2008

Towards an Enterprise Business Process Architecture standard

George Koliadis
University of Wollongong, gk56@uowmail.edu.au

Aditya K. Ghose
University of Wollongong, aditya@uow.edu.au

Srinivas Padmanabhuni
Infosys Technologies Ltd, India

Towards an Enterprise Business Process Architecture standard

Abstract

An effective process architecture helps provide a high-level blueprint of the complexity underlying an enterprise, which is used by executive committees during key decision and change processes. As existing service standards focus on co-ordination, they fall short in describing the motivational structure depicted in such models. In order to progress towards standardization in this area of complexity, we discuss the practicality of a process architecture, and present a set of 22 questions that can be used in the functional evaluation and construction of a process architecture. We use these questions to evaluate the current "state-of-the-art" in business process architecture. We then apply the knowledge gathered during our evaluation and other work to develop the proposal for a general mapping framework that is capable of answering the set of queries we have proposed.

Disciplines

Physical Sciences and Mathematics

Publication Details

Koliadis, G., Ghose, A. K. & Padmanabhuni, S. (2008). Towards an Enterprise Business Process Architecture standard. IEEE Congress on Services (pp. 239-246). USA: IEEE.

Towards an Enterprise Business Process Architecture Standard

George Koliadis and Aditya K. Ghose
Decision Systems Laboratory
School of C.S. and Software Engineering
University of Wollongong, Australia
Email: {gk56, aditya}@uow.edu.au

Srinivas Padmanabhuni
Software Engineering and Technology Labs
Infosys Technologies Ltd
Bangalore, India
Email: srinivas_p@infosys.com

Abstract

An effective process architecture helps provide a high-level blueprint of the complexity underlying an enterprise, which is used by executive committees during key decision and change processes. As existing service standards focus on co-ordination, they fall short in describing the motivational structure depicted in such models. In order to progress towards standardization in this area of complexity, we discuss the practicality of a process architecture, and present a set of 22 questions that can be used in the functional evaluation and construction of a process architecture. We use these questions to evaluate the current "state-of-the-art" in business process architecture. We then apply the knowledge gathered during our evaluation and other work to develop the proposal for a general mapping framework that is capable of answering the set of queries we have proposed.

1 Introduction

Architecture is an engineering activity, aimed at constructing an *Architectural Model* for the purposes of [8]: (1) *establishing a shared understanding* of a system from a high, abstract level; (2) *harvesting component reuse* during evolution and across projects; (3) *constructing implementations* and more detailed refinements; (4) *evolving system structure* supported by a separation between functionality and interconnection; (5) *analyzing high-level design* w.r.t. consistency constraints and quality attributes; and (6) *managing complexity* in large-scale projects and enterprises. During the management of *Business Processes* [21], the importance of architecture is emphasized as core to analyzing and improving value creation and organizational performance [14]. However, a common and practical standard for approaching *Enterprise Business Process Architecture*, or EBPA (which we will occasionally refer to as an enterprise process architecture) is still elusive [12]. In this paper, we outline the functional requirements of a *useful* and *usable* EBPA standard through a list of 22 questions the we, and others [6] [13], believe most analysts and executives ask. Using this set of competence criteria, we evaluate a set of 10 existing process architecture frameworks. This evaluation reveals highly variable coverage and support for these requirements across the frameworks analyzed, pointing to the need for a comprehensive and integrated approach to EBPA. The remainder of the paper describes such an approach. We build on the fundamental premise that an EBPA must involve the correlation of an enterprise model with a set of process models. We use a simplified variant of the i^* framework [23] that permits us to build *enterprise context maps* (in a manner akin to i^* Strategic Dependency diagrams) as well as *enterprise capability maps* (in a manner akin to i^* Strategic Rationale diagrams). Business processes are represented in the BPMN notation. We show how the enterprise model serves as a scaffolding with which processes are correlated, thus surfacing a rich repertoire of insights relevant to EBPA (such as process ownership, cross-actor workflows, inter-relationships between processes, redundancies, unsupported objectives, amongst many others). We also provide methodological guidelines for the building and maintenance of EBPAs. This paper aims to fuel progress on a proposal (in the tradition of the WS-ARCH specification [2]) for an EBPA standard.

