Supporting Development and Evolution of Service-based Processes

Martin Henkel  
Stockholm University and  
Royal Institute of Technology  
Forum 100, 164 40 Kista  
Sweden  
martinh@dsv.su.se

Jelena Zdravkovic  
University of Gävle and  
Royal Institute of Technology  
Forum 100, 164 40 Kista  
Sweden  
jzc@dsv.su.se

Abstract

When designing executable enterprise processes, the alignment between business and technical requirements is still one of the central problems. Process specifications obtained from the pure business perspective commonly cannot be executed “as-is” due to constraints of information systems given in the form of existing software services. In this paper we introduce several concepts targeted towards aligning business processes and service-based systems. The notion of realization levels is introduced to discern the level of business conformance. Furthermore, a model of system constraints is introduced to guide the developer to process realizations with high business conformance. By the combined use of the concepts, structured development and evolution of process realizations is enabled by continuous tracking of the alignment between business and the supporting technology.

1. Introduction

When designing executable business processes, consideration is paid to business requirements i.e. the design should closely resemble the business operations, events and message exchanges that exist in the enterprise. This means that the process (we name it business process) reflects the pure business perspective, as the focus is to “automate the business”.

In a majority of organizations, business processes have to be aligned with existing systems. These systems, represented by software services, may impose specific requirements. For instance, it might be the case that parallel activities must be realized as sequential because of execution dependencies of the services of the back-end systems. Another example is that there might not exist services that support the activities of the business process in their current form. As the result, to realize a business process, the designer must consider the behaviour of existing systems’ services. A final executable process specification (we name it technical process) will reflect both the business and technology perspectives.

It can, therefore, be said that technical processes realize business processes by using existing systems’ services. If there are no constraints put on the realization from services, the technical process will directly correspond to the business process. However, in the majority of cases, business and technical processes will differ.

In our previous work [1] we defined a set of transformation patterns, called realization types, which are utilized when transforming a business process into a technical process. In this paper we put these transformations in context by providing the main concepts needed to support development and evolution of process realizations. Firstly, we introduce the notion of realization levels to discern the quality of a process realization (Section 3). Secondly, we propose modelling concepts that can be used to represent the system constraints that existing services put on a process realization (Section 4). Thirdly, we introduce a traceability model that takes advantage of the previously defined concepts to enable tracking of how changes in the business process and system constraints affect the realised process (Section 5). The notion of realization levels and the model of system constraints can be used as instruments when the designer transform a business process into a technical realization based on existing services.

The problem of non-alignment between business and technical requirements in design of enterprise business processes has been widely studied in the research community ([2], [3], [4], [5], [6]). Our work differs in two aspects. First, we discuss the modelling of executable processes with the focus on the misfit between business and technical perspectives as the consequence of existing systems constraints. Second,
we concentrate our research towards bridging the gap between business- and technology-dependent process specifications, by introducing a set of transformation patterns. In addition, we classify the patterns with respect to the quality of transformation.

The paper is structured as follows. In the next section an example of a business process is presented, as well as how its realization as a technical process is influenced by constraints in existing services. A classification of realizations of business processes and the concepts needed to document system constraints are presented in Section 3 and 4, respectively. Section 5 and 6 concludes the paper by providing a unified model that enables support for process evolution, as well as a discussion of the results.

2. Example case

An example of a business process and its realization as a technical process is presented in Figure 1. The example is based on a process provided by Sandvik, a global industrial materials engineering company. One of the key concerns of the company is coordination of their existing ERP systems in the form of services. The business process in Figure 1(a) depicts an excerpt of a process used to supply customers with product information. The model is described in the Business Process Modeling Notation (BPMN, [7]), a technique used to graphically express rich process semantics which might be directly mapped to executable language specifications (such as BPEL[8]).

The business process in Figure 1(a) starts upon receiving a product information request. The customer profile information including the contact information and the order history are then gathered. Following this, the product drawing, the current stock level and the customer-based price are fetched concurrently. When the product information is merged, it is archived and sent to the customer. Since the product information should not be archived unless the customer receive it, the archive and send activities are enclosed in an atomic transaction [9].

The technical process is based on existing services, provided by the internal ERP systems (CAD, Order, Production and CRM system pools in Figure 1(b)).

