Business Process Integration: Method and Analysis

Evan D. Morrison Alex Menzies George Koliadis Aditya K. Ghose

Decision Systems Lab
School of Computer Science and Software Engineering
University of Wollongong,
Wollongong NSW 2522, Australia,
Email: {edm92, am57, gk56, aditya}@uow.edu.au

Abstract

In the study of business management, process integration has become an interesting area of research that affects analysts studying and working on existing system plans. Process integration aims to investigate relationships across a business compendium to produce classifications and merge similar activities into a standardized system. Integration is the process of merging elements from two similar antecedent processes to create a single process that can be used to replace the original processes. This paper proposes a practical method for process integration and provides a theoretical framework and metrics for business process integration assessment. In the provision of metrics that take into account similarity of activities within processes we are able to offer solutions that provide minimal change *reducing change costs, and minimizing change impact risks.*

Keywords: BPMN, Process Integration, Similarity Matching, SPNets

1 Introduction

The problem of *business process integration* is ubiquitous is a wide variety of domains. Consider, for example, a back-office financial service provider that serves a range of different pension funds. The service provider must support processes for clients (individuals) to be added to the system, new employers to be added to the system, consolidation of pension accounts in various funds into a single account etc. The provider must support different versions of these processes for the different pension funds that it serves, since each fund requires its own process to be followed in each instance. The provider notes that a given process (say that for changing client addresses) varies minimally from fund to fund, and seeks to 'rationalize' these variants to obtain a single process. The problem is one of business process integration, where the fund-specific variants are combined into a single process that achieves the goals of *all* of the original processes.

Consider another example where a smaller insurance company is acquired by a larger insurance company. There is a need for the resulting entity to support a single claims handling process, which requires that the claims handling processes of both the acquiring and acquired companies be integrated into a single consolidated process that achieves the goals/objectives of both prior processes.

In both examples, there would be an implicit requirement that the consolidated process be as 'close' or as 'similar' as possible to the original processes in order to minimize disruption and to protect investments in existing process infrastructure. The business process integration problem can thus be viewed as the problem of identifying a single process that:

1. Achieves all of the goals/objectives of a set of prior processes while
2. Minimizing the extent of change required to the original processes.

The first requirement involves identifying a new goal that the integrated process must achieve, which can be a simple matter of taking the conjunction of the goals of the prior processes (we do not address more complex questions concerning goal merging when the goals are inconsistent, although this is an important problem). The second requirement involves assessing, and ideally measuring, the extent of change affected by a process variant relative to the original process.

We present a novel approach to business process integration that relies on a set of *process proximity metrics*. We assume that processes are represented in the industry-standard BPMN notation. BPMN provides little support for representing the semantics of the processes being modeled (beyond the nomenclature of the tasks involved, and the conditions that label decision gateways). We first describe an approach to supporting *lightweight semantic annotation* of BPMN models by analysts, bearing in mind that insisting on the use of 'heavier' formal methods for annotation, or the translation of BPMN models into formal semantic domains (on which there is no consensus within the community), would find little acceptance in practice. Based on this

Copyright ©2009, Australian Computer Society, Inc. This paper appeared at the Sixth Asia-Pacific Conference on Conceptual Modelling (APCCM 2009), Wellington, New Zealand, January 2009. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 96, Markus Kirchberg and Sebastian Link, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

scheme, we define a uniform graph-based encoding of semantically annotated BPMN models, called *semantic process nets* (or SPNets) (originally introduced in (Ghose & Koliadis 2007)). We then describe a class of process proximity metrics, and show how these can form the basis for an effective business process integration framework. We note that we do not address the problem of data integration (which has been the subject of considerable earlier research), but focus only on behavior integration. Our work improves on several earlier frameworks for process integration that have been discussed in the next section.

2 Background

The principal purpose of any model is to "identify the structural features that have the greatest implications for policy, and thus are worthy of further pursuit" (Fiddaman 1997). Using business process modeling as a means to express the operation of an organizational system based on a combination of artifacts and knowledge extracted from domain experts provides a level of formalism. Maintenance of the formal system can be viewed as problem to be solved within the notation.

2.1 Business Process Modelling

The Business Process Modeling Notation (BPMN) has received strong industry interest and support (White 2006), is highly mature (Becker et al. 2005), but has been found to have some limitations relating to the representation of process state and other ambiguities (Becker et al. 2005). Business processes are represented in BPMN using **flow nodes**: *events*, *activities*, and *decisions*; **connectors**: *control flow links*, and *message flow links*; and **swimlanes**: *pools*, and *lanes* within pools.

We model the quality aspects of a BPMN model using an algebraic scheme developed within the constraint modeling literature (Bistarelli 2001), which defines quality scales as a 5-tuple $\langle \mathbf{A}, \oplus, \otimes, \mathbf{0}, \mathbf{1} \rangle$. Under this scheme, A is a set consisting of numeric, boolean or symbolic preference values, $\oplus$ and $\otimes$ are two commutative and associative operators closed in $\mathbf{A}$, $\mathbf{0}$ is the least preferred value in $\mathbf{A}$, and $\mathbf{1}$ is the most preferred value in $\mathbf{A}$. In addition: $\oplus$ is an idempotent *comparison* operator, which has an absorbing element of $\mathbf{1}$, and unit element of $\mathbf{0}$; and, $\otimes$ is a *combination* operator, which is usually decreasing, distributes over comparison, has an absorbing element of $\mathbf{0}$, and a unit element of $\mathbf{1}$.