2 Functional Evaluation of State-of-the-Art EBPA Frameworks

One of the oldest and most common models advocated for expressing process architecture is the *Value Chain* (VC) [17] applied in many operational reference frameworks such as the *Value Chain Operational Ref-*

978-0-7695-3286-8/08 $25.00 © 2008 IEEE
DOI 10.1109/SERVICES-1.2008.60

erence (VCOR) [20], and extended to a *Value Network* (VN) in [1]. More recently, the notion of a *Strategy Map* (SM)[15] has been presented and a model for *Business Motivation* (BMM - with inter-process relationships) [3] drafted. In addition, ARIS defines a *House of Business Engineering* (HOBE) architecture [18] [19], a *Business Process Architecture Framework* (BPAF) is outlined in [14], and an approach for structuring *Role-Activity Diagrams* (SRAD) is described in [16]. Finally, some prominent frameworks developed in research institutions include: *i** (I*) signifying distributed intentionality) [23]; *e3 Value* (E3) [9]; the *Semantic Object Model* (SOM) [4]; and *Toronto Virtual Enterprise Ontology* (TOVE) [5].

Evaluation Criteria: Within the literature, two prominent techniques have been proposed for evaluating the scope and completeness of an enterprise modeling framework. The first involves the use of an upper-ontology (e.g. BWW [22]) such as in [11], and the second advocates the use of *competency questions* [13] [6], which test the functional completeness (and practicality) of a framework by evaluating support for abstract queries on a possible model instance. The use of an upper ontology in our case is difficult for two reasons: (1) we are working in a very specific domain (i.e. EBPA) where not all concepts may be applicable [10]; (2) some of the frameworks we are evaluating are informally defined. Therefore, a query-based approach will ideally result in a more focused evaluation (but possibly at the expense of completeness). We outline below queries that we, and others [6] [13], believe should ideally be answered by a practical EBPA model. Where relevant, we describe how each of the aforementioned frameworks evaluate against each query using the following scale:

- '++': complete support for answering the query;
- '+': in-direct or limited support, indicating possible completeness issues; and,
- '0': no support for answering the query.

Enterprise Structure:
Q1 - Who are the key enterprise actors (inc. agents, roles, positions)? (++) BMM, HOBE, BPAF, I*, E3, SOM, TOVE; (+) VC, SRAD; (0) SM.
Q2 - What are the structural relationships between enterprise actors (inc. part-of, member-of, is-a)? (++) HOBE, BPAF, E3, SOM, TOVE; (0) VC, SM, BMM, SRAD, I*.
Enterprise Motivation:
Q3 - What are the enterprise-level goals (inc. mission, vision, strategic statements? (++) SM, BMM, HOBE, BPAF, I*, SOM, TOVE; (+) VC, SRAD; (0) E3.
Q4 - How are enterprise-level goals refined into sub-goals (decomposition, alternatives)? (++) I*, TOVE; (+) SM, BMM, HOBE; (0) BPAF, SOM, VC, SRAD, E3.
Q5 - What are the actor level goals? (++) I*, SOM, TOVE; (+) VC, BMM, HOBE, BPAF, SRAD; (0) SM, E3.
Q6 - How do actor level goals support enterprise-level goals? (++) I*, SOM, TOVE; (+) HOBE; (0) VC, BMM, SM, E3, BPAF, SRAD.
Q7 - What are the performance objectives? (++) SM, BMM, HOBE, BPAF, I*, TOVE; (+) VC; (0) SOM, E3, SRAD.
Enterprise Capability:
Q8 - What business processes does the enterprise support? (++) BMM, HOBE, BPAF, I*, TOVE, VC, SOM, E3, SRAD; (+) SM.
Q9 - What business processes do specific enterprise actors support? (++) VC, BMM, HOBE, BPAF, I*, E3, TOVE, SOM; (+) SRAD; (0) SM.
Q10 - What resources do enterprise actors have to participate in each business process? (++) HOBE, SRAD, I*, E3, TOVE, SOM; (0) BPAF, VC, BMM, SM.
Q11 - What are the hierarchical relationships between these business processes? (++) VC, HOBE, BPAF, I*, SOM; (+) SRAD; (0) E3, TOVE, BMM, SM.
Q12 - What business processes does the enterprise have and want? (++) SM, BPAF, SRAD, I*, E3; (0) SOM, VC, HOBE, TOVE, BMM.
Q13 - What are the gaps in the process architecture, i.e., what are the enterprise/actor functionalities that are not supported by corresponding business processes? (++) I*; (0) SM, BPAF, SRAD, E3, SOM, VC, HOBE, TOVE, BMM.
Q14 - What business processes are redundant within the enterprise (i.e., existing processes that do not contribute to any objective/functionality)? (++) I*; (0) SM, BPAF, SRAD, E3, SOM, VC, HOBE, TOVE, BMM.
Enterprise Relationships:
Q15 - How do enterprise actors relate to each other at a functional (as opposed to structural) level (e.g., in relation to tasks, outcomes, resources, quality-of-service)? (++) I*, E3, SOM, TOVE; (+) VC, HOBE, SRAD; (0) SM, BPAF, BMM.
Q16 - Does an enterprise actor depend entirely or partially on another actor to support certain business processes? (++) I*; (0) SM, BPAF, SRAD, E3, SOM, VC, HOBE, TOVE, BMM.
Enterprise Risk
Q17 - What are the critical business processes? (++) SM, BMM, BPAF, I*, E3; (0) SRAD, SOM, VC, HOBE, TOVE. Intuitively, a process is more critical

than another if the failure of the former causes greater disruption than the latter. A similar intuition applies to the notions of critical actors and relationships discussed below.

