Collaboration Viewpoint for Modeling Cross-Organizational Business Concerns

Ayalew Kassahun(✉) and Bedir Tekinerdogan

Information Technology Group, Wageningen University, Hollandseweg 1,
Wageningen, The Netherlands
{ayalew.kassahun,bedir.tekinerdogan}@wur.nl

Abstract. Organizations very often need to collaborate to achieve their business goals. Hereby it is important that the collaboration concerns are properly identified and reflected in their businesses. In practice, the business process design and architecture design are often carried out separately. This often leads to a misalignment between the business process and architecture design, a problem which becomes more severe when multiple collaborating organizations are involved. To address this problem, it is important to provide the proper design abstractions that can be used to detect and correct misalignments. To this end, we propose the architecture collaboration viewpoint that can be used by teams of business analysts and software architects when addressing business collaboration concerns. The collaboration viewpoint uses elements from business process and architecture viewpoints to provide new modeling artifacts for alignment. The design artefacts are mapping tables and workflow pattern diagrams that are used to identify misalignments and redesign the business processes. The viewpoint facilitates the communication between business analysts and architects. We illustrate the collaboration viewpoint for a food supply chain transparency system from a real industrial case study.

Keywords: Architecture viewpoint · Business collaboration · Collaboration viewpoint · Business process modeling · Workflow patterns

1 Introduction

Businesses today rarely operate in isolation but must collaborate with others in a coordinated fashion. To address collaboration concerns business analysts design business process models (BPMs) that integrate business activities across the collaborating organizations. BPMs have to be supported by underlying software systems, and therefore, BPMs will have a direct impact on the required software systems and the corresponding architectural design. Conversely, the architectural design imposes constraints on BPMs, and as a consequence, an inherent, mutual dependency exists between these two sets of designs.

Business collaboration involves BPMs that span multiple organizations – which we hereafter refer to as *business collaboration processes*. When realizing business collaboration processes multiple software systems need to be taken into account. As a result, the mutual alignment of BPMs and architectural designs becomes very

E. Cabello et al. (Eds.): ICSOFT 2016, CCIS 743, pp. 3–21, 2017.
DOI: 10.1007/978-3-319-62569-0_1

cumbersome. We define the difficulties associated in aligning the two designs as *business collaboration concerns*.

The current practice addresses business process concerns and architectural concerns separately, and sequentially—first the BPMs are designed then the software architecture is designed using the BPM models as inputs. This approach is to an extent feasible if applied within the context of an individual organization. However, when dealing with multiple software systems from different organizations the approach becomes infeasible due to the mutual dependency between business process models and the software architecture.

To address the problem we studied the existing modelling approaches. At present, two distinct sets of viewpoints are used to address business collaboration concerns. Various architecture viewpoints are used for modelling the structure of software systems, which we hereafter referred to as *structural viewpoints*. Business process models and notations are used for modelling business processes and are hereafter referred to as *business process viewpoints*. The structural viewpoints do not directly address business process concerns. Likewise, the business process viewpoints do not consider architectural concerns. As a consequence, a business-IT alignment problem arises. The alignment problem has been discussed in the context of individual organizations (Avison et al. 2004; Hong-Mei 2008; Bartens et al. 2014; Aversano et al. 2016) but not in the context of business collaborations.

In this paper we introduce the *collaboration viewpoint* for addressing business collaboration concerns. In the collaboration viewpoint we use architectural and business process viewpoints to provide new kinds of models with the corresponding iterative design process for applying them. We introduce mapping tables and use workflow patterns as a means of identifying misalignment and redesigning the BPMs. The collaboration viewpoint is meant as means of enabling teamwork between software architects and business process analysts. The teamwork ensures that the business process and architecture views are well-aligned and feasible. We illustrate the viewpoint in real industrial case study for which a safety and quality transparency system for food supply chains is designed.

The remainder of this paper is organized as follows. Section 2 provides background information. Section 3 presents the case study used to demonstrate the collaboration viewpoint. Section 4 presents the collaboration viewpoint and a method for applying it. In Sect. 5 the viewpoint is applied to the case study. In Sect. 6 the related work is presented and in Sect. 7 concluding remarks are made.

2 Background

In this section we first discuss the background on software architecture, BPM, and workflow patterns.

2.1 Software Architecture

Software architecture defines the gross-level structure of a software system (ISO/IEC/IEEE 2011). Architecture modeling is important to enhance the understanding of the software system, support the communication among stakeholders, and guide the development process (Tekinerdogan 2014). A common practice to modeling architecture is using different architectural views that address the concerns of a specific group of stakeholders. Architectural views document the architectural design decisions from a specific viewpoint. That means, the designs documented in an architectural view follow the conventions, including models and notations, defined in the corresponding architectural viewpoint. From a given architectural viewpoint one or more architectural views can be designed (Clements et al. 2010; ISO/IEC/IEEE 2011).

In the literature, a number of viewpoints have been identified (Kruchten 1995; Hofmeister et al. 2000; Kruchten 2004; Lattanze 2008; Clements et al. 2010). The Views and Beyond (V&B) approach identifies three major viewpoints: *module*, *component-and-connector (C&C)*, and *allocation*. Module views deal with concerns related to implementation, such as, decomposition and generalization. The C&C and allocation viewpoints are structural viewpoints since they largely refer to the structure of the software system. The C&C views deal with the interaction structure, such as, data flow and message routing. The allocation viewpoint describes how software elements are allocated to the environment of the software system, such as, hardware or development team (Clements et al. 2010).

Recognizing that new viewpoints may be needed to address new kinds of concerns, the ISO/IEC 42010 standard for documenting software architecture (ISO/IEC/IEEE 2011) provides an extensible metamodel for defining new viewpoints.

2.2 BPM

A business process describes how the activities for achieving a particular business outcome are interrelated and how they are executed (Davenport and Short 1998). The process modelling approach has historically gained the attention of businesses when it was effectively used to address inefficiencies in functional organizations (Dumas et al. 2013). At its core, a BPM identifies the events of the business process and the series of activities that are triggered by them (Dumas et al. 2013). In practice, business processes are modeled by business analysts using visual modelling methods. The most prominent business processes modeling language is BPMN (Business Process Model and Notation) (ISO/IEC 2013). BPMs address business requirements, and as such, are inputs for the software architects as requirements that should to be addressed in the architectural design (The Open Group 2013).