![Figure 1. Business process (a) and realized technical process (b)](image-url)
Compared to the business process, the technical process must adhere to a set of system constraints:

- The customer profile is provided as two activities, because the customer contact and order history information is located in different ERP systems.
- The customer price cannot be concurrently obtained with the stock information, because the Order system requires the stock location in to calculate the price.
- Based on the customers' service ability, product information should be sent as either a HTTP message or a FTP file.
- Since the CRM system does not support atomic (two-phase commit) transactions, the stored response to the product request must be explicitly removed if the information does not reach the customer. This is realized by using a long-running (LRT) transaction [10].

The presented example shows how a realization of a business process can be affected by system constraints posed by existing software services. In the next section we define the requirements for a technical process to be considered as a valid realization of a business process.

3. Business process realization levels

Every process is designed to reach a goal. During execution, a process instance passes through a set of states [11]. Each process state fulfills an extent of the final goal, in the form of a subgoal [12]. In the studies [13], [14] the authors define the main aspects that constitute process specification, such as functional, behavioral, informational, organizational and transactional. Looking from that perspective, the process state shows the current condition (status) of all five process aspects. For instance, in the example from Figure 1(a), when the activity “Receive product information request” completes, the process comes into the state which, from the functional aspect, is described as that the required functionality for receiving the request is fulfilled, from the informational aspect that the required information, i.e. the request for a product information is received, and so on.

As explained in Section 1, a technical process is a realization of a business process by the use of existing services. The basic criterion for the alignment between the technical and the business process is that the technical process should be designed to operationalize all subgoals of the business process by tracing its states:

A state in the technical process instance traces a state in the business process instance if, from the business process perspective, those two states are same regarding the statuses of the process aspects - functional, behavioral, informational, organizational and transactional.

In reality, due to constraints of existing services, it is difficult to obtain technical processes that strictly realize the business process specification for all five process aspects. Some business processes cannot be realized without alterations to the business; this is the case when existing services cannot support the process states as required on the business level. In other situations, a business process could be realized by one technical process, or even more if there are services that provide alternative implementations. Finally, the constraints of existing services could lead to a realization of a business process where the technical process only is capable of tracing the states of some of the business process instances. This leads to a situation where it is the actual instance data that determine if the technical process can realize the business process.

Based on the capability of a technical process to trace the states of a business process, we define four possible levels of realization of a business process (B), by using existing system services (S):

- Lossfull-realizable (LFR): The process B cannot be realized as it is, because it is not possible to define a single technical process that would trace the states of the process B.
- Constrained-realizable (CR): The process B is realized with one or more technical processes, where those processes trace the states of the process B but only for some of the required instances.
- Lossless-realizable (LLR): The process B is realized with exactly one technical process, where this process traces the states of the process B for all required instances.
- Exceeded-realizable (ER): The process B is realized with one technical process where this process traces the states of the process B both for the required and some, not required instances; or the process B is realized with two or more technical processes, where each of those processes trace the states of the process B for at least the required instances.

The importance of the distinction between lossfull and lossless realizable business processes is evident – they distinguish “non-feasible” from “ideal” realizations. However, in many circumstances, existing services enable realization of a business process for some instances, while not for others (i.e. the process is constrained realizable). It is also important to detect the situations when existing services provide more capabilities than required. An exceeded realizable

---

1 In general, a business process supports a group of instances; in the rest of the text we name them “required instances”.

---
business process indicates that existing systems are “too-supportive”. Exceeded realizations are discerned when it is significant to identify if an organization engage more resources in service design than necessary. The four realization levels are summarized in Figure 2, below.

<table>
<thead>
<tr>
<th>Realization level</th>
<th>Business process alignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loss-full (LFR)</td>
<td>Does not support all business process states</td>
</tr>
<tr>
<td>Constrained (CR)</td>
<td>Support all business process states, for some instances</td>
</tr>
<tr>
<td>Loss-less (LLR)</td>
<td>Support all business process states</td>
</tr>
<tr>
<td>Exceeded (ER)</td>
<td>Support business process states, and additional states</td>
</tr>
</tbody>
</table>

**Figure 2. The four realization levels**

As we mentioned in Section 1, in our former work [1] we defined the possible transformations from business to technical processes for all five aspects of the process design (i.e. functional, behavioural, organizational, informational and transactional), in the form of realization types. The total set of realization types covers the possible changes applied to a business process during realization. Based on the given classification of realization levels of business processes, in the following, we discern the quality for each realization type.