Q18 - What are the vulnerable business processes? (++) BMM; (0) I*, SM, BPAF, SRAD, E3, SOM, VC, HOBE, TOVE. Intuitively, a process is more vulnerable than another if there greater risk of failure associated with the former than the latter. A similar intuition applies to the notions of vulnerable actors and relationships discussed below.

Q19 - Who are the critical enterprise actors? (++) BMM, E3; (+) BPAF, I*; (0) SM, SRAD, SOM, VC, HOBE, TOVE.

Q20 - What are the critical relationships between enterprise actors? (++) BMM, I*, E3; (0) BPAF, SM, SRAD, SOM, VC, HOBE, TOVE.

Q21 - Who are the vulnerable enterprise actors? (++) BMM; (+) I*; (0) E3, BPAF, SM, SRAD, SOM, VC, HOBE, TOVE.

Q22 - What are the vulnerable relationships between enterprise actors? (++) BMM, I*; (0) E3, BPAF, SM, SRAD, SOM, VC, HOBE, TOVE.

In our evaluation, we found that most frameworks were able to capture knowledge relevant to understanding the following: the constituent actors within an enterprise; enterprise level goals; actor level goals; performance objectives; and enterprise and actor level processes. On the other hand, we found the surveyed frameworks to be deficient for answering the following types of queries: how actor level goals supported enterprise level goals; the capabilities an enterprise does not have, but would like to; the capabilities an enterprise has and does not want to manage; the capabilities an enterprise has and does not want at all; the level of dependence between enterprise actors; and, the business processes that may be vulnerable to failure.

3 Enterprise Mapping with Annotated *i** and BPMN Models

In many frameworks (e.g. VC, SM, I*, E3), satisfaction of commitments/obligations is a key performance indicator, framed as a basis for aligning corporate objectives and capabilities. Therefore, we model an enterprise context map as a set of directed and labeled *Service Relationships* that exist between an organization and external actors including customers, intermediaries and even regulatory agencies as in the example (Figure 1) below. Such contextual/contractual service relationships must be maintained, through enterprise capabilities (i.e. enterprise processes), in order to ensure a stable enterprise process architecture.

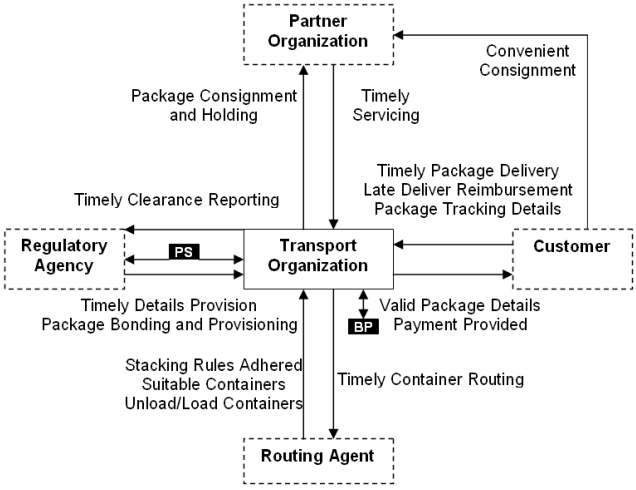

Figure 1. Greater Enterprise Context

In our evaluation, weve found the *i** Framework [23] useful for answering many interesting questions. We will be using portions of *i** as the basis for illustrating a novel approach to enterprise process architecture, which fills many of the gaps in existing approaches. We describe process architecture in two dimensions (1) *Enterprise Context* and (2) *Enterprise Capability*. Below, we illustrate and discuss enterprise context and capability maps for a hypothetical Transport Organization.

Enterprise Context: In the example described in Figure 1, a "Transport Organization" maintains service relationships with its "Customers", "Routing Agents", "Partner Organizations" and "Regulatory Agencies". Service relationships are modeled as arrows pointing from a service requester/consumer to a service provider. For example, the "Transport Organization" (as a requester) requests "Timely Clearance Reporting" (a performance objective) from a "Regulatory Agency" (as a provider). In turn, the "Regulatory Agency" requests "Timely Details Provisioning and Package Bonding/Provisioning" (performance and functional objectives) from the "Transport Organization". Process identifiers appear as dark boxes in the figure - these will be discussed later.

