *New media finish*. These two functions belong to the outgoing branch SEQ_{4a} of OR_4 ; thus *Req. 6* ensures that SEQ_{4a} is activated if at least one of these two functions has been set to ON . *Req. 7* guarantees that at least one finish medium is selected, as *New media Finish* must be set to ON , if no film nor tape finish is desired (i.e. if $OR_4 = SEQ_{4a}$). Finally, *Req. 8* imposes OR_5 to take the same value as OR_4 .

For simplicity’s sake we did not consider those configuration alternatives involving run-time choices (XOR, OR, OPT) and implied the existence of further requirements to avoid such alternatives. Nonetheless, this is an example of how interdependencies over configurable nodes can be complex and intricate when the model refers to a real configuration scenario. In such cases, model-based configurations may turn out to be unacceptably arduous. Moreover, domain experts – supposed to be in charge of the configuration – are likely to be unaware of business process notations, as in the screen business case. In order to tackle these issues, in the next section we propose a new approach to configuration.

4 Approach

4.1 Questionnaire-driven Configuration

We propose to represent choices independently of specific notations or languages, by means of a set of facts, representing the space of possible answers to a set of questions. Questions can be answered solely requiring domain expertise, via an interactive questionnaire that guides the configuration, by posing only the relevant questions in an order consistent with the interdependencies.

Let us examine the approach in detail. To us making a choice corresponds to setting a *fact* within a *question*. Facts are simply statements such as “tape shooting” or “film finish”. Initially they are *unset* while at run-time they can be asserted or negated by setting their value to *true*, resp. *false*. For example, setting “tape shooting” = *false*, would mean that we are not interested in shooting on tape. Each fact features a default value and can be marked as ‘mandatory’ if it needs to be set explicitly. Under certain restrictions, a non-mandatory fact can be left *unset* at configuration-time. In this case its default value is used instead.

Facts are grouped in questions according to their content, so that all the facts of the same group can be set at once by answering the associated question. Each question features at least one fact and the set of questions must cover all the facts. Although a fact can appear in more than one question, its value can be set only the first time, and must be preserved in all the subsequent questions that contain it. However, the value of a fact previously set can still be changed by rolling back the question.

A *facts setting* is any combination of facts values such that all the facts have been set, either explicitly by answering questions or by using their default.

Fig. 2 depicts a possible structure of questions/facts for representing variability in Post-Production. All questions and facts are assigned a unique id and a description. For example, facts f_1 to f_3 refer to typical budget ranges for a Post-Production project, so they all are grouped in question 1 asking for the estimated project budget. Also, these facts are mandatory as we want users to explicitly answer q_1 . Indeed, the choice of budget is rather important as it affects the Post-Production phase overall. Default values have been assigned in order to reflect the typical choices made in a medium budget project, pitched for cinema and home video distribution. Hence, f_2 in q_1 has default value equal to *true* as well as f_4 and f_6 in q_2 , which relates to the primary distribution channels, and so on for the other facts. Other questions would allow users to choose the shooting media (q_3) and the shooting format (q_6, q_7), the type of cut (q_4) and the expected deliverables (q_5).

Questions can be connected via two different types of dependencies. Dependencies determine a partial order for posing questions to users, and can be arbitrary as long as undesired cycles are avoided. A *simple dependency* (dashed arrow in Fig. 2) is used when a question “may” depend on another, whilst a *strict dependency* (plain arrow) is used to model a compulsory order over two questions. For example in Fig. 2, q_3 allows users to choose the shooting media between tape (f_9) and film (f_{10}). This question “simply” depends on q_1 and q_2 ,