2.2 Business Process Integration

In the following sub-sections, we introduce the state-of-the-art in business process matching and integration techniques, and describe how we address some of their drawbacks.

Matching

Process matching is the process of clustering and relating similar activities. These clusters can be derived using various methods each with strengths and weaknesses that can leverage the knowledge stored in a process.

Clustering techniques classify objects (such as business process models) into partitions so that the data in each subset share common traits. A number of clustering methods and functions are outlined in (Huang & Ng 1999) using large set based k-mean algorithms. During the clustering phase each element is massaged into a group of related elements. In cases where data can not be disseminated using large data set averaging methods, the classification of objects in a particular domain can be completed by separating objects into classes based on their attributes, and giving criteria for determining whether a particular object in the domain is in a particular class or not. This is done in bi-clustering (Busygin et al. 2008) where a set of samples are simultaneously partitioned into subsets in a way that they have a high relevance to each other, k-mean clustering can be used with other methods to create of bi-clustered groups.

The problem of using these techniques within an organizational domain is the complexity associated with implementations. Most implementations of data clustering can be seen in large scale projects such as gene mapping and search engine crawling.

Smaller steps can be taken to reduce the complexity of large scale data classification requirements with the use of naming conventions. Activity names should carry clear and concise meanings. Each data set name will provide a significant meaning to the observer. During design analysts define models using meaningful naming conventions to provide clarity in some context. (Kementsietsidis et al. 2003) Kementsietsidis investigates methods for the logical design of data values to promote integration from heterogeneous data sources using data mapping tables. The tables maintain correspondences between, for example, business processes within a process repository. Thus, queries may result in alternate names, retaining knowledge in a particular domain.

An example of classification completed by system users can be seen implemented in the collaborative database schema integration tool SISIBIS (Beynon-Davies et al. 1997) where during the creation of enterprise data schemas, analysts and system users were asked to tag various elements with the semantic meanings (with respect to themselves) and how various data was designed using contextual descriptions.

The use of matching techniques to connect elements from incoming processes helps reveal contextual similarity. Contextual similarity is required in process integration, acting as a mapping function that shows direct similarities between two processes. In (van Dongen et al. 2008) Dongen,et al. presents a vector based proximity metric for contextual similarity. These metric are found by beginning with cluster based semantic similarities across documents (using causal and contextual footprints and combining with semantic scoring), where an organization wide vector of artifacts are contrasted against one another using union operators. This approach allows an analyst to rank semantic equivalences between two or more processes. The adjunct system described in (Mendling & Simon 2006)(Mendling) shows structural integration methods using Event-Driven Process chains against a number of SAP process models. In this work a structure merge operator is defined for use on SAP models that can be used once a semantic similarity between functions has been defined. Mendling also shows a reduction method that can be achieved by eliminating redundant process pathways while keeping EPC based structural integrity. In the preceding research Event-Driven process chains are used to verify structural 'soundness' or non-recoverable errors (van Dongen et al. 2005) as well as reducing complexity within the structure. The problems asso-

ciated with relying on a union combination of various process information (van Dongen et al. 2008) is in considering similarity of functions (defined using synonyms and words, or similarly of footprints) ignoring the frequency of the metrics (Lewis 1998). We address these problems by considering differences and adding together algebraic distances.

Integration

Process integration aims to investigate relationships across a business compendium to produce classifications and merge activities into a standardized system. This involves both matching and merging methods. The process of integrating various activities relies heavily on matching criteria. Once objects are considered close enough to integrate with one another, and if each object is not equal to the other, then the merging process will begin.

Integration is the process of merging elements from two similar antecedent processes to create a single process that can be used to replace the original processes. Integration is broken into two parts, aggregation and regression. Aggregation is the process of combining data elements after detecting common elements or common relations (Wang & Miller 2005). This is done in its simplest form by combining common elements from two antecedent processes.

Hinke (Hinke 1988) shows a further depth to aggregation by comparing general cardinality aggregations and inference aggregation in which predictions of inference can be made from data. Here not only are similar activities from antecedent processes joined in an integrated output, there is also a case where if an antecedent activity has a relation to an activity that does not have a direct role in a process but acts as a constraint on a future activity within the process, then the activity is included during integration as an inference activity. For example, consider a process where there is an activity of 'stamping letters in a mail room'. During the integration of two processes that describe *'sending a letter to a customer'*, we must consider *'stamping a letter'* as a constraint to be satisfied before *'mailing the letter'*. This activity should be included in the integrated output process even if it is not explicitly defined in one of the antecedent processes.

Regression is a stage within an integration system that involves reduction of the possible resulting process solution space while maintaining consistency. As a model of a process is aggregated a number of possible solutions can be generated. It is during regression that duplicate and structurally unnecessary data and information is removed to form explicit processes. These processes can then be analysed and a potential candidate implementation process can be chosen. In (Morimune & Hoshino 2008) Morimune offers a number of regression testing methods that can be used to for the creation of these candidate processes, using homogenous constraints. Regression in the use of process integration is useful for selecting optimal solutions.

Research into the area of business process space has resulted in interesting work, where many of the technical aspects of integration have been addressed. In (Grossmann et al. 2004), a method for business process integration is presented, which relies on the introduction of detailed and explicit process states, inter-process dependencies, and synchronizations as integration criteria. In comparison, the work in this paper presents a goal and proximity-directed criterion (relying on minimal analyst intervention) allowing analysts to explore candidate integrations that maintain structural and semantic similarity to their antecedents. In (Mendling & Simon 2006), a (database) view integration-inspired business process integration method, achieved via a view-merge operator, identity/ordering relations, and restructuring (or simplification rules) is presented. In comparison, we outline criteria that help establish identity relations, and minimize structural and (some) semantic differences during integration.