Once modelled graphically, these relationships can be elaborated on using a template that includes fields for describing:

(1) The conditions leading to the *fulfillment* of the service relationship (e.g. Regulatory Authority knows the source and destination address of each Package received by the Transport Organization for a functional objective, or Clearance reports have been received no later than one hour prior to Container Routing for a perfor-

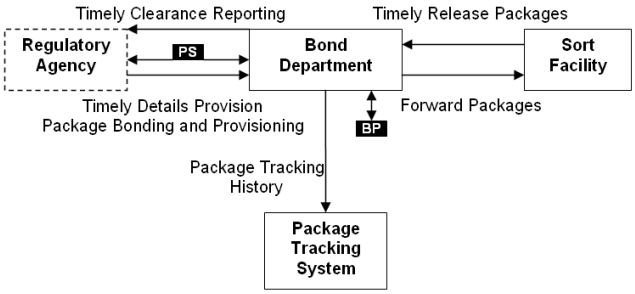

Figure 2. Sub-Enterprise Context

mance objective);
(2) The specific enterprise processes that *require the service relationships to be fulfilled* (e.g. the Regulatory Agency customs clearance process requires Package Bonding and Provisioning to occur);
(3) The specific enterprise processes *required to fulfil service relationships* by a provider (e.g. the Package Screening process is required to meet Package Bonding and Provisioning obligations);
(4) The *risks* (likelihood, impact, mitigation) that could lead to service relationship failure (e.g. Network Outage could lead to a violation of the Timely Clearance Reporting objective).

In addition to modelling the greater context of the organization in relation to external actors, the delegation of each service relationship to internal organizational roles/actors can be modelled using the same notation at a deeper level and between internal roles and/or functional units (e.g. indicating that Bond Department will be responsible for Package Bonding and Provisioning). Such models (e.g. Figure 2) indicate the configuration of enterprise actors required to meet contextual service relationships or obligations.

Enterprise Capability: An enterprise capability map lists the *Service Outcomes* that an actor is capable of fulfilling. Service outcomes are local to a specific actor and may be aggregated into composite outcomes, or refined into alternative outcomes that may fulfill a similar purpose. Figure 3 lists the composite and alternative service outcomes that lead to the "Bond Department" achieving the service outcome "Bonding Duties Handled". Note that our use of the notion of service outcome unifies the distinct notions of "goal"' and "task"' used in the i^* notation (3 is also an augmented version of an i^* SR diagram).

Once listed, each service outcome can be elaborated in a similar way to service relationships in the previous section. For example, describing the service relationships a service outcome is required for, and the service relationships it requires (e.g. "The Bonding

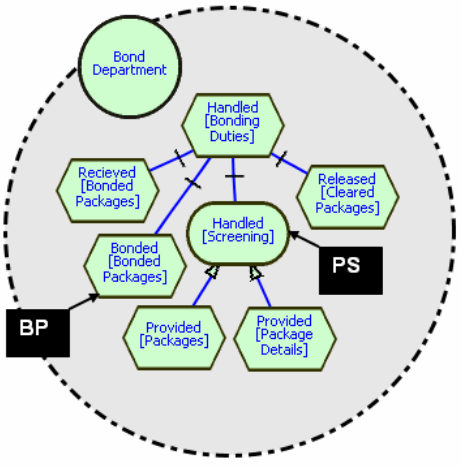

Figure 3. Enterprise Capability

Duties Handled composite service outcome is required for the Bonding and Provisioning relationship that is maintained with the Regulatory Authority"). Other fields of interest would also include the conditions leading the fulfil.

Business Process Modeling with Annotated BPMN

An *annotated BPMN model*, for the purposes of this paper, is one in which every task (atomic, loop, compensatory or multi-instance) and every sub-process has been annotated with descriptions of its immediate effects. We evaluate the relationship of nodes on a process model with service outcomes on an enterprise capability map. However, before doing this, we require that an analyst accumulates effects throughout the process to provide a local in-context description of the *cumulative effect* at task nodes in the process.

We define a process for *pair-wise effect accumulation*, which, given an ordered pair of tasks with effect annotations, determines the cumulative effect after both tasks have been executed in contiguous sequence. The procedure serves as a methodology for analysts to follow if only informal annotations are available. We assume that the effect annotations have been represented in conjunctive normal form or CNF. Simple techniques exist for translating arbitrary sentences into the conjunctive normal form.