The **functional aspect** describes how a process is decomposed i.e. what activities are to be executed. Based on [14], we describe the functionality of an activity with three elements: an activity name describes the goal to be fulfilled, input and output messages define information exchange, and constraints describe pre- and post-conditions. For instance, in the activity “get customer price” (Figure 1(a)) the goal is to obtain the customer-based product price, the input message is the product price and the customer order history, and the output message is a customized price. A constraint put on the output of this activity might be that the price must be calculated in the customer’s local currency.

When a business process (B) is realized, the technical process (T) may require changes in functionality, due to constraints of existing services (S). Changes made during realization of the functional aspect can be classified into three realization types [1]:

- **Specialization.** An activity from the process B is supported with a set of system services S, where those services share the same pre/post conditions and the message exchange as the activity, but have different goals due to specialization to a technical system or a protocol.

- **Aggregation.** An activity from the process B is supported with one or more system services S, where these services jointly exchange the same messages and have same pre/post conditions as the activity thus achieving the same goal; or, several activities from the process B is supported by a single system service S, where this service achieves the same goal, exchange the same messages and have same pre/post conditions as those activities, jointly.

- **Constraint Mapping.** An activity from the process B is supported with a system service, where the activity and the service exchange the same messages and have similar goals but differ in pre/post conditions.

The realization types described above are used to transform the functional aspect of a business process into the functional aspect of a technical process. To realize a business process existing services must implement the process activities such that the business process states can be monitored. However, existing system constraints can affect the use of the functional realization types such that the business process is lossful-, constrained-, lossless- or exceeded-realizable.

In Table 1 we describe under which conditions the use of the functional realization types leads to a particular realization level.

<table>
<thead>
<tr>
<th>Aggregation</th>
<th>Specialization</th>
<th>Constraint mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>LFR</td>
<td>The message exchange of a system service S corresponds to more than one activity in the process B. The process B states cannot be monitored because several activities are realized with a single system service.</td>
<td>The specializations of system service S do not fulfill the goal of the process B activity, because they are in conflict.</td>
</tr>
<tr>
<td>CR</td>
<td>For some instances high granularity system services S that correspond to the message exchange of multiple business activities are invoked.</td>
<td>The given specializations of a process B activity support certain required instances, but not others (i.e. some of the required specializations are supported).</td>
</tr>
</tbody>
</table>
The message exchange of a system service \( S \) contributes to the message exchange of most one activity in the process \( B \). This ensures that the state of the process \( B \) can be monitored on a per-activity basis.

<table>
<thead>
<tr>
<th>Aggregation</th>
<th>Specialization</th>
<th>Constraint mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>LLR</td>
<td>The system service ( S ) provides alternative message aggregations, where all those aggregations support a lossless realization.</td>
<td>The system services ( S ) specialize a process ( B ) activity to several technical systems (protocols), where some of them are not needed by the required instances.</td>
</tr>
<tr>
<td>ER</td>
<td>The system service ( S ) specializes an activity from the process ( B ), while the goal of activity is fulfilled (i.e. the specializations are coherent with the requirements of all the instances).</td>
<td>The constraints of a service ( S ) are same as for an activity in the process ( B ). This ensures that the pre- and post-conditions of the process ( B ) activity are supported.</td>
</tr>
</tbody>
</table>

Example

- **Reordering**: a group of ordered (sequenced or parallel) activities in the process \( B \) is supported by the equivalent, the weaker or the stronger ordering of system services \( S \), due to existing execution dependencies.
- **Condition mapping**: a conditional branch in the process \( B \) is supported by stricter, equivalent or wider branching conditions in the system service \( S \), depending on design and availability of services.

A business processes may be realized in the behavioural aspect if existing services may provide a required composition of the business process activities. In Table 2 we discern under which conditions the use of the realization types leads to a particular realization level.

<table>
<thead>
<tr>
<th>Table 2. Realization levels of the behavioural aspect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Reordering</strong></td>
</tr>
<tr>
<td>LLR</td>
</tr>
<tr>
<td>CR</td>
</tr>
<tr>
<td>ER</td>
</tr>
<tr>
<td>Example</td>
</tr>
</tbody>
</table>

The informational aspect concerns the concepts needed for representing data that a process exchanges with the external environment. When the business process is realized, the business concepts need to be
represented by information structures of existing services; those structures may include, in addition, technical concerns such as transaction and system identifiers. Thus, when realizing a business process (B) by using system services (S), the following realization type is identified:

Information Mapping: business concepts as specified in the process B are mapped to information structures from the system services S.