In the following sections we provide a conceptual framework that can be relatively easily implemented in decision-support tools to detirmine degree of similarity of process model integration options. A key challenge with BPMN is that it provides relatively little by way semantics of the processes being modeled. Another challenge is that there is no consensus on how the semantics of BPMN might be defined, although several competing formalisms have been proposed. Since integration clearly requires more information than is available in a pure BPMN process model, we propose a lightweight, analyst-mediated approach to semantic annotation of BPMN models, in particular, the annotation of activities with effects. Model checking is an alternative approach, but it requires mapping BPMN process models to state models, which is problematic and ill-defined. We encode BPMN process models into semantically-annotated digraphs called Semantic Process Networks (or SPNets). We then define a class of proximity relations that permit us to compare alternative modifications of process models in terms of how much they deviate from the original process model.

3 Semantic Process Nets (SPNets)

Semantic Process Nets (SPNets) (Ghose & Koliadis 2007) provide us with a uniform structural and semantic encoding of a BPMN model to which we will be developing our theory for business process integration.

Definition 1 *A Semantic Process Network (SPNet) is a graph* $\langle V, E, s, t, l_V, l_E \rangle$ *such that:* V *is a set of nodes;* E *a set of edges;* $s, t : E \rightarrow V$ *are source and target node mappings;* $l_V : V \rightarrow \Omega_V$ *maps nodes to node labels; and,* $l_E : V \rightarrow \Omega_E$ *maps edges to edge labels. Each label in* Ω_V *and* Ω_E *is of the form* $\langle id, type, value \rangle$.

We note that a unique SPNet exists for each model in BPMN. This can be determined objectively through transformation. Each event, activity or gateway in a BPMN model maps to a node, with the $type$ element of the label indicating whether the node was obtained from an event, activity or gateway in the BPMN model. Actors also map as nodes, with the value label referring to the name of the role associated with the pool and lane of the actor. The $type$ element of an edge label can be either *control*, *message*, *assignment*, *immediate effect*, or *cumulative effect* depending on whether the edge represents a control flow, message flow, task assignment, immediate effect, or cumulative effect descriptor. The $value$ element of edge labels are: guard conditions (for control edges); message descriptors (for message edges); actor names (for assignment edges); post conditions (for immediate effect edges); or, context descriptors (for cumulative effect edges). Note, $s(e) = t(e)$ for an immediate effect, or cumulative effect edge $e \in E$.

The $value$ elements for immediate effect, and cumulative effect edges are triples of the form

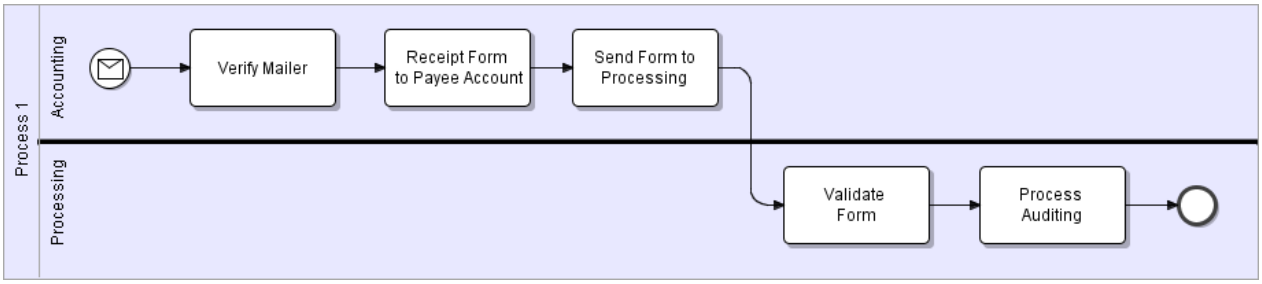

Figure 1. Example Process 1

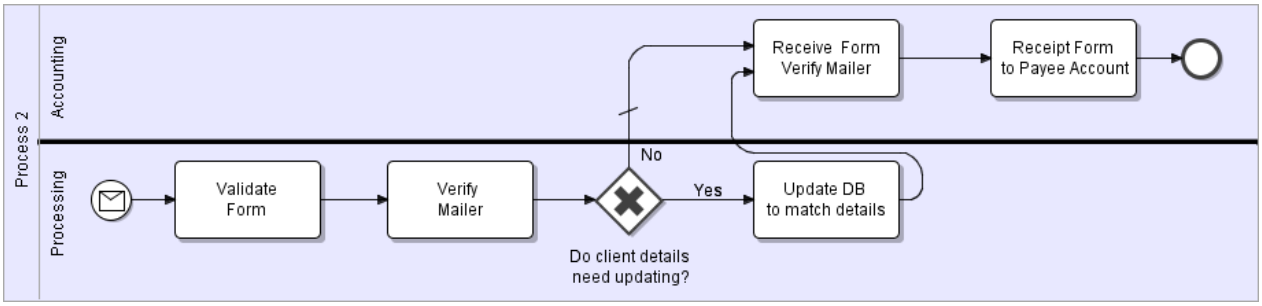

Figure 2. Example Process 2

$\langle id, function, quality \rangle$. The id element of an immediate effect edge corresponds to the source node id label element. The id element of a cumulative effect edge is a scenario identifier (a vector) where each element is either: a node identifier; or, a set whose elements are (recursively) scenario identifiers. A scenario identifier describes the precise path that would have to be taken through the process model to achieve the cumulative effect in question.