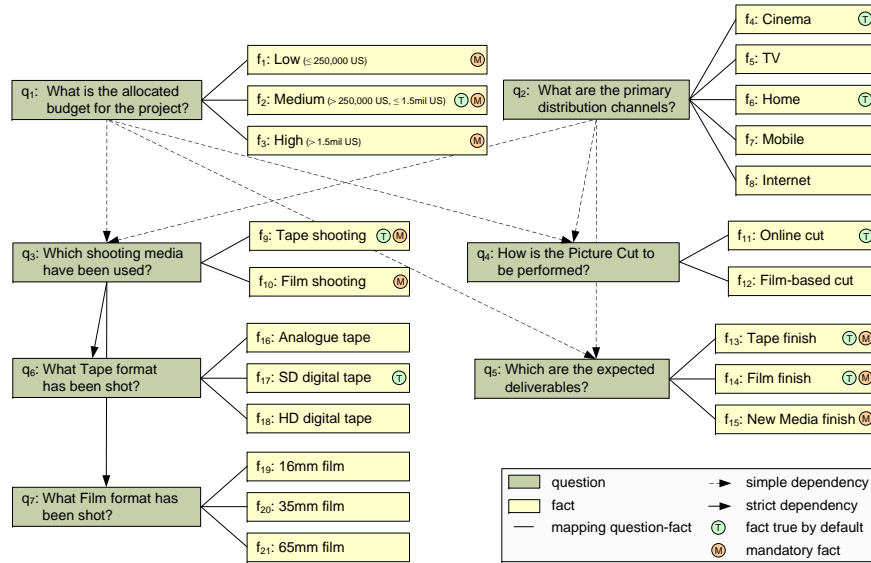


Fig. 2. A possible structure of questions/facts for the Post-Processing example.

viz., it can be posed only after answering at least one of q_1 and q_2 . On the other hand, q_6 – where the tape format is determined – strictly depends on q_3 , as it is reasonable to make this choice only after deciding on the shooting media which includes tape as possible alternative (f_9). Although not shown in this example, dependencies can also be defined over facts, by following the same rules.

Dependencies provide a means for ordering questions but do not affect facts values. Answering a question in a given way may restrict the allowed answers to subsequent questions, and not all combinations of answers may lead to valid facts settings. We model interdependencies over facts values as a set of constraints in propositional logic, used to restrict the number of possibilities. A facts setting is thus a *configuration* if and only if it complies with the constraints.

The following constraints, drawn from the analysis of the Post-Production phase, refer to the facts of Fig. 2:¹

$$\begin{array}{lll}
 \text{C1: } f_1 \dot{\vee} f_2 \dot{\vee} f_3 & \text{C2: } f_1 \Rightarrow \neg(f_{10} \vee f_{14}) & \text{C3: } f_2 \Rightarrow \neg f_{10} \\
 \text{C4: } f_4 \vee f_5 \vee f_6 \vee f_7 \vee f_8 & \text{C5: } f_4 \Rightarrow f_{14} & \text{C6: } f_5 \Rightarrow f_{13} \\
 \text{C7: } f_6 \Rightarrow (f_{13} \vee f_{15}) & \text{C8: } (f_7 \vee f_8) \Rightarrow f_{15} & \text{C9: } f_9 \vee f_{10} \\
 \text{C10: } f_{11} \vee f_{12} & \text{C11: } \neg f_{10} \Rightarrow \neg f_{12} & \text{C12: } f_{13} \vee f_{14} \vee f_{15} \\
 \text{C13: } (f_{16} \dot{\vee} f_{17} \dot{\vee} f_{18}) \Leftrightarrow f_9 & \text{C14: } \neg(f_{16} \vee f_{17} \vee f_{18}) \Leftrightarrow \neg f_9 & \text{C15: } f_{12} \Rightarrow f_{14} \\
 \text{C16: } (f_{19} \dot{\vee} f_{20} \dot{\vee} f_{21}) \Leftrightarrow f_{10} & \text{C17: } \neg(f_{19} \vee f_{20} \vee f_{21}) \Leftrightarrow \neg f_{10}. &
 \end{array}$$

Let us go through some of them. For example, C1 ensures that exactly one fact is asserted in q_1 , as a project places itself only in a specific budget range. On the other hand, due to C4, more than one distribution channel can be selected in q_2 , as it makes sense for a project to have multiple releases (e.g. *TV* and *Home*).