2.3 Workflow Patterns

Workflow patterns are recurring problem-solution pairs that have been frequently used in business process modeling (Russell et al. 2006). In fact, BPMs can be viewed as being composed of workflow patterns. Since workflow patterns represent well-known problem-solution pairs, it is easier to describe, discuss and redesign a BPM by manipulating its constituent workflow patterns.

In the past, more than a hundred workflow patterns have been identified, categorized and cataloged (van der Aalst and ter Hofstede 2011). The most prominent categories are *control-flow*, *data-flow* and *resource-flow* workflow patterns (Van der Aalst et al. 2003). Control-flow patterns model the execution ordering of activities and are the basis for the patterns in the other categories. The data-flow patterns model how data flows along the flow of control. The resource-flow patterns model how work is assigned to resources (*e.g.* devices, people) following the flow of control. A short summary of workflow pattern categories and the workflow patterns in each category is provided in a previous publication (Kassahun and Tekinerdogan 2016).

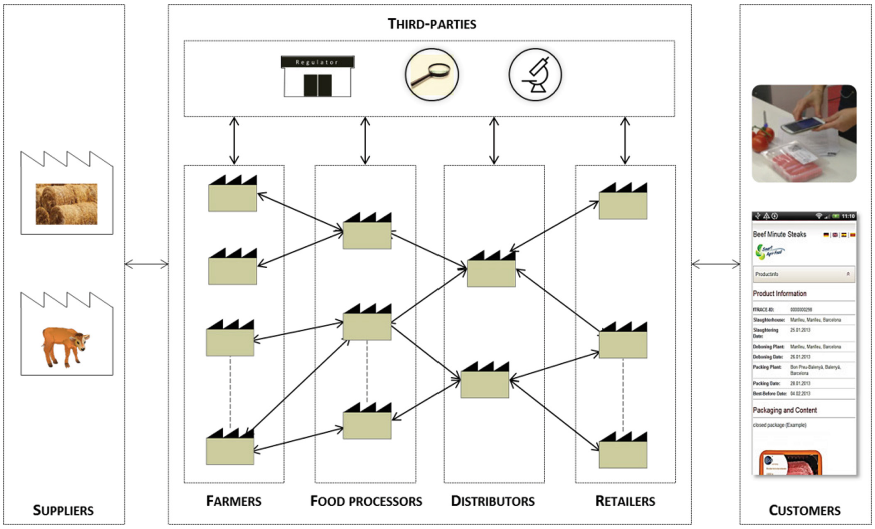
3 Illustrative Case and Problem Statement

In this section we use a case study from the FIspace business collaboration research project (Verdouw et al. 2014) to illustrate collaboration concerns and describe the problem statement.

3.1 Case: Transparency in Food Supply Chains

A food supply chain network is a collaboration linkage of a series of food operators that transform agricultural input products into finished food products. The food operators involved include farmers, a series of food processors and distributors, and retailers. In addition, mandated by food regulations, various third-parties are involved to guarantee the safety and quality of food. In Europe, for instance, recurring food scandals and crises have led to regulations that mandate centralized animal registry systems (EC 2000; EC 2004; EC 2015) and procedures for tracking and tracing of food products (EC 2002; EC 2007; EC 2011). Guaranteeing the safety and quality of food requires, among other things, the smooth flow of transparency data. Transparency in food supply chains refers to the ability to track and trace input, intermediate and finished food products along the supply chain. A conceptual model of a food supply chain network is depicted in Fig. 1.

Transparency involves two basic business processes: *data capture* and *data query*. These business processes are implemented within the individual food operators (*internal transparency*) as well as across the supply chain (*external transparency*). A software system that realizes internal transparency is referred to as *Internal Transparency System (ITS)*; the integration of internal transparency systems that realizes external transparency is referred to as *External Transparency System (ETS)*. Recently, the GS1 system architecture is increasingly being adopted (GS1 2015) in realizing both internal and external transparency systems. The EPCIS (Electronic Product Code Information System) specification (EPCglobal 2014), which is part of the GS1 System Architecture, provides generic data models and interface definitions for both data capture and data query business processes.



Beef Minute Steaks

Product Information

ITRACE-ID:	0000000298
Slaughterhouse:	Manlleu, Manlleu, Barcelona
Slaughtering Date:	25.01.2013
Deboning Plant:	Manlleu, Manlleu, Barcelona
Deboning Date:	26.01.2013
Packing Plant:	Bon Preu-Balenyà, Balenyà, Barcelona
Packing Date:	28.01.2013
Best-Before Date:	04.02.2013

Packaging and Content

closed package (Example)

Fig. 1. A conceptual model of food supply chain networks. (Arrowed lines represent the flow of information through the network.).

We elaborate business collaboration concerns using the data query BPM depicted in Fig. 2. The BPM complies with the EPCIS specification, and is considered the preferred scenario. However, many food operators cannot support it. In the following, we first describe the BPM and then state the collaboration concern related to the model. The data query BPM is initiated when an end-user takes a food product—which can be input, semi-finished or end product—at a food operator and requests transparency data from the food operator's ITS. For the sake of simplicity we assume that each individual food product item has a unique ID and the ID is obtained by scanning the barcode of the product item. The end-user obtains transparency data using a barcode scanner or a smartphone application (*End-User App*). Upon scanning a barcode, the end-user app makes a query request and displays the transparency data returned. When the end-user scans a product item, the app requests transparency data from the food operator (indicated as *focal*). The ITS of the food operator determines where the product data reside. If the data reside locally it fetches the data from its own database; otherwise, it looks up the service address of the food operator (indicated as *partner*) that has the required data at a third-party discovery service. It then makes a query request to the partner food operator ITS, upon which the partner ITS returns the data it has about the item. Since the product may have passed through many food operators—and since transparency data about the ingredients are also part of the transparency data of a product item—this process is repeated until no more transparency data is desired or no more transparency data can be obtained. The focal and partner food operators are identical but drawn in two separate lanes to be able to show the interactions among the food operators clearly. Note, the focal food operator lane represents the one food operator that received the request from the end-user; the partner food operator lane represents all other food operators involved. After all data is gathered, the focal food operator sends the aggregated data to the app, which displays the data to the user.