The $function$ element of an immediate effect, or cumulative effect edge label is a set of assertions, whereas the $quality$ element is a vector of QoS evaluations. The $function$ and $quality$ elements of an immediate effect annotation edge label can be viewed as a context-independent specification of its functional and non-functional effects. These must be accumulated over an entire process to be able to specify, at the end of each activity, the contextual $function$ and $quality$ elements of cumulative effect annotation labels. These labels indicate the functional and non-functional effects that a process would have achieved had it executed up to that point.

3.1 Accumulating Functional Effects

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. In the case of formal annotations, we assume effects have been represented in Conjunctive Normal Form (CNF) where each clause is also a *prime implicate*, thus providing a non-redundant canonical form. Cumulative effect annotation involves a left-to-right pass through a participant lane. Activities which are not connected to any preceding activity via a control flow link are annotated with the cumulative effect $\{e\}$ where e is the immediate effect of the task in question. The process of obtaining cumulative effect annotations from a BPMN model annotated with immediate effects can be automated in the instance of formal or controlled natural language annotations. We note that this approach to obtaining functional effect descriptions comes with no guarantee of completeness. In other words, the quality of the descriptions that we obtain is a function of the quality of immediate effects and goals specified by analysts. Our experience suggests that the approach is nonetheless useful in providing an approximately adequate basis for change management.

Let $\langle t_i, t_j \rangle$ be an ordered pair of tasks connected via a sequence flow such that t_i precedes t_j, let e_i be an effect scenario associated with t_i and e_j be the immediate effect annotation associated with t_j.

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 resulting cumulative effect, denoted by $acc(e_i, e_j)$, is $\{e_i \cup e_j\}$. Else, $acc(e_i, e_j) = \{e_j \cup e | (e \subseteq e_i)$ and $(e \cup e_j$ is consistent) and (there does not exist $e'[(e' \cup e_j$ is consistent) and $(e \subset e')])\}$.

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-joins only, for brevity. The procedure generalizes in a straightforward manner for *n*-way joins.

In the following, let t_1 and t_2 be the two tasks immediately preceding a 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 an-

notation, and E the cumulative effect annotation of a task t immediately following the join.

For an *AND-join*, we define $E = \{a_i \cup a_j | a_i \in acc(es_{1i}, e)$ and $a_j \in acc(es_{2j}, e)$ and $es_{1i} \in E_1$ and $es_{2j} \in E_2$ and $\{es_{1i}, es_{2j}\}$ are compatible$\}$. A pair of effect scenarios are compatible if and only if their identifiers (representing the path and decisions taken during construction of the scenario) are consistent (the outcomes of their decisions match). 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 $ANDacc(E_1, E_2, e)$.

For an *XOR-join* (denoted by $XORacc(E_1, E_2, e)$), we define $E = \{a_i | a_i \in acc(es_i, e)$ and $(es_i \in E_1$ or $es_i \in E_2)\}$.

For an *OR-join*, the result of effect accumulation is denoted by $ORacc(E_1, E_2, e) = ANDacc(E_1, E_2, e) \cup XORacc(E_1, E_2, e)$. The role of guard conditions within effect annotations is also important. Consider the first activity t on an outgoing sequence flow from an OR- or XOR-split.

Let E be the set of effect scenarios annotating the activity immediately preceding the XOR-split and let $E' \subseteq E$ such that each effect scenario E' is consistent with the guard condition c associated with that outgoing flow. Then the set of effect scenarios of t is given by $\{a \mid a \in acc(e \wedge c, e_t)$ and $e \in E'\}$, where e_t is the immediate effect annotation of t and $e \wedge c$ is assumed without loss of generality to be represented as a set of prima implicates.

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.

3.2 Accumulating Non-Functional Effects

We use scenario identifiers (see Section 3) to compute cumulative QoS measures. This leads to a cumulative measure per effect scenario. Recall that a scenario identifier is a sequence composed of activity identifiers or sets consisting (recursively) or scenario identifiers. We use the sets in the label to describe parallel branches. We therefore need to use our algebraic *parallel accumulation operator* ($\ddot{\otimes}$), one for each QoS factor, to specify how cumulative QoS measures, propagated along parallel branches, get combined together at a join gateway.

4 Semantic Process Net Integration

Business process proximity is used during integration to establish a distance measure between two SPNets. Intuitively, this measure is used to ensure that the integrated model is as similar as possible to its two antecedents. In other words, we would like to minimize the deviation of an integrated model from its ancestors, thereby utilizing the previous legacy configuration and minimizing effort during integration.

Definition 2 *Associated with each SPNet is a* proximity metric*: $d(p_i, p_j)$; which given an integrated process p_i, and one of its antecedents p_j, computes the distance of p_i from p_j w.r.t. a combination of structural and semantic criteria, alternatively defined as either (or by combining):*

- $d_V(p_i, p_j) + d_E(p_i, p_j) + d_S(p_i, p_j)$;
- $w_V d_V(p_i, p_j) + w_E d_E(p_i, p_j) + w_S d_S(p_i, p_j)$;
- $\frac{d_V(p_i,p_j)}{D_V(p_i,p_j)} + \frac{d_E(p_i,p_j)}{D_E(p_i,p_j)} + \frac{d_S(p_i,p_j)}{D_S(p_i,p_j)}$;

such that: d_V, d_E, and d_S are node, edge, and semantic (effect) proximity metrics; w_V, w_E, and w_S are weights for each metric; and, D_V, D_E, and D_S indicate the maximum hypothetical distance.