¹ $\dot{\vee}$ indicates the exclusive disjunction (*XOR*), a commutative and associative relation.

We said that due to the costs involved, a film-based cut would be worthwhile if it implied a subsequent finish on film. This is captured by C15, affecting the way q_4 and q_5 can be answered. In truth, as per C2, a low budget choice ($f_1 = true$) implies no shooting on film nor release on film is possible ($f_{10}, f_{14} = false$). As a result, for low budgets a film-based cut is not allowed either (C11).

Dependencies and constraints are not overlapping concepts. They rather complement each other, as shown by C11 over f_{10} and f_{12} . These two facts occur in q_3 resp. q_4 , but the questions do not depend on each other. Thus, by setting f_{10} to *false*, f_{12} is forced to *false* too, although we could have already negated the latter by answering q_4 before q_3 . In this case, only a constraint is used to achieve the desired behavior. On the other hand, as per C13, (exactly) one tape format can be chosen in q_6 , if and only if tape has been selected as shooting medium in q_3 ($f_9 = true$). Otherwise, no tape format can be specified (C14). Anyhow, q_6 cannot be answered before q_3 . In such a case, dependencies and constraints work together to ensure the shooting format being decided only after the shooting medium, and according to its type.

When such a structure is used for configuring a configurable reference process model, a mapping between facts and variation points needs to be defined, so that certain actions are performed on the model in order to reflect the values chosen for facts. Therefore, we hereafter formalize the above concepts towards a rigorous definition of the mapping.

4.2 Formal Definition of Configuration Models

Due to space limitations, this section presents only a reduced definition of *Configuration Model (CM)*, which is the formal underpinning to our approach. The complete definitions, technical details and proofs can be found in [14].

Definition 4 (Reduced Configuration Model). *A reduced configuration model is a six-tuple $rCM = (F, F_D, F_M, Q, map_{QF}, CS)$ where:*

- F is a finite, non-empty set of facts where a fact is a boolean variable,
- $F_D \subseteq F$ is the default setting, i.e. the set of facts whose default is asserted,
- $F_M \subseteq F$ is the set of mandatory facts,
- Q is a finite (non-empty) set of questions,
- $map_{QF} \in Q \rightarrow \mathcal{P}(F) \setminus \{\emptyset\}$ is a function mapping questions onto sets of facts, such that $\bigcup_{q \in Q} map_{QF}(q) = F$,²
- $CS \subseteq \mathcal{P}(F)$ is the set of the allowed settings of the facts in F , such that $F_D \in CS$, i.e. the default setting is always allowed.

Elements of CS are those facts settings that satisfy all the constraints, where only the facts asserted are present in each element. Hence, if a fact is not contained in a clause of CS , it follows that the fact is negated in that setting. Also, as the default setting must be always allowed, set CS is non-empty. The definition of set CS has been used to construct a set of functions for detecting possible

² \mathcal{P} indicates a power set.

conflicts over facts constraints at design-time, and for dynamically restricting the space of available configurations at run-time. Those definitions can be found in [14]. An implementation of these concepts is discussed in Section 5.

A *configuration* σ of CM is thus a facts setting whose truth values form exactly an element of cs , i.e. a facts setting that does not violate the constraints.

4.3 Mapping C-EPCs to Configuration Models

We propose a simple method to define a mapping between a C-EPC and a questionnaire-driven configuration model. The idea is to assign a boolean function over the facts of the configuration model, to each configuration a variation point can assume in the C-EPC. For example, the first configurable node in the C-EPC process of Fig. 1, OR_1 , according to the type of shooting medium, can be set to *AND* (for both tape and film), SEQ_{1a} (for tape only) and SEQ_{1b} (for film only). In the configuration model of Fig. 2 this would correspond to answer q_3 (*Which shooting media have been used?*) with both $f_9, f_{10} = true$ in the first case, with only $f_9 = true$ in the second case, and with only $f_{10} = true$ in the third (where f_9 is *Tape shooting* and f_{10} is *Film shooting*). This is equivalent to checking whether $f_9 \wedge f_{10}$, or $f_9 \wedge \neg f_{10}$, or $\neg f_9 \wedge f_{10}$ holds against a given configuration over facts, obtained by answering the questions shown in Fig. 2. Thus, we assign each of these functions to the corresponding configuration of OR_1 . The remaining configuration alternatives of OR_1 , not allowed by the model (i.e. OR, XOR), are simply given a *false* function.