Contiguous Tasks: Let $\langle t_i, t_j \rangle$ be the ordered pair of tasks, and let e_i and e_j be the corresponding pair of (immediate) effect annotations. Let $e_i = \{c_{i1}, c_{i2}, \ldots, c_{im}\}$ and $e_j = \{c_{j1}, c_{j2}, \ldots, c_{jn}\}$ (we can view CNF sentences as sets of clauses, without loss of generality). If $e_i \cup e_j$ is consistent, then the re-

sulting cumulative effect is $e_i \cup e_j$. Else, we define $e'_i = \{c_k | c_k \in e_i$ and $\{c_k\} \cup e_j$ is consistent$\}$ and the resulting cumulative effect to be $e'_i \cup e_j$. In other words, the cumulative effect of the two tasks consists of the effects of the second task plus as many of the effects of the first task as can be consistently included. We remove those clauses in the effect annotation of the first task that contradict the effects of the second task. The remaining clauses are undone, i.e., these effects are overridden by the second task. In the following, we shall use $acc(e_1, e_2)$ to denote the result of pair-wise effect accumulation of two contiguous tasks t_1 and t_2 with (immediate) effects e_1 and e_2.

Effects are only accumulated within participant lanes. In addition to the effect annotation of each task, we annotate each task t with a cumulative effect E_t. E_t is defined as a set $\{es_1, es_2, \ldots, es_p\}$ of alternative *effect scenarios*. Alternative effect scenarios are introduced by OR-joins or XOR-joins, as we shall see below. Cumulative effect annotation involves a left-to-right pass through a participant lane. Tasks which are not connected to any preceding task via a control flow link are annotated with the cumulative effect $\{e\}$ where e is the immediate effect of the task in question. We accumulate effects through a left-to-right pass of a participant lane, applying the pair-wise effect accumulation procedure on contiguous pairs of tasks connected via control flow links. The process continues without modification over splits. Joins require special consideration. In the following, we describe the procedure to be followed in the case of 2-way joins only, for brevity. The procedure generalizes in a straightforward manner for n-way joins.

AND-joins: Let t_1 and t_2 be the two tasks immediately preceding an AND-join. Let their cumulative effect annotations be $E_1 = \{es_{11}, es_{12}, \ldots, es_{1m}\}$ and $E_2 = \{es_{21}, es_{22}, \ldots, es_{2n}\}$ respectively (where es_{ts} denotes an effect scenario, subscript s within the cumulative effect of some task, subscript t). Let e be the immediate effect annotation, and E the cumulative effect annotation of a task t immediately following the AND-join. We define $E = \{acc(es_{1i}, e) \cup acc(es_{2j}, e) | es_{1i} \in E_1$ and $es_{2j} \in E_2\}$. Note that we do not consider the possibility of a pair of effect scenarios es_{1i} and es_{2j} being inconsistent, since this would only happen in the case of intrinsically and obviously erroneously constructed process models. The result of effect accumulation in the setting described here is denoted by $ANDacc(E_1, E_2, e)$.

XOR-joins: Let t_1 and t_2 be the two tasks immediately preceding an XOR-join. Let their cumulative effect annotations be $E_1 = \{es_{11}, es_{12}, \ldots, es_{1m}\}$ and $E_2 = \{es_{21}, es_{22}, \ldots, es_{2n}\}$ respectively. Let e be the immediate effect annotation, and E the cumulative effect annotation of a task t immediately following the XOR-join. We define $E = \{acc(es_i, e) | es_i \in E_1$ or $es_i \in E_2\}$. The result of effect accumulation in the setting described here is denoted by $XORacc(E_1, E_2, e)$.

OR-joins: Let t_1 and t_2 be the two tasks immediately preceding an OR-join. Let their cumulative effect annotations be $E_1 = \{es_{11}, es_{12}, \ldots, es_{1m}\}$ and $E_2 = \{es_{21}, es_{22}, \ldots, es_{2n}\}$ respectively. Let e be the immediate effect annotation, and E the cumulative effect annotation of a task t immediately following the OR-join. The result of effect accumulation in the setting described here is denoted by $ORacc(E_1, E_2, e) = ANDacc(E_1, E_2, e) \cup XORacc(E_1, E_2, e)$.

We note that the procedure described above does not satisfactorily deal with loops, but we can perform approximate checking by partial loop unraveling. We also note that some of the effect scenarios generated might be infeasible. Our objective is to devise decision-support functionality in the compliance management space, with human analysts vetting key changes before they are deployed.