In order to compute our structural (node and edge) distance metrics, we consider sets of nodes $V(p)$ and edges $E(p)$ of each model p in the following way: $d_V(p_i, p_j) = |V(p_i)\Delta V(p_j)|$; and, $d_E(p_i, p_j) = |E(p_i)\Delta E(p_j)|$. Note, that for an SPNet encoding of a BPMN model, we only consider edges of *type*: *control*, *message*, and *assignment*. In addition: $D_V(p_i, p_j) = |V(p_i)| + |V(p_j)|$, and $D_E(p_i, p_j) = |E(p_i)| + |E(p_j)|$. These measures are used in the last instance to help reduce the dominance of any one structural or semantic proximity metric.

Computing semantic proximity d_S is somewhat more complicated as it relies on the possible end effect (outcome or scenario) of either process. Firstly, we require a mechanism for matching the end effects of either process. Let e be some effect scenario, let E be a set of candidate effect scenario matches, and let $m_S(e, E) = \{e_k \in E \mid |e\Delta e_k| \leq |e\Delta e_j|$ forall $e_j \in E\}$ denote the set of min-cardinality different elements of the set of candidate scenarios E w.r.t. the scenario e. Thus, $\delta_S(e, E) = \delta \in \{e\Delta e_k \mid e_k \in m_S(e, E)\}$ denotes a non-deterministically chosen min-cardinality difference. Let $\Delta_S(E_i, E_j) = \{\delta_S(e_i, E_j) | e_i \in E_i\}$ denote the asymmetric difference between the set of scenarios E_i and E_j for corresponding processes p_i and p_j, and $d_S(p_i, p_j) = \sum |\delta|$, forall $\delta \in \Delta_S(E_i, E_j)$, where E_i and E_j correspond to the end effect scenarios of process p_i and p_j respectively. Therefore, $d_S(p_i, p_j)$ computes the sum cardinality of each difference between an end effect scenario in p_i and a matching end effect scenario in p_j. We note that symmetric versions of this metric exist, but omit their details, along with their proofs, for future work.

In addition, cost metrics could also be incorporated into our calculation of proximity in order to incorporate the cost associated with making changes to either antecedent of an integration.

4.1 Semantic Process Net Integration Criteria

Any approach to process integration should view both the process state (Mendling & Simon 2006), the domain knowledge (Fankhauser et al. 1991) (Beynon-Davies et al. 1997) (Li et al. 2006), as well co-ordination characteristics (Heinz & Eder 1999) (Grossmann et al. 2005). Under our integration scheme we provide a framework for process integration based on process structural and semantic descriptions. This framework may work in combination with one of the aforementioned approaches.