The only requirement for a mapping between a C-EPC and configuration model to be unambiguous is that, for any facts configuration, exactly one boolean function should evaluate to *true* for each variation point in the model. This way we avoid a facts configuration that may lead to zero or more than one alternative for a variation point. In the above example this is enforced by C9, which excludes the fourth combination $\neg f_9 \wedge \neg f_{10}$ as a possible condition of facts values.

Once each variation point has been configured with a proper configuration value, an action, attached to that value, has to be performed on the C-EPC net so as to reflect the values chosen for facts.

A formalization of the mapping is given in Definition 5.

Definition 5 (CF-Mapping, Valid CF-Mapping, Actions, CA-Mapping).

Let $C\text{-EPC} = (E, F, C, l, A, F^C, C^C, O^C, R^C, G^C)$ be a configurable EPC, $l^C \in (F^C \rightarrow \{ON, OFF, OPT\}) \cup (C^C \rightarrow CT \cup CTS)$ a configuration of C-EPC, and let $rCM = (F, F_D, F_M, Q, map_{QF}, CS)$ be a reduced configuration model. For each configurable node $cn \in C^C \cup F^C$:

- $L^{cn} = \{l^C(cn) \mid valid(l^C)\}$ is the set of all the configurations of C-EPC for a given cn ,
- $map_{CF}^{cn} \in L^{cn} \rightarrow \mathbb{B}_F$ is a CF-Mapping, i.e. a function mapping a configuration of cn to a boolean function defined over F ,
- $\varphi \equiv \bigvee_{v \in CS} (\bigwedge_{f \in v} f \wedge \bigwedge_{f \in F \setminus v} \neg f)$ is a boolean function evaluating to true if there exist constraints over F ,

- $\text{valid}(\text{map}_{CF}^{cn})$ holds iff, under the assumption that constraints over F exist, for every configurable node exactly one configuration holds, i.e. iff $\varphi \Rightarrow \bigvee_{e \in L^{cn}} (\text{map}_{CF}^{cn}(e))$ is a tautology,
- Act is a finite set of actions, corresponding to modifications in the C-EPC model in order to reflect choices made over facts,
- $\text{map}_{CA}^{cn} \in L^{cn} \rightarrow Act$ is a function assigning an action to each configuration l^C of cn .

Given a configuration σ of CM , for each configurable node cn of C-EPC, an action a related to a configuration l^C of cn is performed if and only if the boolean function associated to l^C evaluates to *true* taking the values assigned to facts in σ . As functions exclude each other, only one action per variation point can be executed.

The following table shows for each configurable node in the C-EPC example of Fig. 1, the associated boolean function defined over the facts of Fig 2.