To realize the informational aspect, it should be possible to trace information concepts from a business process to their equivalent concepts in the technical process. This leads to the following conditions for discerning the levels of realizations (Table 3):

Table 3. Realization levels of the informational aspect

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LFR</td>
<td>The system services S do not support all information concepts required by activities in the process B.</td>
</tr>
<tr>
<td>CR</td>
<td>The system services S support all information concepts required by activities in the process B, but only for some process instances.</td>
</tr>
<tr>
<td>LLR</td>
<td>The system services S support all information concepts as required by activities in the process B. In addition, extensions to handle technical issues might exist.</td>
</tr>
<tr>
<td>ER</td>
<td>The system services S implement all information concepts required by the process B and, in addition, support some non-required business concepts.</td>
</tr>
</tbody>
</table>

Example: Investigation of the customer profile document required in the business process in Figure 1(a), shows that needed information concepts are supported in the documents exchanged by services for retrieving customer contact and orders (i.e. the realization lossless). In the same example, the process would be lossfull realizable if, for instance, the customer’s e-mail address is required in the business process, while the customer contact document retrieved by the corresponding system service does not contain this information.

The organizational aspect defines the distribution of responsibility (i.e. roles) for executing activities. When designing business processes, roles are assigned to business actors. On the system level, roles for invocation of services are assigned to systems and further to organization units that own those systems. When realizing a business process (B) by using a system services (S), the following realization type is identified:

Role Mapping. Roles in the process B are mapped to the ownership of a set of system services S.

A business processes may be realized in the organizational aspect if, for each (business) activity, it is possible to identify a corresponding system service S, owned by a business partner responsible for execution of the business activity. Following this, the levels of the process B realizations are discerned in Table 4:

Table 4. Realization levels of the organizational aspect

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LFR</td>
<td>The system services S that implement business activities from the process B are not owned by the parties in the process B that are responsible for those business activities (in the form of business roles).</td>
</tr>
<tr>
<td>CR</td>
<td>Responsibility for executing an activity in the process B by one of several parties is determined dynamically. For those instances where a party is assigned to the corresponding business role, the responsibility for execution is traced; for those others, it is not.</td>
</tr>
<tr>
<td>LLR</td>
<td>The system services S that implement business activities from the process B are owned by the parties in the process B that are responsible for those business activities (in the form of business roles).</td>
</tr>
<tr>
<td>ER</td>
<td>Several business parties may take responsibility for an activity in the process B. All parties provide supporting system services.</td>
</tr>
</tbody>
</table>

Example: In the example in Figure 1, Sandvik is responsible for executing all activities in the business process. Thus, the realization is lossless, because there is a clear role mapping between the business roles and the services, since Sandvik owns all systems that implement the activities in the technical process.

The transactional aspect concerns consistent execution of a set of activities. Process transactions may comply with two different models. The atomic transaction (AT) model [9] is used to administer a set of shorter activities, that all agree to enforce a common outcome by two-phase commit. In case of errors, the activities are commonly automatically rolled back. In contrast, long-running transaction (LRT) model [10] rules more durable activities, where each activity enforces a globally visible outcome independently of the others. In case of errors, the transaction is rolled-back by compensating the activities that had
successfully completed. Due to the capabilities of existing services a technical process may enforce the use of a transactional model other than that required by the business process. Thus, when realizing transactional aspect of a business process (B) by using system services (S), the following realization type is identified:

**Model Mapping**: a transaction model (atomic or long-running) in the process B is supported by the same or a different model in the system service S.

To support realization of the transactional aspect, system services must have such transactional properties that support the transactional states required by the transaction model in the business process. This leads to the following conditions for discerning the realization levels of the process B (Table 5):

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LFR</td>
<td>The system services S enclosed in a transaction, differ in transactional properties from the corresponding activities in the process B.</td>
</tr>
<tr>
<td>CR</td>
<td>The system services S differ in transactional properties from the corresponding activities in the process B, but for some customer instances, the transactional requirements are satisfied (such as, for instance, visibility of intermediate results).</td>
</tr>
<tr>
<td>LLR</td>
<td>The system services S enclosed in a transaction have same transactional properties as required for the corresponding activities in the process B. This ensures the equivalence of transaction models in the process B and a final technical process.</td>
</tr>
<tr>
<td>ER</td>
<td>The system services S enclosed in a transaction, support both atomic and long-running properties.</td>
</tr>
</tbody>
</table>

The levels of realization as defined in this section, give the possibility to assess the realization of a business process. The process can be examined on a per-aspect basis to discern if it is lossfull-, constrained-, lossless- or exceeded realizable.