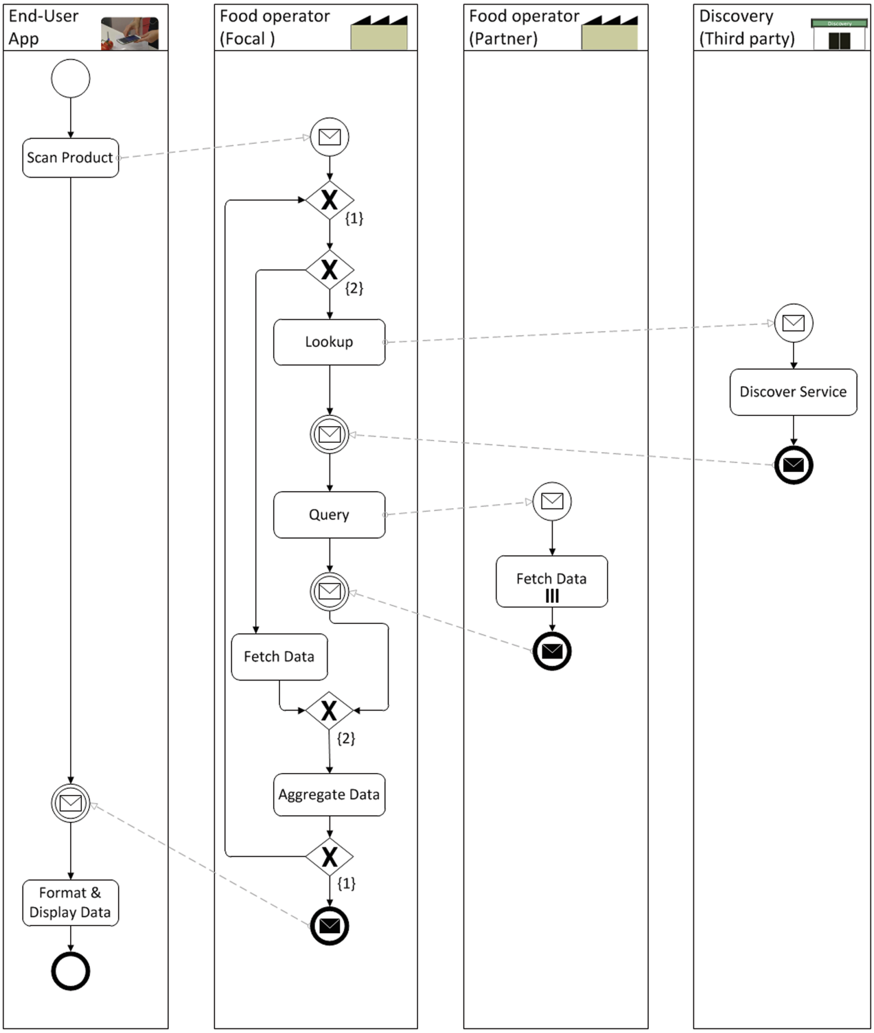


Fig. 2. A BPM showing how transparency data is queried across a food supply chain.

The BPM shown in Fig. 2 has to be implemented by all the four types of food operators shown in Fig. 1. The end-user app should also be provided by the food operators. However, in practice, many of the food operators do not support most of the activities the BPM and cannot provide end-user apps.

3.2 Problem Statement

In the previous sub-section we have described food supply chains and illustrated an inherent business collaboration concern they face regarding transparency. In the case study we identified a number of problems in aligning the BPMs representing the preferred scenario and the software systems that realistically can be realized by the collaborating partners. Specifically, we can define the following problems:

- *Difficulty in realizing business collaboration processes*

The elements of BPMs have to be supported by businesses depending on their roles. That is, process elements, such as, events, tasks and gateways have to be realized by architectural elements, such as, modules, components and nodes of the software systems that are distributed across many businesses. It turns out that the mapping of BPMs to the diverse software systems is not straightforward. For example, the BPM shown in Fig. 2 spans many food operators, many of which are, in practice, not capable of fulfilling all the steps. Particularly, many of the small food operators (mainly farmers) cannot afford to deploy the required software systems.

- *Lack of a common model for supporting the interaction between business analyst and architects*

Faced with the problem stated above business analysts and software architects from the various businesses come together to address the problem. However, the two stakeholder types use two separate sets of models hampering the communication between them. Business analysts use BPMs to define business processes. On the other hand, software architects use architecture viewpoints that mainly address concerns related to the structure of the software system. For the given case study, it was required early on to know which activities *can* be fulfilled by which food operators. Neither the business process models nor the software architecture views provide this information. A common model that depicts the business collaboration concerns (a model that maps elements of BPMs to elements of architectural design) would help to support the communication and the design rationale.

- *Early validation of the business process-architecture alignment is difficult*

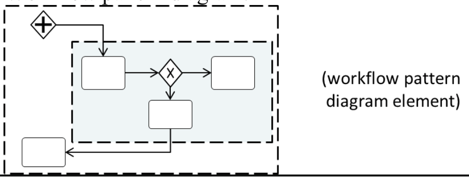
Too often BPMs are validated after the software system is realized creating major risks. For example the BPM of Fig. 2 has an impact on the software components that need to be deployed at each food operator node. Given only this BPM and the corresponding architectural designs, it is not easy to validate that the two are aligned and feasible.

In light of the above obstacles we formulated the following general research question: How can we support software architects and business analysts to design BPMs and the corresponding software architecture as a team and minimize the mismatch between the two designs?

4 Collaboration Viewpoint

Adapting the template for documenting architecture viewpoints proposed in the ISO/IEC standard mentioned before we propose a collaboration viewpoint shown in Table 1. The key stakeholders for the viewpoint are identified as *software architects* and *business analysts*. In the collaboration viewpoint we adopt the BPMN modelling

Table 1. Collaboration viewpoint documentation guide.