Configurable node cn	Configuration $l^C(cn)$	Boolean function $\text{map}_{CF}^{cn}(l^C(cn))$
OR_1	AND	$f_9 \wedge f_{10}$
	SEQ _{1a}	$f_9 \wedge \neg f_{10}$
	SEQ _{1b}	$\neg f_9 \wedge f_{10}$
	OR	<i>false</i>
	XOR	<i>false</i>
OR_2	AND	$f_{11} \wedge f_{12}$
	SEQ _{2a}	$f_{11} \wedge \neg f_{12}$
	SEQ _{2b}	$\neg f_{11} \wedge f_{12}$
	OR	<i>false</i>
	XOR	<i>false</i>
OR_3	same as OR_2	
OR_4	AND	$(f_{13} \wedge f_{14}) \vee (f_{12} \wedge \neg f_{13} \wedge f_{15})$
	SEQ _{4a}	$(f_{13} \wedge \neg f_{14}) \vee (\neg f_{13} \wedge \neg f_{14} \wedge f_{15})$
	SEQ _{4b}	$(\neg f_{13} \wedge f_{14} \wedge \neg f_{15}) \vee (\neg f_{12} \wedge \neg f_{13} \wedge f_{14})$
	OR	<i>false</i>
	XOR	<i>false</i>
OR_5	same as OR_4	
<i>Telecine transfer</i>	ON	$(\neg f_{11} \wedge f_{13}) \vee (\neg f_{11} \wedge f_{15})$
	OFF	$\neg((\neg f_{11} \wedge f_{13}) \vee (\neg f_{11} \wedge f_{15}))$
	OPT	<i>false</i>
<i>Record Digital Film Master</i>	ON	$\neg f_{12} \wedge f_{14}$
	OFF	$\neg(\neg f_{12} \wedge f_{14})$
	OPT	<i>false</i>
<i>Tape finish</i>	ON	f_{13}
	OFF	$\neg f_{13}$
	OPT	<i>false</i>
<i>New media finish</i>	ON	f_{15}
	OFF	$\neg f_{15}$
	OPT	<i>false</i>

Table 1. Mapping configuration alternatives for the Post-Production example.

We can see that some facts have a 1-1 mapping with C-EPC variation points (e.g. f_{13}, f_{15}). In general though, a fact can have a wider impact on the process model (1-N). Consider for example f_1 (low budget). Although this fact does not appear in any function in the above table, if asserted, it would affect a number of configuration nodes due to C2 and C11. Namely all the variation points whose boolean functions feature $\neg f_{10}, \neg f_{12}, \neg f_{14}$. This would lead to the following configuration: $OR_1 = SEQ_{1a}, OR_2 = OR_3 = SEQ_{2a}, OR_4 = OR_5 = SEQ_{4a}$, where all the branches of the C-EPC model involving an activity related to film have been denied.

In general, the more impact a fact has on successive facts, the more variation points in the process model are likely to be affected. This depends on the way both constraints over facts and boolean functions have been defined.

Facts 16 to 21 – regarding the shooting formats for tape and film – are the only ones that do not affect any variation point, neither directly nor indirectly. Thus, they do not appear in any boolean function of Tab. 1. Indeed these facts influence those variation nodes occurring in the sub-processes of functions *Prepare Tape for edit* and *Prepare Film for edit*, that are not shown in Fig. 1.

It is up to the designer to consider all the configuration requirements of a C-EPC model, in order to build the right set of boolean functions. In the following section we present an implementation which, among others, ensures the consistency of boolean functions with respect to the constraints over the facts of a configuration model.

5 Tool Support

With the purpose of validating our questionnaire-driven approach from a practical perspective, we implemented a set of tools during the course of this research. Each tool is a stand-alone application responsible for specific tasks in the configuration process. However, when combined, the tools provide end-to-end support for reference process model configuration, from the collection of the answers via questionnaires, to the release of a configured process model. So far we only support the C-EPC language, but the architecture is such that new modules for new configurable process notations can be easily plugged in. Due to space limitations we just present an overview of the architecture, without entering into the details of the implementation.³ Fig. 3 depicts the architecture.

The *Quaestio* tool takes an XML serialization of a configuration model as input and guides the configuration interactively, by posing only the relevant questions in an order consistent with the interdependencies. The implementation embodies an existing, very efficient BDD calculator,⁴ based on Binary Decision Diagrams (BDDs) [4], to dynamically check answers against the constraints. Questions can be answered by users or automatically by the system (using defaults), and they can be rolled back.