By following the discussion of the Sandvik’s case (given in the Example part in Tables 1-5) we conclude the following: the realization of the functional aspect is constrained, the behavioural, organizational and informational aspect are lossless realized, while the transactional aspect is lossfull realizable. We conclude therefore that the business process in Figure 1(a) is lossfull realizable with the existing ERP services (i.e. the process in Figure 1(b) does not realize the original process).

As seen above, the use of the realization concept and its dissemination into four levels of quality is strongly governed by capabilities of existing services. Thus, in order to realize a business process the designer must have access to documentation of system constraints. In the next section we propose a set of concepts that are specifically targeted towards helping the process designer to select appropriate realization types.

### 4. Modeling system constraints

Limitations in system capabilities can affect all of the five aspects of process design. In this section we will examine how systems can be described in order to provide a structured overview of system service capabilities.

The starting point for structuring the service capabilities is the five aspects of process design. For each aspect, the designer of a technical process needs to select and apply the realization types that transform the business process into a technical process. This design process requires that existing service capabilities are documented in a way that helps the designer select the appropriate realization type. In particular, it is beneficial if the documented service capabilities can guide the designer to create a lossless realization of the business process, rather than a constrained, for instance.

In this section we examine the concepts needed to document service capabilities. The concepts were identified and unified in a conceptual model, by examining the documentation needed to support the procedure of transferring a business process to a technical process, using realization types.

When realizing the **functional aspect** the designer must match the (business) activities to system services that can perform the desired functionality. During realization the designer can select to use aggregation, specialization and constraint mapping realization types as described earlier.

The aggregation realization type can be used to bridge granularity gaps between the business and existing services. As we strive for lossless realization, it is important that the designer can find system services that are close to the granularity of business
process activities. Hence, a classification of the services into granularity levels is desired to support the designer when searching appropriate services. Furthermore, documenting possible service aggregations make it possible for the designer to quickly find multiple services that implement a single activity in the business level. To support the designer in the use of the aggregation realization type, we thus include two concepts in the system constraint model: service granularity level and service aggregation (see Figure 3, below).

System services might be designed to be executed in a certain order. This is common, for example, for the services in a single system interface. Services that where designed to be executed in a sequence, cannot be executed in parallel, or in the reverse order. The designer must be aware of this when transforming the business process into a technical process using the reordering realization type. We therefore introduce the concept of a sequence dependency in the system constraint model. Note that a sequence dependency between two services only indicates that the services should be executed in a certain sequence. Unlike a control flow dependency, a sequence dependency allows the designer to insert additional services between two services that have a sequence dependency.

The lossless use of the branch mapping realization type is dependant on the existence of the system services that support business-level process branches. Therefore, the designer can take advantage of the system constraint model concepts of services and aggregation of services to find appropriate services to implement each branch.

The informational aspect of a technical process is defined by applying the information mapping realization type. A vital part of the design is to ensure that all the information structures that exist on the business level are supported by the system services. The information structures are represented by the input and output messages of the system services. To enable the designer to browse the information structures and to select the services that support the business level information needs, the system constraint model contains the message concept. In the model in Figure 3, the structure of each input and output parameter may be represented by a nested structure of messages. More advanced models might include the use in generalization and the notion of primitive data types to further refine the message structure definition.

Designing the organizational aspect of a process is done by using the role mapping realization type. The role mapping realization type maps a business role to a system that is to execute services on behalf of the business role. Deciding if a service should be part of a technical process is therefore not just a matter of matching service messages and goals. The service must also conform to a correct role-mapping realization type. This means that a service must execute within a system owned by an organizational unit that corresponds to a business role. To help the designer ensure that the service is owned by the appropriate organization the system constraint model contains the concepts of system and organization. System is used as a convenience concept, to depict groups of services.
belonging to the same organization unit. The organizational concept is used to depict legal entities, such as the organizational departments in a company.