4 Building Enterprise Process Architectures

An enterprise process architecture relies on a *portfolio* of processes and an *enterprise model* as the initial inputs. As the preceding discussion suggests, the crux to building and maintaining enterprise process architectures is the exercise of relating processes to enterprise models. At a more fundamental level, processes need to be related to enterprise capabilities, given that a capability is either realized by a process, or contributes to a process (the difference stemming from the level of abstraction at which the capabilities and processes in question have been modelled). We establish that relationship via the cumulative effects of processes (as discussed above) and descriptions of service outcomes. These can be separately specified, or obtained from the relationships (i^* dependencies) that these outcomes participate in. In the FormalTROPOS [7] framework, every dependency is annotated with a creation, invariant and fulfillment condition. Fulfillment conditions can form the basis of descriptions of service outcomes (i^* tasks/goals) on the dependee side of a dependency. The challenge now is to devise means for relating processes to service outcomes. We will consider two approaches. The first involves labeling *normative realization links* specified by analysts with a repertoire of labels that denote varying degrees of confidence in the normative link being an accurate description of the ac-

tual state of affairs. The second approach involves an analyst-mediated exercise in top-down refinement of service outcomes, coupled with a search procedure that seeks out combination of processes from an existing portfolio of processes that might achieve a service outcome.

Building an EBPA via realization links: A normative realization link established by an analyst between a process and a service outcome represents what might be a viewed a good guess by the analyst that the service outcome in question is indeed realized by the process in question. Such a normative statement must then be verified via reference to service outcome descriptions and process effects. In the following, we assume that there is a single description of each service outcome, while the cumulative effects of a process might be described via a set of mutually exclusive effect scenarios (note that this approach can be easily generalized to admit multiple mutually exclusive service outcome descriptions as well). Each normative realization link can be assigned one of the following labels via a process of analysis:

Strongly inconsistent: All effect scenarios of the process are inconsistent with the service outcome description.

Weakly inconsistent: Some, but not all, effect scenarios of the process are inconsistent with the service outcome description.

Consistent but unrealized: All effect scenarios of the process are consistent with the service outcome description, but none entail the service outcome description.

Weakly realized: Some, but not all, effect scenarios of the process entail the service outcome description. In other words, some instances of the process will lead to the service outcome being achieved.

Strongly realized: All effect scenarios of the process entail the service outcome description. In other words, some instances of the process will lead to the service outcome being achieved.

The list of labels above describe a spectrum of levels of confidence in the normative realization link describing the true state of affairs. At one extreme, the strongly inconsistent label suggests that the realization link is definitively incorrect. The weakly inconsistent label suggests that the link is very likely to be incorrect, but leaves open the possibility that some instances might potentially be modified to realize the service outcome. The "consistent but unrealized"' label suggests a neutral stance - while the process in question might be modified to obtain a realization of the service outcome in question, no evidence exists to suggest that it does realize the outcome in its current form. The weakly realized and strongly realized labels indicate progressively higher degrees of confidence that the process does realize the service outcome. The exercise of labeling realization links leads ultimately to an enterprise architecture represented via enterprise capability and enterprise context maps annotated with process identifiers. For instance, Figure 2 represents the Package Screening process as being realized via the interoperation of the Bond Department actor and the Regulatory Agency. The Bond Package process is represented as being realized entirely by the Bond Department actor. The enterprise capability map represented in Figure 3 shows the specific service outcomes within the Bond Department actor that are realized by these two processes.

A top-down methodology for EBPA: An enterprise process architecture might contain unrealized service outcomes, i.e., outcomes which are not supported by underlying processes. Such an unrealized outcome might represent an enterprise goal that is actually unrealized, or might be an artefact of an inadequate understanding of the EBPA. We outline below a methodology that can support analysts in identifying existing processes that might be used, either singly, or in a combined fashion, to support these service outcomes. This methodology also forms the basis for incrementally constructing EBPAs, by starting with a process portfolio and an EBPA containing unrealized service outcomes, which are progressively related to sets of processes. Th methodology involves three high-level steps: (1) *outcome realization via process composition*, (2) *service outcome decomposition* and (3)*process refinement*. In the first step, we iterate from $k = 1$ to n where n is the number of distinct processes in the process portfolio. In the initial iteration (with $k = 1$), we seek any single process such that all of its (cumulative) final effect scenarios entail the service outcome description in question. If no such process exists, in the next iteration, we seek (sequential) compositions of 2 processes such that all of the final effect scenarios entail the service outcome description. Thus, in the k-th iteration, we seek compositions of k processes such that all the effect scenarios entail the service outcome description. In our description of this methodology, we focus on sequential process compositions. More complex process compositions, via a variety of (AND, OR, XOR) splits and joins can be dealt with, but we omit these details here for brevity. This exercise might be impeded, however, by abstraction mis-matches, i.e., we might not be able to obtain entailment relationships between process effect scenarios and service outcome descriptions because these might have used disjoint vocabularies at different levels of abstraction. We may therefore have to interleave the *service outcome decomposition* step and the *process refinement* steps eventually obtain in-