³ Downloadable from <http://sky.fit.qut.edu.au/~dumas/ConfigurationTool.zip>

⁴ Downloadable from <http://www-verimag.imag.fr/~raymond/tools/bddc-manual>

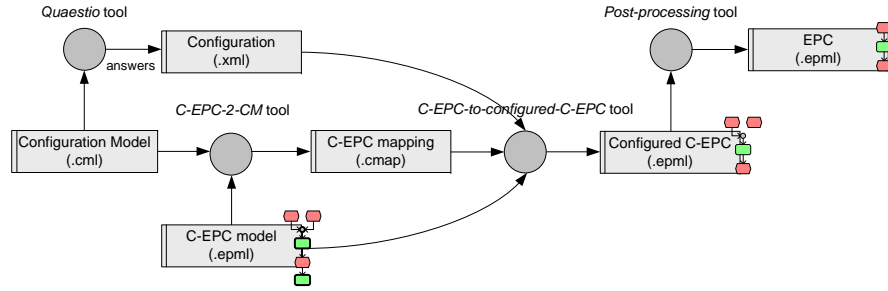


Fig. 3. The software architecture of the tools implemented.

The *C-EPC-2-CM* tool allows designers to define boolean functions over the facts of a configuration model, and to link them to the variation points of a C-EPC net, whose EPML⁵ serialization is taken as input. This tool uses the BBD calculator to check whether the functions assigned to the configuration values of each variation point are in mutual exclusion, so as to generate only valid mappings.

The *C-EPC-to-configured-C-EPC* tool takes as input a configuration over facts generated by *Quaestio*, the EPML of the C-EPC model, and the mapping linking the C-EPC to the corresponding configuration model. It gives as output an EPML representation of a configured C-EPC, where each configurable node has been marked with a configuration value, according to the EPML syntax for C-EPC presented in [10]. This artifact is then post-processed by a tool implementing the derivation algorithm presented in [11]. The output of the latter is a syntactically correct EPC model.

Fig. 4 depicts the EPC model resulting from the application of configuration $\sigma = \{f_1, \neg f_2, \neg f_3, \neg f_4, \neg f_5, \neg f_6, f_7, f_8, f_9, \neg f_{10}, f_{11}, \neg f_{12}, \neg f_{13}, \neg f_{14}, f_{15}, \dots\}$ to the Post-Production C-EPC model.⁶ This configuration corresponds to a low budget project shooting on tape, performing an online cut and releasing on the new media *Mobile* and *Internet*. According to the mapping of Tab. 1, the C-EPC variation points assume the following configuration values: $OR_1 = SEQ_{1a}$, $OR_2 = OR_3 = SEQ_{2a}$, $OR_4 = OR_5 = SEQ_{4a}$, *Telecine transfer* = *OFF*, *Record Digital Film Master* = *OFF*, *Tape finish* = *OFF*, *New media finish* = *ON*.

6 Related Work

Conceptual support for adapting reference models is not prevalent in the field of IS. Existing approaches exhibit a heterogeneous set of methods for reusing reference models but do not provide formalized support to abstract from the model during the configuration process. Process alternatives are depicted as process specializations in [18]. For their activation, these specializations are linked to

⁵ <http://wi.wu-wien.ac.at/~mendling/EPML>

⁶ The model is shown in the *EPC Tools*, <http://wwwcs.upb.de/cs/kindler/Forschung/EPCTools>.

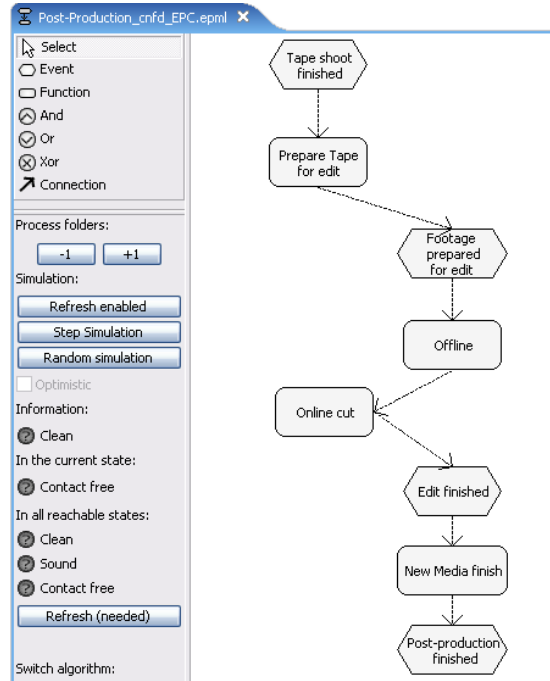


Fig. 4. The EPC for low-budget Post-Productions, obtained from the C-EPC of Fig. 1.