<table>
<tr><td><i>Element</i></td><td><i>Description</i></td></tr>
<tr><td><i>Name</i></td><td>Collaboration Viewpoint</td></tr>
<tr><td><i>Stakeholders</i></td><td>• Business Analysts
• Software Architects</td></tr>
<tr><td><i>Elements</i></td><td>• Elements of BPMN 2.0
• Elements from structural viewpoints (mainly Allocation and C&C)
• Workflow patterns</td></tr>
<tr><td><i>Relations</i></td><td>• Maps to—BPMN models to collaborating partners
• Maps to—structural models to collaborating partners
• Maps to—BPMN models to workflow patterns</td></tr>
<tr><td><i>Properties and constraints</i></td><td>• Constraints as defined business process and structural viewpoints, and workflow patterns</td></tr>
<tr><td><i>Notation</i></td><td>• BPMN and notations used in structural viewpoint models
• Tables for mapping model elements to collaborating partners:
<table>
<tr><td>Collaborating partners
BPMN Elements</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Events</td><td colspan="4"></td></tr>
<tr><td>…</td><td>…</td><td>…</td><td>…</td><td>…</td></tr>
<tr><td>Activities</td><td colspan="4"></td></tr>
<tr><td>…</td><td>…</td><td>…</td><td>…</td><td>…</td></tr>
<tr><td>Gateways</td><td colspan="4"></td></tr>
<tr><td>…</td><td>…</td><td>…</td><td>…</td><td>…</td></tr>
</table>
<table>
<tr><td>Collaborating partners
Structure Elements</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Components</td><td colspan="4"></td></tr>
<tr><td>…</td><td>…</td><td>…</td><td>…</td><td>…</td></tr>
<tr><td>Nodes</td><td colspan="4"></td></tr>
<tr><td>…</td><td>…</td><td>…</td><td>…</td><td>…</td></tr>
</table>
• Tables for mapping workflow patterns to model elements
• Workflow pattern diagram:


</td></tr>
<tr><td><i>Relation to other views/viewpoints</i></td><td>BPMN 2.0 specification, workflow patterns catalogue, allocation and C&C viewpoints.</td></tr>
<tr><td>Examples</td><td>See section V</td></tr>
</table>

method to represent BPMs. BPMN is widely used among business analysts and is also easily understandable for software architects. BPMN models are used for three reasons. First, we use them to represent business collaboration. Second, we map BPMN elements to organizations in mapping tables so that we can reallocate them to a different organization during redesign. Third, we map fragments of BPMN models to workflow patterns so that we can redesign the business collaboration process based on well-understood patterns.

In addition to the business concerns the collaboration viewpoint uses elements of the *C&C* and *allocations* views. Hereby, we consider only the elements of the models of these structural views as modelling elements in the corresponding mapping table. The architectural elements we consider most relevant are *components* and *nodes*.

The mappings of business process and architectural elements are made using two tables shown in Table 1. The first table captures how business process elements are allocated across the collaborating partners; the second table captures how architectural elements are allocated across the collaborating partners. The tables are used for both redesign and validation purposes.

Workflow patterns are represented using a *workflow pattern diagram* which can also be represented as workflow mapping table. The workflow pattern diagram is a BPMN diagram on which the BPMN elements that belong to distinct workflow patterns are delineated using dashed-line blocks. To delineate the BPMN elements the BPM diagram will mostly require simplification. The creation and application of workflow pattern diagram is demonstrated in Sect. 5.

4.1 Method for Applying the Viewpoint

Figure 4 shows the method for applying the collaboration viewpoint. The method is started by business analysts; they first design the business collaboration models as BPMN models (step 1) and subsequently identify the relevant workflow patterns (step 2). The two steps are displayed sequentially but, in reality, they are intertwined. Next, in step 3, software architects model the structural views of the software architecture.

In step 4 the business analysts and the software architects work as a team to allocate elements of the BPMN and architectural views to collaborating partners using mapping tables. They use the workflow pattern diagrams to facilitate the allocation. In this step they identify misalignments and determine if redesign is required. If redesign is required the next (in step 5) they identify possible redesign business process elements, architectural elements and workflow patterns based on the insights gained from the mapping tables. In fact, the mapping tables are used reallocate elements. Then, either the entire process or part of it is repeated until no redesign is required. Finally (in step 6), the BPMs, the workflow pattern diagrams and the mapping tables are documented in collaboration views following the documentation outline proposed in the next sub section (Fig. 3).

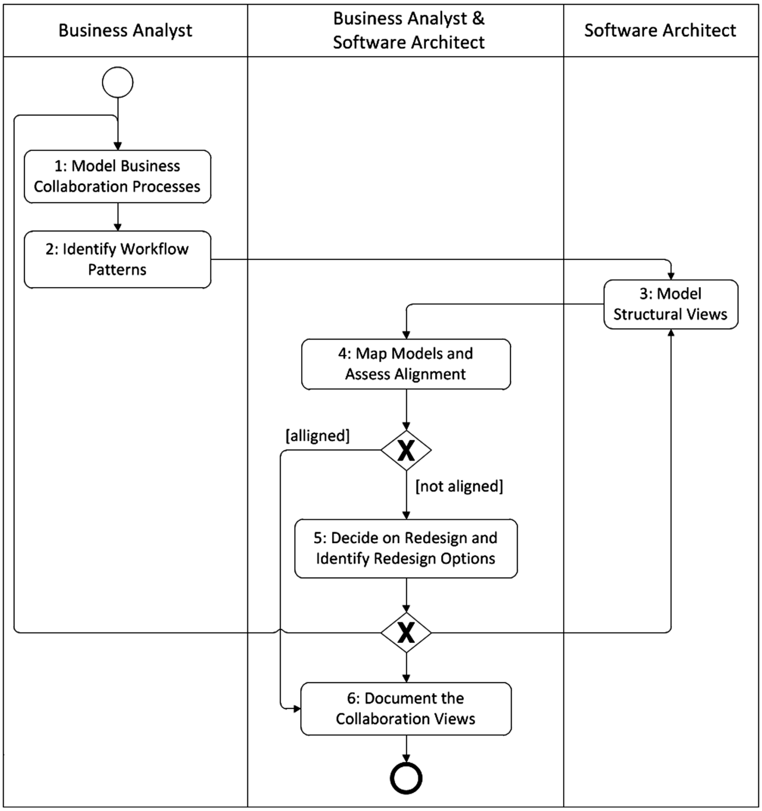

Business Analyst
Business Analyst & Software Architect
Software Architect
1: Model Business Collaboration Processes
2: Identify Workflow Patterns
3: Model Structural Views
4: Map Models and Assess Alignment
[alligned]
[not aligned]
5: Decide on Redesign and Identify Redesign Options
6: Document the Collaboration Views