tersecting (if not identical) vocabularies. We note that this methodology can also guide analysts in identifying opportunities for process reuse in supporting newly introduced service outcomes. We omit these details here for brevity. This exercise might be impeded, however, by abstraction mis-matches, i.e., we might not be able to obtain entailment relationships between process effect scenarios and service outcome descriptions because these might have used disjoint vocabularies at different levels of abstraction. We may therefore have to interleave the *service outcome decomposition* step and the *process refinement* steps eventually obtain intersecting (if not identical) vocabularies. We note that this methodology can also guide analysts in identifying opportunities for process reuse in supporting newly introduced service outcomes.

Maintaining an EBPA: A variety of changes can impact an EBPA. Enterprise actors, service outcomes and business processes might be added or removed. Service relationships might be modified. Maintaining an EBPA in the face of these changes involves ensuring that the following two constraints continue to be satisfied: a *realization* constraint for every service outcome (which ensures that every service outcome is realized via some existing process) and a *non-redundancy* constraint for every process in the process portfolio (which ensures that no process is supported that does not help realize some service outcome/objective). An EBPA that satisfies these constraints, for each service outcome and process respectively, may be viewed as a *stable* EBPA. If a service outcome is identified that violates the realization constraint, we need to design and deploy a process or processes (either from scratch, or by re-using existing processes) that realize the outcome in question. If a process is found to be redundant, it must be deleted.

Competence of Enterprise Mapping

We discuss below how our framework performs in relation the key questions listed earlier. Enterprise actors (Q1) are the focus of both maps, and their structural relationships (Q2) are represented during the refinement of the enterprise motivation map from context to configuration and within the description of an enterprise capability map. Enterprise-level goals/objectives (Q3) are represented as service relationships among and between internal and external actors on and enterprise motivation map. Their refinement (Q4) occurs within the enterprise capability map as actor-level service outcomes (Q5, Q6). Both service relationships and outcomes may refer to performance objectives (Q7). Coarse-grained enterprise capabilities are represented within the motivation map as service relationships between internal and external actors (Q8), and actor level

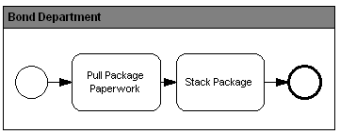

Figure 4. A Bond Package Process(BP)

functionalities are mapped onto the capability map (Q9). Resources (Q10) may be annotated to service outcomes in a capability map. Service outcomes are hierarchically structured (Q11) in an enterprise capability map, leading to corresponding hierarchic relationships between the supporting business processes. Correlating processes to enterprise context and capability maps, as shown in Figures 1, 2 and 3, helps answer questions Q12 through Q14. For instance, it is easy to determine which processes in a process portfolio support a given service outcome (Q12) a well as which processes do not participate in supporting any service outcome (Q14). We may view a service outcome as *supported* if and only if at least one of the following hold: (1) the service outcome is directly correlated with a process, (2) the service outcome participates in a service relationship as a requester, such that the outcome at the provider end of the relationship is supported, (3) the service outcome is OR-decomposed into a set of finer-grained outcomes of which at least one is supported and (4) the service outcome is AND-decomposed into a set of finer-grained outcomes which are all supported. Identifying the set of *unsupported* outcomes helps answer Q13. Functional dependence (Q15) and the level of dependence (Q16) between actors is explicit within an enterprise motivation map in correlation with an enterprise capability map. Critical (Q17) and vulnerable (Q18) business processes can be determined by analysing the level of functional dependence for a service outcome, or via additional risk related annotations. Some simple metrics can be used. For instance, we might deem an actor to be more critical than another if the former plays the role of provder in a greater number of service relationships than the latter (Q19). Similarly, an actor might be deemed to be more vulnerable than another if the former plays the role of requester in a greater number of relationships than the latter (Q21). Criticality and vulnerability of relationships (Q20, Q22) can be determined by following a chain of relationships and incoporating the criticality (resp. vulnerability) measures of the actors in the chain (details omitted for brevity). Processes can also inherit criticality (Q17) and vulnerability (Q18) measures from the actors and relationships they support (details again omitted for brevity).