conditions expressed in questions. However, dependencies and constraints between questions cannot be expressed. Following a domain engineering approach, [13] defines so-called stereotypes to specify the multiple appearance of model elements. The model instantiation is not supported by any abstraction from the actual model. There are also no means to depict dependencies and constraints.

Reference model adaptation mechanisms have been introduced in [5]. Among others, these include logical terms and attributes that are linked to model elements to indicate which sections are not relevant for a specific application scenario and have thus to be removed. The approach supports the configuration apart from the actual model. However, it does not address the definition of management questions, particularly not their dependencies and relationships. Consequently, existing approaches to reference model adaptation support configuration but still require users to be model experts in order to perform it. However, as we shown in this paper, there are scenarios where this is unfeasible, due to the complexity of interdependencies of configuration decisions and the degree of variability.

Our work is also related to questionnaire systems. A range of commercial products supports the creation of online questionnaires. They rely on the notion of question flows and support dependencies among questions but lack support to capture constraints among them [12]. Form definition languages like XForms [6] support both dependencies and constraints but are impractical for our purpose as

they define constraints in a first-order logic language which hampers an efficient computational analysis.

The field of Software Configuration Management deals with models and languages to capture how a collection of available options impact the way a software system is built from a set of components. For a comparison between our proposal and existing work in this area, refer to [14].

7 Conclusion

In this paper we showed how a formalized approach to questionnaire-driven model configuration can be applied to the configuration of reference process models. We do not claim that questionnaire-driven configuration will substitute configurable reference models. Rather, we suggest the combined use of both.

While modeling languages for configurable reference modeling such as C-EPCs support the specification of variation points and alternatives, interactive questionnaires facilitate the configuration process itself by abstracting from the actual model. To demonstrate this, we have implemented a toolset that generates interactive questionnaires for model configuration, and a mapping between C-EPCs and facts gathered by these questionnaires.

Future research has to show that our approach can be applied to other modeling languages as well. As an example, we are currently developing a new configurable process language, based on the YAWL environment [2]. This is a suitable area to apply the results of our research, due to YAWL's comprehensive support for the workflow control-flow patterns [3].

The need for abstracting the configuration from actual models can be argued as follows. First, the model user is not required to have extensive knowledge of both the domain and the reference model. Second, when variation points are added to a model, configuration complexity increases dramatically and, thus, configuring the model without any means of abstraction gets close to unmanageable. Regarding this research, some limitations have to be pointed out. First, the actual impact of questionnaire-based configuration on the modeling process has not been empirically investigated. Second, in this paper we have not elaborated on the relationship between the construction of configurable reference models and the construction of interactive questionnaires. The question still remains whether the questionnaire or the reference model should be constructed first. Alternatively, this could be an iterative process in which the reference model construction influences the questionnaire and vice versa.

Acknowledgments. The authors wish to thank Florian Gottschalk and Michael Rosemann for their valuable comments and Wil M. P. van der Aalst for his crucial contribution to the formalization of the approach.

References

1. W. M. P. van der Aalst. Formalization and Verification of Event-driven Process Chains. *Information and Software Technology*, 41(10): 639–650, 1999.