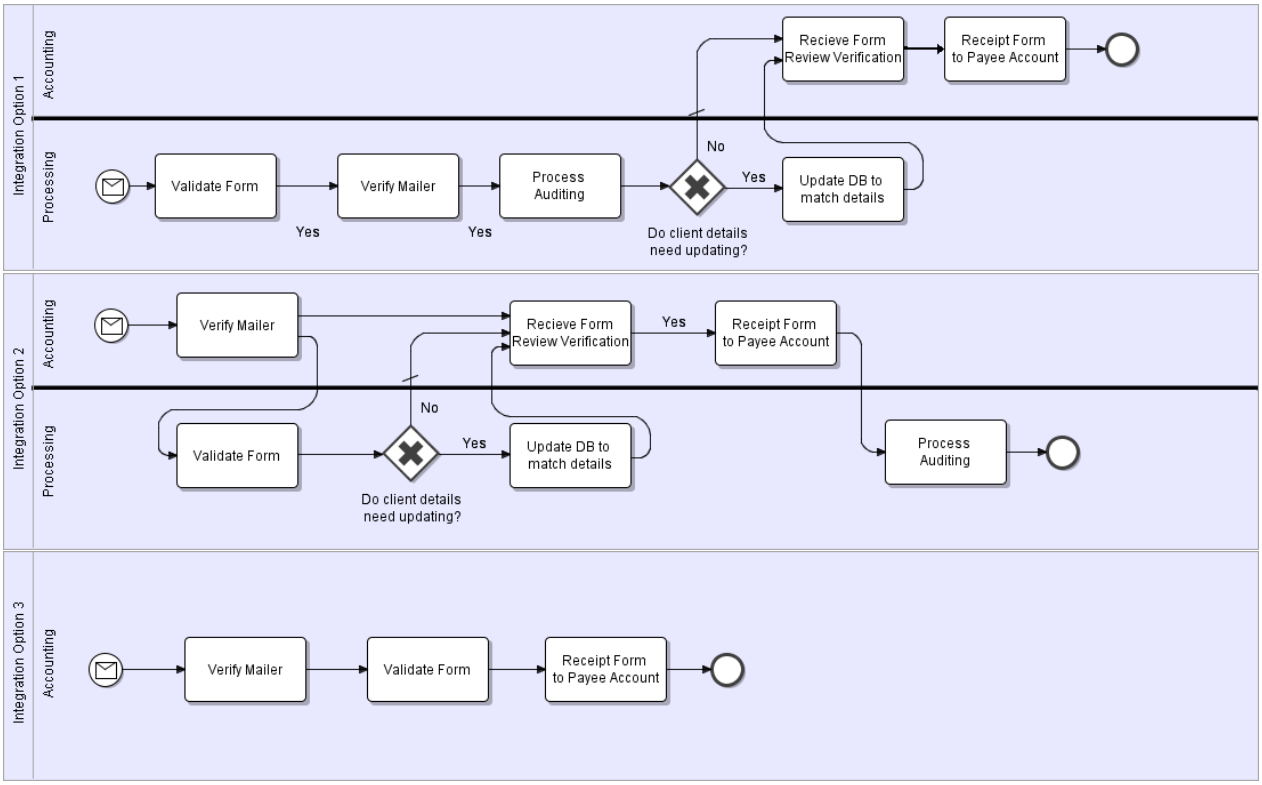

Figure 3. Example Integrations

Definition 3 *An SPNet p represents the integration of an SPNet p_1 and SPNet p_2 iff all of the following hold:*

1. *SPNet p achieves G_p (the goals associated with p) where $G_p = G_{p_1} \wedge G_{p_2}$ and G_{p_i} is the goal achieved by process p_i;*
2. *there exists no p' such that p' achieves G_p and the following holds: $d(p_1, p') + d(p_2, p') < d(p_1, p) + d(p_2, p)$. Here d is a distance function between two processes. This defines an integration solution where the closest integration super process has no closer potential solution p'.*

Note, that our definition of goals above applies to both the functional and non-functional properties of an SPNet.

4.2 Semantic Process Net Integration Methodology

Business process integration in practice requires some effort on behalf of an analyst, during both process matching and selection of candidate integrations. The criteria we have outlined in the previous sections allow us to reduce analyst effort during the matching and selection steps (as outlined in the discussion below).

Step 1: Business Process Matching.
Prior to and during integration, matching is required to determine the likelihood that two business processes, or activities within a business process, share similarities or are equivalent. This may involve the use of three techniques. The first involves evaluating the labels of business processes and activities using linguistic techniques (mediated with an ontology) as in (Ehrig et al. 2007). This may help in, for example, determining that a *Package Consignment* process is semantically similar to a *Package Receiving* process. Another technique that may be applied (also in combination with an ontology) is the evaluation of the effect (or functionality) of a business process or activity. Here, the semantic aspect to our proximity metric can be re-used effectively. Finally, as processes may be represented at varying levels of abstraction, we can apply the aforementioned techniques to detect the part-whole relations among and within business processes in order to initially resolve abstraction conflicts. These three approaches to matching may be either completely automated, involve some automation, or be a simple guide applied to a completely manual integration.

Step 2: Business Process Integration Goals.
We firstly require a goal (describing a set of criteria) for the integrated business process to be determined. In this approach, the goal can be either given or determined by merging the effect scenarios of each business process to be integrated using an automated or semi-automated strategy. An automated strategy might involve: conjoining consistent effect scenarios; and/or, extracting the most common effects among effect scenarios. As these strategies only

Table 1. Example Process Effects

Accumulation Effects	Description Process 1	Description Process 2
Validation of Form	In processing after receipting	In Processing
Verification of Identity	In Accounting	In Processing
Receipting of Funds	In Accounting	In Accounting
Auditing of Work	In Processing	
Update of DB		After verification and validation
The correct payee is known to all (after verify)	In Accounting	In Processing
Database Updated	Completed after verification	

Table 2. Example Integration Effects

Accumulation Properties	Description Integration 1	Description Integration 2	Description Integration 3
Validation of Form	In processing	In processing	In accounting
Verification of Identity	In processing	In Accounting	In accounting
Receipting of Funds	In Accounting	In Accounting	In accounting
Auditing of Work	In processing	In processing	
Update of DB	Completed if needed after verification and verification	Completed after verification and validation	
The correct payee is known to all (after verify)	Account information given to accounting after validation and verification	Accounting passes verification information to processing and keeps information on their records	

lead to some approximate baseline, analysts will need to provide input. The requirement to firstly establish the common business goal for an integration step allows us to reduce the complexity of ad-hoc integration, as well as separating concerns and roles during the process. As discussed, the integration goal can be either computed in a bottom-up or top-down manner, and provides a concise description of the requirements for the integrated model.

Step 3: Business Process Integration.

Business process integration involves a search through a space of possible integration options that is directed by our integration characteristics. One way to search for the most proximally efficient integration, can be to follow a local generate and test strategy. Consider an algorithm sketch: whose input is R (a repository of SPNets to be integrated); and, manipulates a set V of $\langle spn, history \rangle$ pairs. The algorithm would 1: $V = \{\langle spn, \langle\rangle\rangle\}$ (initialize with the model to be manipulated (possibly the intersection of nodes and edges among models); 2: $While(!Accepted(V))\ V = Step(V, R)$ (step through changes until an acceptable integration is identified). An implementation of the $Step$ function would apply a single admissible addition or removal of a node or edge (possibly from elements of R). The history would allow: poor candidates to be forgotten; ensure complementary operations are not applied to single models; ensure uniqueness is maintained across models; and provide a local basis for evaluating proximity and other heuristics. Firstly, termination could be an issue due to the infinite (gateways) nature of R, although results are anytime. There is also a large branching factor, although the metrics we have defined guide search.