Fig. 3. A process diagram representing the process of modeling a collaboration view.

5 Applying the Collaboration Viewpoint

In this section we illustrate how the approach shown in Fig. 4 is applied in the real industrial case mentioned in Sect. 3. The first step of designing the BPMs is already demonstrated in the business collaboration model shown in Fig. 2. The second step is identifying the workflow patterns.

Figure 4 shows the main workflow patterns of the business collaboration model we identified, which are: *sequence* (*cf-1*), *exclusive choice* (*cf-4*), *simple merge* (*cf-5*), *multiple instances without synchronization* (*cf-12*) and s*tructured loop* (*cf-21*). A further analysis shows that the workflow patterns *cf-4 and cf-5* belong together. Similarly, the workflow patterns *cf-21 and cf-12* belong together. Therefore, we identify three workflow patterns, two of which are composite patterns.

The third step of the approach is to capture the existing software architecture that is already in place. For the sake of simplicity we distinguish between two major groups of food operators in terms of their existing software systems, *i.e.* their ITSs: *small food operator* (*FO_{small}*) and *large food operator* (*FO_{large}*), and a single *third party* (*3P*). Similarly, we identify three components of an ITS: a *data query component*, a *data*

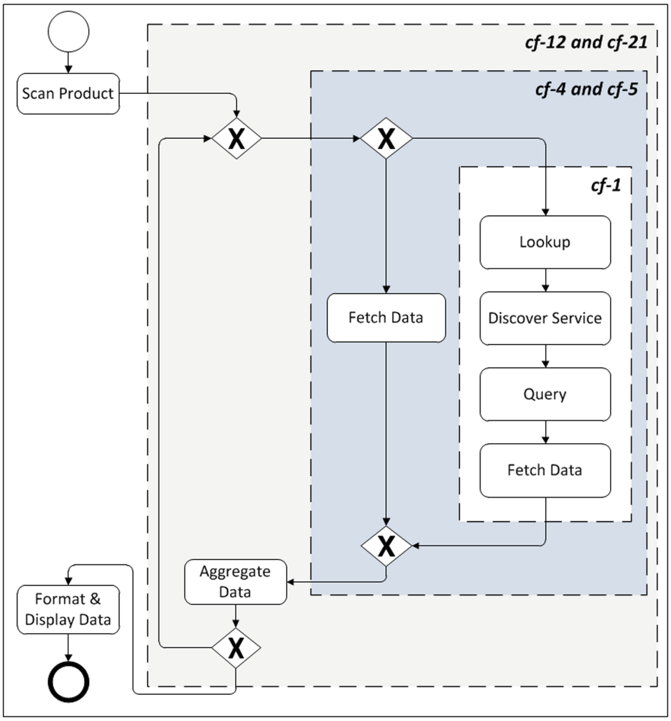


Fig. 4. A workflow pattern diagram of the query business collaboration process.

aggregator component and a *product data repository service*. In relation to the business collaboration process shown in Fig. 2 the data query component implements the *lookup* and *query* tasks and the 2nd XOR decision; the data aggregator component implements the aggregate data task and the 1st XOR decision; the data retrieval service implements the *fetch data* task.

The next step, step 4, is mapping the allocation of the elements of the business collaboration model and the architectural design to the collaborating partners. Table 2 shows how the BPM elements are allocated across the two types of food operators; Table 3 does the same but for architectural elements. The tables are interpreted as follows. A '+' sign in a cell implies that the business process or the architectural element is allocated to the corresponding collaboration partners and the collaboration partner indeed supports the element. For instance, the *scan product* task should be supported in *large food operator* nodes and it is indeed supported. A '–' sign implies that the business process or the architectural element is allocated to the collaboration partner but the collaboration partner fails to support the element. Using the above example, the *scan product* task should have been supported by *small food operator* nodes but it is not. An empty cell implies that the element is not relevant for the specific collaboration partner. A '∼' sign implies that the business process or the architectural element is not allocated to the collaboration partner according to the models but in reality the collaboration partner supports the element. For instance, in the given supply

Table 2. Mapping of business process elements to the corresponding collaborating partners.

BPMN elements	Collaboration partners		
	FO_{small}	FO_{large}	3P
Events			
Start	–	+	∼
End	–	+	∼
Gateways			
XOR {1}	–	–	∼
XOR {2}	–	–	
Tasks			
Scan product	–	+	∼
Lookup	–	–	
Discover service	–	–	–
Query	–	–	
Fetch data	–	+	∼
Aggregate data	–	+	∼
Format and display data	–	+	∼

Table 3. Mapping architectural elements to the corresponding collaborating partners.

Structural elements	Collaboration partners		
	FO_{small}	FO_{large}	3P
End-user app	–	+	∼
Query component	–	–	
Data aggregator component	–	–	∼
Product data repository service	–	+	∼
Discovery service			+

chain a third party provides, for part of the supply chain, an end-user transparency app and transparency system that supports a number of tasks. An empty cell implies that the element is not relevant for the specific collaboration partner. Typically, these tables require knowledge of the state of affairs in all collaboration partners, which could be many, and it may require more fine-grained attributes than the simple +, –, ∼ and *blank* entries. As shown in the table, it turns out that small food operators implement none of the required architectural elements adequately, large food operators provide only part, and a third party seems to fill the gap left by the food operators, albeit partly.