2. W. M. P. van der Aalst, L. Aldred, M. Dumas, and A. H. M. ter Hofstede. Design and Implementation of the YAWL System. In A. Persson and J. Stirna, editors, *Advanced Information Systems Engineering, Proceedings of CAiSE'04*, vol. 3084, pp. 142–159, 2004.
3. W. M. P. van der Aalst, A. H. M. ter Hofstede, B. Kiepuszewski, and A.P. Barros. Workflow Patterns. *Distributed and Parallel Databases*, 14(1): 5–51, 2003.
4. S. B. Akers. Binary Decision Diagrams. *IEEE Trans. Computers*, 27(6): 509–516, 1978.
5. J. Becker, P. Delfmann, and R. Knackstedt. Adaptive Reference Modeling: Integrating Configurative and Generic Adaptation Techniques for Information Models. In *Reference Modeling Conference 2006*, Passau, 2006.
6. J. Boyer, D. Landwehr, R. Merrick, T. Raman, M. Dubinko, and L. Klotz. XForms 1.0 se, W3C Recommendation, 2006. <http://www.w3.org/MarkUp/Forms>.
7. P. Fettke and P. Loos. Classification of Reference Models: A Methodology and its Application. *Information Systems and e-Business Management*, 1(1): 35–53, 2003.
8. U. Frank. Conceptual Modelling as the Core of the Information Systems Discipline. In *Proceedings of the AMICS 1999*, pp. 695–698, Milwaukee, 1999.
9. G. Keller, M. Nüttgens, and A. W. Scheer. Semantische Prozessmodellierung auf der Grundlage Ereignisgesteuerter Prozessketten (EPK). Veröffentlichungen des Instituts für Wirtschaftsinformatik, University of Saarland, Saarbrücken, 1992.
10. J. Mendling, J. Recker, M. Rosemann, and W. M. P. van der Aalst. Towards the Interchange of Configurable EPCs. In *EMISA 2005*, pp. 8–21, 2005.
11. J. Mendling, J. Recker, M. Rosemann, and W. M. P. van der Aalst. Generating Correct EPCs from Configured C-EPCs. In *SAC 2006*, pp. 1505–1510, 2006.
12. K. Morton, C. Carey-Smith, and K. Carey-Smith. The QUEST Questionnaire System. In *Proceedings of the 2nd ANNES*, pp. 214–217. IEEE Computer Society, 1995.
13. I. Reinhartz-Berger, P. Soffer, and A. Sturm. A Domain Engineering Approach to Specifying and Applying Reference Models. In J. Desel and U. Frank, editors, *Workshop Enterprise Modelling and Information Systems Architectures*, vol. 75 of *LNI*, pp. 50–63. German Informatics Society, 2005.
14. M. La Rosa, W.M.P. van der Aalst, M. Dumas, and A.H.M. ter Hofstede. Generating Interactive Questionnaires from Configuration Models. 2006. QUT ePrints, <http://eprints.qut.edu.au/archive/00005284/01/5284.pdf>.
15. M. Rosemann and W. M. P van der Aalst. A Configurable Reference Modelling Language. *Information Systems*, 32(1): 1–23, 2007.
16. A.-W. Scheer and M. Nüttgens. ARIS Architecture and Reference Models for Business Process Management. In Wil M. P. van der Aalst, J. Desel, and A. Oberweis, editors, *Business Process Management*, vol. 1806 of LNCS, pp. 376–389. Springer, Berlin, Heidelberg, 2000.
17. S. Seidel, M. Rosemann, A.H.M. ter Hofstede, and L. Bradford. Developing a Business Process Reference Model for the Screen Business - A Design Science Research Case Study. In *17th ACIS*, Adelaide, 2006. www.screenbusiness.org.
18. P. Soffer, B. Golany, and D. Dori. ERP Modeling: A Comprehensive Approach. *Information Systems*, 28(6): 673–690, 2003.
19. S. Stephens. The Supply Chain Council and the SCOR Reference Model. *Supply Chain Management - An International Journal*, 1(1):9–13, 2001.
20. C. Taylor and C. Probst. Business Process Reference Model Languages: Experiences from BPI Projects. In *Proceedings of INFORMATIK 2003*, pp. 259–263. Jahrestagung der Gesellschaft für Informatik e. V. (GI), 2003.