5 Semantic Process Net Integration: An Example

In order to demonstrate the framework described within this paper, we will present a worked example of process integration, given two processes as in Fig. 1 and Fig. 2 . Each process describes a way in which a business completes the task of *'receipting cheques into the organization'*. For the two given processes each activity contributes a number of effects that work to achieve a global organizational goal. These Effects are shown in Table. 1.

To compute the Node Proximity Metric we consider each node in each BPMN process diagram. Using Integration 1 from Fig. 3 we compute the delta with Process 1 from Fig. 1. In process one there are 7 BPMN process nodes (including events). In integration solution 1 there are 9 BPMN process nodes (including events and gateways). We follow the same process to compute the similarity of edges. In order to consider values for our semantic metric we have in the case of our example considered differences in the effect of completing each activity as a representation as described in Table.1 and Table.2 (this could as easily be replaced with word average per activity name, annotations, or effects).

Once these effects have been found and listed an analyst or automated task may then proceed to generate various integrated process schema's. We have provided 3 possible integration models in Fig. 3. These models vary in terms of Actors rolls and ordering. During this stage the naming conventions have remained the same (as such there is little semantic deviation), and relative ordering based on compliance constraints have been left untouched.

Table 3. Example Integration Effects: Node Proximity

Node Proximity	Process 1 (P_1)	Process 2 (P_2)
Integration(I_1)	$\frac{nd_{P_1}(7)\Delta nd_{I_1}(9)}{\|P_1\|+\|I_1\|} = \frac{1}{8}$	$\frac{nd_{P_2}(8)\Delta nd_{I_1}(9)}{\|P_1\|+\|I_1\|} = \frac{1}{17}$
Integration(I_2)	$\frac{nd_{P_1}(7)\Delta nd_{I_2}(9)}{\|P_1\|+\|I_2\|} = \frac{1}{8}$	$\frac{nd_{P_2}(8)\Delta nd_{I_2}(9)}{\|P_1\|+\|I_2\|} = \frac{1}{17}$
Integration(I_3)	$\frac{nd_{P_1}(7)\Delta nd_{I_3}(5)}{\|P_1\|+\|I_3\|} = \frac{1}{6}$	$\frac{nd_{P_2}(8)\Delta nd_{I_3}(5)}{\|P_1\|+\|I_3\|} = \frac{3}{13}$

Table 4. Example Integration Effects: Edge Proximity

Edge Proximity	Process 1 (P_1)	Process 2 (P_2)
Integration(I_1)	$\frac{ed_{P_1}(6)\Delta ed_{I_1}(9)}{\|P_1\|+\|I_1\|} = \frac{1}{5}$	$\frac{ed_{P_2}(8)\Delta ed_{I_1}(9)}{\|P_1\|+\|I_1\|} = \frac{1}{17}$
Integration(I_2)	$\frac{ed_{P_1}(6)\Delta ed_{I_2}(10)}{\|P_1\|+\|I_2\|} = \frac{1}{4}$	$\frac{ed_{P_2}(8)\Delta ed_{I_2}(10)}{\|P_1\|+\|I_2\|} = \frac{1}{9}$
Integration(I_3)	$\frac{ed_{P_1}(6)\Delta ed_{I_3}(4)}{\|P_1\|+\|I_3\|} = \frac{1}{5}$	$\frac{ed_{P_2}(8)\Delta ed_{I_3}(4)}{\|P_1\|+\|I_3\|} = \frac{1}{3}$

Table 5. Example Integration Effects: Semantic Proximity

Semantic Proximity*	Process 1 (P_1)	Process 2 (P_2)
Integration(I_1)	$\frac{sd_{P_1}(7)\Delta sd_{I_1}(9)}{\|P_1\|+\|I_1\|} = \frac{1}{8}$	$\frac{sd_{P_2}(8)\Delta sd_{I_1}(9)}{\|P_1\|+\|I_1\|} = \frac{1}{17}$
Integration(I_2)	$\frac{sd_{P_1}(7)\Delta sd_{I_2}(9)}{\|P_1\|+\|I_2\|} = \frac{1}{8}$	$\frac{sd_{P_2}(8)\Delta sd_{I_2}(9)}{\|P_1\|+\|I_2\|} = \frac{1}{17}$
Integration(I_3)	$\frac{sd_{P_1}(7)\Delta sd_{I_3}(5)}{\|P_1\|+\|I_3\|} = \frac{1}{6}$	$\frac{sd_{P_2}(8)\Delta sd_{I_3}(5)}{\|P_1\|+\|I_3\|} = \frac{3}{13}$

Once these alternate integration option have been created, we now use ratio proximity relations to compare nodes, edges, and semantics between the original processes and the integrated possibilities. In the tables. 3, 4, 5 we have computed the various proximity relations in order to find an overall solution. After analyzing the results obtained, we have found the Integration 2 is the prime candidate for the role of an integration of process 1 and process 2.

In this example we have shown that using proximity metrics across node, edge and semantic values we are able chose an appropriate integration solution. These metrics can be further formulated using the weighted metrics for each candidate depending on the application. For a business wishing to integrate processes with a goal of broader knowledge consistency (language based), attention to structural deviation may be modulated to have a lesser impact in the decisional stages.

6 Summary and Conclusions

The interesting element of our method is in the use of *minimal change* of processes. This acts in favor of a business implementing a change management solution in terms of costs minimization (as it costs less to change less), and also in the reduction of change risks. This risk is of growing concern for compliance reasons, as with strict regulative control acting on many businesses it is assumed that broad innovative changes to processes as the result of any integration activity may leave an organization vulnerable to breaks in the value chain or penalties bought about by uncompleted activity steps.