From Tables 2 and 3 it is clear that the desired business processes are not aligned with the existing architecture. The next step, step 5, is redesigning the business collaboration process by identifying better fitting workflow patterns, structural components and allocations. Obviously improved versions of Tables 2 and 3 are required. For instance, though small food operators do not fulfill the allocated tasks, it turned out that

Table 4. New allocation of architectural elements to collaborating partners.

Structural elements	Collaboration partners		
	FO_{small}	FO_{large}	3P
End-user app			+
Query component			
Data aggregator component			+
Product data repository service		+	+
Discovery service			+

they are, however, willing to (and usually do) delegate the tasks to a third party and pass the required transparency data to it that enables it to perform the delegated tasks. This is also consistent with some aspects of the food laws described in Sect. 3.1 that require centralized repositories of transparency data to be managed by third-parties or regulatory authorities.

In Table 4 we show the improved allocation of architectural elements that eliminates the misalignment identified in Table 3. (Similarly, a new allocation table for Table 2 can be produced but is not included for brevity.) The new allocation allows all food operators (small and large) to comply with the EPCIS specification by formalizing the roles that the third party was playing. However, it raises a new issue related to data capture. Because food operators have to pass transparency data to the third-party that enables it to perform the new tasks assigned to it, the data capture (which so far was local and trivial) now becomes a collaboration concern.

Now that we identified redesign options, we start a new iteration to improve the business collaboration model and associated software architecture. We start by rede-signing the workflow pattern diagram because the workflow patterns identified earlier seem to capture the fundamental essence of the query BPM and may not need substantial modifications. The BPM, on the other hand, may change substantially. In Fig. 5 we provide the improved workflow pattern diagram that contains the same three workflow patterns but in a slightly different configuration. The change in the configuration of the workflow patterns is a direct consequence of the new allocation. The details of the consequences of the new allocation are shown in the new BPM provided in Fig. 6. As in the previous business collaboration process the new business process is triggered by the end-user app but all query requests are always sent to the third party. Instead of all food operators, the new model involves only large food operators in the query business process. Small food operators no longer need to maintain their own transparency data and to support the fetch data task, because the third-party supports this task on their behalf. When these and other business process redesign issues are resolved the software architects (re)design the software architecture. Then new mapping tables are produced to see if there are any misalignments that have to be addressed.

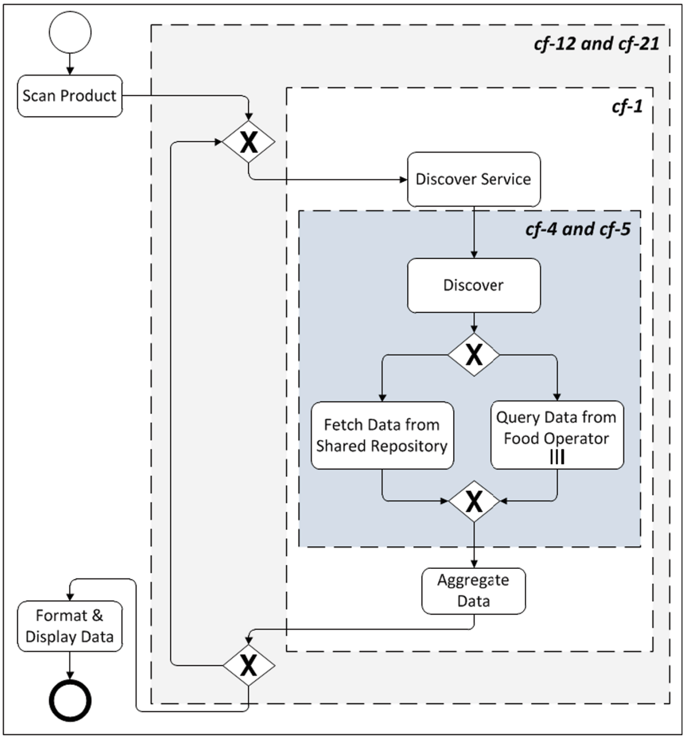


Fig. 5. Improved workflow pattern diagram of the query business collaboration process.

6 Related Work

The prominent way addressing business processes and software architecture concerns along with other concerns, such as general vision for the system, concerns related to technology, *etc.* in a consistent manner is to follow guidance provided by an enterprise architecture framework. The Zachman (Rational Software 2001) and TOGAF/ArchiMate (The Open Group 2013) frameworks are probably the most widely used and include the modeling of business processes and the designing of software architecture as part of the larger enterprise architecture. This framework use largely fixed categories of perspectives and concerns (*e.g.* vision, business concerns, software architecture concerns, *etc.*) Moreover, they follow a hierarchical conceptualization of models in which requirements cascade from vision, to BPMs, to software architecture and finally to technology architecture. A hierarchical approach suggests the use of elaborate methods to get the design at a higher hierarchical level before moving to the next. There are for instance extensive methods for analyzing the *as-is* BPMs and designing elaborate *to-be* BPMs (Sharp and McDermott 2009) before a large scale architectural design process commences. These approaches do not directly address