The transactional aspect depicts the transactional model used by the activities in a business process, and the services in a technical process. The model mapping realization type is used to map transactional models from the business to the technical process. In order to achieve a lossless realization, the designer needs to select services that enforce the same transactional model as the business level. To enable the selection of these services the system constraints model includes the simple concept of transactional model. The concept has the extent {no transaction support, atomic, long-running}.

The outlined system constraint model is derived from the information required when applying realization types to transform a business process to a technical process. Clearly, this kind of model is an aid in the development of new technical processes. In the next section, we discuss how a combination of the concepts of system constraints, realization types and realization levels enable a continuous process evolution.

6. Process development and evolution

The concepts presented in this paper can be used to support both the development of new technical processes, as well as to support the evolution of those processes. Developing new processes consists of selecting the transformations (realization types) and utilizing the system services that yields the highest possible realization level. Later, in process evolution, it must be possible to track how changes of system constraints affect an existing business process realization.

The key issue to enable this kind of tracking is to represent the use of realization types and the system constraints that affected the selection of the realization types in a unified model. A high-level version of such a model, based on the concepts presented in this paper, is shown in Figure 4. The system constraints of each system service presented in the model are expressed in a uniform and structured way, as described in the previous section. The applied realization types are documented in the form of realization cases. Each realization case has a level of realization (lossless, constrained, etc).

When creating a process realization based on existing system services the aim is to select and use those services that provide the highest realization level. As a first step in developing technical processes the system constraint model (Figure 3) can be used to select appropriate services in a stepwise manner:

- The concepts of organizations and systems are used as an entry point to find the services that a business partner provides, for example the customer in Figure 1 might already have predefined services for receiving product information. The granularity levels can further help the designer to browse the existing services using a top-down approach, starting with high-level services.
- The candidate services selected in step 1 can be further examined by using the models concepts of transaction model and the concept of specialization groups can be used last, to ensure proper transactional behavior and, if possible, provide alternative implementations of the business activity.

By applying the above outlined steps, it is possible to find services that match each activity on the business level. The services selected for inclusion in the process are documented by registering a set of realization cases (Figure 4, below).

![Figure 4. High-level traceability model](image-url)

Once the business process is realized, the model from Figure 4 may be continuously used to check the consistency between the business processes and system constraints. When changing existing system services, according to technology updates, organizational restructuring, etc., the model is used to control the validity of the existing business process realization. We distinguish the following ways along which system constraints may be changed:

- **Extension** – when a new service is added to the system. For instance, in the example case in Section 2, adding a “send order” service to the existing system with the specialization to the SMTP protocol, would lead to the upgrade of the realization case from the lossless to exceeded (as none of the customers’ instances require SMTP protocol). The realization upgrade is detected by relating the service with adequate specialization group (Figure 3).
- **Update** – when existing service is changed in some of its capabilities. In the case example, updating the existing services to support the atomic transactional model for the required transaction, would improve the related realization case to the lossless level.
- **Remove** – when an service is removed from the system. The change may only downgrade level of related realization cases. For instance, removing the service for sending the product information by the FTP protocol in the example case, will downgrade the related realization case from the lossless to a constrained.

Taken together, the presented concepts are instruments to the quality controlled development of new processes as well as their future evolution.

### 6. Conclusion

In this paper we set out to bridge the gap between business processes and their realization and evolution in a technical system environment. The solution as discussed in this paper consists of four interrelated parts. Firstly, we based our work on a set of realization types that depicts possible transformations that can be applied to business processes to transforms them into their implementation counterpart, technical processes. The realization types are identified by the use of a description framework that constitutes five main aspects of process design. Secondly, we identified four realization levels, which represent different levels of business process conformance. Thirdly, to support the process designer in the selection of realization types we proposed a model of system constraints, which provides the basic concepts needed for selecting realization types. Last, we discussed a basic framework that utilizes the aforementioned concepts to track changes in business and technical processes. The concepts presented in this paper together forms a basis for providing structured tool support for the development and evolution of processes that need to take advantage of existing assets in the form of software services.

This work can be extended to provide further enhancement and support for both the technical and business side of process design. An extension targeted towards business analysts would be to introduce means to express flexibility of business processes, based on realization types. A more flexible, or loose definition of a business process will result in a wider range of possible realization types, and therefore would more likely lead to a lossless, or even exceeded lossless realization.

### References