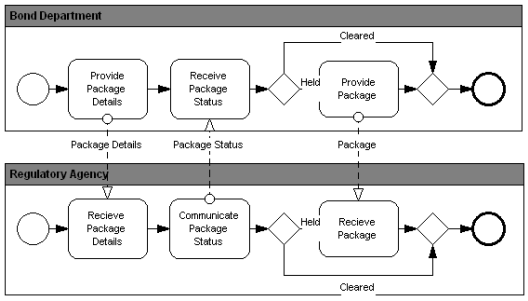

Figure 5. A Package Screening Process(PS)

5 Discussion and Conclusion

In this report, weve provided some meaningful questions to help assess the practicality of an process architecture standard. We have surveyed a sample of architecture frameworks to better understand how support for our questions is provided. We have also illustrated a simple framework for mapping an enterprise process architecture that conforms to our analysis criteria.

References

[1] V. Allee. *The Future of Knowledge: Increasing Prosperity through Value Networks*. Butterworth-Heinemann, 2002.

[2] D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris, and D. Orchard. Web services architecture (ws-arch) - w3c working group note. Technical report, World Wide Web Consortium (W3C), 11 February 2004.

[3] BRG. The business motivation model, release 1.2. Technical report, Business Rules Group, http://www.businessrulesgroup.org/bmm.shtml, 2005.

[4] O. K. Ferstl and E. J. Sinz. Modeling of business systems using the semantic object model - a methodological framework. Technical report, Otto-Friedrich-Universitat Bamberg, 1997.

[5] M. S. Fox. The tove project towards a common-sense model of the enterprise. In *IEA/AIE '92: Proceedings of the 5th international conference on Industrial and engineering applications of artificial intelligence and expert systems*, pages 25–34, London, UK, 1992. Springer-Verlag.

[6] M. S. Fox, M. Barbuceanu, M. Gruninger, and J. Lin. An organizational ontology for enterprise modeling. *Simulating Organizations: Computational Models of Institutions and Groups*, pages 131–152, 1998.

[7] A. Fuxman, L. Liu, J. Mylopoulos, M. Pistore, M. Roveri, and P. Traverso. Specifying and analyzing early requirements in tropos. *Requirements Engineering*, 9(2):132150, 2004.

[8] D. Garlan. Software architecture: a roadmap. In A. Finkelstein, editor, *The Future of Software Engineering*. ACM Press, 2000.

[9] J. Gordijn. *Value-based Requirements Engineering: Exploring Innovative e-commerce Ideas*. PhD thesis, Vrije Universiteit Amsterdam, 2002.

[10] P. Green, M. Indulska, and M. Rosemann. A reference methodology for conducting ontological analyses. In *Proceedings of the International Conference on Conceptual Modelling (ER 2004)*, pages 8–11, 2004.

[11] P. Green, M. Rosemann, and M. Weske. Integrated process modeling: An ontological evaluation. *Information Systems*, 25(2):73–87, 2000.

[12] S. Green and M. A. Ould. The primacy of process architecture. In *CAiSE'04 Workshops in connection with The 16th Conference on Advanced Information Systems Engineering, Riga, Latvia, 7-11 June, 2004, Knowledge and Model Driven Information Systems Engineering for Networked Organisations, Proceedings, Vol. 2*, pages 154–159, 2004.

[13] M. Gruninger and M. S. Fox. The role of competency questions in enterprise engineering. In *IFIP WG5.7 Workshop on Benchmarking - Theory and Practice*, 1994.

[14] P. Harmon. *Business Process Change*. Morgan Kaufmann, San Francisco, CA, 2003.

[15] R. S. Kaplan and D. P. Norton. *Strategy Maps: Converting Intangible Assets into Tangible Outcomes*. Harvard Business School Press, 2004.

[16] M. A. Ould. *Business Process Management: A Rigorous Approach*. Meghan-Kiffer Press, 2005.

[17] M. E. Porter. *Competitive Advantage: Creating and Sustaining Superior Performance*. Free Press, 1998.

[18] A.-W. Scheer and M. Nuttgens. Aris architecture and reference models for business process management. In *Business Process Management, Models, Techniques, and Empirical Studies*, pages 376–389, London, UK, 2000. Springer-Verlag.

[19] A. W. Sheer. *ARIS - Business Process Frameworks*. Springer, 1999.

[20] I. The Value Chain Group. The value chain operational reference model - vcor. Technical report, http://www.value-chain.org/, 2007.

[21] W. van der Aalst, A. ter Hofstede, and M. Weske. Business process management: A survey. In *BPM'03 - International Conference on Business Process Management*, pages 1–12, Berlin, 2003. Springer-Verlag, Lecture Notes in Computer Science.

[22] Y. Wand and R. Weber. An ontological model of an information system. *IEEE Transactions on Software Engineering*, 16(11):1282 – 1292, 1990.

[23] E. Yu. *Modelling Strategic Relationships for Process Reengineering*. PhD thesis, Graduate Department of Computer Science, University of Toronto, Toronto, 1995.