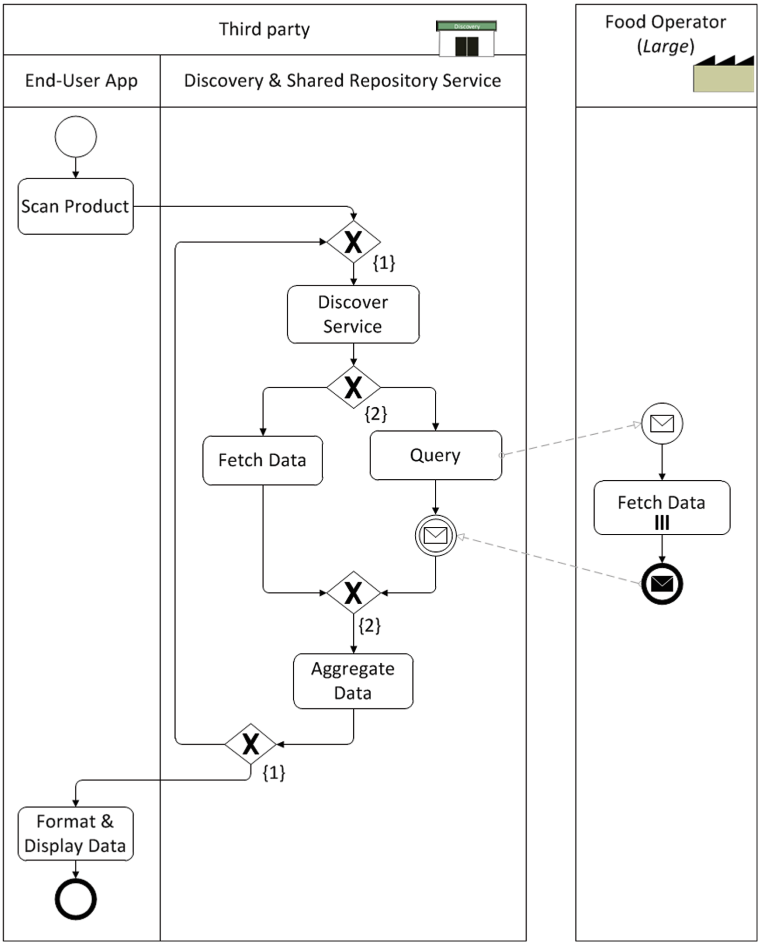


Fig. 6. Improved query business process model.

business collaboration concerns that often arise when different organizations are involved.

Business collaboration concerns could probably be addressed generically as cross cutting quality concerns across different viewpoints. In this respect business collaboration concerns could be viewed as concerns that cut across business process and architecture viewpoints. In this regard the concept of architectural perspectives is suggested that include a collection of activities, tactics and guidelines to be used across a number of the architectural views to address quality concerns (Woods and Rozanski 2005). In this context, Rozanski and Wood define several architectural perspectives for selected quality concerns such as security, performance, scalability, availability and evolution. In order to capture the system-wide quality concerns, each relevant perspective is applied to some or all views. In this way, the architectural views provide the description of the architecture, while the architectural perspectives can help to analyze and modify the architecture to ensure that system exhibits the desired quality properties. However, no architectural perspective for addressing business collaboration concerns has been addressed yet.

In this paper we have defined a collaboration viewpoint which is defined on top of structural viewpoints. Similarly, in our earlier work, we have considered the explicit modelling of viewpoints for quality concerns (Tekinerdogan and Sözer 2011) from viewpoints that address functional concerns. We have shown that quality concerns do not easily match the architectural elements that are primarily functional in nature. As a result, the communication and analysis of these quality concerns becomes more problematic in practice. We have introduced a general and practical approach for supporting architects to model quality concerns by extending the architectural viewpoints of the so-called V&B approach and illustrated the approach for defining recoverability and adaptability viewpoints (Sözer et al. 2013). In this paper we have focused on collaboration concerns which could also be seen as a non-functional concern. Like other quality concerns collaboration concerns require a dedicated viewpoint which we have discussed in this paper.

The collaboration viewpoint concerns the mutual alignment of BPMs and architectural designs and in this respect is closely related to architectural consistency analysis. Architecture consistency analysis has been mainly investigated in relation to consistency between software code and software architecture. Hereby, architecture consistency implies that the architecture design elements can be mapped to the implementation elements. In case the relationships between the architecture and implementation do not correspond then these are called architectural violations. If the relations that are present in the architecture are also found in the implementation then this is convergent relation. In case the architecture relation is not present in the implementation then this is called an absence relation. A successful design recovery technique that is used for architecture consistency checking is the reflexion modeling approach as proposed by Murphy et al. (Murphy et al. 2001). In this paper we have also focused on consistency of the architecture but now from a business model perspective in which we focused on business collaboration concerns.

In our earlier work (Tekinerdogan 2015), we have proposed to enhance existing reflexion modeling approaches using architecture viewpoints. We introduced the *architecture reflexion viewpoint* that can be used to define reflexion model for different architecture views. The viewpoint includes both a visual notation and a notation based on design structure matrices. The design structure reflexion matrices (DSRMs) that we have defined provide a complementary and succinct representation of the architecture and code for supporting qualitative and quantitative analysis, and likewise the refactoring of the architecture and code. For this we introduce the notion of *design structure reflexion matrices* (DSRM) and a generic reflexion modeling approach based on DSRMs.

In service-oriented architecture business collaboration concerns are addressed using choreography languages. In this respect recent research show that it is possible to automate the generation software from business processes choreography models (Autili et al. 2015). However, design heuristics for integrating misaligned business process and IT systems are largely missing.

7 Conclusion

The problem of business-IT alignment has been broadly addressed in the literature and several solutions have been provided for this. In this paper we have focused on the business process alignment with the architecture design. Further we have explicitly considered the alignment within the context of collaborating organizations. To this end we have identified three key collaboration concerns: ensuring that the BPMs are indeed supported by software components, ensuring that business analyst can communicate effectively with software architects in search of better design solutions, and validating the architecture with respect to the BPMs. The architecture collaboration viewpoint that we have proposed is novel from both the software architecture design perspective as well as the business process modeling perspective. We have shown that the viewpoint can support the communication among the business analysts and architects, and likewise help to align the business process models and the software architecture of the collaboration system. This has been justified by the application of the viewpoint to a real industrial case study on food supply chains. In our future work we will apply the viewpoint for other industrial cases.