In this work we have presented an innovative method of process integration using Semantic Process Networks. This framework can be used a means to complete process integration. As a continuation of our work into this area we would like to explore additional instantiations of our proximity metrics and validate our approach in a controlled setting.

References

Becker, J., Indulska, M., Rosemann, M. & Green, P. (2005), Do process modelling techniques get better?, *in* 'Proceedings of the 16th Australasian Conference on Informaion Systems'.

Beynon-Davies, P., Bonde, L., McPhee, D. & Jones, C. (1997), 'A collaborative schema integration system', *Computer Supported Cooperative Work (CSCW)* **6**, 1–18. **URL:** *http://dx.doi.org/10.1023/A:1008627102073*

Bistarelli, S. (2001), Soft Constraint Solving and Programming: a General Framework, PhD thesis, Computer Science Department, University of Pisa.

Busygin, S., Prokopyev, O. & Pardalos, P. M. (2008), 'Biclustering in data mining', *Comput. Oper. Res.* **35**(9), 2964–2987.

Ehrig, M., Koschmider, A. & Oberweis, A. (2007), Measuring similarity between semantic business process models, *in* 'Proc. of the Fourth Asia-Pacific Conf. on Conceptual Modelling'.

Fankhauser, P., Kracker, M. & Neuhold, E. J. (1991), 'Semantic vs. structural resemblance of classes', *SIGMOD Rec.* **20**(4), 59–63.

Fiddaman, T. (1997), Feedback Complexity in Integrated Climate-Economy Models, Ph.d. thesis, MIT Sloan School of Management.
URL: *http://metasd.com/papers/dissabstract.html*

Ghose, A. & Koliadis, G. (2007), Auditing business process compliance, *in* 'Proceedings of the International Conference on Service-Oriented Computing', Springer LNCS, pp. 169–180.
URL: *http://dx.doi.org/10.1007/978-3-540-74974-5_14*

Grossmann, G., Ren, Y., Schrefl, M. & Stumptner, M. (2005), *Behavior Based Integration of Composite Business Processes*, Springer Berlin / Heidelberg, pp. 186–204.
URL: *http://dx.doi.org/10.1007/11538394_13*

Grossmann, G., Schrefl, M. & Stumptner, M. (2004), Classification of business process correspondences and associated integration operators, Springer Berlin / Heidelberg, pp. 653–666.
URL: *http://www.springerlink.com/content/5l0qh8f1dnf96m93*

Heinz, F. & Eder, J. (1999), Towards an automatic integration of statecharts, *in* 'ER '99: Proceedings of the 18th International Conference on Conceptual Modeling', Springer-Verlag, pp. 430–444.

Hinke, T. (1988), Inference aggregation detection in database management systems, *in* 'Security and Privacy, 1988. Proceedings., 1988 IEEE Symposium on', pp. 96–106.

Huang, Z. & Ng, M. (1999), 'A fuzzy k-modes algorithm for clustering categorical data', *Fuzzy Systems, IEEE Transactions on* **7**, 446–452.

Kementsietsidis, A., Arenas, M. & Miller, R. J. (2003), Mapping data in peer-to-peer systems: semantics and algorithmic issues, *in* 'SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data', ACM, New York, NY, USA, pp. 325–336.

Lewis, D. D. (1998), Naive (Bayes) at forty: The independence assumption in information retrieval., *in* C. Nédellec & C. Rouveirol, eds, 'Proceedings of ECML-98, 10th European Conference on Machine Learning', number 1398, Springer Verlag, Heidelberg, DE, Chemnitz, DE, pp. 4–15.
URL: *citeseer.ist.psu.edu/lewis98naive.html*

Li, Q., Shan, Z., Hung, P. C. K., Chiu, D. K. W. & Cheung, S. C. (2006), Flows and views for scalable scientific process integration, *in* 'InfoScale '06: Proceedings of the 1st international conference on Scalable information systems', ACM, New York, NY, USA, p. 30.

Mendling, J. & Simon, C. (2006), Business process design by view integration, *in* 'Proceedings of the BPM 2006 Workshops, Workshop on Business Process Design BPD 2006', Vol. 4103 of *Lecture Notes in Computer Science*, Springer-Verlag, Vienna, Austria, pp. 55–64.

Morimune, K. & Hoshino, Y. (2008), 'Testing homogeneity of a large data set by bootstrapping', *Math. Comput. Simul.* **78**(2-3), 292–302.

van Dongen, B., Dijkman, R. & Mendling, J. (2008), Measuring similarity between business process models, *in* 'Advanced Information Systems Engineering', Springer, pp. 450–464.
URL: *http://www.springerlink.com/content/xv3503k264370475*

van Dongen, B., van der Aalst, W. & Verbeek, H. (2005), Verification of epcs: Using reduction rules and petri nets, *in* 'Advanced Information Systems Engineering', Springer, pp. 272–286.
URL: *http://www.springerlink.com/content/g8cpj5rqbtehb6a1*

Wang, G. & Miller, S. (2005), Intelligent aggregation of purchase orders in e-procurement, *in* 'EDOC Enterprise Computing Conference, 2005 Ninth IEEE International', pp. 27–36.

White, S. (2006), Business process modeling notation (bpmn),, Technical report, OMG Final Adopted Specification 1.0 (http://www.bpmn.org).