References

Autili, M., Inverardi, P., Tivoli, M.: Automated synthesis of service choreographies. IEEE Softw. **32**(1), 50–57 (2015)

Aversano, L., Grasso, C., Tortorella, M.: Managing the alignment between business processes and software systems. Inf. Soft. Technol. **72**, 171–188 (2016)

Avison, D., Jones, J., Powell, P., Wilson, D.: Using and validating the strategic alignment model. J. Strateg. Inf. Syst. **13**(3), 223–246 (2004)

Bartens, Y., Schulte, F., Voss, S.: E-business IT governance revisited: an attempt towards outlining a novel bi-directional business/IT alignment in COBIT5. In: 2014 47th Hawaii International Conference on System Sciences (HICSS) (2014)

Clements, P., Bachmann, F., Bass, L., Garlan, D., Ivers, J., Little, R., Merson, P., Nord, R., Stafford, J.: Documenting Software Architectures: Views and Beyond. Addison-Wesley, Boston (2010)

Davenport, T.H., Short, J.E.: The new industrial engineering: information technology and business process redesign. IEEE Eng. Manage. Rev. **26**(3), 46–60 (1998)

Dumas, M., La Rosa, M., Mendling, J., Reijers, H.: Introduction to business process management. In: Dumas, M., La Rosa, M., Mendling, J., Reijers, H. (eds.) Fundamentals of Business Process Management, pp. 1–31. Springer, Heidelberg (2013)

EC: Regulation (EC) No 1760/2000 of the European Parliament and of the Council of 17 July 2000 establishing a system for the identification and registration of bovine animals and regarding the labelling of beef and beef productsand repealing Council Regulation (EC) No 820/97. Off. J. Eur. Communities **L204**, 1–10 (2000)

EC: Regulation (EC) No 178/2002 of the European Parliament and of the Council of 28 January 2002 laying down the general principles and requirements of food law, establishing the European Food Safety Authority and laying down procedures in matters of food safety. Off. J. Eur. Communities **L31**(1), 1–24 (2002)

EC: Regulation No. 911/2004 of 29 April 2004 implementing Regulation (EC) No 1760/2000 of the European Parliament and of the Council as regards eartags, passports and holding registers. Off. J. Eur. Union **L163**, 65–70 (2004)

EC: Factsheet: tracing food through the production and distribution chain to identify and address risks and protect public health, Brussels (2007)

EC: Commission Implementing Regulation (EU) No 931/2011 of 19 September 2011 on the traceability requirements set by Regulation (EC) No 178/2002 of the European Parliament and of the Council for food of animal origin. Off. J. Eur. Union **L242,** 1–2 (2011)

EC: Regulation (EU) No 2015/262 of 17 February 2015 laying down rules pursuant to Council Directives 90/427/EEC and 2009/156/EC as regards the methods for the identification of equidae (Equine Passport Regulation). Off. J Eur. Union **L59**, 51 (2015)

EPCglobal: EPC Information Services (EPCIS) Version 1.1 Specification. GS1 Standard Version 1.1, May 2014, Brussels, Belgium, GS1 AISBL (2014)

GS1: GS1 General Specifications, version 15, no 2, January 2015, GS1: 490 (2015)

Hofmeister, C., Nord, R., Soni, D.: Applied Software Architecture. Addison-Wesley Professional, Boca Raton (2000)

Hong-Mei, C.: Towards service engineering: service orientation and business-IT alignment. In: Proceedings of the 41st Annual Hawaii International Conference on System Sciences (2008)

ISO/IEC:. Information technology - Object Management Group Business Process Model and Notation. ISO/IEC 19510:2013 (2013)

ISO/IEC/IEEE: Systems and software engineering – architecture description. ISO/IEC/IEEE Standard 42010:2011 (2011)

Kassahun, A., Tekinerdogan, B.: Architecture viewpoint for modeling business collaboration concerns using workflow patterns. In: The 11th International Joint Conference on Software Technologies, Lisbon, 1: ICSOFT-EA, pp. 27–38 (2016)

Kruchten, P.: The Rational Unified Process: An Introduction. Addison-Wesley Professional, Boca Raton (2004)

Kruchten, P.B.: The 4 + 1 view model of architecture. IEEE Softw. **12**(6), 42–50 (1995)

Lattanze, A.J.: Architecting Software Intensive Systems: A Practitioners Guide. CRC Press, Boca Raton (2008)

Murphy, G.C., Notkin, D., Sullivan, K.J.: Software reflexion models: bridging the gap between design and implementation. IEEE Trans. Softw. Eng. **27**(4), 364–380 (2001)

Rational Software: The Zachman framework for enterprise architecture and rational best practices and products, a Rational Software White paper (2001). http://www.rational.com/

Russell, N., van der Aalst, W.M., Mulyar, N.: Workflow control-flow patterns: a revised view. BPM Center Report BPM-06-22 (2006)

Sharp, A., McDermott, P.: Workflow modeling: tools for process improvement and applications development. Artech House, Norwood (2009)

Sözer, H., Tekinerdoğan, B., Akşit, M.: Optimizing decomposition of software architecture for local recovery. Softw. Qual. J. **21**(2), 203–240 (2013)

Tekinerdogan, B.: Software architecture. In: Computing Handbook, Third edn., pp. 1–16. Chapman and Hall/CRC (2014)

Tekinerdogan, B.: Architectural drift analysis using design structure reflexion matrices. In: Sofware Quality Assurance in Large Scale and Complex Software-Intensive Systems, pp. 221–236. Elsevier (2015)

Tekinerdogan, B., Sözer, H.: Defining architectural viewpoints for quality concerns. In: Crnkovic, I., Gruhn, V., Book, M. (eds.) ECSA 2011. LNCS, vol. 6903, pp. 26–34. Springer, Heidelberg (2011). doi:10.1007/978-3-642-23798-0_3

The Open Group: ArchiMate 2.1 Specification (2013). http://pubs.opengroup.org/architecture/archimate2-doc/toc.html van der Aalst, Wil., M.P., ter Hofstede, A.H.M.: Workflow Patterns (2011). http://www.workflowpatterns.com/. Accessed 23 Dec 2015

van der Aalst, W.M.P., ter Hofstede, A.H.M., Kiepuszewski, B., Barros, A.P.: Workflow patterns. Distrib. Parallel Databases **14**(1), 5–51 (2003)

Verdouw, C., Beulens, A., Wolfert, S.: Towards software mass customization for business collaboration. In: 2014 Annual SRII Global Conference (SRII) (2014)

Woods, E., Rozanski, N.: Using architectural perspectives. In: 5th Working IEEE/IFIP Conference on software architecture WICSA 2005 (2005)